
BASIC ROM Routines

Page Last Altered: undefined

BASIC ROM Routines

by Christopher Dewhurst

BASIC IV ROM Routines

By Steve Fewell

I decided to begin something I've been wanting to do for a long time, that is to produce listings of BASIC 4 ROM's routines, together with detailed descriptions of how they work.

Please let me know what you think, and whether you'd like to see more like it. If anyone else wants to have a go at writing up with me then please feel free to post (added CJR: send them to 8BS) your own descriptions of routines from BASIC 4 (or another version of BASIC).

If I've made a mistake, and someone wants to correct/expand on something, then please do so.

Latest addtions:

19/Jan/2004	Addition of keywords: FOR, NEXT, ON, REPEAT and UNTIL
06/Jan/2004	Addition of the 'Get Line Number & find Program Address of the Line Number' routine and keywords: TRACE, GOSUB, RETURN, GOTO, RESTORE, READ and IF
28/ Dec/2003	Greetings during this Christmas and new year period! This update contains descriptions for the Trigonometry functions! Addition of keywords: TAN, ACS, ASN, ATN, SIN and COS

05/Oct/2003	Addition of the 'BASIC error handler' routine and keywords: CHAIN, RUN, LOAD, END, CALL, DELETE, RENUMBER, AUTO, USR, EXT, =PTR, BGET, OPENIN, OPENOUT, OPENUP, EOF, INSTR, STRING$, CHR$, SAVE, OSCLI, CLOSE and BPUT
12/Sep/2003	Addition of "'[' - Begin Assembly", "Execute '*'-command", "Execute next command line / program statement", 'skip end of program line' and 'Display current Line Number to screen [for TRACE]' routines and keywords: OLD, EXT= and PTR=
20/ Aug/2003	Addition of 'Print Line Number on screen', 'Calculate next Random Number Seed value', 'Output ASCII character/BASIC Token (in ASCII text)' routines and keywords: TIME=, TIME$=, MOVE, DRAW, PLOT, REPORT, VDU, RND, POS, VPOS, EVAL, INT, INKEY, INKEY$, SGN, POINT(, ADVAL, =TIME, =TIME$, LEFT$(, RIGHT$(, MID$(, EDIT, LIST, LISTO
01/ Aug/2003	Addition of 'BASIC ROM Startup Initialisations', 'Find positon of Program Line', 'Tokenise command line text', 'Insert line into Program', 'Remove line from program', 'Check program can be read correctly' and 'Detokenise Line Number' routines and keywords HIMEM=, LOMEM=, PAGE=, CLEAR, GCOL, COLOUR and MODE
10/Jun/2003	Addition of 'Assemble Assembly Statement', 'Output character to screen' and 'Extract next field' routines and keywords PRINT, PRINT#, CLG, SOUND, ENVELOPE, WIDTH, INPUT and INPUT#
01/ May/2003	Addition of Numeric to ASCII (NUMASC) routine and keywords: NOT, VAL, ASC, ABS, LEN, =PAGE, TOP, COUNT, =LOMEM, =HIMEM, ERL & ERR
09/Apr/2003	Addition of Disassembler program to the analysis disc images, Added LET & DIM keywords, and added &AD36 (get value routine)
24/ Mar/2003	Addition of Reference Disk Images, SQR & routine [&9909] to evaluate Variables and Arrays (and get address of the required value/variable parameter block).
28/Feb/2003	First of the Floating-Point Series calculations for functions such as LOG, LN and EXP

BASIC IV Routines

Routine Name	Start Address	Address Range	Comments
BASIC ROM header	8000	8000-8022	
BASIC ROM Startup Initialisations	802B	802B-8074	
Get Address of Variable	8085	8085-80CC	
Search for Program Line	80CD	80CD-80F8	
Integer Division	80F9	80F9-8171	

Convert Integer to Floating-Point	8185	8185-81CB	Also includes: 81E0-81E1
Set FWA to 1-byte value	81D5	81D5-81DF	
Normalise FWA#2	81E2	81E2-81F6	Also includes: 81CC-81D4
Normalise FWA#1	81F7	81F7-8241	
Convert Floating-Point to Integer	8242	8242-826E	Also includes: 8272-8274; 827E-82BC; 82BD-82BF; 82C0-82C3
Integer Reverse Order Complement	82C4	82C4-82DF	
Split FWA (into Integer/Fractional part)	82E0	82E0-830C	Also includes: 8275-827D
Calculate next Random Number Seed value	831E	831E-8348	
Copy FWB to FWA	8349	8349-8367	
Floating-point addition	8368	8368-8455	
BASIC Keyword List	8456	8456-8768	
BASIC Execution Address List	8769	8769-884C	
'[' Begin Assembly	8920	891A-89D3	
Assemble Assembly instruction/statement	89EB	89D4-8CD6	Also includes: 884D-891B
Replace untokenised ASCII value with a 1-byte token	8CEB	8CEB-8D03	
Tokenise Line Number	8D04	8D04-8D83	

Check for Variable name character or digit (in A)	8D84	8D84-8D9A	
Increment [and read] pointer (&37, &38)	8DA0	8DA0-8DAE	
Tokenise Command Line	8DB2	8DAF-8ED4	Also includes: 8D9B-8D9F
Next non-space Char PTRB	8ED5	8ED5-8EDF	Also includes: 8EEB-8EF0
Next non-space Char PTRA	8EE0	8EE0-8EEA	Also includes: 8CD7-8CEA
CHAIN	8EFB	8EFB-8EFF	
OLD	8F00	8F00-8F11	
RUN	8F12	8F12-8F1F	
LOAD	8F20	8F20-8F24	
END	8F25	8F25-8F2C	
BASIC ROM Startup initialisations (part 2)	8F2D	8F2D-8F7C	
NEW	8F7D	8F7D-8F82	Also includes: BEFE-BF13
Prompt for command line and execute the entered command (s)	8F83	8F83-8FA3	
Execute '*'-Command	8FA4	8FA4-8FAD	
Execute next command line / program statement	8FAE	8FAE-9049	
LET	904A	904A-905F	Also includes: 9072-9085

STOP	9086	9086-9088	
Set String variable	90AB	90AB-9140	Also includes: BE25-BE32
PRINT#	9141	9141-918C	Also includes: BA3C-BA57
PRINT	918D	918D-9229	Also includes: 927A-9293
(PRINT) TAB(	9241	922A-925A	Also includes: 9840-9844
(PRINT) SPC	925B	925B-9266	
(PRINT) quote (')	9267	9267-926E	
Get Result of expression from BASIC Text pointer A & convert to Integer	926F	926D-9279	
CALL	92BE	92BE-9313	
DELETE	9317	9314-934C	
RENUMBER	9384	934D-9488	
AUTO	9489	9489-94B8	
DIM	9534	9534-9604	Also includes: 94B9-9502, 9AF6-9B1B, BC43-BC50, BEEF-BEFD
HIMEM=	960F	960F-961F	Also includes: 96B9-96BD
LOMEM=	9620	9620-9633	
PAGE=	9634	9634-963D	

CLEAR	963E	963E-9645	
TRACE	9646	9646-9678	
TIME=	9679	9679-968D	
TIME$=	968E	968E-96A3	
Get Integer result of expression and check for closing bracket	96A7	96A7-96AB	
Get Integer result of expression	96AF	96AC-96B3	Also includes: 8EF1-8EF5
Get Integer value at PTR B	96B4	96B4-96B8	
Check if Integer and Convert if Float	96BE	96BE-96D9	
Get & Check Float value	96DA	96D7-96E3	
GCOL	9741	9741-9754	
COLOUR	9755	9755-975E	
MODE	975F	975F-97A1	
MOVE	97A2	97A2-97A5	
DRAW	97A6	97A6-97B0	
PLOT	97B1	97B1-97DF	
CLG	97E0	97E0-97E6	

CLS	97E7	97E7-97F3	
REPORT	97F4	97F4-9807	
VDU	980D	9808-983F	
Create new variable name in Variable Pointer table	9854	9854-9882	
Allocate space for new variable	9883	9883-98AA	
Evaluate variable name & Create if new variable	98AE	98AB-98BF	
'?' and '!' address peek/poke operators	98D1	98D1-98DB	
'$' address peek/poke operator	98DC	98DC-98EA	
Evaluate Variable name and return value address	9909	9909-99AD	Also includes: 98C1-98D0
'!' and '?' address modifier operators	99AE	99AE-99D4	
Get address of specified Array element	99FE	99FE-9AE9	Also includes: 9503-952B
Detokenise the requested Line Number and Set IWA to the Line Number value	9B1C	9B1C-9B45	
Check for '=', evaluate expression & check end of statement	9B52	9B52-9B5F	Also includes: 9B8E-9B95
Check for End of Statement	9BA6	9BA6-9BCE	Also includes: 9B96-9B99
Check & skip end of program line	9BCF	9BCF-9C04	
IF	9C08	9C05-9C4A	

Display current Line number to screen [for TRACE]	9C4B	9C4B-9C64	
Compare Float values	9C82	9C65-9CC5	
Compare Integer values	9CC9	9CC6-9D01	
Compare String values	9D02	9D02-9D2E	
Expression Handler	9D3B	9D2F-9D4B	Also includes: 9D7B-9D88, 9DA9-9DB4, 9E4C-9E57, 9FC1-9FDA, A00F-A026
'OR' Operator	9D4C	9D4C-9D65	
'EOR' Operator	9D66	9D66-9D7A	
'AND' Operator	9D89	9D89-9DA8	
'=' and other relational operators	9DB5	9DB5-9DCC	
'<' Operator	9DCD	9DCD-9DE0	
'<=' Operator	9DE1	9DE1-9DEB	
'<>' Operator	9DEC	9DEC-9DF4	
'>' Operator	9DF5	9DF5-9E06	
'>=' Operator	9E07	9E07-9E0F	
String Addition	9E22	9E22-9E4B	
'+' Operator	9E58	9E58-9E64	Also includes: 9E91-9EBC

Integer Addition	9E65	9E65-9E90	Also includes: 9E4F-9E57
'-' Operator	9EBD	9EBD-9EC9	Also includes: 9EE7-9F11; ACC7-ACD6
Integer Subtraction	9ECA	9ECA-9EE6	
'*' Operator	9F3B	9F12-9F63	
Integer Multiplication	9F64	9F64-9FDA	
'/' Operator	9FDB	9FDB-9FF4	
Integer MOD routine	9FF5	9FF5-9FFC	
Integer DIV routine	9FFD	9FFD-A00E	
'^' Operator (raise to Power)	A027	A027-A07F	
Print Line Number on screen	A085	A081-A0C9	Also includes: 8021-802A
Multiply FWA Mantissa by 10	A26C	A26C-A2BB	
NUMASC (convert Numeric value to ASCII value)	A118	A0CA-A2D9	Also Includes:
ASCNUM: Extract Number at PTRB	A2E1	A2DA-A35C	
ASCNUM: Handle Exponental values & complete number conversion	A35D	A35D-A3F1	
Floating-point Sign	A3F2	A3F2-A40A	
Copy FWA to FWB	A40B	A40B-A427	

Floating-point multiply by 10	A436	A436-A477	Also Includes: A428-A435
Floating-point divide by 10	A478	A478-A4DF	
Unpack FP Variable to FWB	A4E0	A4E0-A50C	
Store FWA to Temporary Variable	A50D	A50D-A518	
Pack FWA to Variable	A519	A519-A538	
Unpack FP Variable to FWA	A541	A539-A56F	
Clear FWB	A570	A570-A57E	
TAN	A59B	A59B-A5BD	
Raise FWA to the power of the integer value in A	A5BE	A5BE-A5E1	
Floating-Point Reciple	A5E9	A5E9-A5ED	Also Includes: A589-A591
Floating-Point Division Entry Point	A5EE	A5EE-A5F9	
Floating-Point Division	A5FA	A5FA-A689	Also includes: &A5E5 to &A5E8
Floating-Point Subtract Entry Point	A68A	A68A-A68C	
Floating-Point Addition Entry Point	A68D	A68D-A694	
Round FWA Mantissa to 4 bytes	A695	A695-A6A5	Also includes &A6AB to &A6B3.
Floating-Point Multiply Entry Point	A6A6	A6A6-A6AA	
Clear FWA	A6B4	A6B4-A6C4	

Floating-Point Multiplication	A6CF	A6CF-A745	
LN	A746	A746-A7B4	
SQR	A7B5	A7B5-A860	
Evaluate continued-fraction expansions Series	A861	A861-A89B	Also includes: A57F-A588 and A592-A59A
ACS	A89C	A89C-A8A0	
ASN	A8A1	A8A1-A8C2	
ATN	A8C3	A8C3-A90C	
SIN	A90D	A90D-A90D	
COS	A90E	A90E-A9AC	
RAD	A9C8	A9C8-A9CE	
LOG	A9CF	A9CF-A9D7	
DEG	A9D8	A9D8-A9DE	
EXP	A9DF	A9DF-AA11	
RND	AA73	AA1E-AA7F	
Load Integer from Zero Page Address	AA80	AA80-AA92	
NOT	AA93	AA93-AAA2	
POS	AAA3	AAA3-AAA8	

USR	AAA9	AAA9-AABB	
VPOS	AABC	AABC-AAC4	
EXT	AAC5	AAC5-AAC8	
=PTR	AAC9	AAC9-AAD6	
BGET	AAD7	AAD7-AADE	
OPENIN	AADF	AADF-AAE2	
OPENOUT	AAE3	AAE3-AAE6	
OPENUP	AAE7	AAE7-AAFE	
PI	AAFF	AAFF-AB04	
EVAL	AB05	AB05-AB39	
VAL	AB49	AB46-AB4D	
Ascnum (ASCII String to Binary Number)	AB4E	AB4E-AB88	Also includes:AB3A-AB45
INT	AB8A	AB8A-ABB2	Also includes: 830D-831D
ASC	ABB3	ABB3-ABC1	
INKEY	ABC2	ABC2-ABCB	Also includes: AA12-AA1D
EOF	ABCF	ABCF-ABDA	
TRUE	ABDB	ABDB-ABDC	

FALSE (Clear IWA)	ABE8	ABDD-ABEB	
SGN	ABF5	ABEC-AC0D	
POINT(	AC0E	AC0E-AC35	
INSTR(	AC36	AC36-ACB3	
ABS	ACB7	ACB4-ACBD	Also includes: ACC4-ACC6
Integer Positive	ACBE	ACBE-ACC3	
Floating-Point Compliment	ACCA	ACCA-ACD6	
Compliment Result	ACD7	ACD7-ACDD	
Integer Compliment	ACDE	ACDE-ACF7	
Extract next field	ACF8	ACF8-AD18	
Extract String	AD19	AD11-AD35	
Evaluate Variable/Value/BASIC Function/Open Bracket	AD36	AD36-AD8B	
Evaluate expression and check for closing bracket	ADAC	ADAC-ADB6	
Extract Hex Integer	ADB7	ADB7-ADEB	
ADVAL	ADEC	ADEC-ADF8	
TOP	ADF9	ADF9-AE07	
=PAGE	AE08	AE08-AE0D	

LEN	AE11	AE0E-AE17	
IWA = 8-bit or 16-bit Integer	AE1A	AE18-AE24	
COUNT	AE25	AE25-AE28	
=LOMEM	AE29	AE29-AE2E	
=HIMEM	AE2F	AE2F-AE34	
ERL	AE35	AE35-AE3A	
ERR	AE3B	AE3B-AE3E	
GET	AE3F	AE3F-AE43	
=TIME	AE44	AE44-AE56	
=TIME$	AE57	AE57-AE68	
GET$	AE69	AE69-AE72	Also includes: AE8F-AE93
LEFT$(	AE73	AE73-AE73	
RIGHT$(	AE74	AE74-AEB2	Also includes: 96A4-96A6
INKEY$	AEB3	AEB3-AEBE	
MID$(	AEC5	AEBF-AF1B	
STR$	AF1C	AF1C-AF46	
STRING$	AF47	AF47-AF81	

Load Variable	B1A0	B1A0-B1A9	
Load IWA with Integer from Address [iin]	B1AA	B1AA-B1C1	
Load IWA with 1-byte Integer value	B1C2	B1C2-B1C6	
Load FWA with Float Variable (ain)	B1C7	B1C7-B1F6	
Load SWA with String Value	B1F7	B1F7-B22E	
CHR$	B22F	B22F-B236	
BASIC Error handler	B278	B237-B2A5	
Reset ON ERROR code pointer	B2A6	B2A6-B2C7	
SOUND	B2C8	B2C8-B2EB	
ENVELOPE	B2EC	B2EC-B316	
WIDTH	B317	B317-B324	
Set Numeric variable	B32B	B325-B346	Also includes: B360-B388
Save Integer to Address	B347	B347-B35F	
EDIT	B393	B389-B399	
LIST	B39A	B39A-B3DC	Also: B3F3-B4F0
LISTO	B3DD	B3DD-B3F2	
NEXT	B4F1	B4F1-B5F7	

FOR	B618	B618-B6D8	
GOSUB	B6D9	B6D9-B6F2	
RETURN	B707	B707-B71C	
GOTO	B71D	B71D-B738	
ON	B75B	B739-B829	
Get Line Number & find Program Address of the Line Number	B82A	B82A-B83B	
INPUT#	B847	B83C-B8B1	
INPUT	B8B6	B8B2-B94C	Also includes: 9299-92BD, BA70-BA91
RESTORE	B94D	B94D-B974	
READ	B97D	B975-B9F0	Also includes: BA13-BA16
UNTIL	BA17	BA17-BA3B	
REPEAT	BA58	BA58-BA6F	
Start new output line	BA92	BA92-BA97	
Remove Line Number (specified in the IWA) from the Program	BA98	BA98-BAEA	
Tokenise Command Line and Insert Line into Program	BAEB	BAEB-BBAB	
Initialise Page 7 & reset Variable pointers, etc...	BBAC	BBAC-BBE7	Also includes: BF14 to BF23
Pop Float from Stack (to &4A,&4B)	BBE8	BBE8-BBF9	

Push FWA to Stack	BBFA	BBFA-BC21	
Push Integer to BASIC Stack	BC26	BC22-BC42	
Push String to Stack	BC51	BC51-BC69	
Pop String from Stack	BCD2	BCD2-BCE5	
Pop Integer from BASIC Stack	BCE6	BCE6-BD05	
Pop Integer from BASIC Stack (Zp)	BD08	BD06-BD1D	
Check for Stack clash with Heap	BD1E	BD1E-BD30	Also includes: BD34-BD36
Output ASCII character or BASIC Token (in ASCII text) to the screen	BD37	BD37-BD6B	Also includes: BD34-BD36
Output character to the screen	BD92	BD6C-BDC5	Also includes: BD31-BD36
Save Integer to Zero Page Address	BDC6	BDC6-BDD6	
Load/Save named file	BDD7	BDD7-BDE4	
Check Program can be read correctly	BDE5	BDE5-BE24	Also includes BECF-BEE1
Get Filename and set parameter block Filename and Load address	BE41	BE33-BE54	
SAVE	BE55	BE55-BE86	
OSCLI	BE87	BE87-BE92	
EXT =	BE93	BE93-BE96	
PTR =	BE97	BE97-BEAD	

CLOSE	BEAE	BEAE-BEBC
BPUT	BEBD	BEBD-BECE
Read byte from I/O processor memory location (at the address contained in the IWA)	BEE2	BEE2-BEEE
Floating Point Constant Table	BF24	BF24-BFFA

Error Messages

Character Set

Memory Map

Glossary

Disassembly

8000 to 9000 HTML TEXT CSV
9000 to A000 HTML TEXT CSV
A000 to B000 HTML TEXT CSV
B000 to C000 HTML TEXT CSV

Reference Disc images

BASIC ROM Routines

Page last modified: undefined

BASIC ROM ROUTINES
==================
By Christopher Dewhurst

The Basic Rom is that dark and mysterious area of memory that lies beyond the screen, starting at &8000 and stretching up to &BFFF. It's the backstage department containing the machine code needed to interpret your Basic commands. But have you ever wondered if we can use some of that machine code in our own programs? Have you, for instance, struggled to write an assembler routine to print a number on the screen, when one must surely exist somewhere in the Basic Rom?

Well, wonder no more, because there is indeed such a routine, and in this article we'll be exploring that and a lot more besides.

Before we go any further, however, a word of caution. The best machine code is specific machine code written for a specific job; Basic Rom routines are general-purpose routines, and are not the answer to everything. Having said that, if speed is not your main priority, then the Rom routines are ideal. They make your programs smaller and smarter, provided you use them properly - and this usually involves some fairly tricky setting up - so listen carefully.

I learnt a lot about the Basic Rom by exploring around it and experimenting with it myself. I also picked up a few tips from 'The Advanced Basic Rom User Guide' by Colin Pharo (Cambridge Micro Centre, 1984). Roland Waddilove also presented a series of excellent articles on the subject in 'Electron User'; if you still have these paper beauties, dig out the November 1988 for a rundown on mathematical Rom routines. However, I will be concentrating on routines which print numbers in hex or decimal, the random number generator, and printing strings of text.

In case you're wondering, BBC Master owners won't be left out of the discussion this time. I have done

quite a bit of disassembling of the Basic 4 Rom to find out where equivalent routines to Basic 2 reside. Basic 2 is the Rom fitted in the BBC B and Electron, and Basic 4 is the one fitted in the Master. (Like the Plus 2, for some reason Basic 3 never was.)

When I talk about a Rom routine, I will specify both the Basic 2 and Basic 4 addresses - together with examples and commentaries on how to use them - so it is up to you to use the correct one depending on which computer you have.

If you experience any difficulties - or if you have additional hints and tips - just email me.

Right, down to business. We must first get to know what is called the Integer Work Area, or IWA for short. This is just a sequence of four bytes in zero page, located at &2A-&2D. Before Basic can work on an integer variable, be it adding a number to it or printing it out, it must be put into the IWA. Fortunately, life is made easier with the help of a couple of routines which copy an integer variable, either from zero page or from the main memory, to the IWA:

1. Routine: Copy 4-byte integer from zero page to the IWA
Basic 2 address: &AF56
Basic 4 address: &AA80

Entry: X = zero page offset at which the integer to be copied is located.
Exit: IWA contains the integer.
Ex.: LDX #&70 \integer at &70-3
JSR &AF56 \copy to IWA

2. Routine: Copy 4-byte integer from memory to the IWA
Basic 2: &B336
Basic 4: &B1AA

Entry: &2A/&2B contain address of the integer.
Exit: IWA contains the integer.
Ex.: LDA #integer MOD 256
STA &2A
LDA #integer DIV 256
STA &2B
JSR &B336
...
.integer EQUD &12345678

There are also two routines which do the opposite of above. The one at &BE44 (Basic 2)/&BDC6 (Basic 4) copies the IWA to a zero-page location, X being set to the zero page location on entry. The routine at

&B4C6 (Basic 2)/ &B347 (Basic 4) copies the IWA to a location in main memory whose address is held in &37/&38.

3. Routine: Print a string
Basic 2: &BFCF
Basic 4: &BECF

Entry: The string must follow the JSR &BFCF instruction, and be terminated by a byte of value &80 or greater.
Ex.: JSR &BFCF
EQUS "Hello there.":NOP
...

Notice how I've used the NOP instruction to terminate the string. The NOP opcode has a value of &EA, which satisfies the condition of being &80 or greater. The important point to remember is that program execution continues AFTER that NOP instruction. In machine code, every time a JSR instruction is executed the current address is pushed onto the stack. Basic pulls this address from the stack, stores it in zero page locations &37/&38, and uses indirect addressing to get the bytes of the string. By the time the string has been printed, &37/&38 contains the address of the next instruction after the string in the program that called the routine. The disadvantage of this routine, however, is that while you can include control codes (to turn off the cursor for instance) you can't print out a string of graphics characters because they have an ASCII value of &80 or above which, as we said, is used to terminate the string.

4. Routine: Print A in hex
Basic 2: &B545
Basic 4: &BD6C

Entry: The Accumulator contains the byte to be printed in hex
Ex.: LDA #&CD
JSR &B545

This one can be quickly demonstrated from Basic, if you really want, by typing A%=&CD: CALL&B545.

5. Routine: Print 16-bit number in decimal
Basic 2: &991F
Basic 4: &A081

Entry: &2B/&2C (the 2 least significant bytes of the IWA) should contain the number to be printed.
Ex.: LDA #1023 MOD 256 \put the number 1023
STA &2B
LDA #1023 DIV 256 \onto the two lsb's of the IWA

```
STA &2C
JSR &991F
```

This is the routine which I promised we would discuss at the beginning of this article, so let's take some time going through it in detail. If you have a disassembler then you could look at the actual machine code, which in English goes something like this. You first of all see how many times 10,000 can be subtracted from the given number before it becomes negative. For example, you can subtract 10,000 six times from the number 60,000. This is the 10,000s count. Then you see how many times 1,000 can be subtracted from the remainder, then how many times 100 can be taken away from the remainder of that, and so on down to the 1s count. In order to do this, we need a table of two-byte values: 10,000, 1,000, 100, 10 and 1. There are two tables in Rom; the first table contains the low bytes, and the second table contains the high bytes:

```
Basic 2 Low bytes: High bytes
&996B [&01] &99B9 [&00] &0001 = 1
&996C [&0A] &99BA [&00] &000A = 10
&996D [&64] &99BB [&00] &0064 = 100
&996E [&E8] &99BC [&03] &03E8 = 1000
&996F [&10] &99BD [&27] &2710 =
10000
```

```
Basic 4 Low bytes High bytes
&8026 [&01] &8021 [&00] &0001 = 1
&8027 [&0A] &8022 [&00] &000A = 10
&8028 [&64] &8023 [&00] &0064 = 100
&8029 [&E8] &8024 [&03] &03E8 = 1000
&802A [&10] &8025 [&27] &2710 = 10000
```

If you haven't got a disassembler, then the program below demonstrates how it works:

```
10 FORI%=0TO2STEP2
20 P%=&900
30 [OPTI%
40 LDX #&50 \copy integer from zero page
50 JSR &AF56 \to IWA
60
70 LDX #4
80 .loop LDA#0
90 STA &3F,X
100 SEC
110 .loop2 LDA&2A
120 SBC &996B,X \&8026 for BBC M
```

```
130 TAY
140 LDA &2B
150 SBC &99B9,X \&8021 for Basic 4
160 BCC skip
170 STA &2B
180 STY &2A
190 INC &3F,X
200 BNE loop2
210 .skip DEX
220 BPL loop
270
280 LDX #5 \suppress leading zeroes
290 .loop3 DEX \by indexing to first
300 BEQ print \non-zero number
310 LDA &3F,X
320 BEQ loop3
330 .print LDA &3F,X
340 ORA #&30
350 JSR &FFEE
360 DEX
370 BPL print
380 RTS
390 ]
400 NEXT
410 INPUT !&50
420 CALL &900
```

The section of code from line 280 to 320 suppresses leading zeroes. This just means that if you had the number 234, then it will be printed as 234 and not 00234. Sometimes you might not care for leading zero suppression. In most games, for instance, your score is displayed as 00000 at the start, then changes to 00010 when you score some points and so on. In this case, you can dispense with lines 290-320 in the above program and replace line 280 with LDX #4.

6. Routine: Convert number in IWA to ASCII decimal or hexadecimal
Basic 2: &9EFF
Basic 4: &A138

Entry: IWA should contain number to be converted location &15 = &FF for hexadecimal ASCII or &15 = 0 for decimal ASCII

The previous routine only allowed 16-bit numbers (numbers in the range 0-65535) to be printed. This routine helps you print 32-bit numbers in decimal or hex. When we speak of "hexadecimal" or "decimal ASCII", it means that a string containing ASCII codes is made for the given number. For example, the

four ASCII decimal codes for &9EFF are 57, 69, 70, and 70 (ignoring the ampersand which Basic doesn't print anyway). Basic puts these ASCII codes into the String Work Area, or SWA for short. We can then use another routine to print out the contents of the SWA:

7. Routine: Print the SWA
Basic 2: &8E12
Basic 4: &921B

Entry: location &30 must contain the length of the string the SWA must contain the string location &A must = 0.

```
Ex.: LDX #&50 \copy integer at &50-3
JSR &AF56 \to SWA
LDA #&FF:STA &15 \number to be in hex
JSR &9EFF \convert IWA to ASCII codes
LDA #0:STA &A
JMP &8E12 \and print the number
```

8. Routine: Random number generator
Basic 2: &AF51
Basic 4: &AA7B

No entry requirements. On exit, the IWA contains a 4-byte random integer.

```
Ex.: JSR &AF51
LDA &2A
JSR alien
```

I find this Rom routine extremely useful for getting random numbers in games, and it's the only decent way of getting fairly unpredictable numbers in machine code.

Conclusion
If you use any of the above routines, don't forget to use the correct address for your version of Basic. Now that you've seen what the Basic Rom can do, hopefully you'll want to start exploring other parts. If you find anything useful, do write in and let us know!

Christopher Dewhurst
24 September, 2000

BASIC IV ROM Routines

Page Last Altered: undefined

B618 FOR

Submitted by Steve Fewell

Description:

Call routine &98AE to evaluate the variable name at the BASIC Text Pointer A location, and to create it if it doesn't exist, and return the variable value's memory address in locations &2A-&2B and its type in &2C.
If the zero flag is set on return from &98AE, then the variable name is not a valid variable name, so issue 'FOR variable' error.
If the carry flag is set on return from &98AE, then the variable is a String variable, so issue 'FOR variable' error, as a FOR variable must be numeric.

Call routine &BC43 to store the variable information data (&2A-&2C) to the 6502 stack.
Call routine &9B86 to check whether the next non-space character is '=' (equals), if not then issue 'Mistake' error; otherwise, skip over the '=' character.

Call routine &B328 to evaluate the expression (after the '=' character), retrieve the variable information from the 6502 stack and set the Numeric FOR variable to the result of the expression.
If the next non-space character at BASIC Text Pointer B is not the 'TO-token', then issue 'No TO' error, as the TO-Keyword must follow the expression giving the variable's start value.

Load Y with the current FOR level (location &26).
If the current FOR level is greater than or equal to #&96 (150) then issue 'Too many FORs' error, as only a maximum of 10 FOR loops can be nested (each FOR loop uses 15 bytes on the FOR stack - and increases the FOR level by 15).

Add #&0F (15) to the current FOR level, and update location &26 with the new value.
Now, the &26 location has been updated, but Y still contains the original FOR level (before the addition).
Store the variable value address (&37-&38) in locations &0528 + Y and &0529 + Y; and store the variable type (&39) in location &052A + Y.

If the variable type (&39) is Floating-Point (i.e. it is #&05) then:

* [&B6A1] Call routine &9D3B to evaluate the expression after the TO-Keyword
* Call routine &96DD to convert the result to a Floating-point value if it is Integer, or issue 'Type mismatch' error if it is a String value.
* Add #&21 (33) to the current FOR level (location &26) and store the result in location &4A.
* Store #&05 in location &4B
* Store the TO-value (the value in the FWA) in the 5-byte location pointed to by argp (&4A-&4B).
* Call routine &A5D8 to set the FWA to 1.0 (the default FOR STEP value)
* Get the next non-space character after the TO-expression (at the location pointed to by BASIC text pointer B)
* If the character is the 'STEP-token' then: Call routine &9D3B to evaluate the expression after the STEP-Keyword, Call &96DD to convert the result to Floating-Point (if it was an integer, or issue 'Type mismatch' error if it was a String) and Set Y to the BASIC text pointer B offset.
* [&B6C7] Update BASIC Text Pointer A to point to the end of the FOR-statement (by setting it to the value of Y (which is the current position on the Program Line)).
* Add #&1C to the current FOR level (location &26) and store the result in location &4A
* Set location &4B to #&05
* Call &A519 to store the STEP value (the value in the FWA) to the location pointed to by argp (&4A-&4B)
* Jump to &B68F to Check and skip the end of the program statement (issuing 'Syntax error' if the end of statement character (':','<cr>','ELSE') wasn't found), Store the current BASIC Text pointer A value to &0526 + Y - &0527 + Y (where Y is the FOR level - from location &26), Jump to &900B to continue executing the program - starting at the next program statement after the FOR-Statement.

Otherwise, the variable type (&39) must be Integer (i.e. it is #&04), so:

* [&B64D] Call routine &96AF to evaluate the expression after the TO-Keyword and convert the expression result to an Integer (if it was a Floating-Point value), or issue 'Type mismatch' error if the result was a String value.
* Set Y to the current FOR level (from location &26)
* Store the TO-value (in the IWA) in locations &0521 + Y to &0524 + Y (LSB first)
* Set the IWA to 1 (the default FOR STEP value)
* Get the next non-space character after the TO-expression (at the location pointed to by BASIC text pointer B)
* If the character is the 'STEP-token' then: Call routine &96AF to evaluate the expression after the TO value, and convert the result to Integer (if it was Floating-Point, or issue 'Type mismatch' error if the result was a String value), Set Y to the BASIC text pointer B offset value (&1B).
* [&B677] Update BASIC Text Pointer A to point to the end of the FOR-statement (by setting it to the value of Y (which is the current position on the Program Line)).
* Set Y to the current FOR level (from location &26).
* Store the STEP value (the current IWA value) to the FOR stack at starting at location &051C + Y (for the LSB byte - &2A) and ending at location &051F + Y (for the MSB byte - &2D).
* [&B68F] Check and skip the end of the program statement (issuing 'Syntax error' if the end of

statement character (':','<cr>','ELSE') wasn't found) - also reset the BASIC text pointer A base value (&0B-&0C) to take into account the offset value (&0A).

* Store the current BASIC Text pointer A value to &0526 + Y and &0527 + Y (where Y is the FOR level - from location &26)
* Jump to &900B to continue executing the program - starting at the next program statement after the FOR-statement.

The memory layout of the FOR Stack (&0528-&05CB) is as follows:
The stack contains details for a maximum of 10 FOR loops. Each FOR block is 15 bytes long.
In each FOR block, the bytes have the following meaning:
Byte 1 (&0528) is the variable location - LSB.
Byte 2 (&0529) is the variable location - MSB
Byte 3 (&052A) is the variable type
Byte 4 (&052B) - Byte 8 (&052F) is the STEP value (Float (5-byte) / Integer (4-byte))
Byte 9 (&0530) - Byte 13 (&0534) is the TO value (Float (5-byte) / Integer (4-byte))
Byte 14 (&0535) is the program location of the start of the first statement within the FOR loop - LSB
Byte 15 (&0536) is the program location of the start of the first statement within the FOR loop - MSB

Disassembly for the FOR routine

B618		032 174 152	20 AE 98	JSR &98AE Evaluate variable name & create it if it's a new variable
B61B		240 219	F0 DB	BEQ -37 --> &B5F8 'FOR variable' error
B61D		176 217	B0 D9	BCS -39 --> &B5F8 'FOR variable' error
B61F	C	032 067 188	20 43 BC	JSR &BC43 Push &2A, &2B & &2C (the variable info) to the Stack
B622		032 134 155	20 86 9B	JSR &9B86 Check whether the next non-space character is '=' ('Mistake' error if not)
B625	(	032 040 179	20 28 B3	JSR &B328 Evaluate expression and set Numeric variable
B628		032 213 142	20 D5 8E	JSR &8ED5 Get next non-space character (PTR B)
B62B		201 184	C9 B8	CMP#&B8
B62D		208 226	D0 E2	BNE -30 --> &B611 'No TO' error
B62F	&	164 038	A4 26	LDY &26
B631		192 150	C0 96	CPY#&96
B633		176 207	B0 CF	BCS -49 --> &B604 'Too many FORs' error
B635		152	98	TYA
B636	i	105 015	69 0F	ADC#&0F
B638	&	133 038	85 26	STA &26
B63A	7	165 055	A5 37	LDA &37
B63C	(	153 040 005	99 28 05	STA &0528,Y
B63F	8	165 056	A5 38	LDA &38

```
B641    )   153 041 005     99 29 05    STA &0529,Y
B644    9   165 057         A5 39       LDA &39
B646    *   153 042 005     99 2A 05    STA &052A,Y
B649        201 005         C9 05       CMP#&05
B64B    T   240 084         F0 54       BEQ 84 --> &B6A1
B64D        032 175 150     20 AF 96    JSR &96AF Get expression result & convert it to Integer
B650    &   164 038         A4 26       LDY &26
B652    *   165 042         A5 2A       LDA &2A
B654    !   153 033 005     99 21 05    STA &0521,Y
B657    +   165 043         A5 2B       LDA &2B
B659    "   153 034 005     99 22 05    STA &0522,Y
B65C    ,   165 044         A5 2C       LDA &2C
B65E    #   153 035 005     99 23 05    STA &0523,Y
B661    -   165 045         A5 2D       LDA &2D
B663    $   153 036 005     99 24 05    STA &0524,Y
B666        169 001         A9 01       LDA#&01
B668        032 024 174     20 18 AE    JSR &AE18 Set IWA to the 8-bit value in A
B66B        032 213 142     20 D5 8E    JSR &8ED5 Get next non-space character (PTR B)
B66E        201 136         C9 88       CMP#&88
B670        208 005         D0 05       BNE 5 --> &B677
B672        032 175 150     20 AF 96    JSR &96AF Get expression result & convert it to Integer
B675        164 027         A4 1B       LDY &1B
B677        132 010         84 0A       STY &0A
B679    &   164 038         A4 26       LDY &26
B67B    *   165 042         A5 2A       LDA &2A
B67D        153 028 005     99 1C 05    STA &051C,Y
B680    +   165 043         A5 2B       LDA &2B
B682        153 029 005     99 1D 05    STA &051D,Y
B685    ,   165 044         A5 2C       LDA &2C
B687        153 030 005     99 1E 05    STA &051E,Y
B68A    -   165 045         A5 2D       LDA &2D
B68C        153 031 005     99 1F 05    STA &051F,Y
B68F        032 207 155     20 CF 9B    JSR &9BCF Check & skip end of program line
B692    &   164 038         A4 26       LDY &26
B694        165 011         A5 0B       LDA &0B
```

```
B696    &  153 038 005    99 26 05     STA &0526,Y
B699       165 012        A5 0C        LDA &0C
B69B    '  153 039 005    99 27 05     STA &0527,Y
B69E    L  076 011 144    4C 0B 90     JMP &900B Process next BASIC program statement
B6A1    ;  032 059 157    20 3B 9D     JSR &9D3B Evaluate expression at BASIC Text pointer B
B6A4       032 221 150    20 DD 96     JSR &96DD Check Float value (Convert if Integer, or 'Type mismatch' error if String)
B6A7    &  165 038        A5 26        LDA &26
B6A9       024            18           CLC
B6AA    i! 105 033        69 21        ADC#&21
B6AC    J  133 074        85 4A        STA &4A
B6AE       169 005        A9 05        LDA#&05
B6B0    K  133 075        85 4B        STA &4B
B6B2       032 025 165    20 19 A5     JSR &A519 Store FWA's value to argp address (&4A-&4B)
B6B5       032 216 165    20 D8 A5     JSR &A5D8 Set FWA to 1.0
B6B8       032 213 142    20 D5 8E     JSR &8ED5 Get next non-space character (PTR B)
B6BB       201 136        C9 88        CMP#&88
B6BD       208 008        D0 08        BNE 8 --> &B6C7
B6BF    ;  032 059 157    20 3B 9D     JSR &9D3B Evaluate expression at BASIC Text pointer B
B6C2       032 221 150    20 DD 96     JSR &96DD Check Float value (Convert if Integer, or 'Type mismatch' error if String)
B6C5       164 027        A4 1B        LDY &1B
B6C7       132 010        84 0A        STY &0A
B6C9    &  165 038        A5 26        LDA &26
B6CB       024            18           CLC
B6CC    i  105 028        69 1C        ADC#&1C
B6CE    J  133 074        85 4A        STA &4A
B6D0       169 005        A9 05        LDA#&05
B6D2    K  133 075        85 4B        STA &4B
B6D4       032 025 165    20 19 A5     JSR &A519 Store FWA's value to argp address (&4A-&4B)
B6D7       128 182        80 B6        BRA -74 --> &B68F
```

BASIC IV ROM Routines

Page Last Altered: undefined

98AE Evaluate variable name & Create if new variable

Submitted by Steve Fewell

Routine:CreateNewVariable
Name: Evaluate variable name & Create if new variable
Starting Address: &98AE
Entry criteria: The BASIC Text Pointer A points to the start of a variable name.
Exit: The Variable has been created (or no action if it already exists).
Zero flag is set if the variable name was not valid (Syntax error).
C = 0 if the variable is numerical; otherwise, C = 1 (for String variable).

Description:

Call routine &98F5 to reset the BASIC Text pointer B to the BASIC Text pointer A location, and evaluate the variable name at that location.
If the variable name is already allocated then exit [A = non zero, BNE).
If the variable name is a direct memory access (? or !) then exit [A = non zero].
If the variable name is not a valid name then exit [Carry flag is set].

Othwise, the variable doesn't exist yet, so it needs to be created.
So, call &9854 to add the variable name to the variable pointer table.
Next, If the variable type is not 5 then call &9883 with X = 4 (Integer/String).
Otherwise, increment X to 5, and call &9883. &9883 will allocate parameter block space for the new variable.

Next, call &98F5 again to evaluate the variable name again and set &2A, &2B to point to the variable's value and set &2C to the variable type.
We need to call &98F5 again, as we would not have the variable vaue address in &2A-&2B if we didn't.

Disassembly for the Evaluate variable name & Create if new variable routine

98AB	032 131 152	20 83 98	JSR &9883 Allocate space for variable
98AE	032 245 152	20 F5 98	JSR &98F5 PtrB=PtrA & evaluate variable name & obtain address of value
98B1	208 029	D0 1D	BNE 29 --> &98D0 [RTS]
98B3	176 027	B0 1B	BCS 27 --> &98D0 [RTS]

98B5	T	032 084 152	20 54 98	JSR &9854 Create new variable name in variable pointer table
98B8		162 005	A2 05	LDX#&05
98BA	,	228 044	E4 2C	CPX &2C
98BC		208 237	D0 ED	BNE -19 --> &98AB
98BE		232	E8	INX
98BF		128 234	80 EA	BRA -22 --> &98AB

BASIC IV ROM Routines

Page Last Altered: undefined

9883 Allocate space for new variable

Submitted by Steve Fewell

Routine:allocspace
Name: Allocate space for new variabler
Starting Address: &9983
Entry criteria: Y contains the length of the variable name (which has
just been copied to the VARTOP location) + 1.
X contains the number of bytes required for the variable parameter block.
Exit: The Variable parameter block is complete, and VARTOP has been updated
to point to the next free location.

Description:

Place a zero at the end of the variable name (to mark the end of the name).

Zero X number of bytes following the variable name location (starting at VARTOP + Y + 1).
This will clear the new variable parameter block.

Calculate new VARTOP (&02-&03) location (which will point to the next free variable location (i.e. the location after the variable name and parameter block info which has been stored for the current variable)). But, don't update VARTOP with the new address yet.

If the new VARTOP location would be >= the BASIC Stack pointer, then we have run into the stack, so store 0 for the Next variable MSB address field of the previous variable's parameter block (&3A,1), to set that variable as being the last variable in the list, and generate a 'No Room' error.

Otherwise Store the updated VARTOP position back in &02-&03 and exit.

Disassembly for the Allocate space for new variable routine

9883	169 000	A9 00	LDA#&00
9885	145 002	91 02	STA (&02),Y
9887	200	C8	INY
9888	202	CA	DEX
9889	208 250	D0 FA	BNE -6 --> &9885
988B	024	18	CLC

988C		152	98	TYA
988D	e	101 002	65 02	ADC &02
988F		144 002	90 02	BCC 2 --> &9893
9891		230 003	E6 03	INC &03
9893		164 003	A4 03	LDY &03
9895		196 005	C4 05	CPY &05
9897		144 015	90 0F	BCC 15 --> &98A8
9899		208 004	D0 04	BNE 4 --> &989F
989B		197 004	C5 04	CMP &04
989D		144 009	90 09	BCC 9 --> &98A8
989F		169 000	A9 00	LDA#&00
98A1		160 001	A0 01	LDY#&01
98A3	:	145 058	91 3A	STA (&3A),Y
98A5	L	076 161 144	4C A1 90	JMP &90A1 No Room error
98A8		133 002	85 02	STA &02
98AA	`	096	60	RTS

BASIC IV ROM Routines

Error Messages

Submitted by Steve Fewell

This page contains all of the error messages issued by the BASIC IV ROM. A number surrounded by pointed brackets, i.e. <138>, indicates reference to a BASIC KEYWORD. These references are automatically expanded into the actual keywords when BASIC prints the error message. The Keywords relating to each number are shown on the character table page.

Note: The error messages "Bad Program" and "Failed at" are not included, as they aren't really error messages, because they are not handled via the BRK vector. Instead, the BASIC ROM just displays these texts directly on the screen, without causing a BRK event to occur.

The Error Messages are listed in Numerical order by ERR number:

Error 0 - LINE space

```
BB61   BRK
BB62   EQUB 0
BB63   EQUS "<134> space"
BB6A   EQUB 0
```

Error 0 - No room

```
90A1   BRK
90A2   EQUB 0
90A3   EQUS "No room"
90AA   EQUB 0
```

Error 0 - RENUMBER space

```
93B3   BRK
93B4   EQUB 0
93B5   EQUS "<204> space"
93BC   EQUB 0
```

Error 0 - Silly

```
93BC   BRK
93BD   EQUB 0
93BE   EQUS "Silly"
93C3   EQUB 0
```

Error 0 - STOP

```
9089   BRK
908A   EQUB 0
908B   EQUS "<250>"
908C   EQUB 0
```

Error 1 - Out of range

```
8ACC   BRK
8ACD   EQUB 1
8ACE   EQUS "Out of range"
8ADA   EQUB 0
```

Error 2 Byte

```
8AF9   BRK
8AFA   EQUB 2
8AFB   EQUS "Byte"
8AFF   EQUB 0
```

Error 3 - Index

```
8B3C   BRK
8B3D   EQUB 3
8B3E   EQUS "Index"
8B43   EQUB 0
```

Error 4 - Mistake

```
9B60   BRK
9B61   EQUB 4
9B62   EQUS "Mistake"
9B69   EQUB 0
```

Error 5 - Missing ,

```
8EF6   BRK
8EF7   EQUB 5
8EF8   EQUS "<141>,"
8EFA   EQUB 0
```

Error 6 - Type mismatch

```
9092   BRK
9093   EQUB 6
9094   EQUS "Type mismatch"
90A1   EQUB 0
```

Error 7 - No FN

```
908C   BRK
908D   EQUB 7
908E   EQUS "No <164>"
9092   EQUB 0
```

Error 8 - $ range

```
98EB   BRK
98EC   EQUB 8
98ED   EQUS "$ range"
98F4   EQUB 0
```

Error 9 - Missing "

```
9294   BRK
9295   EQUB 9
9296   EQUS "<141>""""
9298   EQUB 0
```

Error 10 - Bad DIM

```
952C   BRK
952D   EQUB 10
952E   EQUS "Bad <222>"
9533   EQUB 0
```

Error 11 - DIM space

```
9605   BRK
9606   EQUB 11
9607   EQUS "<222> space"
960E   EQUB 0
```

Error 12 - Not LOCAL

```
9732   BRK
9733   EQUB 12
9734   EQUS "Not <234>"
9739   EQUB 0
```

Error 13 - No PROC

```
9B77   BRK
9B78   EQUB 13
9B79   EQUS "No <242>"
9B7D   EQUB 0
```

Error 14 - Array

```
99F6   BRK
99F7   EQUB 14
99F8   EQUS "Array"
99FD   EQUB 0
```

Error 15 - Subscript

```
9AEA   BRK
9AEB   EQUB 15
9AEC   EQUS "Subscript"
9AF5   EQUB 0
```

Error 16 - Syntax error

```
9B69   BRK
9B6A   EQUB 16
9B6B   EQUS "Syntax error"
9B77   EQUB 0
```

Error 17 - Escape

```
9B7D   BRK
9B7E   EQUB 17
9B7F   EQUS "Escape"
9B85   EQUB 0
```

Error 18 - Division by zero

```
8172   BRK
8173   EQUB 18
8174   EQUS "Division by zero"
8184   EQUB 0
```

Error 19 - String too long

```
9E10   BRK
9E11   EQUB 19
9E12   EQUS "String too long"
9E21   EQUB 0
```

Error 20 - Too big

```
A6C5   BRK
A6C6   EQUB 20
A6C7   EQUS "Too big"
A6CE   EQUB 0
```

Error 21 - -ve root

```
A75B   BRK
A75C   EQUB 21
A75D   EQUS "-ve root"
A765   EQUB 0
```

Error 22 - Log range

```
A750   BRK
A751   EQUB 22
A752   EQUS "Log range"
A75B   EQUB 0
```

Error 23 - Accuracy lost

```
A9AD   BRK
A9AE   EQUB 23
A9AF   EQUS "Accuracy lost"
A9BC   EQUB 0
```

Error 24 - Exp range

```
A9BC   BRK
A9BD   EQUB 24
A9BE   EQUS "Exp range"
A9C7   EQUB 0
```

Error 25 - Bad MODE

```
9739   BRK
973A   EQUB 25
973B   EQUS "Bad <235>"
9740   EQUB 0
```

Error 26 - No such variable

```
AD8C   BRK
AD8D   EQUB 26
AD8E   EQUS "No such variable"
AD9E   EQUB 0
```

Error 27 - Missing)

```
AD9E   BRK
AD9F   EQUB 27
ADA0   EQUS "<141>)"
ADA2   EQUB 0
```

Error 28 - Bad Hex

```
ADA2   BRK
ADA3   EQUB 28
ADA4   EQUS "Bad Hex"
ADAB   EQUB 0
```

Error 29 - No such FN/PROC

```
AF89   BRK
AF8A   EQUB 29
AF8B   EQUS "No such <164>/<242>"
AF96   EQUB 0
```

Error 30 - Bad call

```
B00C   BRK
B00D   EQUB 30
B00E   EQUS "Bad call"
B016   EQUB 0
```

Error 31 - Arguments

```
B134   BRK
B135   EQUB 31
B136   EQUS "Arguments"
B13F   EQUB 0
```

Error 32 - No FOR

```
B532   BRK
B533   EQUB 32
B534   EQUS "No <227>"
B538   EQUB 0
```

Error 33 - Can't match FOR

```
B523  BRK
B524  EQUB 33
B525  EQUS "Can't match <227>"
B532  EQUB 0
```

Error 34 - FOR variable

```
B5F8  BRK
B5F9  EQUB 34
B5FA  EQUS "<227> variable"
B604  EQUB 0
```

Error 35 - Too many FORs

```
B604  BRK
B605  EQUB 35
B606  EQUS "Too many <227>s"
B611  EQUB 0
```

Error 36 - No TO

```
B611  BRK
B612  EQUB 36
B613  EQUS "No <184>"
B617  EQUB 0
```

Error 37 - Too many GOSUBs

```
B6F3  BRK
B6F4  EQUB 37
B6F5  EQUS "Too many <228>s"
B700  EQUB 0
```

Error 38 - No GOSUB

```
B700  BRK
B701  EQUB 38
B702  EQUS "No <228>"
B706  EQUB 0
```

Error 39 - ON syntax

```
B7EA  BRK
B7EB  EQUB 39
B7EC  EQUS "<238> syntax"
B7F4  EQUB 0
```

Error 40 - ON range

```
B7E1   BRK
B7E2   EQUB 40
B7E3   EQUS "<238> range"
B7EA   EQUB 0
```

Error 41 - No such line

```
B7F4   BRK
B7F5   EQUB 41
B7F6   EQUS "No such line"
B802   EQUB 0
```

Error 42 - Out of DATA

```
B9F1   BRK
B9F2   EQUB 42
B9F3   EQUS "Out of <220>"
B9FB   EQUB 0
```

Error 43 - No REPEAT

```
B9FB   BRK
B9FC   EQUB 43
B9FD   EQUS "No <245>"
BA01   EQUB 0
```

Error 44 - Too many REPEATs

```
BA05   BRK
BA06   EQUB 44
BA07   EQUS "Too many <245>s"
BA12   EQUB 0
```

Error 45 - Missing #

```
BA01   BRK
BA02   EQUB 45
BA03   EQUS "<141>#"
BA05   EQUB 0
```

BASIC IV ROM Routines

Characters

Submitted by Steve Fewell
Page Last Altered: undefined

This page contains a listing of all of the ASCII values that are meaningful to the BASIC ROM.

The Characters are listed in ASCII order:

ASCII value	ASCII value (Hex)	Display Character/ Keyword
13	D	<CR> Carriage Return
32	20	< > Space
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)

42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I

74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_
96	60	£
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i

106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	<DEL>
128	80	AND
129	81	DIV
130	82	EOR
131	83	MOD
132	84	OR
133	85	ERROR
134	86	LINE
135	87	OFF
136	88	STEP
137	89	SPC

138	8A	TAB(
139	8B	ELSE
140	8C	THEN
141	8D	Missing
142	8E	OPENIN
143	8F	PTR
144	90	PAGE
145	91	TIME
146	92	LOMEM
147	93	HIMEM
148	94	ABS
149	95	ACS
150	96	ADVAL
151	97	ASC
152	98	ASN
153	99	ATN
154	9A	BGET
155	9B	COS
156	9C	COUNT
157	9D	DEG
158	9E	ERL
159	9F	ERR
160	A0	EVAL
161	A1	EXP
162	A2	EXT
163	A3	FALSE
164	A4	FN
165	A5	GET
166	A6	INKEY
167	A7	INSTR(
168	A8	INT
169	A9	LEN

170	AA	LN
171	AB	LOG
172	AC	NOT
173	AD	OPENUP
174	AE	OPENOUT
175	AF	PI
176	B0	POINT(
177	B1	POS
178	B2	RAD
179	B3	RND
180	B4	SGN
181	B5	SIN
182	B6	SQR
183	B7	TAN
184	B8	TO
185	B9	TRUE
186	BA	USR
187	BB	VAL
188	BC	VPOS
189	BD	CHR$
190	BE	GET$
191	BF	INKEY$
192	C0	LEFT$(
193	C1	MID$(
194	C2	RIGHT$(
195	C3	STR$
196	C4	STRING$(
197	C5	EOF
198	C6	AUTO
199	C7	DELETE
200	C8	LOAD
201	C9	LIST

202	CA	NEW
203	CB	OLD
204	CC	RENUMBER
205	CD	SAVE
206	CE	EDIT
207	CF	PTR
208	D0	PAGE
209	D1	TIME
210	D2	LOMEM
211	D3	HIMEM
212	D4	SOUND
213	D5	BPUT
214	D6	CALL
215	D7	CHAIN
216	D8	CLEAR
217	D9	CLOSE
218	DA	CLG
219	DB	CLS
220	DC	DATA
221	DD	DEF
222	DE	DIM
223	DF	DRAW
224	E0	END
225	E1	ENDPROC
226	E2	ENVELOPE
227	E3	FOR
228	E4	GOSUB
229	E5	GOTO
230	E6	GCOL
231	E7	IF
232	E8	INPUT
233	E9	LET

234	EA	LOCAL
235	EB	MODE
236	EC	MOVE
237	ED	NEXT
238	EE	ON
239	EF	VDU
240	F0	PLOT
241	F1	PRINT
242	F2	PROC
243	F3	READ
244	F4	REM
245	F5	REPEAT
246	F6	REPORT
247	F7	RESTORE
248	F8	RETURN
249	F9	RUN
250	FA	STOP
251	FB	COLOUR
252	FC	TRACE
253	FD	UNTIL
254	FE	WIDTH
255	FF	OSCLI

BASIC IV ROM Routines

Page Last Altered: undefined

9909 Evaluate Variable Name and return address of value

Submitted by Steve Fewell

Routine:getvar
Name: Evaluate Variable Name and return address of value
Starting Address: &98F5/&9909
Entry criteria: A contains the next character. Text PTR B points to the rest of the command line.
Exit: (&2A, &2B) point to the variable's value. &2C contains the value type.
A = 0 if error (variable not found); otherwise, A = &FF (?).
If A = 0 for error condition, then the carry flag has the following meanings:
Carry is clear if variable name was valid but variable doesn't exist (not declared).
Carry is set if no variable was specified (invalid variable name character/Syntax error).
If A is not 0 for variable found condition, then the carry flag has the following meanings:
Carry is clear if the variable contains a numerical value.
Carry is set if the variable is a string (non-numerical) value.

Description:

If called from &98F5 then set Text Pointer B to point to the current Text Pointer A location, and skip any leading spaces in text pointer B.

Firstly this routine checks for a 'peek/poke' address operator ('?', '!', '$').
This needs to be done first, because if a peek operator is found then this provides us with our address value without need to look further.
If a 'peek' operator is found, then exit with the address (to be peeked) placed in (&2A, &2B) and &2C containing the return value type (String for '$', 1-byte Integer for '?' or 4-byte Integer for '!').
Note: After a peek/poke operator is evaluated, this routine is exited. This means that a peek/poke address cannot be modified by an address modifier ('!', '?'), as address modifier operators can only be used with numeric variables.
If the next character was less than '@', but it wasn't a '?', '!' or '$' operator then exit (routine &99A6) with A = 0 and the carry set as the text is not a variable name.

Secondly, the routine checks for use of a resident integer variable (@% to Z%).
This is done second to ensure that resident integer variables are processed as fast as possible, so that the quickest access times an be given.
If a resident Integer variable is found then multiply the first character of the variable name by 4 (to obtain the page &4 starting address of the resident Integer variable).
and store the starting address in &2A. Store 4 in &2B (as this is the page where the variable

is located). Store 4 in &2C (as the return type is a 4-byte Integer value).
Check that the variable is not an Array by checking that the character after the '%' is an open bracket. If it isn't then we have a resident Integer variable, so jump to the exit section (&9998) of the evaluate variable routine.

Variable is not a peek/poke or a resident integer variable, so it is a normal variable
Begin by setting the default return value type and setting (&37, &38) to point to the start of the variable name.

This is done as follows:

* Store #5 in &2C (the variable's return value type). This defaults the return value to Floating-Point if a '%' or '$' is not found at the end of the variable name.
* Set A to point to the Text pointer B Offset.
* Set Y to point to the Text pointer B MSB.
* If the offset (A) is not 0, then add the Text pointer B LSB address value to the offset (A) and increment Y (if page overflowed).
* If the offset (A) is zero then add the Text pointer B LSB address value.
* If the Text pointer B LSB value is zero also, then decrement Y (to point to the previous page (the value in A is decremented to &FF in the next step!).
* Decrement the offset(A), to point to the first character of the variable name.
* Set zero page location &37 to the LSB address of the first character of the variable name.
* Set zero page location &38 to the MSB address of the first character of the variable name.

[9949] Check the first character of the variable name, as follows:
Load the first character of the variable name. Set Y to 1.
If the first character is between 'A' and 'Z' then increment the pointers (X and Y) to point to the next character and test the next character [995B].
If the first character is between '_' and 'z' then increment the pointers (X and Y) to point to the next character and test the next character [995B]. Note: this includes the characters '_', '£' and 'a'-'z'.
If the first character is between '0' and '9' then increment the pointers (X and Y) to point to the next character and test the next character [995B]. Note: his will not be the case as '0' to '9' are already rejected as being the first character in the variable name in routine &98C1.
If the first character is anything else then goto 9977 to indicate the end of the variable name has been reached.

[995B] Check the next character of the variable name, as follows:
Get the next character.
If the next character is between 'A' and 'Z' then increment the pointers (X and Y) to point to the next character and test the next character [995B].
If the next character is between '_' and 'z' then increment the pointers (X and Y) to point to the next character and test the next character [995B]. Note: this includes the characters '_', '£' and 'a'-'z'.
If the next character is between '0' and '9' then increment the pointers (X and Y) to point to the next character and test the next character [995B].
If the next character is anything else then goto 9977 to indicate the end of the variable name has been reached.

[9977] The end of the variable name has been reached, so check the following:
If Y = 1 then no valid characters were found (the first character was invalid), so exit [&99A6] with A = 0 and Carry set.
If the invalid character (last character) was '$' then goto &99DA to handle String variables.

String variables are handled as follows:

* Decrement &2C (the variable value type, from 5 to 4). This tricks the Get Array element routine (&99FE), if it is called, to handle Integer values. As String variables have a 4-byte parameter block, this works well as it tells the Get Array element address routine to work with 4-byte values.
* Increment the pointers (X and Y) (to point to the character after the '$')

* If the next character is an open bracket '(', then call &99FE to get the address of the specified element.
Otherwise, call &8085 to get the address of the variable block for the specified variable & store the text pointer B offset (X) back to &1B; & if the variable wasn't found (Zero flag set) then exit via (&99AA --> A = 0 and Carry clear).
* Store #&81 in &2C - to specify that the return type is a string value.
* exit with the carry set (String value) and A = &81. Note with string variables we do not need to check for address modifier operators '!' and '?' [as done by routine &9998 for numeric values] as a String variable cannot contain an address value in which to modify!

Now String variables have been dealt with ('$'), so the following code deals only with Integer and Floating-Point variables.
If the invalid character (last character) was '%' then decrement &2C (from 5 to 4) as the return type is now an Integer, Increment the pointers (Y and X) and load the next character.

If the next character is an open bracket, '(', then call &99FE (the Get Array element address routine). Otherwise: call &8085 to get the address of the variable block for the specified variable & store the text pointer B offset (X) back to &1B; & if the variable wasn't found (Zero flag set) then exit via (&99AA --> A = 0 and Carry clear).

Load the next character, and exit via routine &9998 to check for address modifier operators ('!' and '?').

9998 Exit the evaluate variable routine:
Now we have the address for a numerical variable value, we need to check whether the variable is followed by an address modifier operator ('!' or '?'). If it is then the numeric value of the variable needs to be modified by the number of bytes specified after the address modifier operator. I.e. the expression could be 'var!4', meaning that the value of variable 'var' needs to be adjusted by 4, and the return value type is a 4-byte Integer.

If an address modifier is found then the appropriate routine is called (&99AE for '!' or &99B0 for '?'). Otherwise, clear the carry (as numerical value), store the pointer B offset back in &1B and exit with A=&FF (as routine succeeded).

Disassembly for the Evaluate Variable Name and return address of value routine

98F5		165 011	A5 0B	LDA &0B
98F7		133 025	85 19	STA &19
98F9		165 012	A5 0C	LDA &0C
98FB		133 026	85 1A	STA &1A
98FD		164 010	A4 0A	LDY &0A
98FF		136	88	DEY
9900		200	C8	INY
9901		132 027	84 1B	STY &1B
9903		177 025	B1 19	LDA (&19),Y
9905		201 032	C9 20	CMP#&20
9907		240 247	F0 F7	BEQ -9 --> &9900
9909	@	201 064	C9 40	CMP#&40
990B		144 180	90 B4	BCC -76 --> &98C1 Check for '!', '?' or '$' address peek operators
990D	[	201 091	C9 5B	CMP#&5B

```
990F       176 026     B0 1A     BCS 26 --> &992B
9911       010         0A        ASL A
9912       010         0A        ASL A
9913   *   133 042     85 2A     STA &2A
9915       200         C8        INY
9916       177 025     B1 19     LDA (&19),Y
9918   %   201 037     C9 25     CMP#&25
991A       208 015     D0 0F     BNE 15 --> &992B
991C       169 004     A9 04     LDA#&04
991E   +   133 043     85 2B     STA &2B
9920       162 004     A2 04     LDX#&04
9922   ,   134 044     86 2C     STX &2C
9924       200         C8        INY
9925       177 025     B1 19     LDA (&19),Y
9927   (   201 040     C9 28     CMP#&28
9929   m   208 109     D0 6D     BNE 109 --> a href="#9998">&9998 Exit the Evaluate Variable routine
992B       162 005     A2 05     LDX#&05
992D   ,   134 044     86 2C     STX &2C
992F       024         18        CLC
9930       164 026     A4 1A     LDY &1A
9932       165 027     A5 1B     LDA &1B
9934       170         AA        TAX
9935       208 008     D0 08     BNE 8 --> &993F
9937   :   058         3A        DEC A
9938   e   101 025     65 19     ADC &19
993A       176 009     B0 09     BCS 9 --> &9945
993C       136         88        DEY
993D       128 006     80 06     BRA 6 --> &9945
993F   :   058         3A        DEC A
9940   e   101 025     65 19     ADC &19
9942       144 001     90 01     BCC 1 --> &9945
9944       200         C8        INY
9945   7   133 055     85 37     STA &37
9947   8   132 056     84 38     STY &38
9949       160 001     A0 01     LDY#&01
994B   7   177 055     B1 37     LDA (&37),Y
994D   A   201 065     C9 41     CMP#&41
994F       176 026     B0 1A     BCS 26 --> &996B
9951   0   201 048     C9 30     CMP#&30
9953   "   144 034     90 22     BCC 34 --> &9977
9955   :   201 058     C9 3A     CMP#&3A
9957       176 030     B0 1E     BCS 30 --> &9977
9959       232         E8        INX
995A       200         C8        INY
```

```
995B  7  177 055          B1 37       LDA (&37),Y
995D  A  201 065          C9 41       CMP#&41
995F     176 010          B0 0A       BCS 10 --> &996B
9961  0  201 048          C9 30       CMP#&30
9963     144 018          90 12       BCC 18 --> &9977
9965  :  201 058          C9 3A       CMP#&3A
9967     144 240          90 F0       BCC -16 --> &9959
9969     128 012          80 0C       BRA 12 --> &9977
996B  [  201 091          C9 5B       CMP#&5B
996D     144 234          90 EA       BCC -22 --> &9959
996F  _  201 095          C9 5F       CMP#&5F
9971     144 004          90 04       BCC 4 --> &9977
9973  {  201 123          C9 7B       CMP#&7B
9975     144 226          90 E2       BCC -30 --> &9959
9977     192 001          C0 01       CPY#&01
9979  +  240 043          F0 2B       BEQ 43 --> &99A6 Error: Set carry
997B  $  201 036          C9 24       CMP#&24
997D  [  240 091          F0 5B       BEQ 91 --> &99DA Handle String variables
997F  %  201 037          C9 25       CMP#&25
9981     208 006          D0 06       BNE 6 --> &9989
9983  ,  198 044          C6 2C       DEC &2C
9985     232              E8          INX
9986     200              C8          INY
9987  7  177 055          B1 37       LDA (&37),Y
9989  (  201 040          C9 28       CMP#&28
998B  H  240 072          F0 48       BEQ 72 --> &99D5 Handle Numeric Array
998D     032 133 128      20 85 80    JSR &8085 Get address of variable
9990     240 024          F0 18       BEQ 24 --> &99AA Error: Clear carry
9992     134 027          86 1B       STX &1B
9994     164 027          A4 1B       LDY &1B
9996     177 025          B1 19       LDA (&19),Y
9998  !  201 033          C9 21       CMP#&21
999A     240 018          F0 12       BEQ 18 --> &99AE '!' address modifier operator
999C  I? 073 063          49 3F       EOR#&3F
999E     240 016          F0 10       BEQ 16 --> &99B0 '?' address modifier operator
99A0     024              18          CLC
99A1     132 027          84 1B       STY &1B
99A3     169 255          A9 FF       LDA#&FF
99A5  `  096              60          RTS
99A6     169 000          A9 00       LDA#&00
99A8  8  056              38          SEC
99A9  `  096              60          RTS
99AA     169 000          A9 00       LDA#&00
```

99AC		024	18	CLC
99AD	`	096	60	RTS

Handle numeric array

99D5	032 254 153	20 FE 99	JSR &99FE Get address of required Array element
99D8	128 186	80 BA	BRA -70 --> &9994

Handle String variables

99DA	,	198 044	C6 2C	DEC &2C
99DC		232	E8	INX
99DD		200	C8	INY
99DE	7	177 055	B1 37	LDA (&37),Y
99E0	(	201 040	C9 28	CMP#&28
99E2		240 013	F0 0D	BEQ 13 --> &99F1
99E4		032 133 128	20 85 80	JSR &8085 Get address of variable
99E7		240 193	F0 C1	BEQ -63 --> &99AA Error: Clear carry
99E9		134 027	86 1B	STX &1B
99EB		169 129	A9 81	LDA#&81
99ED	,	133 044	85 2C	STA &2C
99EF	8	056	38	SEC
99F0	`	096	60	RTS
99F1		032 254 153	20 FE 99	JSR &99FE Get address of required Array element
99F4		128 245	80 F5	BRA -11 --> &99EB

Check for '!', '$' or '?' address peek operators

98C1	!	201 033	C9 21	CMP#&21
98C3		240 012	F0 0C	BEQ 12 --> &98D1 '!' address peek operator
98C5	$	201 036	C9 24	CMP#&24
98C7		240 019	F0 13	BEQ 19 --> &98DC '$' address peek operator
98C9	I?	073 063	49 3F	EOR#&3F
98CB		240 006	F0 06	BEQ 6 --> &98D3 '?' address peek operator
98CD		169 000	A9 00	LDA#&00
98CF	8	056	38	SEC
98D0	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8085 Get Address of Variable

Submitted by Steve Fewell

Routine:varaddr
Name: Get Address of Variable
Starting Address: &8085
Entry criteria: &37 and &38 point to the variable name. Y = Variable name length.
Exit: &2A and &2B = the address of the variable block for the variable name [LSB first].
A = &00 if variable not found. &39 = Length of variable Name (Y).

Description:

Store the offset to the end of the variable name in &39 (variable name length).
Load the first character of the variable name. Multiply the ASCII value of the first character by 2 and store in Y. Y now points to the page &4 offset for the variable pointer table.
i.e.: If first Char = "A" [ASCII value=&41], then the variable pointer table for "A" variables is located at &0482.
Note: The resident integer variables (@%, A%, B%, etc...) are handled separately.

Load the MSB of the address stored at the specified variable pointer table location.
If the MSB of the address is zero, then no "A" variables have been defined, so exit with A=&00.

Store the MSB of the variable pointer table address in &2B.
Store the LSB of the variable pointer table address in &2A.

&80A6: Load the 3rd byte of the variable lookup table (&2A, &2B). The 3rd byte is the beginning of the rest of the variable name [as the variable name is stored without its first character].

If the variable name character is zero, then we have reached the end of the variable name, so check whether the length of the variable we are searching for (&39) is the same

as the variable we have found (Y). If not then jump to **&8099** to look at the next variable in the pointer table. If equal, then we have found our variable, so jump to **&80C3**.
Otherwise (end of name not reached); [**&80B6:**] Compare the variable name character (from lookup table) with the next character pointed to by &37, &38.
If the characters don't match then this is not our variable, so goto 8099 to look at next variable in the pointer table.
Increment Y, and check whether the variable length (&39) has been reached, if not then (**&80B2**) load the next character of the variable name in the pointer table (and jump to **&8099** (look at next variable), if the the end of the name was reached (while we still had more characters in the variable name we were looking for)); otherwise compare this next character [**&80B6**].
If the end of the variable we are looking for has been reached then check that the next character of the variable name in the lookup table is zero (indicating the end of the name); if it isn't, then jump to **&8099** to look at the next variable in the lookup table.

Otherwise -> **[&80C3:] We have found our variable.**
Add length of variable to the address in &2A, &2B -> so that &2A, &2B now points to the variable details/variable contents section of the variable block (which is located directly after the variable name). And exit (updating the MSB if any overflow occurred).

[&8099:] Look at next variable in lookup table.
This routine checks byte 2 of the variable block, if it is zero then there is no next variable address, so exit (with A=00) as the variable doesn't exist.
Otherwise, store the next variable address in &2A (LSB) and &2B and jump to **&80A6** to compare the variable name with the one we are looking for.

Disassembly for the Get Address of Variable routine

8085	9	132 057	84 39	STY &39
8087		160 001	A0 01	LDY#&01
8089	7	177 055	B1 37	LDA (&37),Y
808B		010	0A	ASL A
808C		168	A8	TAY
808D		185 001 004	B9 01 04	LDA &0401,Y
8090	:	240 058	F0 3A	BEQ 58 --> &80CC
8092	+	133 043	85 2B	STA &2B
8094		185 000 004	B9 00 04	LDA &0400,Y
8097		128 011	80 0B	BRA 11 --> &80A4
8099		160 001	A0 01	LDY#&01
809B	*	177 042	B1 2A	LDA (&2A),Y
809D	-	240 045	F0 2D	BEQ 45 --> &80CC
809F		168	A8	TAY
80A0	*	178 042	B2 2A	LDA (&2A)

80A2	+	132 043	84 2B	STY &2B
80A4	*	133 042	85 2A	STA &2A
80A6		160 002	A0 02	LDY#&02
80A8	*	177 042	B1 2A	LDA (&2A),Y
80AA		208 010	D0 0A	BNE 10 --> &80B6
80AC	9	196 057	C4 39	CPY &39
80AE		208 233	D0 E9	BNE -23 --> &8099
80B0		128 017	80 11	BRA 17 --> &80C3
80B2	*	177 042	B1 2A	LDA (&2A),Y
80B4		240 227	F0 E3	BEQ -29 --> &8099
80B6	7	209 055	D1 37	CMP (&37),Y
80B8		208 223	D0 DF	BNE -33 --> &8099
80BA		200	C8	INY
80BB	9	196 057	C4 39	CPY &39
80BD		208 243	D0 F3	BNE -13 --> &80B2
80BF	*	177 042	B1 2A	LDA (&2A),Y
80C1		208 214	D0 D6	BNE -42 --> &8099
80C3		152	98	TYA
80C4	e*	101 042	65 2A	ADC &2A
80C6	*	133 042	85 2A	STA &2A
80C8		144 002	90 02	BCC 2 --> &80CC
80CA	+	230 043	E6 2B	INC &2B
80CC	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

99AE '!' and '?' address modifier operators

Submitted by Steve Fewell

Routine:addrmodify
Name: '!' and '?' address modifier operators
Starting Address: &99AE (for '!') or &99B0 with A=0 (for '?')
Entry criteria: &2A and &2B point to the contents of the variable.
The Text Pointer B points to the next character of the program line.
Exit:(&2A, &2B) contain the adjusted address (variable value + modifier value).
&2C contains the return type (of the value to obtain from the new address).

Description:

Store the return value type on the stack (0 for 1-byte Integer or 4 for 4-byte Integer).
Store back the offset in &1B, as we have correctly evaluated the variable address.
Load the variable value to obtain the base-memory address to modify.
If the variable is a string, then it is not a Memory address so generate a Type mismatch error.

Store the Integer base-address MSB <&2B) on the Stack.
Store the Integer base-address LSB (&2A) on the Stack.
Get the Integer value at PTR B. This provides the value to modify the base-address by.

Add the Integer value to the Base memory address (from the Stack). Now (&2A, &2B) contain the modified address.
Exit with Carry clear and A = #&FF (meaning that the routine ended sucessfully).

Disassembly for the '!' and '?' address modifier operators routine

99AE		169 004	A9 04	LDA#&04
99B0	H	072	48	PHA
99B1		200	C8	INY
99B2		132 027	84 1B	STY &1B
99B4		032 160 177	20 A0 B1	JSR &B1A0 Load Variable
99B7		032 191 150	20 BF 96	JSR &96BF Check if Integer and Convert if Float
99BA	+	165 043	A5 2B	LDA &2B
99BC	H	072	48	PHA

99BD	*	165 042	A5 2A	LDA &2A
99BF	H	072	48	PHA
99C0		032 180 150	20 B4 96	JSR &96B4 Get Integer value at PTR B
99C3		024	18	CLC
99C4	h	104	68	PLA
99C5	e*	101 042	65 2A	ADC &2A
99C7	*	133 042	85 2A	STA &2A
99C9	h	104	68	PLA
99CA	e+	101 043	65 2B	ADC &2B
99CC	+	133 043	85 2B	STA &2B
99CE	h	104	68	PLA
99CF	,	133 044	85 2C	STA &2C
99D1		024	18	CLC
99D2		169 255	A9 FF	LDA#&FF
99D4	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Check Number and Convert to Integer if Float

Submitted by Steve Fewell

Routine: CheckNum&ConvFloat
Name: Check Number and Convert to Integer if Number is Floating-Point
Starting Address: &96BE
Entry criteria: The Accumulator specifies which type of value has just been processed.
Exit: If the value was Floating-Point, then the FWA's value will be converted to Integer and stored in the IWA.

Description:
Transfers the current Variable Type [stored in A] to Y in order to update the status flags.

If the current Variable Type = 0 (String) then a Type Mismatch error is given.

If the current Variable Type = positive (i.e. &40, Integer) then nothing to do as the number is already integer, so exit with the IWA's value intact.

Otherwise, the current Variable Type must be negative (i.e. &FF, Floating-Point), so the number is floating-point. So we need to convert the number into an Integer and return it in the IWA.

The Convert Float to Integer routine is used to return a Two's-complement Integer value in the FWA Mantissa (stored Most significant byte first, least significant byte last). Now, store the Integer Number in the IWA - stored the right way around (least significant byte first, and Most significant byte last), and exit.

Disassembly for the Check Number and Convert to Integer if Float routine

96BE		168	A8	TAY
96BF		240 022	F0 16	BEQ 22 --> &96D7
96C1		016 019	10 13	BPL 19 --> &96D6
96C3	B	032 066 130	20 42 82	JSR &8242 Convert Float to Integer
96C6	1	165 049	A5 31	LDA &31
96C8	-	133 045	85 2D	STA &2D
96CA	2	165 050	A5 32	LDA &32
96CC	,	133 044	85 2C	STA &2C
96CE	3	165 051	A5 33	LDA &33
96D0	+	133 043	85 2B	STA &2B
96D2	4	165 052	A5 34	LDA &34
96D4	*	133 042	85 2A	STA &2A
96D6	`	096	60	RTS
96D7	L	076 146 144	4C 92 90	JMP &9092 Type Mismatch error 6

BASIC IV ROM Routines

Glossary

Submitted by Steve Fewell
Page Last Altered: undefined

A

A usually refers to the Accumulator. This is an 8-bit register within the processor, which is written/read via many assembler commands.

argp

The argp is the argument pointer - a pointer to a floating-point variable. Its location is usually &4A (for low byte) and &4B (for high byte).

Carry Flag

The Carry flag is one of the processor flags, which can contain the values 0 or 1. It is often used to implement carrying and borrowing functionality when calculating with numbers larger than 8-bits in length. If it has a value of 1 then this indicates that a carry has occurred. This tells the command ADC to include the carry in its calculations. If it has a value of 0 then this indicates that a borrow has occurred. This tells the command SBC to include the borrow in its calculations. Programs usually reset this flag to 0 before beginning any add calculations, and set it to 1 before beginning any subtract calculations, as this clears the carry/borrow condition.

Error Vector

The BASIC error vector is located in zero-page locations &16 (low byte) and &17 (high byte). This vector contains the address of the BASIC error code which BASIC will execute after an error condition occurs. This vector will point to the code located after an ON ERROR statement in a program. The BASIC instructions pointed to by the error vector are executed by the BASIC interpreter when an error condition occurs.

FWA
The FWA is the Floating-point working area 'A', this is an 8 byte zero-page location for the temporary storage of one floating-point number. Its location is &2E for the sign byte, &2F for the Exponent Overflow, &30 for the number's Exponent and &31 (most significant byte) to &35 (least significant byte) for the number's Mantissa. Byte 5 of the mantissa (&35) is a rounding byte, as it doesn't form part of the number when it is copied from the FWA and stored to a floating-point variable. It allows extra precision during calculations while the number is in the FWA. There is also a Floating-point working area 'B'.

FWB
The FWB is the Floating-point working area 'B', this is a 7 byte zero-page location for the temporary storage of one floating-point number. Its location is &3B for the sign byte, &3C for the number's Exponent and &3D (most significant byte) to &41 (least significant byte) for the number's Mantissa. It does not have an Exponent Overflow byte (like the FWA). Byte 5 of the mantissa (&41) is a rounding byte, as it doesn't form part of the number when it is copied from the FWB and stored to a floating-point variable. It allows extra precision during calculations while the number is in the FWB.

Heap Pointer
The BASIC Heap Pointer is located in zero page memory at locations &02 (low byte) and &03 (high byte). It contains the address of the top of the Heap. The Heap is the work space for the current program. It contains all variables used by the program. The Heap expands (as more items get allocated to memory) upwards starting from the top of the program code. When the Heap clashes with the Stack, there is no memory left, and a "No Room" error is produced.

IWA
The IWA is the Integer working area 'A', this is a 4-byte zero-page location for the temporary storage of one 32-bit Integer number. Its location is &2A (least significant byte) to &2D (most significant byte).

N Flag
The N flag is one of the processor flags. It is set if the processor has just worked on (or calculated) a value that has bit 7 set. In two's compliment this indicates a negative number.

PTRA
PTRA is BASIC's primary Text Pointer. It is located in a 3-byte zero page location: &0B and &0C are the PTRA Base, i.e. The address of the first character in the text string pointed to. And &0A is the PTRA offset (this points to the current character being processed in the Text string). This pointer usually points to the current position in the Command/Program line as the line is processed.

PTRB
PTRB is BASIC's second Text Pointer. It is located in a 3-byte zero page location: &19 and &1A are the PTRB Base, i.e. The address of the first character in the text string pointed to. And &1B is the PTRB offset (this points to the current character being processed in the Text string). This pointer usually points

to the current value/variable being processed.

Reset
To reset a flag, or a bit, usually means giving it a value of 0.

Set
To set a flag, or a bit, usually means giving it a value of 1.

Stack Pointer
The BASIC Stack Pointer is located in zero page memory at locations &04 (low byte) and &05 (high byte). It contains the address of the top item on the stack. In numeric calculations, this is usually the next number to process. When an item is popped from the stack, the Stack Pointer is usually moved up so that it points to the previous item on the Stack. The BASIC Stack grows downwards in memory from HIMEM.

SWA
The SWA is the String working area, this location has a maximum of 255-bytes for the storage of one string. Its location is &600 (First Character), with each subsequent byte representing the subsequent character, until the end of the String (or to the maximum of &6FF). The length of the SWA (pointer to the last character) is stored in zero page location &36. This means that a terminator byte is not needed to signify the end of the string, as this length byte provides the required information.

VARTOP
VARTOP is a pointer to the next free variable storage location. It is located in zero page locations &02-&03. When the value of VARTOP exceeds the value pointed to by the BASIC Stack pointer (&04-&05) then a No Room error occurs as BASIC has run out of storage space for the program's variables. VARTOP is the same as the Heap Pointer.

X
X usually refers to the X register. This is an 8-bit register within the processor, which is written/read via many assembler commands. It is often used as an index.

Y
Y usually refers to the Y register. This is an 8-bit register within the processor, which is written/read via many assembler commands. It is often used as an index in address lookups.

Zero Flag
The Zero flag is one of the processor flags. It is set if the processor has just worked on (or calculated) a value of zero.

BASIC IV ROM Routines

Page Last Altered: undefined

Convert Floating-Point to Integer

Submitted by Steve Fewell

Routine: ConvFWAtoInt
Name: Convert Floating-Point to Integer
Starting Address: &8242
Entry criteria: The FWA contains a Floating-Point number.
Exit: The FWA value has been truncated, and the Integer part of it's value has been retained.

Description:
Load A with the FWA Exponent Byte, if it isn't negative (offset by &80) then the Floating-Point value is less than or equal to zero, so jump to Clear FWA and exit with the Integer value set to zero.

If the FWA Mantissa byte 1 is zero then assume that the FWA contains the value zero [?], so exit by doing an Integer Reverse Order Complement of the FWA's Mantissa value. I'm not sure why this is necessary though, as if the number zero doesn't need to be stored in two's complement form.

Now, keep on moving the bite in the FWA Mantissa right one position, until we are left with the Integer part of the number (Stored most significant Byte first, and least significant Byte in ?&34). This achieves two goals, first it gets rid of the fractional part of the number, and second it right-aligns the Integer value and prefixes it with leading zeros (The Integer value will still be stored the opposite way around to an Integer in the IWA though!).
Here is a comparison showing how a Two's-Complement Integer value is stored in the IWA, and how it is stored in the FWA:

Integer in IWA		**Integer in FWA**
Byte	**Description**	**Byte**
&2A	Least Significant Byte	&34
&2B	Integer Byte 2	&33
&2C	Integer Byte 3	&32

&2D	Most Significant Byte	&31

Note: The Floating-Points exponent is the number of bits which make up the Integer part of the number offset by &80 (128). Additionally, the FWA Mantissa Rounding Byte is ignored during this conversion.

To achieve the two goals stated above, BASIC does the following:
Keep dividing the FWA Mantissa by 2 (Moving the bits right one position, making the first bit zero, and loosing the last bit) and incrementing the exponent (Stored in the Accumulator), so that the exponent still points to the imaginary decimal point which separates the Integer part from the fractional part, until the exponent is 32 (&A0). I.e. This occurs when the start of the fraction has just been moved out of the end of the Mantissa and lost.
This operation (divide Mantissa by 2 and Increment Exponent) is done once at the beginning, but then if more than 7 bits need to be shifted right, then the alternative quicker method (Divide Mantissa by 16 and Add 8 to Exponent) is used [i.e. move the Mantissa bytes along one place, loosing the last (?&34) byte]. If less than 8 bits need to be shifted, then the single-bit method is used.

If at any point the exponent goes over &A0, then a Too Big error is produced (as the number is > 32 bits long), otherwise, when the exponent is equal to &A0, the Exponent [A] is stored back in the FWA Exponent Byte and an Integer Reverse Order Complement is done (which converts the Integer into Two's-complement notation if a negative result is required - i.e. if the FWA's Sign Byte is negative).

Disassembly for the Convert Floating-Point to Integer routine

8242	0	165 048	A5 30	LDA &30
8244	,	016 044	10 2C	BPL 44 --> &8272
8246	1	164 049	A4 31	LDY &31
8248	4	240 052	F0 34	BEQ 52 --> &827E Move the fractional value from the FWA to the FWB
824A	F1	070 049	46 31	LSR &31
824C	f2	102 050	66 32	ROR &32
824E	f3	102 051	66 33	ROR &33
8250	f4	102 052	66 34	ROR &34
8252		026	1A	INC A
8253	h	240 104	F0 68	BEQ 104 --> &82BD Too big error
8255		201 160	C9 A0	CMP#&A0
8257	g	176 103	B0 67	BCS 103 --> &82C0 Check Exponent and make Two's Complement Integer (if necessary)
8259		201 153	C9 99	CMP#&99
825B		176 237	B0 ED	BCS -19 --> &824A
825D	i	105 008	69 08	ADC#&08
825F	3	164 051	A4 33	LDY &33
8261	4	132 052	84 34	STY &34
8263	2	164 050	A4 32	LDY &32

8265	3	132 051	84 33	STY &33
8267	1	164 049	A4 31	LDY &31
8269	2	132 050	84 32	STY &32
826B	d1	100 049	64 31	STZ &31
826D		128 230	80 E6	BRA -26 --> &8255

Jump to Clear FWA

8272	L	076 180 166	4C B4 A6	JMP &A6B4 Clear FWA

Move the fractional value from the FWA to the FWB

827E	D	240 068	F0 44	BEQ 68 --> &82C4 Integer Reverse Order Complement
8280	F1	070 049	46 31	LSR &31
8282	f2	102 050	66 32	ROR &32
8284	f3	102 051	66 33	ROR &33
8286	f4	102 052	66 34	ROR &34
8288	f=	102 061	66 3D	ROR &3D
828A	f>	102 062	66 3E	ROR &3E
828C	f?	102 063	66 3F	ROR &3F
828E	f@	102 064	66 40	ROR &40
8290		026	1A	INC A
8291	*	240 042	F0 2A	BEQ 42 --> &82BD Too big error
8293		201 160	C9 A0	CMP#&A0
8295	)	176 041	B0 29	BCS 41 --> &82C0 Check Exponent and make Two's Complement Integer (if necessary)
8297		201 153	C9 99	CMP#&99
8299		176 229	B0 E5	BCS -27 --> &8280
829B	i	105 008	69 08	ADC#&08
829D	?	164 063	A4 3F	LDY &3F
829F	@	132 064	84 40	STY &40
82A1	>	164 062	A4 3E	LDY &3E
82A3	?	132 063	84 3F	STY &3F
82A5	=	164 061	A4 3D	LDY &3D
82A7	>	132 062	84 3E	STY &3E
82A9	4	164 052	A4 34	LDY &34
82AB	=	132 061	84 3D	STY &3D
82AD	3	164 051	A4 33	LDY &33
82AF	4	132 052	84 34	STY &34
82B1	2	164 050	A4 32	LDY &32

82B3	3	132 051	84 33	STY &33
82B5	1	164 049	A4 31	LDY &31
82B7	2	132 050	84 32	STY &32
82B9	d1	100 049	64 31	STZ &31
82BB		128 214	80 D6	BRA -42 --> &8293

Jump to "Too Big" error

82BD	L	076 197 166	4C C5 A6	JMP &A6C5 Too Big error number 20

Check Exponent and make Two's Complement Integer (if necessary)

82C0		208 251	D0 FB	BNE -5 --> &82BD
82C2	0	133 048	85 30	STA &30
82C4		...		Integer Reverse Order Complement

BASIC IV ROM Routines

Page Last Altered: undefined

Integer Reverse Order Complement (compliment Integer in FWA)

Submitted by Steve Fewell

Routine: icompFWA
Name: Test and Complement Integer in FWA
Starting Address: &82C4
Entry criteria: The FWA contains an Integer number.
Exit: The Integer Number in the FWA has been complemented (FWA = zero - FWA)

Description:
Complements an Integer stored in the Floating-Point A Mantissa (which is stored the opposite way to Integers, that is: most significant byte first and least significant byte last; whereas Integers are stored least significant byte first, most significant byte last [as two's complement is used]).

If the FWA Sign Bit is positive, then no compliment is required (as we already have a positive Integer), so exit.

[82C8] As the FWA Sign Bit is negative (i.e. &FF), we need to complement the Integer, by subtracting each of the bytes (?&34 -> ?&31) from zero, using the carry flag to keep track of any borrows.

Disassembly for the Test and Compliment Integer in FWA routine

82C4	.	165 046	A5 2E	LDA &2E
82C6		016 023	10 17	BPL 23 --> &82DF
82C8	8	056	38	SEC

82C9		160 000	A0 00	LDY#&00
82CB		152	98	TYA
82CC	4	229 052	E5 34	SBC &34
82CE	4	133 052	85 34	STA &34
82D0		152	98	TYA
82D1	3	229 051	E5 33	SBC &33
82D3	3	133 051	85 33	STA &33
82D5		152	98	TYA
82D6	2	229 050	E5 32	SBC &32
82D8	2	133 050	85 32	STA &32
82DA		152	98	TYA
82DB	1	229 049	E5 31	SBC &31
82DD	1	133 049	85 31	STA &31
82DF	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Clear FWA

Submitted by Steve Fewell

Routine: aclear
Name: Clear FWA
Starting Address: &A6B4
Entry criteria: None
Exit: Every byte in the FWA contains the value 0 (this indicates the number 0).

Description:
Sets every byte of the FWA (locations &2E to &35) to zero. This uniquely identifies the number zero.

This routine can be jumped in at address A6B8, in this case all bytes of the FWA are cleared, except for the Exponent and Mantissa byte 1 (the most significant byte of the mantissa).

Disassembly for the Clear FWA routine

A6B4	d0	100 048	64 30	STZ &30
A6B6	d1	100 049	64 31	STZ &31
A6B8	d.	100 046	64 2E	STZ &2E
A6BA	d/	100 047	64 2F	STZ &2F
A6BC	d2	100 050	64 32	STZ &32
A6BE	d3	100 051	64 33	STZ &33
A6C0	d4	100 052	64 34	STZ &34

A6C2	d5	100 053	64 35	STZ &35
A6C4	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

96B4 Get Integer value at PTR B

Submitted by Steve Fewell

Description:

This routine calls the Get value (Keyword, variable, value, open bracket) routine (&AD36)
to obtain the next value on the command line.
If the value is a String, then a Type Mismatch error will be generated.
If this value is a Float then it will be converted to an Integer.
Exit with the Integer value.

Disassembly for the Get Integer Value at PTR B

96B4	6	032 054 173	20 36 AD	JSR &AD36 Get Value (Keyword, variable, value, open bracket)
96B7		128 006	80 06	BRA 6 --> &96BF Check if Integer & convert if float or error if String

---

BASIC IV ROM Routines

Page Last Altered: undefined

AD36 Evaluate Value/Variable/Open Bracket/BASIC Function

Submitted by Steve Fewell

Routine:evalValue
Name: Evaluate Value/Variable/Open Bracket/BASIC Function
Starting Address: &AD36
Entry criteria: &19, &1A & &1B (Text Pointer B) point to the program text location at which we should begin our evaluation.
Exit: Any variable/value found will be loaded, or the function will have been executed, Or the value inside the brackets ['(', ')'] will have been evaluated.

Description:

Load the ASCII value of the next character into the accumulator & increment the Text pointer B offset (keeping a previous copy of it in Y). Skip any spaces in the text by incrementing the Text Pointer B offset (past the space character) and loading the next character.

Now, we have our first non-space character in the accumulator.
If the character is '-', then we have a leading [unary] '-', so call &ACD7 to evaluate the expression after the '-', and complement the result (or Type mismatch error if result is a string).
If the character is '"' [open quote], then call &AD19 to extract string into the SWA & exit.
If the character is '+', leading [unary] '+', then skip any spaces (after the '+') and load the next non-space character after the '+'.

If the character is a BASIC Statement Keyword (non-function, i.e. >= 198 'AUTO') then generate a 'No such variable' error as a BASIC Statement keyword can only occur at the beginning of a line, and not in the middle of an expression!

If the character is >= 142 &8E (OPENIN) [& < 198 &C5], then this represents a BASIC Function Keyword, so execute the portion of code (in the BASIC ROM) which deals with the appropriate function.
[9019] To do this, the character ASCII Code is multiplied by 2 & added to the base address &874D to form a pointer to the execution address (LSB first, MSB next) of the required function. This resulting address is jumped to.
Example 1: Character = &8E [OPENIN token] = 10001110 multiply by 2 = 00011100 (which is &1C in hex).
So, &874D + &1C = &8769 (The LSB of the execution address for the OPENIN function (&876A is the MSB of the address)).
Example 2: Character = &C5 [EOF token] = 11000101 multiply by 2 = 10001010 (which is &8A in hex).
So, &874D + &8A = &87D7 (The LSB of the execution address for the EOF function (&87D8 is the MSB of the address)).

BASIC Keywords between 128 and 141 are not considered, as these Keywords are used in the middle of statements (and are not functions), so the statements/expression handler will deal with these values. These keywords are as follows: AND, DIV, EOR, MOD, OR, ERROR, LINE, OFF, STEP, SPC, TAB(, ELSE and THEN.

If the character is >&eq;"." and <"?" then call routine &A2E1 to evaluate the numeric value at Text Pointer B. This routine will place the result either into the IWA or FWA (depending on how large the value is, or whether it contains a fractional part).
If the carry flag is clear then the routine failed to convert the ASCII text to a numeric value, so issue a 'No such variable' error. Otherwise exit.

If the character is "&" then call routine &ADB7 to extract the Hex Number and exit.
If the character is "(" then call routine &ADAC to get the result of the expression (inside the brackets) & check for a closing bracket ')' at the end (issue a 'Missing)' error if not present) & exit.

If the character is anything else, then it must be a variable.
So, call routine &9909 to evaluate the variable name (or array reference) and set (&2A, &2B) to the address of the variable's value (or the address of the value of the specified Array element).
If &9909 returns with the zero flag clear, then the variable was evaluated sucessfully so exit via routine &B1A0 to Load the variables value into the appropriate location (IWA/FWA/SWA).
Otherwise (zero flag was set), the variable either wasn't found, or a valid variable name was not found, so check the value of &28, which is the OPT flag.
if Bit 1 of this flag (the second bit from the right) [Relocate is on?] is not set then issue a 'No such variable' error. If the carry flag (on exit from routine &9909) is set then the text was not a valid variable name, so issue a 'No such variable' error.
Otherwise, (the second bit is set, meaning that OPT = Relocate?) Store back the Text pointer B offset in &1B and set the IWA to the 2-byte address value from locations &0440-&0441 (&0440 is the LSB), this is the value of the address stored in variable P%, then exit.

Disassembly for the Evaluate Value/Variable/Open Bracket/BASIC Function routine

AD36		164 027	A4 1B	LDY &1B
AD38		230 027	E6 1B	INC &1B
AD3A		177 025	B1 19	LDA (&19),Y
AD3C		201 032	C9 20	CMP#&20
AD3E		240 246	F0 F6	BEQ -10 --> &AD36
AD40	-	201 045	C9 2D	CMP#&2D
AD42		240 147	F0 93	BEQ -109 --> &ACD7 Complement result of expression
AD44	"	201 034	C9 22	CMP#&22
AD46		240 209	F0 D1	BEQ -47 --> &AD19 Extract String
AD48	+	201 043	C9 2B	CMP#&2B
AD4A		208 003	D0 03	BNE 3 --> &AD4F
AD4C		032 213 142	20 D5 8E	JSR &8ED5 Skip Spaces (Ptr B) & Get next character
AD4F		201 142	C9 8E	CMP#&8E
AD51		144 007	90 07	BCC 7 --> &AD5A
AD53		201 198	C9 C6	CMP#&C6
AD55	5	176 053	B0 35	BCS 53 --> &AD8C No such variable error

AD57	L	076 025 144	4C 19 90	JMP &9019 Jump to BASIC Keyword evaluation routine
AD5A	?	201 063	C9 3F	CMP#&3F
AD5C		176 012	B0 0C	BCS 12 --> &AD6A
AD5E	.	201 046	C9 2E	CMP#&2E
AD60		176 018	B0 12	BCS 18 --> &AD74
AD62	&	201 038	C9 26	CMP#&26
AD64	Q	240 081	F0 51	BEQ 81 --> &ADB7 Extract Hex number
AD66	(	201 040	C9 28	CMP#&28
AD68	B	240 066	F0 42	BEQ 66 --> &ADAC Evaluate expression & check for ')'
AD6A		198 027	C6 1B	DEC &1B
AD6C		032 009 153	20 09 99	JSR &9909 Evaluate variable/array name & return the value's address
AD6F		240 009	F0 09	BEQ 9 --> &AD7A If variable wasn't found
AD71	L	076 160 177	4C A0 B1	JMP &B1A0 Load Variable's value
AD74		032 225 162	20 E1 A2	JSR &A2E1 ASCNUM: Extract ASCII number at PTRB to FWA/IWA
AD77		144 019	90 13	BCC 19 --> &AD8C No such variable error
AD79	`	096	60	RTS
AD7A	(	165 040	A5 28	LDA &28
AD7C	)	041 002	29 02	AND#&02
AD7E		208 012	D0 0C	BNE 12 --> &AD8C No such variable error
AD80		176 010	B0 0A	BCS 10 --> &AD8C No such variable error
AD82		134 027	86 1B	STX &1B
AD84	@	173 064 004	AD 40 04	LDA &0440
AD87	A	172 065 004	AC 41 04	LDY &0441
AD8A	k	128 107	80 6B	BRA 107 --> &ADF7 [BRA &21 -> &AE1A (Load IWA with 2-byte value)]

BASIC IV ROM Routines

Page Last Altered: undefined

ACD7 Complement Result (get result then complement it)

Submitted by Steve Fewell

Routine: resultcomp
Name: Compliment Result
Starting Address: &ACD7

Description:
Calls AD4C to get the result of the expression (AD4C is part of the evaluate expression/get next value from the command line/program routine).

If the return type of the result is 0 then a string was found; so, as this routine requires a numerical result, a Type Mismatch error is generated.
If the return type of the result is negative (i.e. &FF) then a floating-point number was found; so exit the routine via the Float Compliment routine [&ACCA].
Otherwise an integer was found (result type is &40). So, exit the routine via the Integer Complement routine [&ACDE].

Disassembly for the compliment result routine

ACD7	L 032 076 173 20 4C AD		JSR &AD4C
ACDA	240 216	F0 D8	BEQ -40 --> &ACB4 [JMP &9092 - Type Mismatch error]
ACDC 0	048 236	30 EC	BMI -20 --> &ACCA Float Compliment
ACDE			...&ACDE Integer Compliment

BASIC IV ROM Routines

Page Last Altered: undefined

Floating-Point Compliment

Submitted by Steve Fewell

Routine: acomp
Name: Floating-Point Compliment
Starting Address: &ACCA
Entry criteria: The FWA contains a floating-point number.
Exit: FWA has been complimented (FWA = zero - FWA)

Description:
If the FWA Mantissa byte 1 (Most significant byte) contains zero - indicating that the current number in the FWA is zero, then exit as no further operation required.

Otherwise, Exclusive-OR the FWA Sign Byte with 10000000 (&80). This reverses the Most significant Bit of the Sign Byte (so 0 becomes 1, and 1 becomes 0). Then exit with A = 255 (&FF) indicating that a Floating-Point value has just been processed.

Disassembly for the Floating-Point Compliment routine

ACCA	1	165 049	A5 31	LDA &31
ACCC		240 006	F0 06	BEQ 6 --> &ACD4
ACCE	.	165 046	A5 2E	LDA &2E
ACD0	I	073 128	49 80	EOR#&80

ACD2	.	133 046	85 2E	STA &2E
ACD4		169 255	A9 FF	LDA#&FF
ACD6	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Integer Compliment Routine

Submitted by Steve Fewell

Routine: icomp
Name: Integer Compliment
Starting Address: &ACDE
Entry criteria: The IWA contains the integer to compliment.
Exit: IWA contains the compliment integer [i.e. IWA = - IWA]

Description:
Subtracts each byte of the IWA from zero. The least significant byte is processed first. The carry flag is set at the beginning because its reset state (0) indicates that a borrow has occurred. This routine makes clever usage of the Y register to hold the value of zero, and to transfer the value to A before each byte is subtracted. This is a lot faster than if A was loaded with #&00 each time.

On exit, A is loaded with #&40, which indicates that an integer value is being processed.

Disassembly for the integer compliment routine

ACDE	8	056	38	SEC
ACDF		169 000	A9 00	LDA#&00
ACE1		168	A8	TAY
ACE2	*	229 042	E5 2A	SBC &2A
ACE4	*	133 042	85 2A	STA &2A
ACE6		152	98	TYA

ACE7	+	229 043	E5 2B	SBC &2B
ACE9	+	133 043	85 2B	STA &2B
ACEB		152	98	TYA
ACEC	,	229 044	E5 2C	SBC &2C
ACEE	,	133 044	85 2C	STA &2C
ACF0		152	98	TYA
ACF1	-	229 045	E5 2D	SBC &2D
ACF3	-	133 045	85 2D	STA &2D
ACF5	@	169 064	A9 40	LDA#&40
ACF7	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9854 Add new variable name to Variable Pointer table

Submitted by Steve Fewell

Routine:addvar
Name: Add new variable name to Variable Pointer table
Starting Address: &9854
Entry criteria: &37 and &38 point to the start of the variable name.
&39 contains the length of the variable name + 1.
Exit: The Variable name has been stored in a newly created variable parameter block.

Description:

Load the first character of the variable name.
Multiply the ASCII character of the variable name by 2 to get the page &4 address of
the pointer table start position for that character (a 2-byte address is allocated for each character
a variable can start with. This address points to the first defined variable beginning with that
character, or the MSB adress byte contains &00, if no variables have been defined beginning with that character.

Store the variable pointer table start address (for the character we want) in &3A-&3B.
If the variable pointer table location contains a non-zero address, then there are already variable(s)
defined beginning with that character, so, store this address in &3A-&3B and check this address.
The first two bytes of each variable block is the address of the next variable (beginning with that character),
so we just need to keep checking this address, then the location pointed to by that address (etc...),
until we find an address with a zero MSB value.

[9868] When we have found zero address, we know that we are at the last variable in the list
of variables beginning with the character we want [note: the variables are stored as a linked list].
Replace the &0000 address with the value of VARTOP (the next free space for variable storage).
VARTOP will be the first location of the variable block for the variable that we will create.
Store 00 in the seond byte of this variable location (as a MSB adress of &00 represents the *new* end of
the variable list).

If there are no more characters in the variable name (only 1 character name) then exit, as we
do not need to store the variable name, as the first character is already known (by the pointer table
location).
Otherwise, store the rest of the variable name (excluding the first character) at the new variable block location.
Exit when the variable name has been copied.

Disassembly for the Add new variable name to Variable Pointer table routine

9854		160 001	A0 01	LDY#&01
9856	7	177 055	B1 37	LDA (&37),Y
9858		010	0A	ASL A
9859		162 004	A2 04	LDX#&04
985B	:	133 058	85 3A	STA &3A
985D	;	134 059	86 3B	STX &3B
985F	:	177 058	B1 3A	LDA (&3A),Y
9861		240 005	F0 05	BEQ 5 --> &9868
9863		170	AA	TAX
9864	:	178 058	B2 3A	LDA (&3A)
9866		128 243	80 F3	BRA -13 --> &985B
9868		165 003	A5 03	LDA &03
986A	:	145 058	91 3A	STA (&3A),Y
986C		165 002	A5 02	LDA &02
986E	:	146 058	92 3A	STA (&3A)
9870		169 000	A9 00	LDA#&00
9872		145 002	91 02	STA (&02),Y
9874		200	C8	INY
9875	9	196 057	C4 39	CPY &39
9877	1	240 049	F0 31	BEQ 49 --> &98AA [RTS]
9879	7	177 055	B1 37	LDA (&37),Y
987B		145 002	91 02	STA (&02),Y
987D		200	C8	INY
987E	9	196 057	C4 39	CPY &39
9880		208 247	D0 F7	BNE -9 --> &9879
9882	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9534 DIM

Submitted by Steve Fewell

Description:

Get the next non-space character pointed to by BASIC Text pointer A.
Subtract 1 from address pointed to by Text pointer A (plus Pointer A offset) and store the new address in location &37-&38.
Store 5 in &3F (variable trype), as the default variable type (unless a % or $ is specified at the end of the variable name) is a 5-byte Floating-Point value.

Backup the current PTR Offset value in X (to preserve the pointer to the start of the variable name).
Call routine &9AF6 to Check the variable name. This routine does the following:
1* Set Y = 1
2* Read next character from address pointed to by &37-&38.
3* If the next character is < "0" then exit.
4* If the next character is >= ":" and less then "@" then exit.
5* If the next character is "0" to "9" and Y = 1 then exit (variable cannot start with a digit).
6* If the next character is >= "[" and < "_" then exit.
7* If the next character is > "z" then exit.
8* Otherwise, the character is valid for the variable name, so increment Y & X (Ptr A offset) and check next character (2*).

If Y is still 1 then no valid characters were found, so a variable name was not specified after the DIM keyword, so issue a Bad DIM error.

if A (the Character after the last valid character of the variable name) is "(" (open bracket) then jump to &9570 to dimension an Array variable.
If A is "$" or "%" then we have either a String or an Integer variable type, as both of these has a parameter block size of 4 bytes, then decrement &3F (size of the variable parameter block) from 5 (default - Float) to 4. Also, check the next charater pointed to by &37-&38. If this character is "(" then jump to &9570 todimension an Array variable.
If the next character is not "(", or the variable was a Floating-Point value (no "%" or "$" after name), then jump to &94BC to dimension a memory area.

&9570 - DIM Array variable
Increment Y to point to after bracket character (this is the variable name length we will pass to &8085).
Store back X to the PTR A offset (pointer to start of variable name).
Call routine &8085 to get the variable return address details.

If A is not 0 then an array with that name already exists, so issue a Bad Dim error as
we cannot have two arrays with the same name. &8085 sets &39 to the length of the variable name.

Call routine &9854 to add the new array name to the variable parameter block.
Set X to 1 (Number of bytes to reserve for the variable parameter block) and call routine &9883 to Allocate
variable parameter block space for the variable.

Store &3F to the stack (Variable type length - 5 for float, 4 for Integer/String).
Store 1 to the stack (this is the next Dimension subscript position in the variable parameter block).
Call &AE18 to set the IWA to 1.

[9589]
Push the current IWA value to the BASIC stack.
Call routine &926F to get the result of the expression pointed to by BASIC text pointer A
and convert the result to an Integer value (or issue Type Mismatch error if String value).
If any of the bytes &2C or &2D, or either of the top 2 bits of Byte &2B, are set
then the value is either negative or too large for an array subscript (>&3FFF (16383 in decimal)).
Increment the IWA value (routine &BEEF), we we need to define 1 more position that the value stated, in order
to make space for element 0 of the array.

Retrieve the "next Dimension subscript position in the variable parameter block" from the stack
and store &2A and &2B (the next dimension subscript value) to VARTOP + next subscript position in
parameter block value.
Increment the next subscript position by 2 (to obtain the next dimension's (if any) position
and store this value back on the stack.

Call routine &9503 to Multiply the specified subscript by any lower subscript values (the IWA value on the
Stack). This routine issues a Bad DIM error if the total number of elements is >FFFF.
Now, the IWA contains the current total number of elements in the array.ÿ
Check the next non-space character at BASIC Text Pointer A (&8CE5) and if the character is
a comma (','), thengo back to &9589, to store the current number of elements (IWA) on the Stack
and process the next dimension subscript in the array.

Otherwise, if the next non-space character is not a comma, then it needs to be a close bracket ')',
if it isn't a close bracket then there is an error with the structure of the DIM statement,
so issue a Bad DIM error.

Retrieve the Variable return type value, and store this value in location &3F.
*[To do this the variable block offset (pointer to next free location in the Array's variable block header)
is temporarily retrieved, and stored back to the stack again, after we have retrieved the variable type.]*
Set location &40 to zero. Now &3F-&40 contain the number of bytes required by the variable
return type [that is 4 for String/Integer and 5 for Floating-Point].
Multiply the number of elements by the number of bytes required by the variable return type.
This will set the IWA to the total number of bytes required for the array variable data.
Retrieve the Variable parameter block offset value (which points to the location after the last Subscript value)
from the stack, and add this to the IWA.
Now the IWA contains the total number of bytes required for the array storage (including the storage
required to hold the maximum subscript values for each dimension).

Copy the current VARTOP address to &37-&38.
Add the total number of bytes required for ther array's storage (&2A-&2B) to the
address pointed to by VARTOP. If this value exceeds &FFFF then issue a DIM space error.
If the new VARTOP value would run into the BASIC Stack then issue
the DIM space error, as there is not enough variable storage space left for the array.

Update VARTOP to point to the next free location after the array storage space.

Retrieve the offset to the next dimension in the variable parameter block (which actually
is a pointer to the first value, as there are no more dimensions to add) in the first
byte of the variable parameter block (pointed to by &37-&38 - the old VARTOP value!).
This byte specifies how many dimensions the array has.

Next, store zero in the array's element value locations to initialise all elements to blank.
*[To get the value of the first element to byte, we set Y to &37 (Old VARTOP LSB value) + offset
to the first value, so that Y contains the LSB pointer to the first value in the array,
and clear &37, so now (&37),Y will return the first byte of the first array value].*
[Keep storing zero until (&37), Y reaches the new value of VARTOP.]

Jump to &94FB to check for a comma at BASIC Text pointer A location, if there is a comma
there we have more arrays to define, so jump back to &9534.
Otherwise, jump to &9000 to continue processing the current BASIC command/program line.

&94BC - DIM Memory area
Decrement &0A (the BASIC Text pointer A), to disregard the last character we read (we needed to check
whether it was a '(' or not in order to know whether to DIM an array or memory).

Call routine &98AE to check for the variable's existence (and create a new variable if it doesn't
already exist).
If the variable name was not valid (zero flag set) then issue a Bad DIM error.
If the variable is a String variable (carry flag set), then issue a Bad DIM error,
as we need a variable that can hold the address value of the dimensioned memory space.

Call &BC43 to Push &2A, &2B and &2C to the stack (&2A-&2B is the
address of the variable's value, and &2C is the variable's type).
Call &96AF to get the result of the expression and convert the result to an Integer
if it was a Floating-Point value, or issue Type Mismatch error is a string value was returned.
Increment the IWA value (&BEEF), as we need to account for the 0th byte (i.e. DIM a% 0 - reserves
1-byte of storage). Now the IWA contains the number of bytes of memory to reserve.

If &2C or &2D contain a value then the amount of memory to reserve is either negative
or >FFFF, so issue a Bad DIM error.

Add the required number of bytes to reserve (IWA) to the current VARTOP address
(but dont update VARTOP value yet).
If the new VARTOP would run into the Stack then there is not enough room for the
required number of bytes of memory to be reserved, so issue a DIM Space error.
[Note: The variable is still defined even if there is not enough room to reserve the required memory!].

Set &2A-&2B to the previous VARTOP address (the start of the reserved memory).
Update VARTOP to point to the new VARTOP value.
Set &27 and A to #&40 (Integer result value), and call &B32B to set the
variable (whose value address and type are stored on the Stack) to the current result value
(in this case the value of the IWA, as the result type (&27) is Integer).
Call &9275 to reset BASIC Text Pointer A offset (&0A) to the BASIC Text Pointer B offset (&1B).

Continue to &94FB to check for a comma at BASIC Text pointer A location, if there is a comma
there we have more arrays to define, so jump back to &9534.
Otherwise, jump to &9000 to continue processing the current BASIC command/program line.

Disassembly for the DIM routine

9534		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space character pointed to by Ptr A
9537		152	98	TYA
9538		024	18	CLC
9539	e	101 011	65 0B	ADC &0B
953B		166 012	A6 0C	LDX &0C
953D		144 002	90 02	BCC 2 --> &9541
953F		232	E8	INX
9540		024	18	CLC
9541		233 000	E9 00	SBC#&00
9543	7	133 055	85 37	STA &37
9545		138	8A	TXA
9546		233 000	E9 00	SBC#&00
9548	8	133 056	85 38	STA &38
954A		162 005	A2 05	LDX#&05
954C	?	134 063	86 3F	STX &3F
954E		166 010	A6 0A	LDX &0A
9550		032 246 154	20 F6 9A	JSR &9AF6 Check variable name
9553		192 001	C0 01	CPY#&01
9555		240 213	F0 D5	BEQ -43 --> &952C Bad DIM error
9557	(	201 040	C9 28	CMP#&28
9559		240 021	F0 15	BEQ 21 --> &9570 DIM Array variable
955B	$	201 036	C9 24	CMP#&24
955D		240 004	F0 04	BEQ 4 --> &9563
955F	%	201 037	C9 25	CMP#&25
9561		208 010	D0 0A	BNE 10 --> &956D
9563	?	198 063	C6 3F	DEC &3F
9565		200	C8	INY
9566		232	E8	INX
9567	7	177 055	B1 37	LDA (&37),Y
9569	(	201 040	C9 28	CMP#&28
956B		240 003	F0 03	BEQ 3 --> &9570 DIM Array variable
956D	L	076 188 148	4C BC 94	JMP &94BC DIM Memory area

DIM Array

9570		200	C8	INY
9571		134 010	86 0A	STX &0A
9573		032 133 128	20 85 80	JSR &8085 Get variable address
9576		208 180	D0 B4	BNE -76 --> &952C Bad DIM error
9578	T	032 084 152	20 54 98	JSR &9854 Add new variable name to Variable Pointer table
957B		162 001	A2 01	LDX#&01
957D		032 131 152	20 83 98	JSR &9883 Allocate space for new variable
9580	?	165 063	A5 3F	LDA &3F
9582	H	072	48	PHA

9583		169 001	A9 01	LDA#&01
9585	H	072	48	PHA
9586		032 024 174	20 18 AE	JSR &AE18 Set IWA to 1-byte (A) [i.e. 1]
9589	&	032 038 188	20 26 BC	JSR &BC26 Push IWA to Stack
958C	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
958F	+	165 043	A5 2B	LDA &2B
9591	)	041 192	29 C0	AND#&C0
9593	,	005 044	05 2C	ORA &2C
9595	-	005 045	05 2D	ORA &2D
9597		208 147	D0 93	BNE -109 --> &952C Bad DIM error
9599		032 239 190	20 EF BE	JSR &BEEF Increment IWA
959C	z	122	7A	PLY
959D	*	165 042	A5 2A	LDA &2A
959F		145 002	91 02	STA (&02),Y
95A1		200	C8	INY
95A2	+	165 043	A5 2B	LDA &2B
95A4		145 002	91 02	STA (&02),Y
95A6		200	C8	INY
95A7	Z	090	5A	PHY
95A8		032 003 149	20 03 95	JSR &9503 Multiply upper dimension value by lower dimenson subscript
95AB		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space PTRA character with ','
95AE		240 217	F0 D9	BEQ -39 --> &9589
95B0	)	201 041	C9 29	CMP#&29
95B2		208 194	D0 C2	BNE -62 --> &9576 [Bad DIM error]
95B4		250	FA	PLX
95B5	h	104	68	PLA
95B6		218	DA	PHX
95B7	?	133 063	85 3F	STA &3F
95B9	d@	100 064	64 40	STZ &40
95BB		032 008 149	20 08 95	JSR &9508 Multiply upper dimension value by the lower dimension subscript
95BE	h	104	68	PLA
95BF	H	072	48	PHA
95C0	e*	101 042	65 2A	ADC &2A
95C2	*	133 042	85 2A	STA &2A
95C4		144 002	90 02	BCC 2 --> &95C8
95C6	+	230 043	E6 2B	INC &2B
95C8		165 003	A5 03	LDA &03
95CA	8	133 056	85 38	STA &38
95CC		165 002	A5 02	LDA &02
95CE	7	133 055	85 37	STA &37
95D0		024	18	CLC
95D1	e*	101 042	65 2A	ADC &2A
95D3		168	A8	TAY

95D4	+	165 043	A5 2B	LDA &2B
95D6	e	101 003	65 03	ADC &03
95D8	+	176 043	B0 2B	BCS 43 --> &9605 DIM Space error
95DA		170	AA	TAX
95DB		196 004	C4 04	CPY &04
95DD		229 005	E5 05	SBC &05
95DF	$	176 036	B0 24	BCS 36 --> &9605 DIM Space error
95E1		132 002	84 02	STY &02
95E3		134 003	86 03	STX &03
95E5	h	104	68	PLA
95E6	7	146 055	92 37	STA (&37)
95E8	e7	101 055	65 37	ADC &37
95EA		168	A8	TAY
95EB		169 000	A9 00	LDA#&00
95ED	d7	100 055	64 37	STZ &37
95EF		144 002	90 02	BCC 2 --> &95F3
95F1	8	230 056	E6 38	INC &38
95F3	7	145 055	91 37	STA (&37),Y
95F5		200	C8	INY
95F6		208 002	D0 02	BNE 2 --> &95FA
95F8	8	230 056	E6 38	INC &38
95FA		196 002	C4 02	CPY &02
95FC		208 245	D0 F5	BNE -11 --> &95F3
95FE	8	228 056	E4 38	CPX &38
9600		208 241	D0 F1	BNE -15 --> &95F3
9602	L	076 251 148	4C FB 94	JMP &94FB Check if there are further Arrays to DIMension

DIM Memory Area

94B9	L	076 005 150	4C 05 96	JMP &9605 DIM Space error
94BC		198 010	C6 0A	DEC &0A
94BE		032 174 152	20 AE 98	JSR &98AE Evaluate variable name & create new variable
94C1	i	240 105	F0 69	BEQ 105 --> &952C Bad DIM error
94C3	g	176 103	B0 67	BCS 103 --> &952C Bad DIM error
94C5	C	032 067 188	20 43 BC	JSR &BC43 Push &2A, &2B & &2C to Stack
94C8		032 175 150	20 AF 96	JSR &96AF Get expression result & convert it to Integer
94CB		032 239 190	20 EF BE	JSR &BEEF Increment IWA
94CE	-	165 045	A5 2D	LDA &2D
94D0	,	005 044	05 2C	ORA &2C
94D2	X	208 088	D0 58	BNE 88 --> &952C Bad DIM error
94D4		024	18	CLC
94D5	*	165 042	A5 2A	LDA &2A
94D7	e	101 002	65 02	ADC &02
94D9		168	A8	TAY

94DA	+	165 043	A5 2B	LDA &2B
94DC	e	101 003	65 03	ADC &03
94DE		170	AA	TAX
94DF		196 004	C4 04	CPY &04
94E1		229 005	E5 05	SBC &05
94E3		176 212	B0 D4	BCS -44 --> &94B9 [DIM Space error]
94E5		165 002	A5 02	LDA &02
94E7	*	133 042	85 2A	STA &2A
94E9		165 003	A5 03	LDA &03
94EB	+	133 043	85 2B	STA &2B
94ED		132 002	84 02	STY &02
94EF		134 003	86 03	STX &03
94F1	@	169 064	A9 40	LDA#&40
94F3	'	133 039	85 27	STA &27
94F5	+	032 043 179	20 2B B3	JSR &B32B Set Numeric variable
94F8	u	032 117 146	20 75 92	JSR &9275 Ptr A offset = Ptr B offset
94FB		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space PTRA character with ','
94FE	4	240 052	F0 34	BEQ 52 --> &9534 DIMension next array/variable
9500	L	076 000 144	4C 00 90	JMP &9000 Check for end of statement & process next BASIC program Statement

Check variable name

9AF6		160 001	A0 01	LDY#&01
9AF8	7	177 055	B1 37	LDA (&37),Y
9AFA	0	201 048	C9 30	CMP#&30
9AFC		144 024	90 18	BCC 24 --> &9B16
9AFE	@	201 064	C9 40	CMP#&40
9B00		176 012	B0 0C	BCS 12 --> &9B0E
9B02	:	201 058	C9 3A	CMP#&3A
9B04		176 016	B0 10	BCS 16 --> &9B16
9B06		192 001	C0 01	CPY#&01
9B08		240 012	F0 0C	BEQ 12 --> &9B16
9B0A		232	E8	INX
9B0B		200	C8	INY
9B0C		208 234	D0 EA	BNE -22 --> &9AF8
9B0E	_	201 095	C9 5F	CMP#&5F
9B10		176 005	B0 05	BCS 5 --> &9B17
9B12	[	201 091	C9 5B	CMP#&5B
9B14		144 244	90 F4	BCC -12 --> &9B0A
9B16	`	096	60	RTS
9B17	{	201 123	C9 7B	CMP#&7B
9B19		144 239	90 EF	BCC -17 --> &9B0A
9B1B	`	096	60	RTS

Store &2A-&2C on Stack

BC43	z	122	7A	PLY
BC44		250	FA	PLX
BC45	*	165 042	A5 2A	LDA &2A
BC47	H	072	48	PHA
BC48	+	165 043	A5 2B	LDA &2B
BC4A	H	072	48	PHA
BC4B	,	165 044	A5 2C	LDA &2C
BC4D	H	072	48	PHA
BC4E		218	DA	PHX
BC4F	Z	090	5A	PHY
BC50	`	096	60	RTS

Increment IWA value

BEEF	*	230 042	E6 2A	INC &2A
BEF1		208 010	D0 0A	BNE 10 --> &BEFD
BEF3	+	230 043	E6 2B	INC &2B
BEF5		208 006	D0 06	BNE 6 --> &BEFD
BEF7	,	230 044	E6 2C	INC &2C
BEF9		208 002	D0 02	BNE 2 --> &BEFD
BEFB	-	230 045	E6 2D	INC &2D
BEFD	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Next non-space Char PTRA

Submitted by Steve Fewell

Routine: NextCharPTRA
Name: Next non-space Char PTRA
Starting Address: &8ED5
Exit: Move past spaces in BASIC's Text Pointer A and return the next Character in the string (A).

Description:
Reads past any space characters found at the current position in the PTRA. Returns with the next non-space Character from PTRA in A, and with the PTRA Offset pointing to this value.

Routine &8CD7 returns the next non-space character in PTRA, and also sets the zero flag is the character is 'X' or 'x', or clears the zero flag is the character is not 'X' or 'x'. Bit 5 of the ASCII value is cleared to return any lower case ASCII characters as upper case.

Routine &8CDF returns the next non-space character in PTRA, and also sets the zero flag is the character is a hash ('#'), or clears the zero flag is the character is not a hash.

Routine &8CE5 returns the next non-space character in PTRA, and also sets the zero flag is the character is a comma (','), or clears the zero flag is the character is not a comma.

Disassembly for the Next non-space Char PTRA routine

8EE0		164 010	A4 0A	LDY &0A
8EE2		230 010	E6 0A	INC &0A
8EE4		177 011	B1 0B	LDA (&0B),Y
8EE6		201 032	C9 20	CMP#&20
8EE8		240 246	F0 F6	BEQ -10 --> &8EE0
8EEA	`	096	60	RTS

Get Next non-space Char PTRA and compare with 'X' or 'x'

8CD7		032 224 142	20 E0 8E	JSR &8EE0
8CDA	)	041 223	29 DF	AND#&DF
8CDC	X	201 088	C9 58	CMP#&58
8CDE	`	096	60	RTS

Get Next non-space Char PTRA and compare with '#'

8CDF		032 224 142	20 E0 8E	JSR &8EE0
8CE2	#	201 035	C9 23	CMP#&23
8CE4	`	096	60	RTS

Get Next non-space Char PTRA and compare with ','

8CE5		032 224 142	20 E0 8E	JSR &8EE0
8CE8	,	201 044	C9 2C	CMP#&2C
8CEA	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

IWA = 8-bit or 16-bit Integer

Submitted by Steve Fewell

Routine: i16-bit
Name: IWA = 8-bit or 16-bit Integer
Starting Address: &AE1A for 16-bit value [or AE18 for 8-bit]
Entry criteria: A contains the least significant byte of the 16-bit number, Y contains the most significant byte of the 16-bit number.
[Y is always set to 0 when called from AE18, so only need to supply A].
Exit: IWA contains the 8-bit or 16-bit number [(256 * Y) + A].

Description:
Stores an 8-bit [if called from AE18] or 16-bit [if called from AE1A] number (The 16-bit value is represented by 2 individual bytes) into the IWA and sets the 2 most significant bytes (or in the case or an 8-bit value, the 3 most significant bytes) of the IWA to zero.

Disassembly for the IWA = 8-bit or 16-bit Integer routine

AE18		160 000	A0 00	LDY#&00
AE1A	*	133 042	85 2A	STA &2A
AE1C	+	132 043	84 2B	STY &2B
AE1E	d,	100 044	64 2C	STZ &2C
AE20	d-	100 045	64 2D	STZ &2D
AE22	@	169 064	A9 40	LDA#&40
AE24	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Push Integer to BASIC Stack

Submitted by Steve Fewell

Routine: pushi
Name: Push Integer to BASIC Stack
Starting Address: &BC26
Entry criteria: The IWA contains the 32-bit Integer to store on the BASIC Stack.
Exit: A copy of the IWA has been pushed to the Stack or a "No Room" error occurs if insufficient memory.

Description:
[If called from BC22 then the type of value will be tested and if the value is a String or Float then the appropriate Push routine will be called; otherwise push Integer (as below)].

Subtract the required number of bytes (that is 4, as we are about to push an Integer number) from the Stack Pointer Low Byte. Leaving the result in A, call the routine to Check for Stack clash with Heap (which will decrement the high byte of the Stack Pointer, if a page boundary is crossed). Store the IWA on the Stack.

Disassembly for the Push Integer to BASIC Stack routine

BC22	-	240 045	F0 2D	BEQ 45 --> &BC51 Push String to Stack
BC24	0	048 212	30 D4	BMI -44 --> &BBFA Push FWA to Stack
BC26		165 004	A5 04	LDA &04
BC28	8	056	38	SEC
BC29		233 004	E9 04	SBC#&04
BC2B		032 030 189	20 1E BD	JSR &BD1E Check Stack clash with Heap
BC2E		160 003	A0 03	LDY#&03
BC30	-	165 045	A5 2D	LDA &2D
BC32		145 004	91 04	STA (&04),Y
BC34		136	88	DEY
BC35	,	165 044	A5 2C	LDA &2C
BC37		145 004	91 04	STA (&04),Y
BC39		136	88	DEY
BC3A	+	165 043	A5 2B	LDA &2B
BC3C		145 004	91 04	STA (&04),Y
BC3E	*	165 042	A5 2A	LDA &2A
BC40		146 004	92 04	STA (&04)

BC42 ` 096 60 RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Check for Stack clash with Heap

Submitted by Steve Fewell

Routine: checkstackclash
Name: Check for Stack clash with Heap
Starting Address: &BD1E
Entry criteria: A = The BASIC Stack Pointer Low Byte, after the required number of bytes of storage space has been subtracted. C (Carry flag) = 0 if overflow occurred during the subtraction of the required number of bytes.

Description:
Store A back to the Stack Pointer Low Byte. Decrement &05 (The Stack Pointer High Byte), if necessary.

If the Stack Pointer High Byte is now less than the Heap high byte then there is no room in memory, so error (as the Stack is stored above the Heap).

If the Stack Pointer High Byte is equal to the Heap high Byte, then test the low bytes. If the Stack Pointer Low Byte is less than the Heap Low Byte then a "No Room" error is produced, otherwise the routine returns successfully.

Disassembly for the Check for Stack clash with Heap routine

BD1E	133 004	85 04	STA &04
BD20	176 002	B0 02	BCS 2 --> &BD24

BD22		198 005	C6 05	DEC &05
BD24		164 005	A4 05	LDY &05
BD26		196 003	C4 03	CPY &03
BD28		144 010	90 0A	BCC 10 --> &BD34
BD2A		208 004	D0 04	BNE 4 --> &BD30
BD2C		197 002	C5 02	CMP &02
BD2E		144 004	90 04	BCC 4 --> &BD34
BD30	`	096	60	RTS
...		...		...
BD34	L	076 161 144	4C A1 90	JMP &90A1 Error: No Room

BASIC IV ROM Routines

Page Last Altered: undefined

BC51 Push String to Stack

Submitted by Steve Fewell

Description:

Check whether there is space for the length of the String in the String Buffer (SWA) plus 1 byte (for length) on the stack. If no space, then generate "No Room" error.

Y = length of the String Buffer (?&36, Length of SWA). If length is zero then store
binary zero on the Stack (length of String) and return, as the first byte retrieved from
the Stack will be the String length, 0 in this case.
Otherwise, store the String on the Stack (in reverse order).
Lastly, the length of the string is stored on the stack (?&36) [last byte of Stack Value] and routine exits.

Disassembly for the Push String to Stack routine

BC51		024	18	CLC
BC52		165 004	A5 04	LDA &04
BC54	6	229 054	E5 36	SBC &36
BC56		032 030 189	20 1E BD	JSR &BD1E Check for Stack clash with Heap
BC59	6	164 054	A4 36	LDY &36
BC5B		240 008	F0 08	BEQ 8 --> &BC65
BC5D		185 255 005	B9 FF 05	LDA &05FF,Y
BC60		145 004	91 04	STA (&04),Y
BC62		136	88	DEY
BC63		208 248	D0 F8	BNE -8 --> &BC5D
BC65	6	165 054	A5 36	LDA &36

```
BC67    146 004     92 04       STA (&04)
BC69    `   096     60          RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

BBFA Push FWA to Stack

Submitted by Steve Fewell

Routine:pushFWA
Name: Push FWA to Stack
Starting Address: &BBFA
Entry criteria: The FWA contains a floating-point value.

Description:

Check whether the Stack has room for 5 bytes. If no space, then generate 'No Room' error; otherwise, update the Stack Pointer to point to the next free location on the Stack.

Store the FWA Exponent byte on the Stack (1st Byte).
Load FWA Sign, EOR with FWA Mantissa byte 1, AND with &80 (to restore the top bit) and EOR with the FWA Mantissa byte 1 again to set the Mantissa byte 1 value. Store result on stack (2nd byte). This is done to pack the FWA Mantissa, as the FWA is pushed to the Stack in its 5-byte packed variable format to save on Stack space.
Store the FWA Mantissa Byte 2 on the stack (3rd byte of Stack Value).
Store the FWA Mantissa Byte 3 on the stack (4th byte of Stack Value).
Store the FWA Mantissa Byte 4 on the stack (5th byte of Stack Value).
all done, so return.

Disassembly for the Push FWA to Stack routine

BBFA		165 004	A5 04	LDA &04
BBFC	8	056	38	SEC
BBFD		233 005	E9 05	SBC#&05
BBFF		032 030 189	20 1E BD	JSR &BD1E Check for Stack Clash with Heap

BC02	0	165 048	A5 30	LDA &30
BC04		146 004	92 04	STA (&04)
BC06		160 001	A0 01	LDY#&01
BC08	.	165 046	A5 2E	LDA &2E
BC0A	E1	069 049	45 31	EOR &31
BC0C	)	041 128	29 80	AND#&80
BC0E	E1	069 049	45 31	EOR &31
BC10		145 004	91 04	STA (&04),Y
BC12		200	C8	INY
BC13	2	165 050	A5 32	LDA &32
BC15		145 004	91 04	STA (&04),Y
BC17		200	C8	INY
BC18	3	165 051	A5 33	LDA &33
BC1A		145 004	91 04	STA (&04),Y
BC1C		200	C8	INY
BC1D	4	165 052	A5 34	LDA &34
BC1F		145 004	91 04	STA (&04),Y
BC21	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

926F Get result of expression from BASIC Text Pointer A & convert to Integer

Submitted by Steve Fewell

Description:

If called from &926D then decrement the BASIC Text Pointer A offset.
Set BASIC Text Pointer B = BASIC Text Pointer A.
Get result of expression.
Convert expression result to Integer - or Type Mismatch error (if String value).
Set BASIC Text Pointer A offset (&0A) = BASIC Text Pointer B offset (&1B) to
set Text Pointer A to point to after the expression.

Disassembly for the Get result of expression from BASIC Text Pointer A routine & convert to Integer

926D	198 010	C6 0A	DEC &0A
926F	/ 032 047 157	20 2F 9D	JSR &9D2F Ptr B = Ptr A & Get result of expression
9272	032 191 150	20 BF 96	JSR &96BF Check value & convert to Integer (if Float), error if String
9275	164 027	A4 1B	LDY &1B
9277	132 010	84 0A	STY &0A
9279	` 096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9D3B Expression Handler

Submitted by Steve Fewell

Description:

If called from &9D2F then set BASIC Text Pointer B = BASIC Text Pointer A.

The lowest level of the expression handler starts at address &9D3B.
There are many levels to the expression handler, each level calls the level above
(to evaluate any higher-level operators) before checking whether any of its operators
need to be evaluated.

The Expression handler is structured as follows:
JSR &9D3B [Entry point (lowest level)] Handle (**OR**, **EOR**)
JSR &9D81 handle (**AND**)
JSR &9DA9 handle (<, =, >)
JSR &9E4C handle (+, -)
JSR &9FC7 handle (*****, /, **MOD**, **DIV**)
JSR &A012 handle (Skip spaces and ^)
JSR &AD36 handle (**leading -**, **Variable name/value**, **Open Bracket**, **BASIC Keyword (functions)**)

Some routines have additional entry points which enable a previous value to be stored
on the stack and/or a new value obtained before evaluating the expression.
This is so that any previous value is not lost during the expression handling,
in situations where more than 1 value may be required to be worked on at the same time.

After evaluating the operator, the routine usually jumps back into the expression handler
at the same level as the previous operator - this checks for further operators with the same
priority (precedence) before carrying on with the lower levels. I.e. an expression can contain
many multiplications, one after the other, all which much be evaluated before the lower level is
returned to (i.e. 2 + X * Y * Z). The exception for this, is the relational operators (<, =, >);

these cannot be located one after the other, as this is not allowed in BASIC.

If there are no more operators/values to evaluate (&9D46), then the PTR B offset is decremented, Y = Value type, and ?&27 is also set to the value type, and then exit.

Note: On exit X contains the next non-space character found after the expression (pointed to by &1B). Many of the routines that call this expression handler use this as a quick way of knowing what the next character is.

Disassembly for the Expression Handler routine

9D3B Expression Handler Entry/Level 1 [OR, EOR]

9D2F		165 011	A5 0B	LDA &0B
9D31		133 025	85 19	STA &19
9D33		165 012	A5 0C	LDA &0C
9D35		133 026	85 1A	STA &1A
9D37		165 010	A5 0A	LDA &0A
9D39		133 027	85 1B	STA &1B
9D3B		032 129 157	20 81 9D	JSR &9D81 Expression Handler level 2 (AND)
9D3E		224 132	E0 84	CPX#&84 Token value for OR
9D40		240 010	F0 0A	BEQ 10 --> &9D4C OR operator
9D42		224 130	E0 82	CPX#&82 Token value for EOR
9D44		240 032	F0 20	BEQ 32 --> &9D66 EOR operator
9D46		198 027	C6 1B	DEC &1B
9D48		168	A8	TAY
9D49	'	133 039	85 27	STA &27
9D4B	`	096	60	RTS

9D81 Expression Handler Level 2 [AND]

9D7B		032 190 150	20 BE 96	JSR &96BE Check for Integer & convert to Integer if Float
9D7E	&	032 038 188	20 26 BC	JSR &BC26 Push Integer to Stack
9D81		032 169 157	20 A9 9D	JSR &9DA9 Expression Handler level 3 (<, =, >)
9D84		224 128	E0 80	CPX#&80 Token value for AND
9D86		240 001	F0 01	BEQ 1 --> &9D89 AND operator
9D88	`	096	60	RTS

9DA9 Expression Handler Level 3 [<, =, >]

9DA9	L	032 076 158	20 4C 9E	JSR &9E4C Expression Handler level 4 (+, -)
9DAC	?	224 063	E0 3F	CPX#&3F '?'
9DAE		176 004	B0 04	BCS 4 --> &9DB4
9DB0	<	224 060	E0 3C	CPX#&3C
9DB2		176 001	B0 01	BCS 1 --> &9DB5 '=' and other Relational operators
9DB4	`	096	60	RTS

9E4C Expression Handler Level 4 [+, -]

9E4C		032 199 159	20 C7 9F	JSR &9FC7 Expression Handler level 5 (*, /, MOD, DIV)
9E4F	+	224 043	E0 2B	CPX#&2B '+'
9E51		240 005	F0 05	BEQ 5 --> &9E58 '+' Operator - Addition
9E53	-	224 045	E0 2D	CPX#&2D '-'
9E55	f	240 102	F0 66	BEQ 102 --> &9EBD '-' Operator - Subtraction
9E57	`	096	60	RTS

9FC7 Expression Handler Level 5 [*, /, MOD, DIV]

9FC1	L;	076 059 159	4C 3B 9F	JMP &9F3B
9FC4	&	032 038 188	20 26 BC	JSR &BC26 Push Integer to Stack
9FC7		032 018 160	20 12 A0	JSR &A012 JSR &A012 Expression Handler level 6 (Skip Spaces, ^)
9FCA	*	224 042	E0 2A	CPX#&2A '*'
9FCC		240 243	F0 F3	BEQ -13 --> &9FC1
9FCE	/	224 047	E0 2F	CPX#&2F '/'
9FD0		240 009	F0 09	BEQ 9 --> &9FDB '/' Operator - Division
9FD2		224 131	E0 83	CPX#&83 Token value for MOD
9FD4		240 031	F0 1F	BEQ 31 --> &9FF5 Integer MOD routine
9FD6		224 129	E0 81	CPX#&81 Token value for DIV
9FD8	#	240 035	F0 23	BEQ 35 --> &9FFD Integer DIV routine
9FDA	`	096	60	RTS

A012 Expression Handler Level 6 [Skip spaces, ^]

A00F	&	032 038 188	20 26 BC	JSR &BC26 Push Integer to Stack

A012	6	032 054 173	20 36 AD	JSR &AD36 Evaluate Variable / Value / BASIC Keyword (Function)/ Open bracket
A015	H	072	48	PHA
A016		164 027	A4 1B	LDY &1B
A018		230 027	E6 1B	INC &1B
A01A		177 025	B1 19	LDA (&19),Y
A01C		201 032	C9 20	CMP#&20 <space>
A01E		240 246	F0 F6	BEQ -10 --> &A016
A020		170	AA	TAX
A021	h	104	68	PLA
A022	^	224 094	E0 5E	CPX#&5E '^'
A024		240 001	F0 01	BEQ 1 --> &A027 '^' operator
A026	`	096	60	RTS

The disassembly for the "AD36 Expression Handler Level 7 [Unary -, Variable/Value, Open bracket, BASIC Keyword (function)]" routine is in the separate description for "AD36 Evaluate Variable/Value"

BASIC IV ROM Routines

Page Last Altered: undefined

9D4C 'OR' operator

Submitted by Steve Fewell

Description:

Convert the first value (current value) to an integer (if it's a Float).
If the first value is a string then issue a ype Mismatch error.
Push the first value to the stack.
[9D7B] Get the next value (expression handler level 2 [AND]). For the next value we only need to search the operators with higher precedence, as these must be evaluated before we have the correct second value that we need. If the value obtained is a float then it is converted to an integer (if it's a string then Type Mismatch error).

OR the Integer on the stack (first value) with the Integer in the IWA (second value) storing the result in the IWA.
We now have the result we require.

Reclaim the stack space, set A = #&40 (as we are handling an Integer) and jump to &9D3E to check for further OR/EOR operators which need to be processed.

Disassembly for the 'OR' Operator routine

9D4C	{	032 123 157	20 7B 9D	JSR &9D7B Convert to Int, push to Stack & Get next value
9D4F		032 190 150	20 BE 96	JSR &96BE Check for Integer & convert if Float
9D52		160 003	A0 03	LDY#&03
9D54		177 004	B1 04	LDA (&04),Y
9D56	*	025 042 000	19 2A 00	ORA &002A,Y
9D59	*	153 042 000	99 2A 00	STA &002A,Y
9D5C		136	88	DEY
9D5D		016 245	10 F5	BPL -11 --> &9D54
9D5F		032 250 188	20 FA BC	JSR &BCFA Move Stack Pointer up 4 bytes (reclaim space)
9D62	@	169 064	A9 40	LDA#&40
9D64		128 216	80 D8	BRA -40 --> &9D3E Expression Handler level 1 [OR, EOR]

BASIC IV ROM Routines

Page Last Altered: undefined

Pop Integer from the BASIC Stack

Submitted by Steve Fewell

Routine: popi
Name: Pop Integer from the BASIC Stack
Starting Address: &BCE6
Entry criteria: None
Exit: IWA contains the value of the most recent Integer number on the BASIC Stack. The Basic Stack Pointer is moved up.

Description:
Loads the IWA with the 32-bit Integer pointed to by the BASIC Stack Pointer. Then move the Stack Pointer up 4 bytes so that is points to the previous item pushed onto the stack.

Disassembly for the Pop Integer from the BASIC Stack routine

BCE6	160 003	A0 03	LDY#&03
BCE8	177 004	B1 04	LDA (&04),Y
BCEA -	133 045	85 2D	STA &2D
BCEC	136	88	DEY
BCED	177 004	B1 04	LDA (&04),Y
BCEF ,	133 044	85 2C	STA &2C
BCF1	136	88	DEY
BCF2	177 004	B1 04	LDA (&04),Y

```
BCF4  +   133 043    85 2B     STA &2B
BCF6      178 004    B2 04     LDA (&04)
BCF8  *   133 042    85 2A     STA &2A
BCFA      024        18        CLC
BCFB      169 004    A9 04     LDA#&04
BCFD  e   101 004    65 04     ADC &04
BCFF      133 004    85 04     STA &04
BD01      144 002    90 02     BCC 2 --> &BD05
BD03      230 005    E6 05     INC &05
BD05  `   096        60        RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

9D66 'EOR' operator

Submitted by Steve Fewell

Description:

[9D7B] Convert the first value (current value) to an integer (if it's a Float).
If the first value is a string then issue a ype Mismatch error.
Push the first value to the stack.
[9D81] Get the next value (expression handler level 2 [AND]). For the next value
we only need to search the operators with higher precedence, as these must be evaluated
before we have the correct second value that we need. If the value obtained is a float
then convert it to an integer (if it's a string then Type Mismatch error).

EOR the Integer on the stack (first value) with the Integer in the IWA (second value)
storing the result in the IWA.
We now have the result we require.

Reclaim the stack space, set A = #&40 (as we are handling an Integer) and jump to
&9D3E to check for further OR/EOR operators which need to be processed.

Disassembly for the 'EOR' Operator routine

9D66	{	032 123 157	20 7B 9D	JSR &9D7B Convert to Int, push to Stack & Get next value
9D69		032 190 150	20 BE 96	JSR &96BE Check for Integer & convert if Float
9D6C		160 003	A0 03	LDY#&03
9D6E		177 004	B1 04	LDA (&04),Y
9D70	Y*	089 042 000	59 2A 00	EOR &002A,Y
9D73	*	153 042 000	99 2A 00	STA &002A,Y
9D76		136	88	DEY
9D77		016 245	10 F5	BPL -11 --> &9D6E
9D79		128 228	80 E4	BRA -28 --> &9D5F Reclaim stack space & Expression handler level 1

BASIC IV ROM Routines

Page Last Altered: undefined

9D89 'AND' operator

Submitted by Steve Fewell

Description:

[96BE] Convert the first value (current value) to an integer (if it's a Float).
If the first value is a string then issue a ype Mismatch error.
Push the first value to the stack.
[9DA9] Get the next value (expression handler level 3 [<, =, >). For the next value we only need to search the operators with higher precedence, as these must be evaluated before we have the correct second value that we need. If the value obtained is a float then convert it to an integer (if it's a string then Type Mismatch error).

AND the Integer on the stack (first value) with the Integer in the IWA (second value) storing the result in the IWA.
We now have the result we require.

Reclaim the stack space, set A = #&40 (as we are handling an Integer) and jump to &9D84 to check for further AND operators which need to be processed. Note: We only need to check for the AND operator, as all higher precedence operators have already been processed.

Disassembly for the 'AND' Operator routine

9D89		032 190 150	20 BE 96	JSR &96BE Check for Integer & convert if Float
9D8C	&	032 038 188	20 26 BC	JSR &BC26 Push Integer to Stack
9D8F		032 169 157	20 A9 9D	JSR &9DA9 Expression handler level 3 (<, =, >)
9D92		032 190 150	20 BE 96	JSR &96BE Check for Integer & convert if Float
9D95		160 003	A0 03	LDY#&03
9D97		177 004	B1 04	LDA (&04),Y
9D99	9*	057 042 000	39 2A 00	AND &002A,Y
9D9C	*	153 042 000	99 2A 00	STA &002A,Y
9D9F		136	88	DEY
9DA0		016 245	10 F5	BPL -11 --> &9D97
9DA2		032 250 188	20 FA BC	JSR &BCFA Move Stack Pointer up 4 bytes (reclaim space)
9DA5	@	169 064	A9 40	LDA#&40

9DA7	128 219	80 DB	BRA -37 --> &9D84 Expression Handler level 2 [AND]

BASIC IV ROM Routines

Page Last Altered: undefined

9DB5 '=' and other relational operators

Submitted by Steve Fewell

Description:

Check which operator (<, = or >) we have. If we have the Less Than operator "<",
then jump to &9DCD to handle Less Than.
If we have the Greater Than operator ">", then jump to &9DF5 to handle Greater Than.
Otherwise, we have '=' Equal To. This is handled as below:

Update the flags with the current value type (in A).
Call &9CCA to Compare the current value with the second value and set the flags
according to which value is greater, or whether the values are equal.

If the zero flag is set then the values are equal, so set the IWA to TRUE; otherwise,
the values aren't equal so set the IWA to FALSE.
Exit with A = #&40 (as we are currently handling an Integer value).

Disassembly for the '=' and other relational operators routine

9DB5		240 022	F0 16	BEQ 22 --> &9DCD '<' Less Than Operator
9DB7	>	224 062	E0 3E	CPX#&3E
9DB9	:	240 058	F0 3A	BEQ 58 --> &9DF5 '>' Greater Than Operator
9DBB		170	AA	TAX
9DBC		032 202 156	20 CA 9C	JSR &9CCA Compare Values
9DBF		208 001	D0 01	BNE 1 --> &9DC2
9DC1		136	88	DEY
9DC2	*	132 042	84 2A	STY &2A
9DC4	+	132 043	84 2B	STY &2B
9DC6	,	132 044	84 2C	STY &2C
9DC8	-	132 045	84 2D	STY &2D
9DCA	@	169 064	A9 40	LDA#&40
9DCC	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9DCD '<' operator

Submitted by Steve Fewell

Description:

Check the next character from the BASIC Input line. If the next character is '=' then
the operator is '<=' Less Than or Equal To, so jump to &9DE1 to handle this operator.
If the next character is '>' then the operator is '<>' Not Equal To, so jump to &9DEC
to handle this operator.
Otherwise the operator is '<' Less Than. which is handled as follows:

Call &9CC9 to Compare the current value with the second value and set the flags
according to which value is greater, or whether the values are equal.

If the carry flag is clear then the first value is less than the second, so set the
IWA to TRUE; otherwise, the first value is not less than the second, so set the IWA to FALSE.
Exit with A = #&40 (as we are currently handling an Integer value).

Disassembly for the '<' Operator routine

9DCD		170	AA	TAX
9DCE		164 027	A4 1B	LDY &1B
9DD0		177 025	B1 19	LDA (&19),Y
9DD2	=	201 061	C9 3D	CMP#&3D '='
9DD4		240 011	F0 0B	BEQ 11 --> &9DE1 '<=' Less Than or Equal Operator
9DD6	>	201 062	C9 3E	CMP#&3E '>'
9DD8		240 018	F0 12	BEQ 18 --> &9DEC '<>' Not Equal Operator
9DDA		032 201 156	20 C9 9C	JSR &9CC9 Compare Values
9DDD		144 226	90 E2	BCC -30 --> &9DC1 Set TRUE
9DDF		128 225	80 E1	BRA -31 --> &9DC2 Set FALSE

BASIC IV ROM Routines

Page Last Altered: undefined

9DE1 '<=' operator

Submitted by Steve Fewell

Description:

Increment the Text Pointer B offset (so that the next character is the character after the '=').

Call &9CC9 to Compare the current value with the second value and set the flags according to which value is greater, or whether the values are equal.

If the zero flag is set then the values are equal, so set the IWA to TRUE.
Otherwise, if the carry flag is clear then the first value is less than the second, so set the IWA to TRUE; otherwise, the first value is not less than or equal to the second, so set the IWA to FALSE.
Exit with A = #&40 (as we are currently handling an Integer value).

Disassembly for the '<=' Operator routine

9DE1	230 027	E6 1B	INC &1B
9DE3	032 201 156	20 C9 9C	JSR &9CC9 Compare Values
9DE6	240 217	F0 D9	BEQ -39 --> &9DC1 Set TRUE
9DE8	144 215	90 D7	BCC -41 --> &9DC1 Set TRUE
9DEA	128 214	80 D6	BRA -42 --> &9DC2 Set FALSE

BASIC IV ROM Routines

Page Last Altered: undefined

9cc9 Compare Integer values

Submitted by Steve Fewell

Starting Address: &9CC9
Entry criteria: if called from &9CCA, the flags indicate the value type we are working with (Int,Float,String); otherwise, (called from &9CC9) X contains this information.
If we are working with an integer value: the IWA contains the value to compare.
BASIC Text pointer points to the expression to compare IWA with.
Exit: Zero Flag = 1 if values are equal. Carry is set if value 1 > value 2; clear otherwise.
Y = 00.

Description:

Test the value type (X). If the value is a string, then do Compare String routine,
If the value is a Float, then do compare Float routine, otherwise continue with this routine as the 1st value to compare is an Integer.

Push the IWA to the Stack (first value to compare).
Gosub &9E4C to get the result of the expression (in the BASIC program text line),
[we need to process expression level (+,-) upwards as any operators below this level should only be evaluated once this compare has been evaluated].
If the second value is a string then generate a Type Mismatch error.
If the second value is a float then goto &9C65, to convert the first value to a Float and continue with the Compare Float values routine (as the values are now Float not Integer).

Reverse the sign of the IWA (second value) and the Integer on the Stack (first value).
This causes positive numbers to correctly be identified as greater than negative values.
Pop the Integer from the stack (first value) and compare each byte with the corresponding byte of the IWA (second value) LSB first.
C = 1 if the first value is > the second value.
&2A contains a value if the two values are not equal, this is returned in A.
Exit with A and C set as appropriate.

Disassembly for the Compare Integer values routine

9CC6	L	076 146 144	4C 92 90	JMP &9092 Type Mismatch error

9CC9		138	8A	TXA
9CCA	6	240 054	F0 36	BEQ 54 --> &9D02 Compare String values
9CCC	0	048 180	30 B4	BMI -76 --> &9C82 Compare Float values
9CCE	-	165 045	A5 2D	LDA &2D
9CD0	H	072	48	PHA
9CD1	,	165 044	A5 2C	LDA &2C
9CD3	H	072	48	PHA
9CD4	+	165 043	A5 2B	LDA &2B
9CD6	H	072	48	PHA
9CD7	*	165 042	A5 2A	LDA &2A
9CD9	H	072	48	PHA
9CDA	L	032 076 158	20 4C 9E	JSR &9E4C Expression handler [(+,-) level and above]
9CDD		168	A8	TAY
9CDE		240 230	F0 E6	BEQ -26 --> &9CC6 Generate Type mismatch error
9CE0	0	048 131	30 83	BMI -125 --> &9C65 Convert 1st Integer to Float and compare Float values
9CE2	-	165 045	A5 2D	LDA &2D
9CE4	I	073 128	49 80	EOR#&80
9CE6	-	133 045	85 2D	STA &2D
9CE8	8	056	38	SEC
9CE9	h	104	68	PLA
9CEA	*	229 042	E5 2A	SBC &2A
9CEC	*	133 042	85 2A	STA &2A
9CEE	h	104	68	PLA
9CEF	+	229 043	E5 2B	SBC &2B
9CF1	*	004 042	04 2A	TSB &2A
9CF3	h	104	68	PLA
9CF4	,	229 044	E5 2C	SBC &2C
9CF6	*	004 042	04 2A	TSB &2A
9CF8	h	104	68	PLA
9CF9		160 000	A0 00	LDY#&00
9CFB	I	073 128	49 80	EOR#&80
9CFD	-	229 045	E5 2D	SBC &2D
9CFF	*	005 042	05 2A	ORA &2A
9D01	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9D02 Compare String values

Submitted by Steve Fewell

Starting Address: &9D02
Entry criteria: SWA contains the value to compare.
BASIC Text pointer points to the expression to compare SWA with.
Exit: Zero Flag = 1 if values are equal. Carry is set if value 1 > value 2; clear otherwise.
Y = 00.

Description:

Push the SWA to the Stack (first value to compare).
Gosub &9E4C to get the result of the expression (in the BASIC program text line),
[we need to process expression level (+,-) upwards as any operators below this level should
only be evaluated once this compare has been evaluated].
If the second value is not a string, then issue a Type Mismatch error.

Set &37 to the number of bytes to compare (the length of the shortest string).
If the number of bytes to compare is 0 then stop comparing [&9D23].
Keep comparing each character in both strings until we either encounter a difference
between the strings, in which case we have our result (C = 1 if first value > second value) [goto &9D27],
or until we have compared the number of bytes that we needed to compare (&37) [goto &9D23].

[9D23:]
Now we have compared the number of bytes we needed to compare, and the values we have
compared so far are equal.
Compare the length of the first value with the length of the second value.
Whichever string is the longer will be the greater. The Zero flag will be set if
the lengths are equal, and thus the strings are equal. (C = 1 if the first value length
is greater than the second value length).

[9D27:]

We have our result, so store result (PHP), restore stack space used by the string, then restore the result value (PLP) and exit.

Disassembly for the Compare String values routine

9D02	Q	032 081 188	20 51 BC	JSR &BC51 Push String to Stack
9D05	L	032 076 158	20 4C 9E	JSR &9E4C Expression Handler [level (+, -) and above]
9D08		168	A8	TAY
9D09		208 187	D0 BB	BNE -69 --> &9CC6 [JMP &9092]
9D0B		178 004	B2 04	LDA (&04)
9D0D	6	197 054	C5 36	CMP &36
9D0F		144 002	90 02	BCC 2 --> &9D13
9D11	6	165 054	A5 36	LDA &36
9D13	7	133 055	85 37	STA &37
9D15	7	196 055	C4 37	CPY &37
9D17		240 010	F0 0A	BEQ 10 --> &9D23
9D19		200	C8	INY
9D1A		177 004	B1 04	LDA (&04),Y
9D1C		217 255 005	D9 FF 05	CMP &05FF,Y
9D1F		240 244	F0 F4	BEQ -12 --> &9D15
9D21		128 004	80 04	BRA 4 --> &9D27
9D23		178 004	B2 04	LDA (&04)
9D25	6	197 054	C5 36	CMP &36
9D27		008	08	PHP
9D28		032 225 188	20 E1 BC	JSR &BCE1 Restore Stack Space used by String
9D2B		160 000	A0 00	LDY#&00
9D2D	(	040	28	PLP
9D2E	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

BCD2 Pop String from Stack

Submitted by Steve Fewell

Description:

Get last byte from Stack, Store in &36 (Length of String, SWA). If length is zero then increase Stack space by 1 (to remove length byte from Stack) & exit.
Otherwise, copy ?&36 number of bytes from the Stack to the SWA and then increase Stack space by ?&36 + 1 bytes (to reclaim the space used by the String) and exit.

Disassembly for the Pop String from Stack routine

BCD2		178 004	B2 04	LDA (&04)
BCD4	6	133 054	85 36	STA &36
BCD6		240 011	F0 0B	BEQ 11 --> &BCE3
BCD8		168	A8	TAY
BCD9		177 004	B1 04	LDA (&04),Y
BCDB		153 255 005	99 FF 05	STA &05FF,Y
BCDE		136	88	DEY
BCDF		208 248	D0 F8	BNE -8 --> &BCD9
BCE1		178 004	B2 04	LDA (&04)
BCE3	8	056	38	SEC
BCE4		128 023	80 17	BRA 23 --> &BCFD End of Pop Integer routine

---

BASIC IV ROM Routines

Page Last Altered: undefined

9C82 Compare Float values

Submitted by Steve Fewell

Starting Address: &9C82 (or &9C65 to convert Integer first)
Entry criteria: FWA contains the value to compare.
BASIC Text pointer points to the expression to compare FWA with.
Exit: Zero flag = 1 if values are equal. Carry is set if value 1 > value 2; clear otherwise.
Y = 00.

Description:

9C65 Convert Integer to Float and initialise Float values:
This routine is called during Compare Integer values if the second value in the compare is a Float value (not an Integer value, as expected).
Pop the IWA value from the processor stack (the first value to compare).
Push the FWA value (second value to compare) to the BASIC Stack.
Convert the IWA (first value to compare) to a Float value.
Copy this result from the FWA to the FWB.
Pop the Float from the BASIC Stack (second value to compare) and unpack the returned variable (pointed to by &4A, &4B) to the FWA.
Now, FWA = second value to compare & FWB = first value to compare. Go to &9C92 to perform the compare.

9C82 Initialise and get the Float values:
Push FWA to stack (first value to compare).
Gosub &9E4C to get the result of the expression (in the BASIC program text line), [we need to process expression level (+,-) upwards as any operators below this level should only be evaluated once this compare has been evaluated].
Copy A to Y [to update processor flags] and call &96DD to check that the result obtained is a Float value - if it is an Integer it will be converted to Float. If it is a String then a Type Mismatch error will be generated.

Pop the Float variable from the stack and unpack this variable to the FWB.
Now, the FWB = the first value to compare, and the FWA = second value to compare.

9C92 Compare the Float values:
Y = #&00
Reset bottom 7 bits of the FWB sign byte.
Load the FWA sign bit & reset the bottom 7 bits of the loaded value, then compare
this sign with the FWB sign byte. If the signs are not equal then exit (with C=1 if FWA>FWB).

Compare the FWB exponent with the FWA exponent (if not equal then stop as we have result).
Compare the FWB Mantissa byte 1 with the FWA Mantissa byte 1 (if not equal then stop as we have result).
Compare the FWB Mantissa byte 2 with the FWA Mantissa byte 2 (if not equal then stop as we have result).
Compare the FWB Mantissa byte 3 with the FWA Mantissa byte 3 (if not equal then stop as we have result).
Compare the FWB Mantissa byte 4 with the FWA Mantissa byte 4 (if not equal then stop as we have result).
If the Mantissa and exponents are equal, then the values are equal, so exit with A = 0.
Otherwise, C = 1 (if FWA > FWB). However, if both values are negative (Sign byte is 1), then
we need to reverse the carry, so that C = 0 (if it was 1), or C = 1 (if it was zero)
this is because the value that was higher is actually lower (as it is negative).
Exit with A = &01 (as values not equal).

Disassembly for the Compare Float values routine

9C65	h	104	68	PLA
9C66	*	133 042	85 2A	STA &2A
9C68	h	104	68	PLA
9C69	+	133 043	85 2B	STA &2B
9C6B	h	104	68	PLA
9C6C	,	133 044	85 2C	STA &2C
9C6E	h	104	68	PLA
9C6F	-	133 045	85 2D	STA &2D
9C71		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9C74		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9C77		032 011 164	20 0B A4	JSR &A40B Copy FWA to FWB
9C7A		032 232 187	20 E8 BB	JSR &BBE8 Pop Float from Stack
9C7D	A	032 065 165	20 41 A5	JSR &A541 Unpack Float variable to FWA
9C80		128 016	80 10	BRA 16 --> &9C92
9C82		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9C85	L	032 076 158	20 4C 9E	JSR &9E4C Expression Handler (+, - level onwards)
9C88		168	A8	TAY
9C89		032 221 150	20 DD 96	JSR &96DD Check Float value (conv int to float, etc.)
9C8C		032 232 187	20 E8 BB	JSR &BBE8 Pop Float from Stack (to &4A, &4B)

9C8F		032 224 164	20 E0 A4	JSR &A4E0 Unpack Float variable to FWB
9C92		160 000	A0 00	LDY#&00
9C94		169 127	A9 7F	LDA#&7F
9C96	;	020 059	14 3B	TRB &3B
9C98	.	165 046	A5 2E	LDA &2E
9C9A	)	041 128	29 80	AND#&80
9C9C	;	197 059	C5 3B	CMP &3B
9C9E		208 030	D0 1E	BNE 30 --> &9CBE
9CA0	<	165 060	A5 3C	LDA &3C
9CA2	0	197 048	C5 30	CMP &30
9CA4		208 025	D0 19	BNE 25 --> &9CBF
9CA6	=	165 061	A5 3D	LDA &3D
9CA8	1	197 049	C5 31	CMP &31
9CAA		208 019	D0 13	BNE 19 --> &9CBF
9CAC	>	165 062	A5 3E	LDA &3E
9CAE	2	197 050	C5 32	CMP &32
9CB0		208 013	D0 0D	BNE 13 --> &9CBF
9CB2	?	165 063	A5 3F	LDA &3F
9CB4	3	197 051	C5 33	CMP &33
9CB6		208 007	D0 07	BNE 7 --> &9CBF
9CB8	@	165 064	A5 40	LDA &40
9CBA	4	197 052	C5 34	CMP &34
9CBC		208 001	D0 01	BNE 1 --> &9CBF
9CBE	`	096	60	RTS
9CBF	j	106	6A	ROR A
9CC0	E;	069 059	45 3B	EOR &3B
9CC2	*	042	2A	ROL A
9CC3		169 001	A9 01	LDA#&01
9CC5	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Convert Integer to Floating-Point

Submitted by Steve Fewell

Routine: ConvIWAtoFWA
Name: Convert Integer to Floating-Point
Starting Address: &8185
Entry criteria: The IWA contains an Integer number.
Exit: The value of the IWA has been transferred to the FWA.

Note: The FWA Mantissa is stored Most significant byte first, least significant byte last (the opposite way to Integers).

Description:
Set the FWA Rounding (Mantissa 5) and Exponent Overflow bytes to zero. Store the IWA Sign Byte (? &2D) in the FWA Sign Byte (The top bit of this byte indicates whether the number is positive or negative, and it's value is directly transferable between the IWA and FWA).

If the Integer in the IWA is negative, then complement the Integer, and load A with the most significant byte of the Integer (We can deal directly with a positive value as the FWA will be stored as a positive number, with its Sign Byte indicating its sign).

If the Most Significant Byte of the IWA [A] isn't zero, then the number is 4 bytes long, so store IWA Byte 3 in FWA Mantissa Byte 2, IWA Byte 2 in FWA Mantissa Byte 3 and IWA Byte 1 in FWA Mantissa Byte 4 (opposite order), then normalise the FWA (Make the Most Significant Bit of the FWA's Mantissa equal to 1, and store A back to FWA Mantissa Byte 1, and Y back to the FWA Exponent), with an initial exponent [Y] value of 160 (that is 2^32, as there are 32 bits in a 4 byte number), and exit.

Otherwise, store zero in FWA Mantissa Byte 4, and check to see if IWA Byte 3 is zero.

If IWA Byte 3 isn't zero, then the number is 3 bytes long, so store IWA Byte 2 in FWA Mantissa Byte 2 and IWA Byte 1 in FWA Mantissa Byte 3 (opposite order), then normalise the FWA (Make the Most Significant Bit of the FWA's Mantissa equal to 1, and store A back to FWA Mantissa Byte 1, and Y back to the FWA Exponent), with A = IWA Byte 3 and an initial exponent [Y] value of 152 (that is 2^24, as there are 24 bits in a 3 byte number), and exit.

Otherwise, store zero in FWA Mantissa Byte 3, and check to see if IWA Byte 2 is zero.

If IWA Byte 2 isn't zero, then the number is 2 bytes long, so store IWA Byte 1 in FWA Mantissa Byte 2, then normalise the FWA (Make the Most Significant Bit of the FWA's Mantissa equal to 1, and store A back to FWA Mantissa Byte 1, and Y back to the FWA Exponent), with A = IWA Byte 2 and an initial exponent [Y] value of 144 (that is 2^16, as there are 16 bits in a 2 byte number), and exit.

Otherwise, store zero in FWA Mantissa Byte 2 and normalise the FWA (Make the Most Significant Bit of the FWA's Mantissa equal to 1, and store A back to FWA Mantissa Byte 1, and Y back to the FWA Exponent), with A = IWA Byte 1 and an initial exponent [Y] value of 136 (that is 2^8, as there are 8 bits in a 1 byte number). The normalisation will decide whether the number is zero, or a 1-byte number, and adjust the FWA accordingly.

Disassembly for the Convert Integer to Floating-Point routine

8185	d5	100 053	64 35	STZ &35
8187	d/	100 047	64 2F	STZ &2F
8189	-	165 045	A5 2D	LDA &2D
818B	.	133 046	85 2E	STA &2E
818D		016 005	10 05	BPL 5 --> &8194
818F		032 222 172	20 DE AC	JSR &ACDE icomp
8192	-	165 045	A5 2D	LDA &2D
8194	&	208 038	D0 26	BNE 38 --> &81BC
8196	d4	100 052	64 34	STZ &34
8198	,	165 044	A5 2C	LDA &2C
819A		208 020	D0 14	BNE 20 --> &81B0
819C	d3	100 051	64 33	STZ &33
819E	+	165 043	A5 2B	LDA &2B
81A0		208 006	D0 06	BNE 6 --> &81A8
81A2	d2	100 050	64 32	STZ &32

81A4	*	165 042	A5 2A	LDA &2A	
81A6	8	128 056	80 38	BRA 56 --> &81E0	
81A8	*	164 042	A4 2A	LDY &2A	
81AA	2	132 050	84 32	STY &32	
81AC		160 144	A0 90	LDY#&90	
81AE	2	128 050	80 32	BRA 50 --> &81E2	
81B0	+	164 043	A4 2B	LDY &2B	
81B2	2	132 050	84 32	STY &32	
81B4	*	164 042	A4 2A	LDY &2A	
81B6	3	132 051	84 33	STY &33	
81B8		160 152	A0 98	LDY#&98	
81BA	&	128 038	80 26	BRA 38 --> &81E2	
81BC	,	164 044	A4 2C	LDY &2C	
81BE	2	132 050	84 32	STY &32	
81C0	+	164 043	A4 2B	LDY &2B	
81C2	3	132 051	84 33	STY &33	
81C4	*	164 042	A4 2A	LDY &2A	
81C6	4	132 052	84 34	STY &34	
81C8		160 160	A0 A0	LDY#&A0	
81CA		128 022	80 16	BRA 22 --> &81E2	Normalise FWA#2

Normalise FWA with an initial exponent of 136 (&88)

81E0	160 136	A0 88	LDY#&88
81E2			Normalise FWA#2

BASIC IV ROM Routines

Page Last Altered: undefined

Normalise FWA#2 [only bit-by-bit method]

Submitted by Steve Fewell

Routine: anorm[2]
Name: Normalise Floating-Point Accumulator (FWA)#2 [only bit-by-bit method]
Starting Address: &81E2
Entry criteria: The FWA contains a floating-point number. Y = Exponent (?&30). A = MSB of FWA's Mantissa (?&31).
Exit: FWA has been normalised

Description:
'OR' the Accumulator with 0 to set the status flags according to the value.

If A = Negative then the top bit is set and the FWA is already normalised, so store A & Y back into &31 and &30 (respectively) and exit.

If A is zero then the number in the FWA is considered to be zero, so the FWA's sign, Exponent and Mantissa byte 1 are set to zero to indicate this, and the routine is exited. This uniquely identifies the number zero.

Otherwise, the top bit of the Mantissa is not 1, so we need to normalise the FWA as follows:
==> Keep on decrementing the exponent (Y) and multiplying the mantissa by 2 (moving its bits left a position), until the top bit is set (A becomes negative).

When the number is normalised, store A & Y back into &31 and &30 (respectively) and exit.

Disassembly for the Normalise FWA#2 [only bit-by-bit method] routine

81E2		009 000	09 00	ORA#&00
81E4	0	048 012	30 0C	BMI 12 --> &81F2
81E6		240 228	F0 E4	BEQ -28 --> &81CC
81E8		136	88	DEY
81E9	4	006 052	06 34	ASL &34
81EB	&3	038 051	26 33	ROL &33
81ED	&2	038 050	26 32	ROL &32
81EF	*	042	2A	ROL A
81F0		016 246	10 F6	BPL -10 --> &81E8
81F2	1	133 049	85 31	STA &31
81F4	0	132 048	84 30	STY &30
81F6	`	096	60	RTS

Complete Zero Number:

81CC	d.	100 046	64 2E	STZ &2E
81CE	d0	100 048	64 30	STZ &30
81D0	d/	100 047	64 2F	STZ &2F
81D2	d1	100 049	64 31	STZ &31
81D4	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Copy FWA to FWB

Submitted by Steve Fewell

Routine: bcopya
Name: Copy FWA to FWB
Starting Address: &A40B
Entry criteria: None
Exit: Copies the value of the FWA to the FWB.

Description:
Sets every byte of the FWB to a copy to the corresponding FWA byte as follows:

Byte Description	FWA		FWB
Sign	&2E	--->	&3B
Overflow	&2F	--->	XX Not copied
Exponent	&30	--->	&3C
Mantissa 1	&31	--->	&3D
Mantissa 2	&32	--->	&3E
Mantissa 3	&33	--->	&3F
Mantissa 4	&34	--->	&40
Mantissa 5 / Rounding	&35	--->	&41

The FWB now holds the same value as the FWA.

Disassembly for the Copy FWA to FWB routine

A40B	.	165 046	A5 2E	LDA &2E
A40D	;	133 059	85 3B	STA &3B
A40F	0	165 048	A5 30	LDA &30
A411	<	133 060	85 3C	STA &3C
A413	1	165 049	A5 31	LDA &31
A415	=	133 061	85 3D	STA &3D
A417	2	165 050	A5 32	LDA &32
A419	>	133 062	85 3E	STA &3E
A41B	3	165 051	A5 33	LDA &33
A41D	?	133 063	85 3F	STA &3F
A41F	4	165 052	A5 34	LDA &34
A421	@	133 064	85 40	STA &40
A423	5	165 053	A5 35	LDA &35
A425	A	133 065	85 41	STA &41
A427	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

BBE8 Pop Float from Stack (and point (&4A,&4B) to popped value)

Submitted by Steve Fewell

Description:

Store the Stack Pointer LSB in &4A and the Stack Pointer MSB in &4B.
Add 5 to the Stack Pointer LSB (&04) (add any carry resulting from this add to the Stack Pointer MSB (&05). This will increase Stack size by 5 bytes (reclaiming the 5 bytes occupied by the Floating-Point value that is now pointed to by (&4A,&4B).

Basically, this routine decreases occupied stack space by 5 bytes and stores old values of the stack pointer in (&4A,&4B - the argp).

Disassembly for the Pop Float from Stack routine

BBE8		165 004	A5 04	LDA &04
BBEA		024	18	CLC
BBEB	J	133 074	85 4A	STA &4A
BBED	i	105 005	69 05	ADC#&05
BBEF		133 004	85 04	STA &04
BBF1		165 005	A5 05	LDA &05
BBF3	K	133 075	85 4B	STA &4B
BBF5	i	105 000	69 00	ADC#&00
BBF7		133 005	85 05	STA &05
BBF9	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A541 Unpack Floating-Point Variable to FWA

Submitted by Steve Fewell

Routine:aunp
Name: Unpack Floating-Point Variable to FWA
Starting Address: &A541 (or &A539 to set argp to &046C first)
Entry criteria: &4A and &4B (the argp) point to a 5-byte Floating-Point variable to unpack.
Exit: The FWA contains the value of the variable.

Description:

If called from &A539 then set argp to point to variable location &046C first.
The FWA value will be extracted from the packed floating-point value at the address pointed to by argp.

Set FWA Rounding byte = 0.
Set FWA Overflow byte = 0.
FWA Mantissa 4 = 5th byte of the packed variable
FWA Mantissa 3 = 4th byte of the packed variable
FWA Mantissa 2 = 3rd byte of the packed variable
FWA Mantissa Sign = 2nd byte of the packed variable
FWA Exponent = 1st byte of the packed variable

If the Exponent is zero, then check whether the Mantissa byte 2, Mantissa byte 3
and Mantissa byte 4 are also zero. If so, then the FWA value = 0.0 so store zero in
FWA Mantissa 1 and exit.
Mantissa 1 is formed by ORing the 2nd byte of the packed variable with #&80 to ensure that
the top bit is set - as the top bit is replaced by the sign bit in the packed form.

Disassembly for the Unpack FP Variable to FWA routine

A539	l	169 108	A9 6C	LDA#&6C
A53B	J	133 074	85 4A	STA &4A
A53D		169 004	A9 04	LDA#&04
A53F	K	133 075	85 4B	STA &4B
A541	d5	100 053	64 35	STZ &35
A543	d/	100 047	64 2F	STZ &2F
A545		160 004	A0 04	LDY#&04
A547	J	177 074	B1 4A	LDA (&4A),Y
A549	4	133 052	85 34	STA &34
A54B		136	88	DEY
A54C	J	177 074	B1 4A	LDA (&4A),Y
A54E	3	133 051	85 33	STA &33
A550		136	88	DEY
A551	J	177 074	B1 4A	LDA (&4A),Y
A553	2	133 050	85 32	STA &32
A555		136	88	DEY
A556	J	177 074	B1 4A	LDA (&4A),Y
A558	.	133 046	85 2E	STA &2E
A55A		168	A8	TAY
A55B	J	178 074	B2 4A	LDA (&4A)
A55D	0	133 048	85 30	STA &30
A55F		208 009	D0 09	BNE 9 --> &A56A
A561		152	98	TYA
A562	2	005 050	05 32	ORA &32
A564	3	005 051	05 33	ORA &33
A566	4	005 052	05 34	ORA &34
A568		240 003	F0 03	BEQ 3 --> &A56D
A56A		152	98	TYA
A56B		009 128	09 80	ORA#&80
A56D	1	133 049	85 31	STA &31
A56F	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

96DA Get & Check Float Value

Submitted by Steve Fewell

Description:

This routine gets the result of the expression (in the SWA) and checks the result; or if called at &96DD, it just checks the latest result.
If the returned type is String (0), then Type Mismatch error.
If the returned type is Float (&FF), then ok, so return.
If the returned type is Integer (&40), then perform Integer to Floating-Point conversion (&8185).

Disassembly for the Get & check Float Value routine

96D7	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
96DA	6	032 054 173	20 36 AD	JSR &AD36 Get result of expression
96DD		240 248	F0 F8	BEQ -8 --> &96D7
96DF	0	048 245	30 F5	BMI -11 --> &96D6 [RTS]
96E1	L	076 133 129	4C 85 81	JMP &8185 Integer to Float

BASIC IV ROM Routines

Page Last Altered: undefined

A4E0 Unpack Floating-Point Variable to FWB

Submitted by Steve Fewell

Routine:bunp
Name: Unpack Floating-Point Variable to FWB
Starting Address: &A4E0
Entry criteria: &4A and &4B (the argp) point to a 5-byte Floating-Point variable to unpack.
Exit: The FWB contains the value of the variable.

Description:

Set FWB Rounding byte = 0.
Set FWB Overflow byte = 0.
FWB Mantissa 4 = 5th byte of the packed variable
FWB Mantissa 3 = 4th byte of the packed variable
FWB Mantissa 2 = 3rd byte of the packed variable
FWB Mantissa Sign = 2nd byte of the packed variable
FWB Exponent = 1st byte of the packed variable

If the Exponent is zero, then check whether the Mantissa byte 2, Mantissa byte 3 and Mantissa byte 4 are also zero. If so, then the FWB value = 0.0 so store zero in FWB Mantissa 1 and exit.
Mantissa 1 is formed by ORing the 2nd byte of the packed variable with #&80 to ensure that the top bit is set - as the top bit is replaced by the sign bit in the packed form.

Disassembly for the Unpack FP Variable to FWB routine

A4E0	dA	100 065	64 41	STZ &41
A4E2		160 004	A0 04	LDY#&04

A4E4	J	177 074	B1 4A	LDA (&4A),Y
A4E6	@	133 064	85 40	STA &40
A4E8		136	88	DEY
A4E9	J	177 074	B1 4A	LDA (&4A),Y
A4EB	?	133 063	85 3F	STA &3F
A4ED		136	88	DEY
A4EE	J	177 074	B1 4A	LDA (&4A),Y
A4F0	>	133 062	85 3E	STA &3E
A4F2		136	88	DEY
A4F3	J	177 074	B1 4A	LDA (&4A),Y
A4F5	;	133 059	85 3B	STA &3B
A4F7		168	A8	TAY
A4F8	J	178 074	B2 4A	LDA (&4A)
A4FA	<	133 060	85 3C	STA &3C
A4FC		208 009	D0 09	BNE 9 --> &A507
A4FE		152	98	TYA
A4FF	>	005 062	05 3E	ORA &3E
A501	?	005 063	05 3F	ORA &3F
A503	@	005 064	05 40	ORA &40
A505		240 003	F0 03	BEQ 3 --> &A50A
A507		152	98	TYA
A508		009 128	09 80	ORA#&80
A50A	=	133 061	85 3D	STA &3D
A50C	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9DEC '<>' operator

Submitted by Steve Fewell

Description:

Increment the Text Pointer B offset (so that the next character is the character after the '>').

Call &9CC9 to Compare the current value with the second value and set the flags according to which value is greater, or whether the values are equal.

If the zero flag is not set then the values are not equal, so set the IWA to TRUE.
Otherwise, set the IWA to FALSE.
Exit with A = #&40 (as we are currently handling an Integer value).

Disassembly for the '<>' Operator routine

9DEC	230 027	E6 1B	INC &1B
9DEE	032 201 156	20 C9 9C	JSR &9CC9 Compare Values
9DF1	208 206	D0 CE	BNE -50 --> &9DC1 Set TRUE
9DF3	128 205	80 CD	BRA -51 --> &9DC2 Set FALSE

BASIC IV ROM Routines

Page Last Altered: undefined

9DF5 '>' operator

Submitted by Steve Fewell

Description:

Check the next character from the BASIC Input line. If the next character is '=' then the operator is '>=' Greater Than or Equal To, so jump to &9E07 to handle this operator. Otherwise the operator is '>' Greater Than. which is handled as follows:

Call &9CC9 to Compare the current value with the second value and set the flags according to which value is greater, or whether the values are equal.

If the zero flag is set then the values are equal, so set the IWA to FALSE, as the first value is not greater than the second value.
If the carry flag is set then the first value is greater than the second, so set the IWA to TRUE; otherwise, the first value is not greater than the second, so set the IWA to FALSE.
Exit with A = #&40 (as we are currently handling an Integer value).

Disassembly for the '>' Operator routine

9DF5		170	AA	TAX
9DF6		164 027	A4 1B	LDY &1B
9DF8		177 025	B1 19	LDA (&19),Y
9DFA	=	201 061	C9 3D	CMP#&3D '='
9DFC		240 009	F0 09	BEQ 9 --> &9E07 '>=' Greater Than or Equal to Operator
9DFE		032 201 156	20 C9 9C	JSR &9CC9 Compare Values
9E01		240 191	F0 BF	BEQ -65 --> &9DC2 Set FALSE
9E03		176 188	B0 BC	BCS -68 --> &9DC1 Set TRUE
9E05		128 187	80 BB	BRA -69 --> &9DC2 Set FALSE

BASIC IV ROM Routines

Page Last Altered: undefined

9E07 '>=' operator

Submitted by Steve Fewell

Description:

Increment the Text Pointer B offset (so that the next character is the character
after the '=').

Call &9CC9 to Compare the current value with the second value and set the flags
according to which value is greater, or whether the values are equal.

if the carry flag is set then the first value is greater than or equal to the second,
so set the IWA to TRUE; otherwise, the first value is not greater than or equal to the second,
so set the IWA to FALSE.
Exit with A = #&40 (as we are currently handling an Integer value).

Disassembly for the '>=' Operator routine

9E07	230 027	E6 1B	INC &1B
9E09	032 201 156	20 C9 9C	JSR &9CC9 Compare Values
9E0C	176 179	B0 B3	BCS -77 --> &9DC1 Set TRUE
9E0E	128 178	80 B2	BRA -78 --> &9DC2 Set FALSE

BASIC IV ROM Routines

Page Last Altered: undefined

9E58 '+' operator - addition

Submitted by Steve Fewell

Description:

Check the value type (in A).
If variable type = 0 then goto the String Addition routine.

If variable type is negative then the first value is a Float, so jump to &9E94, which does the following:
* Push FWA to Stack
* Get the second value from the expression handler (level 5)
* If second value is a String then generate a Type Mismatch error.
* If second value is an Integer then convert it to a Float
* Pop the first Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Addition routine to add the values (&A68D)
* Jump to 9E4F (Expression handler level 4) to check for further + or - operators.

Otherwise, the value is Integer, so push the first value to the stack and call the Expression Handler level 5 routine (&9FC4) to get the second value to add.
If the second value is a string then generate a Type Mismatch error.

If the second value is a Float then jump to &9EB0 to do the following:
* Pop the first value (the Integer) from the Stack
* Push the second value (the Float) to the Stack
* Convert the Integer to a Float (and put it in the FWA).
* Pop the second Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Addition routine to add the values (&A68D)
* Jump to 9E4F (Expression handler level 4) to check for further + or - operators.

Otherwise, both values are Integers, so carry out the Integer Addition routine. This routine automatically jumps back to 9E4F on completion, to check for further + or - operators.

Disassembly for the '+' Operator - Addition routine

9E58		168	A8	TAY
9E59		240 199	F0 C7	BEQ -57 --> &9E22 String Addition
9E5B	07	048 055	30 37	BMI 55 --> &9E94
9E5D		032 196 159	20 C4 9F	JSR &9FC4 Push Integer to stack & Expression handler level 5

9E60		168	A8	TAY
9E61	.	240 046	F0 2E	BEQ 46 --> &9E91 [JMP &9092 Type Mismatch error]
9E63	0K	048 075	30 4B	BMI 75 --> &9EB0
9E65				...iplus (Integer addition)

Disassembly for the Floating Point '+' Operator

9E91	L	076 146 144	4C 92 90	JMP &9092 Type Mismatch error
9E94		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9E97		032 199 159	20 C7 9F	JSR &9FC7 Expression Handler level 5 (*,/,MOD,DIV)
9E9A		168	A8	TAY
9E9B		240 244	F0 F4	BEQ -12 --> &9E91 Type Mismatch error
9E9D	'	134 039	86 27	STX &27
9E9F	0	048 003	30 03	BMI 3 --> &9EA4
9EA1		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9EA4		032 232 187	20 E8 BB	JSR &BBE8 Pop Float to argp
9EA7		032 141 166	20 8D A6	JSR &A68D Floating-Point Addition
9EAA	'	166 039	A6 27	LDX &27
9EAC		169 255	A9 FF	LDA#&FF
9EAE		128 159	80 9F	BRA -97 --> &9E4F Expression handler level 4 (+,-)
9EB0	'	134 039	86 27	STX &27
9EB2		032 230 188	20 E6 BC	JSR &BCE6 Pop Integer from Stack
9EB5		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9EB8		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9EBB		128 231	80 E7	BRA -25 --> &9EA4

BASIC IV ROM Routines

Page Last Altered: undefined

9E22 String Addition

Submitted by Steve Fewell

Routine:splus
Name: SWA = SWA + string expression
Starting Address: &9E22
Entry criteria: SWA contains a string value. The BASIC Text pointer points to the second string to concatenate to the end of the SWA.
Exit: The SWA contains the concatenated strings.

Description:

Frstly, push the existing string value in the SWA to the Stack.
Call &A012 to get the result of expression (as there are no higher priority operators for strings above '+', then the highest level of the expression handler can be used [A012].
Y=A [to set flags]
If the second value is not a string then generate a Type Mismatch error.

Store X (type of variable?) on stack.
Add the new string's length to the length of the first string (on Stack). If the result is > 255 then generate a String Too Long error.
X = resulting length (length of new concatenated string).
Store length of new string on Stack.
Y = Length of secong string.
Move each character of the second string (last character first) to the end of theÿ SWA value, so that the last character of the second string is stored in the SWA at the new string length position, and the first character of the second string is stored in the SWA at the position of the first string length + 1.
Now the second string is in the appropriate place, we just need to put the first string back in to the SWA (starting at position &600), to do this we simply

Pop the SWA String value from the Stack. [Note: This won't overwrite any of the 2nd String, as only the length of the first value is extracted].

Pop the new string length from the Stack, and store it in &36, so that the correct new length of the SWA is specified.
Pop X from the stack (to restore the variable type information?).
exit with A = 00 (as result is a String).

Disassembly for the String Addition routine

9E22	Q	032 081 188	20 51 BC	JSR &BC51 Push SWA to Stack
9E25		032 018 160	20 12 A0	JSR &A012 Get result of expression
9E28		168	A8	TAY
9E29	f	208 102	D0 66	BNE 102 --> &9E91 [JMP &9092 Type Mismatch error]
9E2B		024	18	CLC
9E2C		218	DA	PHX
9E2D		178 004	B2 04	LDA (&04)
9E2F	e6	101 054	65 36	ADC &36
9E31		176 221	B0 DD	BCS -35 --> &9E10 String Too Long error
9E33		170	AA	TAX
9E34	H	072	48	PHA
9E35	6	164 054	A4 36	LDY &36
9E37		185 255 005	B9 FF 05	LDA &05FF,Y
9E3A		157 255 005	9D FF 05	STA &05FF,X
9E3D		202	CA	DEX
9E3E		136	88	DEY
9E3F		208 246	D0 F6	BNE -10 --> &9E37
9E41		032 210 188	20 D2 BC	JSR &BCD2 Pop String from Stack
9E44	h	104	68	PLA
9E45	6	133 054	85 36	STA &36
9E47		250	FA	PLX
9E48		169 000	A9 00	LDA#&00
9E4A		128 003	80 03	BRA 3 --> &9E4F Process any more +'s

BASIC IV ROM Routines

Page last modified: undefined

Integer Addition Routine

Submitted by Steve Fewell

Routine: iplus
Name: Integer Addition
Starting Address: &9E65
Entry criteria: IWA contains the first Integer number. ?&4 and ?&5 [The BASIC stack pointer] points to the second Integer variable. X = The next operator code (or 4, if there is no further operation).
Exit: IWA contains the result of [Integer variable + IWA]

Description:
Adds the 32-bit Integer pointed to by the BASIC Stack Pointer (&4 (lo),&5 (hi)) to the number in the IWA. The least significant byte [&2A] is added first, followed by &2B, &2C and &2D (The most significant byte). The carry flag allows any overflow to be carried forward and updated to the next significant byte of the number.

The code from &9E82 updates the Stack pointer, so that it now points to the address after the added number. To do this, 4 bytes are added to the address pointed to by (&4, &5).

Jump to &9E4F to test the value of X [the next operator]. If X contains the operator code for '+' or '-' then send the result to the appropriate routine, for further calculation, otherwise exit the addition routine.

Disassembly for the integer addition routine

9E65	024	18	CLC

```
9E66        178 004     B2 04       LDA (&04)
9E68  e*    101 042     65 2A       ADC &2A
9E6A  *     133 042     85 2A       STA &2A
9E6C        160 001     A0 01       LDY#&01
9E6E        177 004     B1 04       LDA (&04),Y
9E70  e+    101 043     65 2B       ADC &2B
9E72  +     133 043     85 2B       STA &2B
9E74        200         C8          INY
9E75        177 004     B1 04       LDA (&04),Y
9E77  e,    101 044     65 2C       ADC &2C
9E79  ,     133 044     85 2C       STA &2C
9E7B        200         C8          INY
9E7C        177 004     B1 04       LDA (&04),Y
9E7E  e-    101 045     65 2D       ADC &2D
9E80  -     133 045     85 2D       STA &2D
9E82        024         18          CLC
9E83        165 004     A5 04       LDA &04
9E85  i     105 004     69 04       ADC#&04
9E87        133 004     85 04       STA &04
9E89  @     169 064     A9 40       LDA#&40
9E8B        144 194     90 C2       BCC -62 --> &9E4F
9E8D        230 005     E6 05       INC &05
9E8F        128 190     80 BE       BRA -66 --> &9E4F

9E4F  +     224 043     E0 2B       CPX#&2B
9E51        240 005     F0 05       BEQ 5 --> &9E58
9E53  -     224 045     E0 2D       CPX#&2D
9E55  f     240 102     F0 66       BEQ 102 --> &9EBD
9E57  `     096         60          RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

A68D Floating-Point Addition Entry Point

Submitted by Steve Fewell

Routine:aplus
Name: Floating-Point Addition
Starting Address: &A68D
Entry criteria: &4A and &4B (the argp) point to a 5-byte Floating-Point variable to add.
The FWA contains the other value to add.
Exit: The FWA = [argp] + FWA [Normalised and Rounded].

Description:

Calls &A4E0 to unpack the variable pointed to by argp (&4A, &4B) to the
FWB.
If the unpacked variable is zero (Mantissa byte 1 is 0) then exit as there is nothing to add.
Call the Floating Point Addition routine [FWA = FWA + FWB] to actually do the addition.
Continue to the 'Round FWA Mantissa to 4 bytes' routine at address &A695
so that we can exit with the resulted rounded and the Mantissa rounding byte clear.

If called from &A692 then just the FWA = FWA + FWB calculation is done and the
FWA Mantissa is rounded to 4 bytes before exiting.

Disassembly for the Floating-Point Addition Entry Point routine

A68D	032 224 164	20 E0 A4	JSR &A4E0 Unpack argp variable to FWB
A690	2 240 050	F0 32	BEQ 50 --> &A6C4 [RTS]
A692	h 032 104 131	20 68 83	JSR &8368 Floating-Point Addition
A695			... Round FWA Mantissa to 4-bytes (loose rounding byte)

BASIC IV ROM Routines

Page Last Altered: undefined

A695 Round FWA Mantissa to 4 bytes (loose rounding byte)

Submitted by Steve Fewell

Description:

If FWA Mantissa rounding byte is less than #&80 then jump to A6AE.

Otherwise we need to round-up the current 4-byte Mantissa value.
If the Mantissa rounding byte is equal to #&80 then, the LSB of the FWA Mantissa byte 4 will be set to 1 (if it was zero). No other rounding is done.
[Note: the ROL A instruction in this code will change value 1000 0000 to 0000 0001].
Otherwise (rounding > #&80) to round up the value we increment the FWA Mantissa byte 4 by 1 (if necessary, &A4D3 will be called to follow through the increment (to the rest of the Mantissa) if the increment caused the value of Mantissa byte 4 to overflow (from &FF to &00).
Continue with &A6AE (described below).

&A6AE checks the FWA Exponent overflow byte. If it is zero then the number is fine, so clear FWA Mantissa rounding byte and exit.
Otherwise, FWA exponent overflow byte is positive, so generate a "Too Big" error.

Disassembly for the Round FWA Mantissa to 4 bytes routine

A695	5	165 053	A5 35	LDA &35
A697		201 128	C9 80	CMP#&80
A699		144 019	90 13	BCC 19 --> &A6AE
A69B		240 014	F0 0E	BEQ 14 --> &A6AB
A69D	4	230 052	E6 34	INC &34
A69F		208 013	D0 0D	BNE 13 --> &A6AE

A6A1		032 211 164	20 D3 A4	JSR &A4D3
A6A4		128 008	80 08	BRA 8 --> &A6AE
A6A6-A6AA				Not used by this routine
A6AB	*	042	2A	ROL A
A6AC	4	004 052	04 34	TSB &34
A6AE	/	165 047	A5 2F	LDA &2F
A6B0		240 016	F0 10	BEQ 16 --> &A6C2 [STZ &35:RTS] from aclear
A6B2		016 017	10 11	BPL 17 --> &A6C5 Too Big Error

BASIC IV ROM Routines

Page Last Altered: undefined

Float Divide 10

Submitted by Steve Fewell

Routine: adiv10
Name: Float Divide 10
Starting Address: &A478
Entry criteria: The FWA contains the Floating-point number to divide by 10.
Exit: FWA contains the result of [FWA / 10]

Description:
As BASIC handles input and output of base-10 numbers (decimal), but does it's calculations in base-2 (binary), is needs fast routines to multiple and divide by a factor of 10.

The standard Floating-point '/' and '*' routines could be used for this purpose, and BASIC could provide 10 as the second value in the division and multiplication routines. But this wouldn't be a very satisfactory method - as BASIC requires these frequent calculations to be performed as fast as possible.

The problem with the standard Floating-point '/' and '*' routines is that they require loops (iteration), in order to perform the required action on the two variable numbers. The loops will probably be iterated for different numbers of times depending on the values of both of the variable numbers. BASIC requires multiplication and division by 10 as part of its internal routines, and as such cannot accept an undetermined number of iterations until a result is found.

Hence, the BASIC ROM contains separate routines to multiply and divide floating-point numbers by 10. These routines contain no loops (iteration), and run through a set sequence of commands each time (with little variation).

This page contains the complicated code that BASIC performs in order to achieve a Division by 10. The Multiply by 10 routine can be found here.

The routine executes the following steps:

a) 4 is subtracted from the FWA exponent, this divides the number by 16 (as the number is stored in binary). If a borrow occurred, then also decrement the FWA overflow exponent.

b) Copy the FWA to FWB, and divide the FWB mantissa by 2. Then add the FWB mantissa to the FWA mantissa (Least significant byte first, as usual). If carry if left after adding, then the FWA mantissa overflowed, so divide the FWA mantissa by 2 (making the top bit 1), and increment the FWA exponent & FWA exponent overflow (if necessary) to correct.

c) Copy the FWA to FWB, and divide the FWB mantissa by 16. Then add the FWB mantissa to the FWA mantissa (Least significant byte first, as usual). If carry if left after adding, then the FWA mantissa overflowed, so divide the FWA mantissa by 2 (making the top bit 1), and increment the FWA exponent & FWA exponent overflow (if necessary) to correct.

d) Zero &3D (FWB Mantissa byte 1), and set FWB mantissa byte 2 to FWA mantissa byte 1, FWB mantissa byte 3 to FWA mantissa byte 2, FWB mantissa byte 4 to FWA mantissa byte 3 and FWB mantissa byte 5 to FWA mantissa byte 4. ROL &35 (FWA mantissa byte 5), so that C = mantissa byte 5's top bit. This byte manipulation has resulted in FWB = FWA divided by 256.
Next add the FWB mantissa to the FWA mantissa (Least significant byte first, as usual). If carry if left after adding, then the FWA mantissa overflowed, so divide the FWA mantissa by 2 (making the top bit 1), and increment the FWA exponent & FWA exponent overflow (if necessary) to correct.

e) Zero &3E (FWB Mantissa byte 2, Note: FWB byte 1 is still zero from the previous step), and set FWB mantissa byte 3 to FWA mantissa byte 1, FWB mantissa byte 4 to FWA mantissa byte 2 and FWB mantissa byte 5 to FWA mantissa byte 3. ROL &34 (FWA mantissa byte 4), so that C = mantissa byte 4's top bit. This byte manipulation has resulted in FWB = FWA divided by 65536.
Next add the FWB mantissa to the FWA mantissa (Least significant byte first, as usual). If carry if left after adding, then the FWA mantissa overflowed, so divide the FWA mantissa by 2 (making the top bit 1), and increment the FWA exponent & FWA exponent overflow (if necessary) to correct.

f) If the mantissa overflowed during the previous step, some extra rounding is performed, to correct the mantissa's value (this rounding correction also takes into account the value in the FWA mantissa byte 5).

g) If no overflow from the mantissa has occurred then end, otherwise divide the FWA mantissa by 2, making the top bit equal to 1 (from carry flag) to account for the overflow. The FWA exponent is then incremented to allow for the carry that was added to the mantissa. The FWA overflow exponent is also incremented if the FWA exponent overflowed.

In simpler terms, this routine is doing the following:

> a) FWA = FWA / 16
> b) FWA = FWA + (FWA / 2)
> c) FWA = FWA + (FWA / 16)
> d) FWA = FWA + (FWA / 256)
> e) FWA = FWA + (FWA / 65536)

Each of the division steps results in the following fractions of the original number:

a) 1 / 16	[0.0625]
b) 1 / 32	[0.03125]
c) 3 / 512	[0.0058593748]
d) 51 / 131072	[0.000389099]
e) 6553 / 4294967296	[0.00000152585]

All of these fractions add up to 0.1 (a tenth) - So this routines basically takes a tenth of the original number.

This gives the same result as FWA / 10, except that the calculations are done using binary powers - which are a lot easier (and faster) to handle in 6502 Assembly language, than other decimal numbers.

Disassembly for the Floating Point divide 10 routine

A478	8	056	38	SEC
A479	0	165 048	A5 30	LDA &30
A47B		233 004	E9 04	SBC#&04
A47D	0	133 048	85 30	STA &30
A47F		176 002	B0 02	BCS 2 --> &A483
A481	/	198 047	C6 2F	DEC &2F
A483	(	032 040 164	20 28 A4	JSR &A428 FWB=FWA/2
A486	G	032 071 164	20 47 A4	JSR &A447 Add FWB to FWA
A489	(	032 040 164	20 28 A4	JSR &A428 FWB=FWA/2
A48C	+	032 043 164	20 2B A4	JSR &A42B FWB=FWB/2
A48F	+	032 043 164	20 2B A4	JSR &A42B FWB=FWB/2
A492	+	032 043 164	20 2B A4	JSR &A42B FWB=FWB/2
A495	G	032 071 164	20 47 A4	JSR &A447 Add FWB to FWA

```
A498  d=  100 061          64 3D       STZ &3D
A49A  1   165 049          A5 31       LDA &31
A49C  >   133 062          85 3E       STA &3E
A49E  2   165 050          A5 32       LDA &32
A4A0  ?   133 063          85 3F       STA &3F
A4A2  3   165 051          A5 33       LDA &33
A4A4  @   133 064          85 40       STA &40
A4A6  4   165 052          A5 34       LDA &34
A4A8  A   133 065          85 41       STA &41
A4AA  5   165 053          A5 35       LDA &35
A4AC  *   042              2A          ROL A
A4AD  G   032 071 164      20 47 A4    JSR &A447   Add FWB to FWA
A4B0  d>  100 062          64 3E       STZ &3E
A4B2  1   165 049          A5 31       LDA &31
A4B4  ?   133 063          85 3F       STA &3F
A4B6  2   165 050          A5 32       LDA &32
A4B8  @   133 064          85 40       STA &40
A4BA  3   165 051          A5 33       LDA &33
A4BC  A   133 065          85 41       STA &41
A4BE  4   165 052          A5 34       LDA &34
A4C0  *   042              2A          ROL A
A4C1  G   032 071 164      20 47 A4    JSR &A447   Add FWB to FWA
A4C4  2   165 050          A5 32       LDA &32
A4C6  *   042              2A          ROL A
A4C7  1   165 049          A5 31       LDA &31
A4C9  e5  101 053          65 35       ADC &35
A4CB  5   133 053          85 35       STA &35
A4CD      144 016          90 10       BCC 16 --> &A4DF
A4CF  4   230 052          E6 34       INC &34
A4D1      208 012          D0 0C       BNE 12 --> &A4DF
A4D3  3   230 051          E6 33       INC &33
A4D5      208 008          D0 08       BNE 8 --> &A4DF
A4D7  2   230 050          E6 32       INC &32
A4D9      208 004          D0 04       BNE 4 --> &A4DF
```

```
A4DB  1   230 049      E6 31       INC &31
A4DD      240 136      F0 88       BEQ -120 --> &A467   Handle Mantissa Overflow
A4DF  `   096          60          RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

Float Multiply 10

Submitted by Steve Fewell

Routine: amult10
Name: Float Multiply 10
Starting Address: &A436
Entry criteria: The FWA contains the Floating-point number to multiply by 10.
Exit: FWA contains the result of [FWA * 10]

Description:
As BASIC handles input and output of base-10 numbers (decimal), but does it's calculations in base-2 (binary), is needs fast routines to multiple and divide by a factor of 10.

The standard Floating-point '/' and '*' routines could be used for this purpose, and BASIC could provide 10 as the second value in the division and multiplication routines. But this wouldn't be a very satisfactory method - as BASIC requires these frequent calculations to be performed as fast as possible.

The problem with the standard Floating-point '/' and '*' routines is that they require loops (iteration), in order to perform the required action on the two variable numbers. The loops will probably be iterated for different numbers of times depending on the values of both of the variable numbers. BASIC requires multiplication and division by 10 as part of its internal routines, and as such cannot accept an undetermined number of iterations until a result is found.

Hence, the BASIC ROM contains separate routines to multiply and divide floating-point numbers by 10. These routines contain no loops (iteration), and run through a set sequence of commands each time (with little variation).

This page contains the Multiplication by 10 routine. The Division by 10 routine can be found here.

3 is added to the FWA exponent, this multiplies the number by 8 (as the number is stored in binary). The FWA overflow exponent is also updated, if this causes the FWA exponent to overflow.

Copy the FWA to FWB, and divide the FWB mantissa by 4. Then add the FWB mantissa to the FWA mantissa (Least significant byte first, as usual).

If no overflow from the mantissa has occurred then end, otherwise divide the FWA mantissa by 2, making the top bit equal to 1 (from carry flag) to account for the overflow. The FWA exponent is then incremented to allow for the carry that was added to the mantissa. The FWA overflow exponent is also incremented if the FWA exponent overflowed.

In simpler terms, this routine is doing the following:

> FWA = FWA * 8
> FWA = FWA + (0.25 * FWA)

which is the same as the following:

> FWA = 1.25 * (FWA * 8)

This gives the same result as FWA * 10, except that the calculations are done using binary powers - which are a lot easier (and faster) to handle in 6502 Assembly language, than other decimal numbers.

Disassembly for the Floating Point multiply 10 routine

A436		024	18	CLC
A437	0	165 048	A5 30	LDA &30
A439	i	105 003	69 03	ADC#&03
A43B	0	133 048	85 30	STA &30
A43D		144 002	90 02	BCC 2 --> &A441
A43F	/	230 047	E6 2F	INC &2F
A441	(	032 040 164	20 28 A4	JSR &A428
A444	+	032 043 164	20 2B A4	JSR &A42B
A447	5	165 053	A5 35	LDA &35

A449	eA	101 065	65 41	ADC &41
A44B	5	133 053	85 35	STA &35
A44D	4	165 052	A5 34	LDA &34
A44F	e@	101 064	65 40	ADC &40
A451	4	133 052	85 34	STA &34
A453	3	165 051	A5 33	LDA &33
A455	e?	101 063	65 3F	ADC &3F
A457	3	133 051	85 33	STA &33
A459	2	165 050	A5 32	LDA &32
A45B	e>	101 062	65 3E	ADC &3E
A45D	2	133 050	85 32	STA &32
A45F	1	165 049	A5 31	LDA &31
A461	e=	101 061	65 3D	ADC &3D
A463	1	133 049	85 31	STA &31
A465		144 016	90 10	BCC 16 --> &A477
A467	f1	102 049	66 31	ROR &31
A469	f2	102 050	66 32	ROR &32
A46B	f3	102 051	66 33	ROR &33
A46D	f4	102 052	66 34	ROR &34
A46F	f5	102 053	66 35	ROR &35
A471	0	230 048	E6 30	INC &30
A473		208 002	D0 02	BNE 2 --> &A477
A475	/	230 047	E6 2F	INC &2F
A477	`	096	60	RTS

FWB = FWA / 2 (Sub routine used by amult10):

A428		032 011 164	20 0B A4	JSR &A40B bcopya
A42B	F=	070 061	46 3D	LSR &3D
A42D	f>	102 062	66 3E	ROR &3E
A42F	f?	102 063	66 3F	ROR &3F
A431	f@	102 064	66 40	ROR &40
A433	fA	102 065	66 41	ROR &41

A435 ` 096 60 RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8368 Floating-Point Addition

Submitted by Steve Fewell

Routine:aplus1
Name: Floating-Point Addition (FWA=FWA+FWB Normalised & Unrounded)
Starting Address: &8368
Entry criteria: The FWA and FWB contain the required numbers.
Exit: The FWA contains the result.

Description:

If FWA Mantissa byte 1 = 0 then FWA= 0.0, so return FWB
in FWA (AcopyB routine).

A = FWA Exponent - FWB Exponent
If exponents are equal then goto 83E2 to do the addition.

If the FWA exponent > FWB exponent then goto 83A9 to adjust the exponents to be equal.
This routine does the following:
Inverse and increment (by 1) the exponent difference. This will make it a two's complement value and complement the value (exponent diff=-exponent diff).
If the exponent difference > &24 then the FWA is too small to add, so return FWA=FWB.

Otherwise, Set the Result's exponent [FWA exponent] = FWB exponent.

If the exponent difference is a multiple of 8 (8, 16, 24, 32) then shift the FWA mantissa right a byte [divide FWA Mantissa by 16] and reduce the exponent diference by 8. [Keep doing this until FWA exponent = FWB exponent].

If the exponent difference is zero (it will be if the byte shifting was done (above)) then do the addition [83E2].

Otherwise shift the FWA Mantissa's bits right a bit [divide Mantissa by 2] and decrement the exponent difference. Repeat this until the difference is zero.

Lastly, goto 83E2 to do the additional as the exponents are now equal.

Otherwise, (FWB exponent > FWA exponent) so need to make the exponents equal as follows:
If the exponent difference > &24 then the FWB is too small to add, so return with FWA unchanged.

Otherwise, if the exponent difference is a multiple of 8 (8, 16, 24, 32) then shift the FWB Mantissa right a byte [divide FWA Mantissa by 16] and reduce the exponent diference by 8. [Keep doing this until FWB exponent = FWA exponent].

If the exponent difference is zero (it will be if the byte shifting was done (above)) then do the addition [83E2].
Otherwise shift the FWB Mantissa right one bit [divide Mantissa by 2] and decrement the exponent difference. Repeat this until the difference is zero.

Lastly, goto 83E2 to do the additional as the exponents are now equal.

83E2: Addition (where the exponents are equal):
A = FWA Sign; EOR with FWB Sign.
If different signs then goto 83EC to do subtraction (described below).

Clear carry and jump to &A447 (in Floating-Point Multiply by 10 routine).
A447 does the following:
FWA Mantissa 5 = FWA Mantissa 5 + FWB Mantissa 5
FWA Mantissa 4 = FWA Mantissa 4 + FWB Mantissa 4
FWA Mantissa 3 = FWA Mantissa 3 + FWB Mantissa 3
FWA Mantissa 2 = FWA Mantissa 2 + FWB Mantissa 2
FWA Mantissa 1 = FWA Mantissa 1 + FWB Mantissa 1

If no overflow from adding the Mantissas then end (RTS).
(Note: The exponent and sign bytes are unchanged as they are the same for both numbers.)
Otherwise, divide the Mantissa by 2 (keeping the top bit set) & increment exponent.

83EC: Signs are not equal - do subtraction:
If FWA = FWB then **Clear FWA** to return with FWA = 0.0 (numbers add to zero)
else 840D: If FWA > FWB then goto 8435 (Subtract FWA from FWA) [Described below]

Otherwise, copy the FWB sign to FWA sign (as the result will have the same sign as FWB)
& subtract FWA Mantissa from FWB Mantissa storing the result in the FWA Mantissa.
Jump to &81F9 to Normalise FWA (A = FWA Mantissa 1 on entry)

8435: Subtract FWB Mantissa from FWA Mantissa:
This routine subtracts the FWB value from FWA, keeping the FWA Exponent and FWA sign unaltered as the result will have the same sign as the FWA.

Subtract FWA Mantissa from FWB Mantissa storing the result in the FWA Mantissa.
Jump to &81F9 to Normalise FWA (A = FWA Mantissa 1 on entry)

Disassembly for the Floating-Point Addition routine

8368	1	165 049	A5 31	LDA &31
836A		240 221	F0 DD	BEQ -35 --> &8349 AcopyB
836C	8	056	38	SEC
836D	0	165 048	A5 30	LDA &30
836F	<	229 060	E5 3C	SBC &3C
8371	o	240 111	F0 6F	BEQ 111 --> &83E2
8373	4	144 052	90 34	BCC 52 --> &83A9
8375	%	201 037	C9 25	CMP#&25
8377		176 238	B0 EE	BCS -18 --> &8367 (RTS)
8379		168	A8	TAY
837A	)8	041 056	29 38	AND#&38
837C		240 023	F0 17	BEQ 23 --> &8395
837E	8	056	38	SEC
837F	@	166 064	A6 40	LDX &40
8381	A	134 065	86 41	STX &41
8383	?	166 063	A6 3F	LDX &3F
8385	@	134 064	86 40	STX &40
8387	>	166 062	A6 3E	LDX &3E
8389	?	134 063	86 3F	STX &3F
838B	=	166 061	A6 3D	LDX &3D
838D	>	134 062	86 3E	STX &3E
838F	d=	100 061	64 3D	STZ &3D
8391		233 008	E9 08	SBC#&08
8393		208 234	D0 EA	BNE -22 --> &837F
8395		152	98	TYA
8396	)	041 007	29 07	AND#&07
8398	H	240 072	F0 48	BEQ 72 --> &83E2
839A	F=	070 061	46 3D	LSR &3D
839C	f>	102 062	66 3E	ROR &3E
839E	f?	102 063	66 3F	ROR &3F
83A0	f@	102 064	66 40	ROR &40
83A2	fA	102 065	66 41	ROR &41

83A4	:	058	3A	DEC A
83A5		208 243	D0 F3	BNE -13 --> &839A
83A7	9	128 057	80 39	BRA 57 --> &83E2

83A9: Adjust FWA exponent to equal FWB Exponent (where FWA exp > FWB exp)

83A9	I	073 255	49 FF	EOR#&FF
83AB		026	1A	INC A
83AC	%	201 037	C9 25	CMP#&25
83AE		176 153	B0 99	BCS -103 --> &8349 AcopyB
83B0	<	164 060	A4 3C	LDY &3C
83B2	0	132 048	84 30	STY &30
83B4		168	A8	TAY
83B5	)8	041 056	29 38	AND#&38
83B7		240 023	F0 17	BEQ 23 --> &83D0
83B9	8	056	38	SEC
83BA	4	166 052	A6 34	LDX &34
83BC	5	134 053	86 35	STX &35
83BE	3	166 051	A6 33	LDX &33
83C0	4	134 052	86 34	STX &34
83C2	2	166 050	A6 32	LDX &32
83C4	3	134 051	86 33	STX &33
83C6	1	166 049	A6 31	LDX &31
83C8	2	134 050	86 32	STX &32
83CA	d1	100 049	64 31	STZ &31
83CC		233 008	E9 08	SBC#&08
83CE		208 234	D0 EA	BNE -22 --> &83BA
83D0		152	98	TYA
83D1	)	041 007	29 07	AND#&07
83D3		240 013	F0 0D	BEQ 13 --> &83E2
83D5	F1	070 049	46 31	LSR &31
83D7	f2	102 050	66 32	ROR &32
83D9	f3	102 051	66 33	ROR &33
83DB	f4	102 052	66 34	ROR &34
83DD	f5	102 053	66 35	ROR &35
83DF	:	058	3A	DEC A
83E0		208 243	D0 F3	BNE -13 --> &83D5

83E2: FWA = FWA + FWB [where the exponents are equal]

83E2	.	165 046	A5 2E	LDA &2E
83E4	E;	069 059	45 3B	EOR &3B
83E6	0	048 004	30 04	BMI 4 --> &83EC
83E8		024	18	CLC
83E9	LG	076 071 164	4C 47 A4	JMP &A447
83EC	1	165 049	A5 31	LDA &31
83EE	=	197 061	C5 3D	CMP &3D
83F0		208 027	D0 1B	BNE 27 --> &840D
83F2	2	165 050	A5 32	LDA &32
83F4	>	197 062	C5 3E	CMP &3E
83F6		208 021	D0 15	BNE 21 --> &840D
83F8	3	165 051	A5 33	LDA &33
83FA	?	197 063	C5 3F	CMP &3F
83FC		208 015	D0 0F	BNE 15 --> &840D
83FE	4	165 052	A5 34	LDA &34
8400	@	197 064	C5 40	CMP &40
8402		208 009	D0 09	BNE 9 --> &840D
8404	5	165 053	A5 35	LDA &35
8406	A	197 065	C5 41	CMP &41
8408		208 003	D0 03	BNE 3 --> &840D
840A	L	076 180 166	4C B4 A6	JMP &A6B4 Clear FWA
840D	&	176 038	B0 26	BCS 38 --> &8435
840F	;	165 059	A5 3B	LDA &3B
8411	.	133 046	85 2E	STA &2E
8413	8	056	38	SEC
8414	A	165 065	A5 41	LDA &41
8416	5	229 053	E5 35	SBC &35
8418	5	133 053	85 35	STA &35
841A	@	165 064	A5 40	LDA &40
841C	4	229 052	E5 34	SBC &34
841E	4	133 052	85 34	STA &34
8420	?	165 063	A5 3F	LDA &3F
8422	3	229 051	E5 33	SBC &33
8424	3	133 051	85 33	STA &33
8426	>	165 062	A5 3E	LDA &3E
8428	2	229 050	E5 32	SBC &32
842A	2	133 050	85 32	STA &32

842C	=	165 061	A5 3D	LDA &3D	
842E	1	229 049	E5 31	SBC &31	
8430	1	133 049	85 31	STA &31	
8432	L	076 249 129	4C F9 81	JMP &81F9 Normalise FWA#1	

8435: Subtract FWB Mantissa from FWA Mantissa:

8435	8435	5	165 053	A5 35	LDA &35
8437	A	229 065	E5 41	SBC &41	
8439	5	133 053	85 35	STA &35	
843B	4	165 052	A5 34	LDA &34	
843D	@	229 064	E5 40	SBC &40	
843F	4	133 052	85 34	STA &34	
8441	3	165 051	A5 33	LDA &33	
8443	?	229 063	E5 3F	SBC &3F	
8445	3	133 051	85 33	STA &33	
8447	2	165 050	A5 32	LDA &32	
8449	>	229 062	E5 3E	SBC &3E	
844B	2	133 050	85 32	STA &32	
844D	1	165 049	A5 31	LDA &31	
844F	=	229 061	E5 3D	SBC &3D	
8451	1	133 049	85 31	STA &31	
8453	L	076 249 129	4C F9 81	JMP &81F9 Normalise FWA#1	

BASIC IV ROM Routines

Page Last Altered: undefined

Copy FWB to FWA

Submitted by Steve Fewell

Routine: acopyb
Name: Copy FWB to FWA
Starting Address: &8349
Entry criteria: None
Exit: Copies the value of the FWB to the FWA.

Description:
Sets every byte of the FWA to a copy to the corresponding FWB byte as follows:

Byte Description	FWB		FWA
Sign	&3B	--->	&2E
Overflow	00	--->	&2F
Exponent	&3C	--->	&30
Mantissa 1	&3D	--->	&31
Mantissa 2	&3E	--->	&32
Mantissa 3	&3F	--->	&33
Mantissa 4	&40	--->	&34
Mantissa 5 / Rounding	&41	--->	&35

The FWA now holds the same value as the FWB.

Disassembly for the Copy FWB to FWA routine

8349	;	165 059	A5 3B	LDA &3B
834B	.	133 046	85 2E	STA &2E
834D	d/	100 047	64 2F	STZ &2F
834F	<	165 060	A5 3C	LDA &3C
8351	0	133 048	85 30	STA &30
8353	=	165 061	A5 3D	LDA &3D
8355	1	133 049	85 31	STA &31
8357	>	165 062	A5 3E	LDA &3E
8359	2	133 050	85 32	STA &32
835B	?	165 063	A5 3F	LDA &3F
835D	3	133 051	85 33	STA &33
835F	@	165 064	A5 40	LDA &40
8361	4	133 052	85 34	STA &34
8363	A	165 065	A5 41	LDA &41
8365	5	133 053	85 35	STA &35
8367	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Normalise FWA#1 [bit-by-bit & byte-by-byte methods supported]

Submitted by Steve Fewell

Routine: anorm[1]
Name: Normalise Floating-Point number in the FWA [bit-by-bit and byte-by-byte methods supported]
Starting Address: &81F7
Entry criteria: The FWA contains a floating-point number.
Exit: FWA has been normalised (The top bit of the FWA is set, and the exponent adjusted accordingly)

Description:
Unlike the Normalise FWA#2 routine, this routine does not have any special entry requirements (i.e. certain bytes of the number needing to be copied to the A and Y registers).

If ?&31 is negative (the FWA Mantissa's top bit is already set) then exit as the number is already normalised.

If FWA's Mantissa byte 1 (?&31) is zero then OR ?&31 with ?&32, ?&33, ?&34 and ?&35. If the result is zero, then the whole Mantissa (and hence, the whole number) is zero, so jump to &81CC to zero the FWA's Exponent, Exponent Overflow and Sign bytes, and exit.

[Byte-by-Byte method - Normalise one byte at a time]
If FWA's Mantissa byte 1 (?&31) is zero, but the FWA doesn't not contain zero, then Load A with the FWA's Exponent (?&30), and **[&8209]** move the FWA Mantissa byte 2 to the FWA Mantissa Byte 1, move the FWA Mantissa byte 3 to the FWA Mantissa Byte 2, move the FWA Mantissa byte 4 to the FWA Mantissa Byte 3 and move the FWA Mantissa rounding byte to the FWA Mantissa Byte 4. Set the FWA Mantissa Rounding byte to zero. Subtract 8 from the exponent (as we have moved the number

along 8 bits), and decrement the FWA Exponent Overflow (if the Exponent underflowed). If FWA Mantissa Byte 1 is still zero, then we need to normalise by another 8 bits, so jump back to &8209 to move the Mantissa along another byte.
If by moving the Mantissa along byte-by-byte, we now have a normalised number, store A back in the FWA Exponent byte and exit, otherwise jump to &822C to proceed with the bit-by-bit method.

[Bit-by-Bit method - Normalise one bit at a time]
If FWA's Mantissa byte 1 (?&31) is not zero, but the FWA isn't normalised, then Load A with the FWA's Exponent (?&30), and **[&822C]** subtract 1 from the FWA Exponent byte [A], and decrement the FWA Exponent Overflow byte if the result underflowed, and shift the FWA Mantissa left one bit (multiply by 2) so that the bits move from the ?&35 end of the number to the ?&31 end of it. Keep repeating this until the top bit is set.

Now that the number is normalised, store A back in the FWA Exponent Byte (?&30), and exit.

Disassembly for the Normalise FWA#2 routine

81F7	1	165 049	A5 31	LDA &31
81F9	0	048 217	30 D9	BMI -39 --> &81D4 [RTS]
81FB	-	208 045	D0 2D	BNE 45 --> &822A
81FD	2	005 050	05 32	ORA &32
81FF	3	005 051	05 33	ORA &33
8201	4	005 052	05 34	ORA &34
8203	5	005 053	05 35	ORA &35
8205		240 197	F0 C5	BEQ -59 --> &81CC
8207	0	165 048	A5 30	LDA &30
8209	2	164 050	A4 32	LDY &32
820B	1	132 049	84 31	STY &31
820D	3	164 051	A4 33	LDY &33
820F	2	132 050	84 32	STY &32
8211	4	164 052	A4 34	LDY &34
8213	3	132 051	84 33	STY &33
8215	5	164 053	A4 35	LDY &35
8217	4	132 052	84 34	STY &34
8219	d5	100 053	64 35	STZ &35
821B	8	056	38	SEC
821C		233 008	E9 08	SBC#&08

```
821E        176 002   B0 02    BCS 2 --> &8222
8220  /     198 047   C6 2F    DEC &2F
8222  1     164 049   A4 31    LDY &31
8224        240 227   F0 E3    BEQ -29 --> &8209
8226  0     048 023   30 17    BMI 23 --> &823F
8228        128 002   80 02    BRA 2 --> &822C
822A  0     165 048   A5 30    LDA &30
822C        024       18       CLC
822D        233 000   E9 00    SBC#&00
822F        176 002   B0 02    BCS 2 --> &8233
8231  /     198 047   C6 2F    DEC &2F
8233  5     006 053   06 35    ASL &35
8235  &4    038 052   26 34    ROL &34
8237  &3    038 051   26 33    ROL &33
8239  &2    038 050   26 32    ROL &32
823B  &1    038 049   26 31    ROL &31
823D        016 238   10 EE    BPL -18 --> &822D
823F  0     133 048   85 30    STA &30
8241  `     096       60       RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

9EBD '-' operator - Subtraction

Submitted by Steve Fewell

Description:

Check the value type (in A).
If the value is a string then issue a Type Mismatch error, as you cannot subtract Strings in BASIC.

If variable type is negative then the first value is a Float, so jump to &9EE7, which does the following:
* Push FWA to Stack
* Get the second value from the expression handler (level 5)
* If second value is a String then generate a Type Mismatch error.
* If second value is an Integer then convert it to a Float
* Pop the first Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Subtraction routine to subtract the 2nd value from the 1st (&A68A)
* Jump to 9E4F (Expression handler level 4) to check for further + or - operators.

Otherwise, the value is Integer, so push the first value to the Stack and call the Expression Handler level 5 routine (&9FC4) to get the second value to add.
If the second value is a string then generate a Type Mismatch error.

If the second value is a Float then jump to &9EFF to do the following:
* Pop the first value (the Integer) from the Stack
* Push the second value (the Float) to the Stack
* Convert the Integer to a Float (and put it in the FWA).
* Pop the second Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Subtraction routine to subtract the 1st value from the 2nd (&ACC7)
* If the FWA is not zero, then reverse the sign of the FWA (as the subtraction was performed the opposite way than it should have been.
* Jump to 9E4F (Expression handler level 4) to check for further + or - operators.

Otherwise, both values are Integers, so carry out the Integer Addition routine. This routine automatically jumps back to 9E4F on completion, to check for further + or - operators.

Disassembly for the '-' Operator - Subtraction routine

9EBD	168	A8	TAY
9EBE	240 209	F0 D1	BEQ -47 --> &9E91 [JMP &9092 - Type Mismatch error]

9EC0	0%	048 037	30 25	BMI 37 --> &9EE7
9EC2		032 196 159	20 C4 9F	JSR &9FC4 Push Integer to stack & Expression handler level 5
9EC5		168	A8	TAY
9EC6		240 201	F0 C9	BEQ -55 --> &9E91 [JMP &9092 - Type Mismatch error]
9EC8	05	048 053	30 35	BMI 53 --> &9EFF
9ECA				**iminus start:**

Disassembly for the Floating Point '-' Operator

9EE7		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9EEA		032 199 159	20 C7 9F	JSR &9FC7 Expression Handler level 5 (*,/,MOD,DIV)
9EED		168	A8	TAY
9EEE		240 161	F0 A1	BEQ -95 --> &9E91 [JMP &9092 - Type Mismatch error]
9EF0	'	134 039	86 27	STX &27
9EF2	0	048 003	30 03	BMI 3 --> &9EF7
9EF4		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9EF7		032 232 187	20 E8 BB	JSR &BBE8 Pop Float to argp
9EFA		032 138 166	20 8A A6	JSR &A68A Floating-Point Subtraction
9EFD		128 171	80 AB	BRA -85 --> &9EAA
9EFF	'	134 039	86 27	STX &27
9F01		032 230 188	20 E6 BC	JSR &BCE6 Pop Integer from Stack
9F04		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9F07		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9F0A		032 232 187	20 E8 BB	JSR &BBE8 Pop Float to argp
9F0D		032 199 172	20 C7 AC	JSR &ACC7 Floating-Point Subtraction with reverse sign of result
9F10		128 152	80 98	BRA -104 --> &9EAA

Disassembly for the Floating Point Subtraction (with reverse sign of result) routine

ACC7		032 138 166	20 8A A6	JSR &A68A Floating-Point Subtraction
ACCA	1	165 049	A5 31	LDA &31
ACCC		240 006	F0 06	BEQ 6 --> &ACD4
ACCE	.	165 046	A5 2E	LDA &2E
ACD0	I	073 128	49 80	EOR#&80
ACD2	.	133 046	85 2E	STA &2E
ACD4		169 255	A9 FF	LDA#&FF
ACD6	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Integer Subtraction Routine

Submitted by Steve Fewell

Routine: iminus
Name: Integer Subtraction
Starting Address: &9ECA
Entry criteria: ?&4 and ?&5 [The BASIC stack pointer] points to the first Integer variable. The IWA contains the integer to subtract. X = The next operator code (or 4, if there is no further operation).
Exit: IWA contains the result of [Integer variable - IWA]

Description:
Subtracts the IWA from the 32-bit Integer pointed to by the BASIC Stack Pointer (&4 (lo),&5 (hi)) storing the result in the IWA. The least significant byte [&2A] is subtracted first, followed by &2B, &2C and &2D (The most significant byte). The carry flag allows any overflow to be borrowed forward and updated to the next significant byte of the number. The carry flag is set at the beginning because its reset state (0) indicates that a borrow has occurred.

The routine &9E80 in iplus is jumped to. This routine stores the result in A back to &2D and updates the Stack pointer, so that it now points to the address after the added number. To do this, 4 bytes are added to the address pointed to by (&4, &5) and then jumps to &9E4F to test the value of X [the next operator]. If X contains the operator code for '+' or '-' then send the result to the appropriate routine, for further calculation, otherwise exit the routine.

Disassembly for the integer subtraction routine

9ECA	8	056	38	SEC
9ECB		178 004	B2 04	LDA (&04)
9ECD	*	229 042	E5 2A	SBC &2A
9ECF	*	133 042	85 2A	STA &2A
9ED1		160 001	A0 01	LDY#&01
9ED3		177 004	B1 04	LDA (&04),Y
9ED5	+	229 043	E5 2B	SBC &2B
9ED7	+	133 043	85 2B	STA &2B
9ED9		200	C8	INY
9EDA		177 004	B1 04	LDA (&04),Y
9EDC	,	229 044	E5 2C	SBC &2C
9EDE	,	133 044	85 2C	STA &2C
9EE0		200	C8	INY
9EE1		177 004	B1 04	LDA (&04),Y

9EE3	-	229 045	E5 2D	SBC &2D
9EE5		128 153	80 99	BRA -103 --> &9E80

BASIC IV ROM Routines

Page Last Altered: undefined

A68A Floating-Point Subtract Entry Point

Submitted by Steve Fewell

Routine:aminus
Name: Floating-Point Subtraction
Starting Address: &A68A
Entry criteria: &4A and &4B (the argp) point to a 5-byte Floating-Point variable to subtract from the FWA. The FWA contains the number to subtract from.
Exit: The FWA = [argp] - FWA.

Description:

Calls &ACCA to compliment the FWA. Now the FWB and [argp] values can be added, so, continue to the Floating-Point addition entry point at address &A68D.

Disassembly for the Floating-Point Subtract Entry Point routine

A68A	032 202 172	20 CA AC	JSR &ACCA Compliment FWA
A68D			... Floating-Point Addition Entry Point

BASIC IV ROM Routines

Page Last Altered: undefined

9FDB '/' operator - division

Submitted by Steve Fewell

Description:

Check the value type (in A).
If variable type = 0 (String) then Type Mismatch error [96DD].
If variable type is Integer then Convert to Float [96DD].
Push FWA value to Stack.

Call the expression handler level 6 routine, A012, to get the value of the second operand (the divisor).
Check the value type (in A).
If variable type = 0 (String) then Type Mismatch error [96DD].
If variable type is Integer then Convert to Float [96DD].
Pop Dividend (first value) from Stack and set argp to point to it.
Call the Floating-Point Divide routine (A5EE), to divide argp by FWA and store the result in the FWA.

Set A to &FF as we are handling a float value. Store X in &27 and call &9FDB to check for further *,/,MOD or DIV operators.

Note: The '/' Operator always converts Integer values to Float-Point ones.
The '/' Operator does not call the DIV routine if both of the operands are Integer, because the result may not be Integer. The Addition, Subtraction and Multiplication routines need to handle conversion between and use of Integer routines as well as Floating-Point ones, as the Integer routines are quicker, as we know that integer operands will usually give an Integer result (unless multiplication of large numbers is carried out).

Disassembly for the '/' Operator - Division routine

9FDB	168	A8	TAY
9FDC	032 221 150	20 DD 96	JSR &96DD Check Result Type (& If Int, Convert to Float)
9FDF	032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9FE2	032 018 160	20 12 A0	JSR &A012 Expression handler Level 6 [Skip Spaces, ^]
9FE5	' 134 039	86 27	STX &27
9FE7	168	A8	TAY

9FE8	032 221 150	20 DD 96	JSR &96DD Check Result Type (& If Int, Convert to Float)
9FEB	032 232 187	20 E8 BB	JSR &BBE8 Pop Float to argp
9FEE	032 238 165	20 EE A5	JSR &A5EE Floating-Point Division
9FF1	169 255	A9 FF	LDA#&FF
9FF3	128 200	80 C8	BRA -56 --> &9FBD Store X & Check for further *,/,MOD or DIV operators

---

BASIC IV ROM Routines

Page Last Altered: undefined

A5EE Floating-Point Division Entry Point

Submitted by Steve Fewell

Routine:adiv
Name: Floating-Point Division
Starting Address: &A5EE
Entry criteria: &4A and &4B (the argp) point to a 5-byte Floating-Point variable to be divided. The FWA contains the value to divide by.
Exit: The FWA = [argp] / FWA [Normalised and Rounded].

Description:

If the FWA is zero, then issue a Division by Zero error.
Unpack the argp variable to the FWB. If the FWB is zero then exit with aclear [Clear FWA], as the result will be zero. Otherwise, call the divide FWB by FWA division routine [&A5FA].

Disassembly for the Floating-Point Division Entry Point routine

A5EE	1	165 049	A5 31	LDA &31
A5F0		240 240	F0 F0	BEQ -16 --> &A5E2 [JMP &8172] Division by Zero error
A5F2		032 224 164	20 E0 A4	JSR &A4E0 Unpack variable to FWB
A5F5		208 003	D0 03	BNE 3 --> &A5FA Floating-Point Division
A5F7	L	076 180 166	4C B4 A6	JMP &A6B4 Clear FWA

BASIC IV ROM Routines

Page Last Altered: undefined

Floating-Point Division

Submitted by Steve Fewell

Routine: adivb
Name: Floating-Point Division (FWA=FWB/FWA)
Starting Address: &A5FA
Entry criteria: The FWA contains the Divisor and the FWB contains the Dividend.
Exit: The routine evaluates the expression FWB / FWA and puts the resulting Floating-Point number is the FWA.

Description:
Store the sign of the result in the FWA Sign byte [FWB sign EOR FWA sign]. If FWA & FWB signs are different, then the result will be negative, otherwise it will be positive.

Add #&82 to the FWB exponent, to remove the #&80 offset for the exponent. [E.g. 128+130 = 2], rotate the overflow (if FWB was > =1 then there will be an overflow) into the FWA overflow byte (low bit). Subtract the FWA Exponent from the "FWB Exponent + #&82" result to obtain the result's exponent [this automatically adds the #&80 again] store the result as the FWA exponent. Decrement the FWA overflow if a borrow occurred during the subtraction. The overflow considerations cater for very large and very small numbers. The FWB exponent - the FWA exponent gives the unnormalised exponent of the result.

This routine uses the Binary Division method known as "Shift and Subtract".
At a higher level, the routine is doing the following:

* Set the Quotient to 0
* Align the leftmost digits in the dividend and divisor (already done in BASIC!)
* Repeat:
  * If the portion of the dividend above the divisor is greater than or equal to the divisor then (1)Subtract the divisor from that portion of the Dividend and (2) Concatentate 1 to the right hand end of the quotient; Otherwise, concatentate 0 to the right hand end of the quotient.
  * Shift the divisor to the right for one bit.
* Until the dividend is less than the divisor

* The quotient is correct and the dividend is the remainder.

Description of the Mantissa division:
Initialize some variables: Y = 4 (Number of bytes to process); ?&3C = 4 (backup of Y); A = ?&3D (FWB Mantissa byte 1); and X = 8 (Number of bits left in current byte). For speed optimizations, the first byte of the FWB Mantissa is stored in A. The result will be stored in bytes &43 to &46. &3B is temporary storage for the current byte of the Result.

*1) [A622] Compare the FWB Mantissa (bytes 1 to 4) with the FWA Mantissa (stopping when the bytes differ). If the first differing byte in the FWB Mantissa is less than the first differing byte in the FWA Mantissa then [A64F] (FWB < FWA). Multiply the result byte (&3B) by 2 [shift the bits left a place] (this adds a binary 0 to the LSB end of the value) and go on to the next bit (step *3)).

*2) Otherwise, [A638] Subtract the FWA Mantissa (the divisor) from the FWB Mantissa, multiply the result byte (&3B) by 2 [shift the bits left a place] and add 1. (This tags a 1 on to the end of the result). Then go on to the next bit (Step 3).

*3) To move on to the next bit, the bits in the FWB Mantissa are shifted along one position (loosing the most significant bit). X is decremented. [A620] if X is not zero, and the bit that was just lost from the FWB Mantissa was 1, then go to step *2) to subtract and update the result again.

*4) [A620] If X hasn't reached zero then go back to step *1) to process the next bit.

*5) Decrement Y. If Y is less than zero, then jump to step *7) as all of the required bytes have been processed. Otherwise, store the current result byte (&3B), which is now complete in the next result location (that is location &46 for the first byte, &43 for the last byte, as we calculate the result most significant byte first). Next, store Y in &3C [temporary storage during processing of steps *1) to *4)]. Set X to the number of bits that need processing in the next FWB Mantissa byte - for the first 4 bytes, 8 bits are processed, and in the last (least significant) byte, only 2 bits need to be processed. The number of bytes to process is stored in BASIC ROM locations &A5E5 to &A5E8.

*6) [A620] If the bit that was just lost from the FWB Mantissa was 1, then go to step *2) to subtract and update the result again, otherwise go back to step *1) to process the next bit.

*7) Now, all of the bits have been processed, and we have a result, normalise the result [&81F7].
Next exit by rounding the FWA Mantissa to 4 bytes (loosing the rounding byte).

943.34 / 33.33 = 28.303030303030303030303030

FWA = 3333000000 Exponent = 2
FWB = 9433400000 Exponent = 3

*1) FWB > FWA, so do step *2) subtraction ==>
*2) FWB = 91001, Result = Result + 1 = 1
    FWB = 87668, Result = Result + 1 = 2

FWB = 84335, Result = Result + 1 = 3
FWB = 81002, Result = Result + 1 = 4
FWB = 77669, Result = Result + 1 = 5
FWB = 74336, Result = Result + 1 = 6
FWB = 71003, Result = Result + 1 = 7
FWB = 67670, Result = Result + 1 = 8
FWB = 64337, Result = Result + 1 = 9
FWB = 61004, Result = Result + 1 = 10
FWB = 57671, Result = Result + 1 = 11
FWB = 54338, Result = Result + 1 = 12
FWB = 51005, Result = Result + 1 = 13
FWB = 47672, Result = Result + 1 = 14
FWB = 44339, Result = Result + 1 = 15
FWB = 41006, Result = Result + 1 = 16
FWB = 37673, Result = Result + 1 = 17
FWB = 34340, Result = Result + 1 = 18
FWB = 31007, Result = Result + 1 = 19
FWB = 27674, Result = Result + 1 = 20 @
FWB = 24341, Result = Result + 1 = 21
FWB = 21008, Result = Result + 1 = 22
FWB = 17675, Result = Result + 1 = 23
FWB = 14342, Result = Result + 1 = 24
FWB = 11009, Result = Result + 1 = 25
FWB = 07676, Result = Result + 1 = 26
FWB = 04343, Result = Result + 1 = 27
FWB = 01010, Result = Result + 1 = 28
*3) FWB = 1010, Result = Result * 10 = 280
*1) FWB < FWA, So Result = Result + Carry [3] = 283,
*3) FWB = 010, Result = Result * 10 = 2830
*1) FWB < FWA, So Result = Result + Carry [3] = 28303,
*3) FWB = 10, Result = Result * 10 = 283030
*1) FWB < FWA, So Result = Result + Carry [3] = 2830303,
*3) FWB = 0, Result = Result * 10 = 28303030.

Result Exponent = FWB Exponent (3) - FWA Exponent (2) + 1 = 2, So result is 28.30303030

This disassembly also contains code that is not used within Floating-Point division.
Code at &A68A is the call point for Floating-Point Minus
Code at &A68D is the call point for Floating-Point Addition
&A68A just compliments the FWA and then continues with Floating-Point Addition.
&A68D Unpacks the variable pointed to by (&2A, &2B) into the FWB and then, if FWB is not zero, calls the Floating-Point addition routine (&8368).

Disassembly for the Floating-Point Division routine

A5E5	002	02	EQUB &02

```
A5E6        008          08          EQUB &08
A5E7        008          08          EQUB &08
A5E8        008          08          EQUB &08
A5FA  ;     165 059      A5 3B       LDA &3B
A5FC  E.    069 046      45 2E       EOR &2E
A5FE  .     133 046      85 2E       STA &2E
A600  8     056          38          SEC
A601  <     165 060      A5 3C       LDA &3C
A603  i     105 129      69 81       ADC#&81
A605  &/    038 047      26 2F       ROL &2F
A607  0     229 048      E5 30       SBC &30
A609        176 002      B0 02       BCS 2 --> &A60D
A60B  /     198 047      C6 2F       DEC &2F
A60D  0     133 048      85 30       STA &30
A60F        160 004      A0 04       LDY#&04
A611  <     132 060      84 3C       STY &3C
A613  =     165 061      A5 3D       LDA &3D
A615        162 008      A2 08       LDX#&08
A617        128 009      80 09       BRA 9 --> &A622
A619  C     150 067      96 43       STX &43,Y
A61B        190 229 165  BE E5 A5    LDX &A5E5,Y
A61E  <     132 060      84 3C       STY &3C
A620        176 022      B0 16       BCS 22 --> &A638
A622  1     197 049      C5 31       CMP &31
A624        208 016      D0 10       BNE 16 --> &A636
A626  >     164 062      A4 3E       LDY &3E
A628  2     196 050      C4 32       CPY &32
A62A        208 010      D0 0A       BNE 10 --> &A636
A62C  ?     164 063      A4 3F       LDY &3F
A62E  3     196 051      C4 33       CPY &33
A630        208 004      D0 04       BNE 4 --> &A636
A632  @     164 064      A4 40       LDY &40
A634  4     196 052      C4 34       CPY &34
A636        144 023      90 17       BCC 23 --> &A64F
A638        168          A8          TAY
A639  @     165 064      A5 40       LDA &40
A63B  4     229 052      E5 34       SBC &34
A63D  @     133 064      85 40       STA &40
```

```
A63F  ?   165 063   A5 3F     LDA &3F
A641  3   229 051   E5 33     SBC &33
A643  ?   133 063   85 3F     STA &3F
A645  >   165 062   A5 3E     LDA &3E
A647  2   229 050   E5 32     SBC &32
A649  >   133 062   85 3E     STA &3E
A64B      152       98        TYA
A64C  1   229 049   E5 31     SBC &31
A64E  8   056       38        SEC
A64F  &;  038 059   26 3B     ROL &3B
A651  @   006 064   06 40     ASL &40
A653  &?  038 063   26 3F     ROL &3F
A655  &>  038 062   26 3E     ROL &3E
A657  *   042       2A        ROL A
A658      202       CA        DEX
A659      208 197   D0 C5     BNE -59 --> &A620
A65B  ;   166 059   A6 3B     LDX &3B
A65D  <   164 060   A4 3C     LDY &3C
A65F      136       88        DEY
A660      016 183   10 B7     BPL -73 --> &A619
A662  >   005 062   05 3E     ORA &3E
A664  ?   005 063   05 3F     ORA &3F
A666  @   005 064   05 40     ORA &40
A668      240 001   F0 01     BEQ 1 --> &A66B
A66A  8   056       38        SEC
A66B      138       8A        TXA
A66C  j   106       6A        ROR A
A66D  j   106       6A        ROR A
A66E  j   106       6A        ROR A
A66F  )   041 224   29 E0     AND#&E0
A671  5   133 053   85 35     STA &35
A673  C   165 067   A5 43     LDA &43
A675  4   133 052   85 34     STA &34
A677  D   165 068   A5 44     LDA &44
A679  3   133 051   85 33     STA &33
A67B  E   165 069   A5 45     LDA &45
A67D  2   133 050   85 32     STA &32
A67F  F   165 070   A5 46     LDA &46
```

A681	1	133 049	85 31	STA &31
A683	0	048 016	30 10	BMI 16 --> &A695
A685	*	032 042 130	20 2A 82	JSR &822A part of Normalise FWA
A688		128 011	80 0B	BRA 11 --> &A695 Round FWA Mantissa to 4 bytes

---

BASIC IV ROM Routines

Page Last Altered: undefined

Integer Multiplication Routine

Submitted by Steve Fewell

Routine: imult
Name: Integer Multiplication
Starting Address: &9F64
Entry criteria: IWA contains the first Integer number. Y contains the Most Significant Byte of this Integer number (?&2D). ?&4 and ?&5 point to the second Integer variable. ?&27 = 4, X = &4.
Notes: If the IWA > &FFFF, it is truncated to 16 bits.
Exit: IWA contains the result of [Integer variable * IWA]

Description:
Exclusive-OR the Most significant bytes of both integers together, this determines the sign of the result (if only one of the MSBs is negative then the result is negative, otherwise it is positive).

Make both integers positive [using ipos] (so that we only have to do positive multiplication).

Store the multiplier in zero-page locations (&39 to &3C) and put the multiplicand into the IWA [popi].

Initialise the result to zero (The result is stored in Y [&3D], X [&3E], &3F and &40).

Keep repeating the following loop until the multiplier is zero [only the least significant two bytes of the multiplier are used]:

(1) Divide the multiplier by 2 (and make the result integer)

(2) If an integer result was not obtained, (i.e. Multiplier was an odd number before division) then add the multiplicand to the result. [Dividing the multiplier lost 0.5 of its value, so we add on the multiplicand (which at this stage is worth 0.5 * the actual multiplicand) before it is multiplied to perform the multiplication of the lost 0.5 of the multiplier].

(3) Multiply the multiplicand by 2

Until multiplier is zero ((&39 OR &3A) = zero).

Retrieve the result from the zero page locations (&3D to &40) and make it negative if the sign [at the beginning] was determined to be negative.

Jump to &9FCA to test the value of X [the next operator] when the routine was entered. If X contains the operator code for '*', '/', 'MOD' or 'DIV' then send the result to the appropriate routine, for further calculation, otherwise exit the multiplication routine.

Disassembly for the integer multiplication routine

9F64	Z	90	5A	PHY
9F65		032 190 172	20 BE AC	JSR &ACBE Make IWA value positive [ipos]
9F68	'	134 039	86 27	STX &27
9F6A	9	162 057	A2 39	LDX#&39
9F6C		032 198 189	20 C6 BD	JSR &BDC6 Save Integer (IWA) to zero page location [izpout]
9F6F		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from Stack [popi]
9F72	h	104	68	PLA
9F73	E-	069 045	45 2D	EOR &2D
9F75	7	133 055	85 37	STA &37
9F77		032 190 172	20 BE AC	JSR &ACBE Make the IWA value positive [ipos]
9F7A		160 000	A0 00	LDY#&00
9F7C		162 000	A2 00	LDX#&00
9F7E	d?	100 063	64 3F	STZ &3F
9F80	d@	100 064	64 40	STZ &40
9F82	F:	070 058	46 3A	LSR &3A
9F84	f9	102 057	66 39	ROR &39
9F86		144 021	90 15	BCC 21 --> &9F9D

9F88		24	18	CLC
9F89		152	98	TYA
9F8A	e*	101 042	65 2A	ADC &2A
9F8C		168	A8	TAY
9F8D		138	8A	TXA
9F8E	e+	101 043	65 2B	ADC &2B
9F90		170	AA	TAX
9F91	?	165 063	A5 3F	LDA &3F
9F93	e,	101 044	65 2C	ADC &2C
9F95	?	133 063	85 3F	STA &3F
9F97	@	165 064	A5 40	LDA &40
9F99	e-	101 045	65 2D	ADC &2D
9F9B	@	133 064	85 40	STA &40
9F9D	*	006 042	06 2A	ASL &2A
9F9F	&+	038 043	26 2B	ROL &2B
9FA1	&,	038 044	26 2C	ROL &2C
9FA3	&-	038 045	26 2D	ROL &2D
9FA5	9	165 057	A5 39	LDA &39
9FA7	:	005 058	05 3A	ORA &3A
9FA9		208 215	D0 D7	BNE -41 --> &9F82
9FAB	=	132 061	84 3D	STY &3D
9FAD	>	134 062	86 3E	STX &3E
9FAF	7	165 055	A5 37	LDA &37
9FB1		8	8	PHP
9FB2	=	162 061	A2 3D	LDX#&3D
9FB4		032 128 170	20 80 AA	JSR &AA80 Set the IWA to an Integer value at a zero-page location [izpin]
9FB7	(	40	28	PLP
9FB8		016 003	10 03	BPL 3 --> &9FBD
9FBA		032 222 172	20 DE AC	JSR &ACDE icomp
9FBD	'	166 039	A6 27	LDX &27
9FBF		128 009	80 09	BRA 9 --> &9FCA
9FC1	L;	076 059 159	4C 3B 9F	JMP &9F3B '*' Operator
9FC4	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]

9FC7	032 018 160	20 12 A0	JSR &A012
9FCA *	224 042	E0 2A	CPX#&2A
9FCC	240 243	F0 F3	BEQ -13 --> &9FC1
9FCE /	224 047	E0 2F	CPX#&2F
9FD0	240 009	F0 09	BEQ 9 --> &9FDB '/' Operator
9FD2	224 131	E0 83	CPX#&83
9FD4	240 031	F0 1F	BEQ 31 --> &9FF5 Integer MOD routine
9FD6	224 129	E0 81	CPX#&81
9FD8 #	240 035	F0 23	BEQ 35 --> &9FFD Integer DIV routine
9FDA `	96	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Integer Positive Routine

Submitted by Steve Fewell

Routine: ipos
Name: Integer Positive
Starting Address: &ACBE
Entry criteria: The IWA contains the integer to make positive.
Exit: If the IWA contained a negative value, it will now be positive, otherwise IWA will be unchanged.

Description:
Check the most significant byte of the Integer Number in the IWA. If bit 7 of this byte is set [N flag], the IWA contains a negative number, so the icomp routine is jumped to (to compliment the value).
Otherwise the routine exists with A = #&40, indicating that an Integer number is being processed. [Note: The code for this is located in the icomp routine].

Disassembly for the integer positive routine

ACBE	$-	036 045	24 2D	BIT &2D
ACC0	0	048 028	30 1C	BMI 28 --> &ACDE Integer Compliment
ACC2	1	128 049	80 31	BRA 49 --> &ACF5 [LDA#&40 : RTS]

BASIC IV ROM Routines

Page Last Altered: undefined

Save Integer to Zero Page Address

Submitted by Steve Fewell

Routine: izpout
Name: Save Integer to Zero Page Address
Starting Address: &BDC6
Entry criteria: X is the Zero page address of the first of 4 sequential bytes of memory in which to store a 32-bit Integer variable.
Exit: The zero page locations pointed to by X (That is locations 00,X to 03,X) now contain the 32-bit Integer from the IWA.

Description:
The contents of the IWA are copied to the zero page memory locations pointed to by the X register (The number is stored least significant byte first, one byte is copied at a time). The routine sets A to #&40 to indicate that an Integer number is being processed before exiting.

There is another routine which saves the IWA to any address, but we need this zero page routine because it executing a lot faster than the one which allows longer addresses.

Disassembly for the Save Integer to Zero Page Address routine

BDC6	*	165 042	A5 2A	LDA &2A
BDC8		149 000	95 00	STA &00,X
BDCA	+	165 043	A5 2B	LDA &2B
BDCC		149 001	95 01	STA &01,X

BDCE	,	165 044	A5 2C	LDA &2C
BDD0		149 002	95 02	STA &02,X
BDD2	-	165 045	A5 2D	LDA &2D
BDD4		149 003	95 03	STA &03,X
BDD6	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Load IWA with Integer from Zero Page Address

Submitted by Steve Fewell

Routine: izpin
Name: Load IWA with Integer from Zero Page Address
Starting Address: &AA80
Entry criteria: X is the Zero page address of the Least Significant (first) byte of a 32-bit Integer variable
Exit: IWA contains the Integer number pointed to by X (That is the 32-bit Integer stored in locations 00, X to 03,X).

Description:
Loads the IWA with the 32-bit Integer pointed to by the X register (The number is stored least significant byte first, one byte is copied at a time), and sets A to #&40 to show that an Integer number is being processed.

There is another routine which loads the IWA from any address, but we need this zero page routine due to it executing a lot faster than the one which allows longer addresses.

Disassembly for the Load Integer from Zero Page Address routine

AA80		181 000	B5 00	LDA &00,X
AA82	*	133 042	85 2A	STA &2A
AA84		181 001	B5 01	LDA &01,X
AA86	+	133 043	85 2B	STA &2B
AA88		181 002	B5 02	LDA &02,X

```
AA8A  ,   133 044    85 2C     STA &2C
AA8C      181 003    B5 03     LDA &03,X
AA8E  -   133 045    85 2D     STA &2D
AA90  @   169 064    A9 40     LDA#&40
AA92  `   096        60        RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

9F3B '*' operator - multiplication

Submitted by Steve Fewell

Description:

[9F3B:] Check the value type (in A).
If variable type = 0 then issue a Type Mismatch error.

If variable type is negative then the first value is a Float, so jump to &9F23, which
does the following:
* Push FWA to Stack
* Get the second value from the expression handler (level 6) [A012]
* If second value is a String then generate a Type Mismatch error. [96DD]
* If second value is an Integer then convert it to a Float [96DD]
* Pop the first Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Multiplication routine to multiply the values (&A6A6)
* Set A = #&FF, as we have a Floating-Point result.
* Jump to 9FCA (Expression handler level 5) to check for further *,/,MOD or DIV operators.

Otherwise, the value is Integer.
If the IWA byte 4 or IWA byte 3 is zero or the top bit of IWA byte 2 is set,
then we are dealing with a large number, so we will need to use Floating-Point multiplication
(as the result will most likely not be an integer), so call 9F20. [Note: What happens if the IWA
contains &10FFFFFF, it looks like Floating-Point multiplication will not be used for this!].
9F20 does the following:
* Convert the Integer in the IWA to a Float (FWA).
* Push FWA to Stack
* Get the second value from the expression handler (level 6) [A012]
* If second value is a String then generate a Type Mismatch error. [96DD]
* If second value is an Integer then convert it to a Float [96DD]
* Pop the first Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Multiplication routine to multiply the values (&A6A6)
* Set A = #&FF, as we have a Floating-Point result.
* Jump to 9FCA (Expression handler level 5) to check for further *,/,MOD or DIV operators.

Otherwise, the first value (multiplicand) is an integer which isn't too large,
so push the first value to the stack and call the Expression Handler level 6 routine
(&A00F) to get the second value to multiply (the multiplier).
If the second value is a string then generate a Type Mismatch error.

If the second value is a Float then jump to &9F15 to do the following:
* Pop the first value (the Integer) from the Stack

* Push the second value (the Float) to the Stack
* Convert the Integer to a Float (and put it in the FWA).
* Pop the second Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Multiplication routine to multiply the values (&A6A6)
* Set A = #&FF, as we have a Floating-Point result.
* Jump to 9FCA (Expression handler level 5) to check for further *,/,MOD or DIV operators.

Otherwise, check whether the second value is too large for Integer multiplication routine (it's too large if IWA byte 3 or 4 is not 0, or top bit of byte 2 is 1).
If the second value is too large for Integer Multiplication then jump to &9F12 to do the following:
* Convert the second value (which is Integer) to a Float
* Pop the first value (also an Integer) from the Stack
* Push the second value (the Float) to the Stack
* Convert the Integer to a Float (and put it in the FWA).
* Pop the second Float value from Stack to argp (&4A, &4B)
* Call the Floating-Point Multiplication routine to multiply the values (&A6A6)
* Set A = #&FF, as we have a Floating-Point result.
* Jump to 9FCA (Expression handler level 5) to check for further *,/,MOD or DIV operators.

Otherwise, both Integers are in the required range, so do the Integer Multiplication routine (&9F64).
This routine automatically jumps back to 9FCA on completion, to check for further *,/, MOD or DIV operators.

Disassembly for the '*' Operator - Multiplication routine

9F12		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9F15		032 230 188	20 E6 BC	JSR &BCE6 Pop Integer from Stack
9F18		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9F1B		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9F1E		128 013	80 0D	BRA 13 --> &9F2D
9F20		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
9F23		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
9F26		032 018 160	20 12 A0	JSR &A012 Expression handler Level 6 [Skip Spaces, ^]
9F29		168	A8	TAY
9F2A		032 221 150	20 DD 96	JSR &96DD Check Result Type (& If Int, Convert to Float)
9F2D		032 232 187	20 E8 BB	JSR &BBE8 Pop Float to argp
9F30		032 166 166	20 A6 A6	JSR &A6A6 Floating-Point multiplication
9F33		169 255	A9 FF	LDA#&FF
9F35	L	076 202 159	4C CA 9F	JMP &9FCA Check for further *,/,MOD or DIV operators
9F38	L	076 146 144	4C 92 90	JMP &9092 Type Mismatch error
9F3B		168	A8	TAY
9F3C		240 250	F0 FA	BEQ -6 --> &9F38
9F3E	0	048 227	30 E3	BMI -29 --> &9F23
9F40	-	164 045	A4 2D	LDY &2D
9F42	,	196 044	C4 2C	CPY &2C
9F44		208 218	D0 DA	BNE -38 --> &9F20
9F46	+	165 043	A5 2B	LDA &2B

9F48		010	0A	ASL A
9F49		152	98	TYA
9F4A	i	105 000	69 00	ADC#&00
9F4C		208 210	D0 D2	BNE -46 --> &9F20
9F4E		032 015 160	20 0F A0	JSR &A00F Push Integer & Expression handler Level 6 [^]
9F51		168	A8	TAY
9F52		240 228	F0 E4	BEQ -28 --> &9F38
9F54	0	048 191	30 BF	BMI -65 --> &9F15
9F56	-	164 045	A4 2D	LDY &2D
9F58	,	196 044	C4 2C	CPY &2C
9F5A		208 182	D0 B6	BNE -74 --> &9F12
9F5C	+	165 043	A5 2B	LDA &2B
9F5E		010	0A	ASL A
9F5F		152	98	TYA
9F60	i	105 000	69 00	ADC#&00
9F62		208 174	D0 AE	BNE -82 --> &9F12
9F64				...imult Integer Multiplication

BASIC IV ROM Routines

Page Last Altered: undefined

A6A6 Floating-Point Multiply Entry Point

Submitted by Steve Fewell

Routine:amult
Name: Floating-Point Multiplication
Starting Address: &A6A6
Entry criteria: &4A and &4B (the argp) point to a 5-byte Floating-Point variable to multiply.
The FWA contains the other value to multiply [multiplicand].
Exit: The FWA = [argp] * FWA [Normalised and Rounded].

Description:

Call the Floating Point Multiplication routine [FWA = Argp Variable * FWB] to actually do the addition.
Jump to the 'Round FWA Mantissa to 4 bytes' routine at address &A695
so that we can exit with the resulted rounded and the Mantissa rounding byte clear.

Disassembly for the Floating-Point Multiply Entry Point routine

A6A6	032 207 166	20 CF A6	JSR &A6CF Floating-Point Multiplication
A6A9	128 234	80 EA	BRA -22 --> &A695

BASIC IV ROM Routines

Page Last Altered: undefined

A6CF Floating-Point Multiplication

Submitted by Steve Fewell

Routine:fpmult
Name: Floating-Point Multiplication
Starting Address: &A6CF
Entry criteria: &4A and &4B (the argp) point to a 5-byte Floating-Point variable to multiply the FWB by.
Exit: The FWA contains the normalised result [FWA=argp variable * FWA].

Description:

If FWA is zero, then the result is zero, so just exit.

Unpack the argp variable to the FWB.
If the FWB is zero then call FWA clear routine to return with a zero result.

Add the FWB exponent to the FWA exponent and subtract #&7F (to account for the #&80 offset)
The FWA overflow byte will contain any overflow that occurred.
This gives the result's exponent value, so store it in the FWA exponent.

EOR the FWB sign with the FWA sign byte to obtain the sign of the result and store result's
sign in the FWA sign byte. (i.e. neg_num * neg_num = pos_num, pos_num * pos_num = pos_num
neg_num * pos_num = neg_num, & pos_num * neg_num = neg_num).

Store X on the stack during the routine (as X contains the next operator?).
Move the FWA Mantissa to locations &42 to &45 (in reverse order), and clear out
FWA mantissa value (to zero). (i.e. &45 = &31; &44 = &32; &43 = &33; &42 = &34).
Zero The FWB Exponent, the FWB Sign and location &3A (these bytes form the leading zeros
during the multiplication.

(A735): Load the next byte (&42 for the first time, as Y = 0 and X = &FC; so load LSB of Mantissa first).
If the byte is not zero then perform the multiply operation with the byte (A705).

Repeat for all bytes (&42 to &45).
When complete, store any rounding in the FWA Rounding Byte, then if the FWA Mantissa (Result) Byte 1 top bit is 0 then exit via the normalise FWA routine.
Otherwise, just exit as the number is already normalised.

(A705): Performing the multiplication for the current byte:
X is saved to the stack during this routine and retrived at the end, this is so that the value of X - which is used in the byte loop (above) - is not overwritten.

The multiplication consists of the following steps:
1) Divide the FWB Mantissa by 2 [bytes &3D to &41]
2) Shift the top bit out of the FWA byte we are looking at (Either: &42, &43, &44 or &45, as the FWA Mantissa has been moved to these temporary locations).
--> If the top bit shifted out was 0 then look at the next byte (4).
--> If the top bit shifted out was 1 then add the FWB Mantissa to the Result's Mantissa (FWA)(?).
Add the corresponding FWB byte to Y [rounding value]. (?)
3) look at the next byte. When the top bit of all 4 bytes is processed then go back to A735 to work with the next byte.

Example:
FWA = 718.456 [exp = 8A;Man 1 = B3; Man 2 = 9D; Man 3 = 2F; Man 4 = 1B]
FWB = 47.1173 [exp = 86;Man 1 = BC; Man 2 = 78; Man 3 = 1D; Man 4 = 7E]
Result should be 33851.7068888.

Add exponents: 8A + 86 - 7F = 91.

Move FWA to &45 to &42 and zero some fields, So:
?&45 = &B3 [Temp FWA Mantissa 1]
?&44 = &9D [Temp FWA Mantissa 2]
?&43 = &2F [Temp FWA Mantissa 3]
?&42 = &1B [Temp FWA Mantissa 4]
?&41 = &00 [FWB Rounding]
?&40 = &7E [FWB Mantissa 4]
?&3F = &1D [FWB Mantissa 3]
?&3E = &78 [FWB Mantissa 2]
?&3D = &BC [FWB Mantissa 1]
?&3C = &00
?&3B = &00
?&3A = &00

Process next byte [A735]
Y = 0
X = &FC
A = &1B [?&42], not zero so A705.

Process byte ?&42 [A705]:
Divide FWB by 2: [FWB ?&3D = 5E;?&3E = 3C;?&3F = 0E;?&40 = BF;?&41 = 00]
Multiply Byte (&46,X) by 2 [?&42] = &1B * 2 = &36

Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &2F * 2 = &5E
Top bit wasn't set, so jump to next byte [A731]
A731: X = X + 1 = &FE.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &9D * 2 = &3A
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&40]. Y = &BF.
?&34 = ?&34 + &41,X [&3F] = &00 + &0E = &0E
?&33 = ?&33 + &40,X [&3E] = &00 + &3C = &3C
?&32 = ?&32 + &3F,X [&3D] = &00 + &5E = &5E
?&31 = ?&31 + &3E,X [&3C] = &00 + &00 = &00
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&45] = &B3 * 2 = &66
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&41]. Y = &BF + &00 = &BF.
?&34 = ?&34 + &41,X [&40] = &0E + &BF = &CD
?&33 = ?&33 + &40,X [&3F] = &3C + &0E = &4A
?&32 = ?&32 + &3F,X [&3E] = &5E + &3C = &9A
?&31 = ?&31 + &3E,X [&3D] = &00 + &5E = &5E
A731: X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

Process next byte [A735]
Y = &BF
X = &FC
A = &36 [?&42], not zero so A705.

Process byte ?&42 [A705]:
Divide FWB by 2: [FWB ?&3D = 2F;?&3E = 1E;?&3F = 07;?&40 = 5F;?&41 = 80]
Multiply Byte (&46,X) by 2 [?&42] = &36 * 2 = &6C
Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &5E * 2 = &BC
Top bit wasn't set, so jump to next byte [A731]
A731: X = X + 1 = &FE.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &3A * 2 = &74
Top bit wasn't set, so jump to next byte [A731]
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710

A710: Multiply Byte (&46,X) by 2 [?&45] = &66 * 2 = &CC
Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

Process next byte [A735]
Y = &BF
X = &FC
A = &6C [?&42], not zero so A705.

Process byte ?&42 [A705]:
Divide FWB by 2: [FWB ?&3D = 17;?&3E = 8F;?&3F = 03;?&40 = AF;?&41 = C0]
Multiply Byte (&46,X) by 2 [?&42] = &6C * 2 = &D8
Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &BC * 2 = &78
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3F = 03]. Y = &C2.
?&34 = ?&34 + &41,X [&3E] = &CD + &8F + 1 = &5D
?&33 = ?&33 + &40,X [&3D] = &4A + &17 + 1 = &62
?&32 = ?&32 + &3F,X [&3C] = &9A + &00 = &9A
?&31 = ?&31 + &3E,X [&3B] = &5E + &00 = &5E
A731: X = X + 1 = &FE.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &74 * 2 = &E8
Top bit wasn't set, so jump to next byte [A731]
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&45] = &CC * 2 = &98
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&41 = C0]. Y = &82.
?&34 = ?&34 + &41,X [&40] = &5D + &AF + 1 = &0D
?&33 = ?&33 + &40,X [&3F] = &62 + &03 + 1 = &66
?&32 = ?&32 + &3F,X [&3E] = &9A + &8F = &29
?&31 = ?&31 + &3E,X [&3D] = &5E + &17 + 1 = &76
A731:
X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

Process next byte [A735]
Y = &03
X = &FC
A = &D8 [?&42], not zero so A705.

Process byte ?&42 [A705]:

Divide FWB by 2: [FWB ?&3D = 0B;?&3E = C7;?&3F = 81;?&40 = D7;?&41 = E0]
Multiply Byte (&46,X) by 2 [?&42] = &D8 * 2 = &B0
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3E = C7]. Y = &49.
?&34 = ?&34 + &41,X [&3D] = &0D + &0B + 1 = &19
?&33 = ?&33 + &40,X [&3C] = &66 + &00 = &66
?&32 = ?&32 + &3F,X [&3B] = &29 + &00 = &29
?&31 = ?&31 + &3E,X [&3A] = &76 + &00 = &76
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &78 * 2 = &F0
Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &FE.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &E8 * 2 = &D0
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&40 = D7]. Y = &20.
?&34 = ?&34 + &41,X [&3F] = &19 + &81 + 1 = &9B
?&33 = ?&33 + &40,X [&3E] = &66 + &C7 = &2D
?&32 = ?&32 + &3F,X [&3D] = &29 + &0B + 1 = &35
?&31 = ?&31 + &3E,X [&3C] = &76 + &00 = &76
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&45] = &98 * 2 = &30
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&41 = E0]. Y = &00.
?&34 = ?&34 + &41,X [&40] = &9B + &D7 + 1 = &73
?&33 = ?&33 + &40,X [&3F] = &2D + &81 + 1 = &AF
?&32 = ?&32 + &3F,X [&3E] = &35 + &C7 = &FC
?&31 = ?&31 + &3E,X [&3D] = &76 + &0B = &81
A731:
X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

Process next byte [A735]
Y = &00
X = &FC
A = &B0 [?&42], not zero so A705.

Process byte ?&42 [A705]:
Divide FWB by 2: [FWB ?&3D = 05;?&3E = E3;?&3F = C0;?&40 = EB;?&41 = F0]
Multiply Byte (&46,X) by 2 [?&42] = &B0 * 2 = &60 (Carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3E = E3]. Y = &E3.
?&34 = ?&34 + &41,X [&3D] = &73 + &05 + 0 = &78

?&33 = ?&33 + &40,X [&3C] = &AF + &00 = &AF
?&32 = ?&32 + &3F,X [&3B] = &FC + &00 = &FC
?&31 = ?&31 + &3E,X [&3A] = &81 + &00 = &81
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &F0 * 2 = &E0
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3F = C0]. Y = &A3.
?&34 = ?&34 + &41,X [&3E] = &78 + &E3 + 1 = &5C
?&33 = ?&33 + &40,X [&3D] = &AF + &05 + 1 = &B5
?&32 = ?&32 + &3F,X [&3C] = &FC + &00 = &FC
?&31 = ?&31 + &3E,X [&3B] = &81 + &00 = &81
A731:
X = X + 1 = &FE.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &D0 * 2 = &A0
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&40 = EB]. Y = &8E.
?&34 = ?&34 + &41,X [&3F] = &5C + &C0 + 1 = &1D
?&33 = ?&33 + &40,X [&3E] = &B5 + &E3 + 1 = &99
?&32 = ?&32 + &3F,X [&3D] = &FC + &05 + 1 = &02
?&31 = ?&31 + &3E,X [&3C] = &81 + &00 + 1 = &82
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&45] = &30 * 2 = &60
Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

Process next byte [A735]
Y = &8E
X = &FC
A = &60 [?&42], not zero so A705.

Process byte ?&42 [A705]:
Divide FWB by 2: [FWB ?&3D = 02;?&3E = F1;?&3F = E0;?&40 = 75;?&41 = F8]
Multiply Byte (&46,X) by 2 [?&42] = &60 * 2 = &C0 (Carry 0)
Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &E0 * 2 = &C0 (Carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3F = E0]. Y = &6E.
?&34 = ?&34 + &41,X [&3E] = &1D + &F1 + 1 = &0F

?&33 = ?&33 + &40,X [&3D] = &99 + &02 + 1 = &9C
?&32 = ?&32 + &3F,X [&3C] = &02 + &00 = &02
?&31 = ?&31 + &3E,X [&3B] = &82 + &00 = &82
A731:
X = X + 1 = &FE.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &A0 * 2 = &40
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&40 = 75]. Y = &E3.
?&34 = ?&34 + &41,X [&3F] = &0F + &E0 + 0 = &EF
?&33 = ?&33 + &40,X [&3E] = &9C + &F1 + 0 = &8D
?&32 = ?&32 + &3F,X [&3D] = &02 + &02 + 1 = &05
?&31 = ?&31 + &3E,X [&3C] = &82 + &00 + 0 = &82
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&45] = &60 * 2 = &C0
Top bit wasn't set, so jump to next byte [A731]
A731:
X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

Process next byte [A735]
Y = &E3
X = &FC
A = &C0 [?&42], not zero so A705.

Process byte ?&42 [A705]:
Divide FWB by 2: [FWB ?&3D = 01;?&3E = 78;?&3F = F0;?&40 = 3A;?&41 = FC]
Multiply Byte (&46,X) by 2 [?&42] = &C0 * 2 = &80 (Carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3E = 78]. Y = &5B.
?&34 = ?&34 + &41,X [&3D] = &EF + &01 + 1 = &F1
?&33 = ?&33 + &40,X [&3C] = &8D + &00 + 0 = &8D
?&32 = ?&32 + &3F,X [&3B] = &05 + &00 = &05
?&31 = ?&31 + &3E,X [&3A] = &82 + &00 = &82
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &C0 * 2 = &80 (Carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3F = F0]. Y = &4B.
?&34 = ?&34 + &41,X [&3E] = &F1 + &78 + 1 = &6A
?&33 = ?&33 + &40,X [&3D] = &8D + &01 + 1 = &8F
?&32 = ?&32 + &3F,X [&3C] = &05 + &00 = &05
?&31 = ?&31 + &3E,X [&3B] = &82 + &00 = &82
A731:
X = X + 1 = &FE.

X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &40 * 2 = &80
Top bit wasn't set, so jump to next byte [A731]
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&45] = &C0 * 2 = &80
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&41 = FC]. Y = &47.
?&34 = ?&34 + &41,X [&40] = &6A + &3A + 1 = &A5
?&33 = ?&33 + &40,X [&3F] = &8F + &F0 + 0 = &7F
?&32 = ?&32 + &3F,X [&3E] = &05 + &78 + 1 = &7E
?&31 = ?&31 + &3E,X [&3D] = &82 + &01 + 0 = &83
A731:
X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

Process next byte [A735]
Y = &09
X = &FC
A = &80 [?&42], not zero so A705.

Process byte ?&42 [A705]:
Divide FWB by 2: [FWB ?&3D = 00;?&3E = BC;?&3F = 78;?&40 = 1D;?&41 = 7E]
Multiply Byte (&46,X) by 2 [?&42] = &80 * 2 = &00 (Carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3E = BC]. Y = &03.
?&34 = ?&34 + &41,X [&3D] = &A5 + &00 + 1 = &A6
?&33 = ?&33 + &40,X [&3C] = &7F + &00 + 0 = &7F
?&32 = ?&32 + &3F,X [&3B] = &7E + &00 = &7E
?&31 = ?&31 + &3E,X [&3A] = &83 + &00 = &83
A731:
X = X + 1 = &FD.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&43] = &80 * 2 = &00 (Carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&3F = 78]. Y = &7B.
?&34 = ?&34 + &41,X [&3E] = &A6 + &BC + 0 = &62
?&33 = ?&33 + &40,X [&3D] = &7F + &00 + 1 = &80
?&32 = ?&32 + &3F,X [&3C] = &7E + &00 = &7E
?&31 = ?&31 + &3E,X [&3B] = &83 + &00 = &83
A731:
X = X + 1 = &FE.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&44] = &80 * 2 = &00 (carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&40 = 1D]. Y = &98.
?&34 = ?&34 + &41,X [&3F] = &62 + &78 + 0 = &DA

?&33 = ?&33 + &40,X [&3E] = &80 + &BC + 0 = &3C
?&32 = ?&32 + &3F,X [&3D] = &7E + &00 + 1 = &7F
?&31 = ?&31 + &3E,X [&3C] = &83 + &00 + 0 = &83
A731: X = X + 1 = &FF.
X not 0 yet so jump to A710
A710: Multiply Byte (&46,X) by 2 [?&45] = &80 * 2 = &00 (carry 1)
Top bit was set, so Add FWB to result:
Y = Y + ?&42,X [?&41 = 7E]. Y = &16.
?&34 = ?&34 + &41,X [&40] = &DA + &1D + 1 = &F8
?&33 = ?&33 + &40,X [&3F] = &3C + &78 + 0 = &B4
?&32 = ?&32 + &3F,X [&3E] = &7F + &BC + 0 = &3B
?&31 = ?&31 + &3E,X [&3D] = &83 + &00 + 1 = &84
A731:
X = X + 1 = &00.
X is 0, so X = &FC (old value from stack)

?&45, ?&44, ?&43 and ?&42 are all now zero, so we have our result in FWA: Exponent = 90 (after normalisation)
Mantissa 1 = &4 (minus &80 offset)
Mantissa 2 = &3B
Mantissa 3 = &B4
Mantissa 4 = &F8
Mantissa 5 = &16 (set from Y)
Decimal value = 33851.7069

I think this routine uses the "Binary Multiply - Repeated Shift and Add" method for multiplication. This method is described below:
* Result = 0
* Repeat
* Shift 2nd multiplicand left until rightmost digit is lined up with leftmost 1 in the first multiplicand.
* Add 2nd multiplicand in that position to the result.
* Remove the 1 from the 1st multiplicand (done when the value is shifted left).
* Until 1st multiplicand is zero * the result has now been obtained

Disassembly for the Floating-Point Multiplication routine

A6CF	1	165 049	A5 31	LDA &31
A6D1		240 241	F0 F1	BEQ -15 --> &A6C4 [RTS]
A6D3		032 224 164	20 E0 A4	JSR &A4E0 Unpack (&4A, &4B) var to FWB
A6D6		240 220	F0 DC	BEQ -36 --> &A6B4 Clear FWA
A6D8		024	18	CLC
A6D9	0	165 048	A5 30	LDA &30

A6DB	e<	101 060	65 3C	ADC &3C
A6DD	&/	038 047	26 2F	ROL &2F
A6DF		233 127	E9 7F	SBC#&7F
A6E1	0	133 048	85 30	STA &30
A6E3		176 002	B0 02	BCS 2 --> &A6E7
A6E5	/	198 047	C6 2F	DEC &2F
A6E7	.	165 046	A5 2E	LDA &2E
A6E9	E;	069 059	45 3B	EOR &3B
A6EB	.	133 046	85 2E	STA &2E
A6ED		218	DA	PHX
A6EE		162 248	A2 F8	LDX#&F8
A6F0		160 004	A0 04	LDY#&04
A6F2	9	181 057	B5 39	LDA &39,X
A6F4	t9	116 057	74 39	STZ &39,X
A6F6	A	153 065 000	99 41 00	STA &0041,Y
A6F9		232	E8	INX
A6FA		136	88	DEY
A6FB		208 245	D0 F5	BNE -11 --> &A6F2
A6FD	d<	100 060	64 3C	STZ &3C
A6FF	d;	100 059	64 3B	STZ &3B
A701	d:	100 058	64 3A	STZ &3A
A703	0	128 048	80 30	BRA 48 --> &A735
A705		218	DA	PHX
A706	F=	070 061	46 3D	LSR &3D
A708	f>	102 062	66 3E	ROR &3E
A70A	f?	102 063	66 3F	ROR &3F
A70C	f@	102 064	66 40	ROR &40
A70E	fA	102 065	66 41	ROR &41
A710	F	022 070	16 46	ASL &46,X
A712		144 029	90 1D	BCC 29 --> &A731
A714		024	18	CLC
A715		152	98	TYA
A716	uB	117 066	75 42	ADC &42,X
A718		168	A8	TAY
A719	4	165 052	A5 34	LDA &34
A71B	uA	117 065	75 41	ADC &41,X
A71D	4	133 052	85 34	STA &34
A71F	3	165 051	A5 33	LDA &33

A721	u@	117 064	75 40	ADC &40,X
A723	3	133 051	85 33	STA &33
A725	2	165 050	A5 32	LDA &32
A727	u?	117 063	75 3F	ADC &3F,X
A729	2	133 050	85 32	STA &32
A72B	1	165 049	A5 31	LDA &31
A72D	u>	117 062	75 3E	ADC &3E,X
A72F	1	133 049	85 31	STA &31
A731		232	E8	INX
A732	0	048 220	30 DC	BMI -36 --> &A710
A734		250	FA	PLX
A735	F	181 070	B5 46	LDA &46,X
A737		208 204	D0 CC	BNE -52 --> &A705
A739		232	E8	INX
A73A	0	048 249	30 F9	BMI -7 --> &A735
A73C		250	FA	PLX
A73D	5	132 053	84 35	STY &35
A73F	1	165 049	A5 31	LDA &31
A741	0	048 129	30 81	BMI -127 --> &A6C4 [RTS]
A743	L	076 251 129	4C FB 81	JMP &81FB Normalise FWA

BASIC IV ROM Routines

Page Last Altered: undefined

Integer MOD routine

Submitted by Steve Fewell

Routine: iMOD
Name: Integer MOD routine
Starting Address: &9FF5
Entry criteria: Either IWA or FWA contain the dividend. A contains the variable type (Floating-Point/Integer). The Basic Text Pointer 2 points to an expression (the divisor) [e.g. " A%"]
Exit: The IWA.= The Remainder of IWA / Divisor-expression

Description:
Call the Integer Division routine to evaluate the Divisor-expression and perform the actual division.

Push the sign-bit (&38) of the result to the 6502 stack, load X with &3D (to specify where the result is located, this is done by 9FB2), and call the end of the Integer multiplication routine to put the Result into the IWA, convert it to the correct sign, and to check the next operator to see if any further multiplication/division processing is required.

Disassembly for the Integer MOD routine

9FF5		032 249 128	20 F9 80	JSR &80F9	Integer Division
9FF8	8	165 056	A5 38	LDA &38	
9FFA		008	08	PHP	
9FFB		128 181	80 B5	BRA -75 --> &9FB2	Get Result from Zero Page & check next operator in expression

---

BASIC IV ROM Routines

Page Last Altered: undefined

Integer Division

Submitted by Steve Fewell

Routine: Integer Division
Name: Integer Division
Starting Address: &80F9
Entry criteria: Either the IWA or FWA contain the Dividend (Integer A), The BASIC Expression pointed to by BASIC's second text pointer (&19, &1A) evaluates to the Divisor value (Integer B).
Exit: The routine evaluates the expression Integer A / Integer B. Addresses &39 to &3C & the Carry-flag [so that odd-numbers are handled correctly] contain half of the quotient (result) value. Addresses &3D to &40 contain the remainder.

Description:
Check the current variable type (A), and if the current value is Floating-Point, convert the Floating-Point value to an Integer (IWA).
Next, prepare the Dividend value by pushing its sign-byte (&2D) to the 6502 stack, and making the dividend positive (so that we only need to deal with positive numbers).

Next call BASIC's Integer expression handler &A00F to push the current integer to the BASIC stack, read the BASIC expression from the second Text Pointer (&19, &1A), evaluate it and return a numeric result. Check the variable type (A) of the numeric result, and if the current value is Floating-Point, convert the Floating-Point value to an Integer (IWA).

Retrieve the sign-byte for the Dividend (from the 6502 stack), Store it in &38 - This is the sign of the Remainder. EOR the Dividend's sign-byte with the sign-byte of the Divisor to obtain the sign of the result, store this in &37.

Next, as we already have determined the signs of our remainder and result, prepare the Divisor value by making it positive (so that we only need to deal with division of positive numbers).

Retrieve the Dividend from the BASIC stack to locations &39, &3A, &3B and &3C and clear locations &3D, &3E, &3F and &40 (the remainder). So that the IWA contains the Divisor, &39-&3C contain the Dividend/result

and &3D-&40 contain the remainder.

If all bytes of the Divisor are zero (indicating a zero value), then give a Division by zero error and exit.

Set Y to 32 (this is the number of Dividend digits that we need to process, assume a 32-bit Integer).

&812D-&8139 keep decrementing Y and shifting the Dividend value left (towards the Most significant byte &3C), until we find a bit that is not zero. Now we know exactly how many bits in the Dividend we need to process (as we have removed the leading zeros).

This is the main loop, which performs binary long division on the two numbers:

*1) Shift the Result left 1 place. &813A-8141. (This is the same as the shift in step *2), but is specified here for clarity, as a different value is affected).

*2) Shift the Dividend left 1 place, loosing the top-most bit and shift the bit that was just lost from the Dividend into the lower-end of the Remainder (moving the remainder's other bits left a place). &813A-&8149.

*3) Subtract the Divisor from the Remainder (least significant byte first), storing the 32-bit result in these 4 locations: 6502 stack (2 bytes), in X (1 byte) and in A (1 byte). &814A-&815D.

*4) If a borrow did not occur (the Divisor was successfully subtracted), then [&815E-&816B] store the result as the new remainder (&3D to &40). The next time step *2) is executed, shift 1 into the upper-end [right] of the Dividend, this will form the next bit of the result, when the Dividend's bits are shifted left. BASIC cleverly uses the same 4-bytes to store the result, and the bits which are still left to be processed of the Dividend. Note: as the values are Binary, only 1 subtraction is required.

*5) Decrement Y

Keep looping until Y has reached zero. This indicates that no digits of the Dividend remain to be processed, and that locations &39-&3C contain the result. Note: The result has not been updated with the number of times the Divisor could be subtracted from the Remainder (in step *4)), so it is up to the calling routine to shift the carry-flag value into the bottom-end of the Result, shifting its other bits left a place. The calling routine also needs to convert the result (using &37) and/or Remainder (using &38) to the correct sign.

To simplify what this division routine is dong, here is the same routine using Base-10 Integer numbers:

```
ResultAdd = 0
Result = 0
Remainder = 0
Base = 10
Y = Len(Dividend)
Repeat
   *1) Result = (Result * Base) + ResultAdd
   *2) Remainder = (Remainder * Base) + Next-Digit-of-Dividend
   *3a) Temp = Remainder
```

*3b) ResultAdd = 0
*3c) For Counter = 1 to Base-1
*3d) Temp = Temp - Divisor
*3e) if Temp > 0 then Remainder = Temp : ResultAdd = ResultAdd + 1
*3f) Next Counter
*4) Rem Not required
*5) Y = Y - 1
Until Y = 0

E.g. 324 / 5 would produce the following workings:
Y = 3
Divisor = 5
Dividend = 324
*1) Result = (0 * 10) + 0 = 0
*2) Remainder = (0 * 10) + 3 = 3
*4) Remainder = 3; ResultAdd = 0
*5) Y = 2
*1) Result = (0 * 10) + 0 = 0
*2) Remainder = (3 * 10) + 2 = 32
*4) Remainder = 2; ResultAdd = 6
*5) Y = 1
*1) Result = (0 * 10) + 6 = 6
*2) Remainder = (2 * 10) + 4 = 24
*4) Remainder = 4; ResultAdd = 4
*5) Y = 0

Now to get the correct Result we need to do step *1) again :

*1) Result = (6 * 10) + 4 = 64

So, the result of 324/5 is 64, and the remainder is 4.

Disassembly for the Integer Division routine

80F9		032 190 150	20 BE 96	JSR &96BE	Check if Integer and Convert if Float
80FC	-	165 045	A5 2D	LDA &2D	
80FE	H	072	48	PHA	
80FF		032 190 172	20 BE AC	JSR &ACBE	Integer Positive
8102		032 015 160	20 0F A0	JSR &A00F	Push Integer to Stack and Get result of expression
8105	'	134 039	86 27	STX &27	
8107		032 190 150	20 BE 96	JSR &96BE	Check if Integer and Convert if Float
810A	h	104	68	PLA	
810B	8	133 056	85 38	STA &38	

```
810D  E-  069 045      45 2D     EOR &2D
810F  7   133 055      85 37     STA &37
8111      032 190 172  20 BE AC  JSR &ACBE   Integer Positive
8114  9   162 057      A2 39     LDX#&39
8116      032 008 189  20 08 BD  JSR &BD08   Pop Integer from BASIC Stack Zero-page
8119  d=  100 061      64 3D     STZ &3D
811B  d>  100 062      64 3E     STZ &3E
811D  d?  100 063      64 3F     STZ &3F
811F  d@  100 064      64 40     STZ &40
8121  -   165 045      A5 2D     LDA &2D
8123  *   005 042      05 2A     ORA &2A
8125  +   005 043      05 2B     ORA &2B
8127  ,   005 044      05 2C     ORA &2C
8129  G   240 071      F0 47     BEQ 71 --> &8172   Division by zero error
812B      160 032      A0 20     LDY#&20
812D      136          88        DEY
812E  A   240 065      F0 41     BEQ 65 --> &8171
8130  9   006 057      06 39     ASL &39
8132  &:  038 058      26 3A     ROL &3A
8134  &;  038 059      26 3B     ROL &3B
8136  &<  038 060      26 3C     ROL &3C
8138      016 243      10 F3     BPL -13 --> &812D
813A  &9  038 057      26 39     ROL &39
813C  &:  038 058      26 3A     ROL &3A
813E  &;  038 059      26 3B     ROL &3B
8140  &<  038 060      26 3C     ROL &3C
8142  &=  038 061      26 3D     ROL &3D
8144  &>  038 062      26 3E     ROL &3E
8146  &?  038 063      26 3F     ROL &3F
8148  &@  038 064      26 40     ROL &40
814A  8   056          38        SEC
814B  =   165 061      A5 3D     LDA &3D
```

814D	*	229 042	E5 2A	SBC &2A
814F	H	072	48	PHA
8150	>	165 062	A5 3E	LDA &3E
8152	+	229 043	E5 2B	SBC &2B
8154	H	072	48	PHA
8155	?	165 063	A5 3F	LDA &3F
8157	,	229 044	E5 2C	SBC &2C
8159		170	AA	TAX
815A	@	165 064	A5 40	LDA &40
815C	-	229 045	E5 2D	SBC &2D
815E		144 012	90 0C	BCC 12 --> &816C
8160	@	133 064	85 40	STA &40
8162	?	134 063	86 3F	STX &3F
8164	h	104	68	PLA
8165	>	133 062	85 3E	STA &3E
8167	h	104	68	PLA
8168	=	133 061	85 3D	STA &3D
816A		176 002	B0 02	BCS 2 --> &816E
816C	h	104	68	PLA
816D	h	104	68	PLA
816E		136	88	DEY
816F		208 201	D0 C9	BNE -55 --> &813A
8171	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Pop Integer from the BASIC Stack (Zp)

Submitted by Steve Fewell

Routine: popi0
Name: Pop Integer from the BASIC Stack into the Zero page location pointed to by X
Starting Address: &BD08 (or BD06 to pop Integer to location &37-&3A)
Entry criteria: X contains the first byte of a 4-byte zero page memory location where the Integer from the Stack will be saved.
Exit: The 4-byte zero page memory location pointed to by X (i.e. ?X, ?X+1, ?X+2 and ?X+3) contains the most recent Integer number on the BASIC Stack. The Basic Stack Pointer is moved up.

Description:
If called at address &BD06, then set X to #&37 (this will the starting byte of the
4-byte location that the Integer value on the stack will be retrived to.

Loads the Zero page location pointed to by X with the 32-bit Integer pointed to by the BASIC Stack Pointer. Then move the Stack Pointer up 4 bytes so that is points to the previous item pushed onto the stack.

Disassembly for the Pop Integer from the BASIC Stack (Zp) routine

BD06	7	162 055	A2 37	LDX#&37
BD08		160 003	A0 03	LDY#&03
BD0A		177 004	B1 04	LDA (&04),Y
BD0C		149 003	95 03	STA &03,X
BD0E		136	88	DEY
BD0F		177 004	B1 04	LDA (&04),Y
BD11		149 002	95 02	STA &02,X
BD13		136	88	DEY
BD14		177 004	B1 04	LDA (&04),Y
BD16		149 001	95 01	STA &01,X
BD18		178 004	B2 04	LDA (&04)
BD1A		149 000	95 00	STA &00,X
BD1C		128 220	80 DC	BRA -36 --> &BCFA Move SP up 4 bytes

BASIC IV ROM Routines

Page Last Altered: undefined

Integer DIV routine

Submitted by Steve Fewell

Routine: iDIV
Name: Integer DIV routine
Starting Address: &9FFD
Entry criteria: Either IWA or FWA contain the dividend. A contains the variable type (Floating-Point/Integer). The Basic Text Pointer 2 points to an expression (the divisor) [e.g. " A%"]
Exit: The IWA.= Quotient (Result, IWA = IWA / Divisor-expression).

Description:
Call the Integer Division routine to evaluate the Divisor-expression and perform the actual division.

Multiply the Result by the base (Base 2 - Binary) and add on any value needing to be added to the result. This is not done by the Integer Division routine due to time constraints, as the MOD routine does not need to perform these tasks in order to obtain its required remainder.

Push the sign-bit of the result to the 6502 stack, load X with &39 (to specify where the result is located), and call the end of the Integer multiplication routine to put the Result into the IWA, convert it to the correct sign, and to check the next operator to see if any further multiplication/division processing is required.

Disassembly for the Integer DIV routine

9FFD		032 249 128	20 F9 80	JSR &80F9	Integer Division
A000	&9	038 057	26 39	ROL &39	
A002	&:	038 058	26 3A	ROL &3A	
A004	&;	038 059	26 3B	ROL &3B	
A006	&<	038 060	26 3C	ROL &3C	
A008	$7	036 055	24 37	BIT &37	
A00A		008	08	PHP	
A00B	9	162 057	A2 39	LDX#&39	
A00D		128 165	80 A5	BRA -91 --> &9FB4	Get Result from Zero Page & check next operator in expression

---

BASIC IV ROM Routines

Page Last Altered: undefined

Extract String

Submitted by Steve Fewell

Routine: ExtStr
Name: Extract String
Starting Address: &AD19
Entry criteria: Text Pointer B points to the " character [quote] of the String. Y contains the Text Pointer B Offset (&1B).
Exit: SWA contains the extracted String.

Description:
Set X to 0 (X is a pointer to the next free position in the SWA).

Increment Y to point to the character after the quote. If the current character is a carriage return (ASCII 13), then the end of line has been reached before the closing quote, so a Missing " error is generated. Otherwise, store the character in the next free location in the SWA, and increment the next free location (X), and the Text Pointer B Offset (Y).

If the character that was just stored wasn't a quote ["] then process the next character.

Otherwise, if a quote was just processed, then check the next character. If the next character is not a quote then branch to the end of string routine (&AD11), otherwise increment Y to the next character (ignoring the second quote, as the first quote has already been stored in the SWA), before processing the next character.

End of string routine (&AD11):

When the end of string is reached (a quote which isn't followed by a second quote), the length of the string (X) is decremented by 1 (as the end quote is not part of the string). This length is stored in &36 (The SWA length byte), and the current Text Pointer B Offset position (Y) is stored back in the Text Pointer B working area (&1B), the offset now points to the character after the end quotes. Load the Accumulator with 0 (to signify that a String type has just been processed) and exit.

Disassembly for the Extract String routine

AD19		162 000	A2 00	LDX#&00
AD1B		200	C8	INY
AD1C		177 025	B1 19	LDA (&19),Y
AD1E		201 013	C9 0D	CMP#&0D
AD20		240 017	F0 11	BEQ 17 --> &AD33
AD22		157 000 006	9D 00 06	STA &0600,X
AD25		200	C8	INY
AD26		232	E8	INX
AD27	"	201 034	C9 22	CMP#&22
AD29		208 241	D0 F1	BNE -15 --> &AD1C
AD2B		177 025	B1 19	LDA (&19),Y
AD2D	"	201 034	C9 22	CMP#&22
AD2F		240 234	F0 EA	BEQ -22 --> &AD1B
AD31		208 222	D0 DE	BNE -34 --> &AD11
AD33	L	076 148 146	4C 94 92	JMP &9294 Missing " error

Complete the Extraction of the String

AD11		202	CA	DEX
AD12	6	134 054	86 36	STX &36
AD14		132 027	84 1B	STY &1B
AD16		169 000	A9 00	LDA#&00
AD18	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Next non-space Char PTRB

Submitted by Steve Fewell

Routine: NextCharPTRB
Name: Next non-space Char PTRB
Starting Address: &8ED5
Exit: Move past spaces in BASIC's Text Pointer B and return the next Character in the string. Y is the PTR B offset of the character

Description:
Reads past any space characters found at the current position in the PTRB. Returns with the next non-space Character from PTRB in A, and with the PTRB Offset pointing to this value.

Routine &8EEB returns the next non-space character in PTRB, and also sets the zero flag is the character is a comma (','), or clears the zero flag is the character is not a comma.

Disassembly for the Next non-space Char PTRB routine

8ED5	164 027	A4 1B	LDY &1B
8ED7	230 027	E6 1B	INC &1B
8ED9	177 025	B1 19	LDA (&19),Y
8EDB	201 032	C9 20	CMP#&20
8EDD	240 246	F0 F6	BEQ -10 --> &8ED5

8EDF	`	096	60	RTS

Get Next non-space Char PTRB and compare with ','

8EEB		032 213 142	20 D5 8E	JSR &8ED5
8EEE	,	201 044	C9 2C	CMP#&2C
8EF0	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8FAE Execute next command line / program statement

Submitted by Steve Fewell

Description:

This routine has many entry points (depending on how the last statement finished and what needs to be done to tidy up before the next statement can be executed. The entry points are listed between square brackets ([]).
Therefore, this routine may not necessarily be executed in the order described.

[&8FAE] Skip the rest of the line. This routine is called when a DATA keyword is found (as DATA statements are skipped when reached and not executed (as the READ command controls how they are handled). It is also called when the DEF keyword is found (as function and procedure definitions are not directly executed until they are called by an FN or PROC keyword).
This routine is also called when a REM keyword is found, to skip the rest of the line - as it is a comment line.
Additionally, this routine is also executed when a '*' command is found (the '*' command is first passed to the Operating System OSCLI routine and then this routine skips the line - so that BASIC doesn't try to execute the '*' command.
The rest of the line is skipped as follows:
* Set A to #&0D (the end of line character - '<cr>').
* Set Y to the BASIC Text Pointer A offset (&0A) - 1.
* Keep incrementing Y and checking the character at BASIC Text pointer A (&0B-&0C) plus Y (offset) with the character in A (the '<cr>' character) until the characters match.
* Call routine &9BBC to update the BASIC Text Pointer A address (to include the Y offset value) and to reset the offset (&0A) to 1. (basically, this adds Y to the Text Pointer A address (&0B-&0C)).

[&8FBD] This routine checks whether the last character read (in A) is '<cr>', if it isn't then jump to &8FAE to skip the rest of the line.
Now, we are at the end of the line (i.e. we have reached the '<cr>' character).

[&8FC1] If the BASIC Text pointer MSB address is #&07 then there are no more program lines to execute (as we were executing a statement from the command line - and not from within a program), so jump to &8F86 to prompt for the next command line input.

Set Y to #&01.
Load the next character (offset Y) from the BASIC text pointer A location.
If the character is negative (i.e. #&FF) then we have reached the end of the program, so jump to &8F86 to prompt for the next command line input.

If the TRACE flag (location &20) is on (i.e. it's value is not zero), then:

* Set IWA to the Line number (MSB byte (&2B) = the first byte at BASIC Text pointer A (offset 1) and the LSB byte (&2A) = the second byte at BASIC text pointer A location (offset 2)).
* Call routine &9C4B to display the TRACE line number (in the IWA) on the screen.

[&8FDB] Set the BASIC Text pointer A offset (location &0A) to #&04 (i.e. the first character of the program line - after the line number and line length).
Jump to &900D to execute the statement at the BASIC Text pointer A location.

[&8FEB] Check for special start statement characters '*', '=', 'EXT keyword' and '['
Set Y to the BASIC Text pointer A offset (location &0A) minus 1.
Load the character at the BASIC text pointer A location (plus offset - Y) (this should be the first character of the BASIC statement).
If the character is '*' then jump to &8FA4 to execute the '*'-command.
If the character is '[' then set the OPT flag (location &28) to 3 (default setting) and jump to &8920 to begin the assembly.
If the character is '=' then jump to &9060 to deal with returning from a function and setting the return variable's value.
If the character is 'EXT keyword' then jump to &BE93 to execute the 'EXT =' statement. Note: In BASIC 2, it was not possible to assign a file length (EXT) value, only to read the EXT value; therefore, only 1 BASIC token exists for the 'EXT' keyword (unlike PTR, PAGE, LOMEM and HIMEM which have 2 keyword tokens, one for the setting of the value (where the keyword occurs before the '='), and one for the reading of the value (where the keyword occurs after the '=')).
When BASIC version 4 was written, there was not enough spare BASIC Keyword token values to have two 'EXT' keywords (one for 'EXT =' and the other for '= EXT'), so this workaround directly tests for the 'EXT' keyword appearing at the start of a statement (along with the other special start statement characters '*', '[' and '=') in order to distinguish between the reading of the EXT value and the writing of the EXT value.

[&9000-&9048] Execute the BASIC statement
[&9000] Decrement the BASIC Text pointer A offset (location &0A).
[&9002] Call routine &9BA6 to check for the end of statement ('Syntax error' if ':', '<cr>' or 'ELSE' not found).
[&9005] Load the character pointed to by BASIC Text Pointer A.
If the character is not ':' then jump back to &8FBD to skip the rest of the program line and then proceed to execute the statement on the next line.
[&900B] Set Y to the BASIC Text Pointer A offset (location &0A).
[&900D] Keep incrementing the BASIC Text Pointer A offset (location &0A) until we have found a non-space character. This skips any spaces at the beginning of the BASIC Statement.

If the first non-space character is more than or equal to #&CF (PTR=, PAGE=,...,OSCLI), [i.e. it is not a Command Line-only statement (e.g. OLD, NEW, RENUMBER, EDIT) or a middle of statement keyword (e.g. MID$, ELSE, AND, LEN), but a valid keyword that can occur at the start of a program Statement], then:
[&9019] Jump to the keyword's execution address, as follows:

Multiply the character's ASCII Code by 2 & add the result to the base address &874D to form a pointer to the execution address (LSB first, MSB next) of the required BASIC keyword. This resulting address is jumped to.
Example 1: Character = &CF [PTR= token] = 11001111 multiply by 2 = 10011110 (which is &9E in hex).
So, &874D + &9E = &87EB (The LSB of the execution address for the PTR= keyword (&87EC is the MSB of the address)).
Example 2: Character = &FF [OSCLI token] = 11111111 multiply by 2 = 11111110 (which is &FE in hex).
So, &874D + &FE = &884B (The LSB of the execution address for the OSCLI keyword (&884C is the MSB of the address)).

BASIC Keywords between 128 and 141 are not considered. as these Keywords are used in the middle of statements (and are not functions), so the statements/expression handler will deal with these values. These keywords are as follows: AND, DIV, EOR, MOD, OR, ERROR, LINE, OFF, STEP, SPC, TAB(, ELSE and THEN.

[&901E] Execute the BASIC Command line statement (which can include keywords OLD, NEW, AUTO, EDIT, etc...)
If the first/next non-space character on the command line is more than or equal to #&C6 ('AUTO') then jump back to &9019 to jump to the execution address for the BASIC keyword. This includes all BASIC statement start keywords (but not middle of statement keywords like MID$, LEN, =PTR, AND, etc...).
Otherwise (the character is less than #&C6) continue to &9025 to check for a variable name.

[&9025] Check for variable name
If the character is less than #&CF then check whether it is a variable name as follows:
* Set BASIC Text pointer B location to the BASIC Text Pointer A location. (&19=&0B, &1A=&0C, &1B=Y)
* Call routine &9909 to evaluate a variable name at the BASIC text pointer B location.
* If routine &9909 returns with a value other than zero then jump to the LET keyword routine (address &904F).
  to assign a value to the variable, as the variable was found and the address of the variable's value was found

* Otherwise, (routine &9909 returned zero) the variable is either invalid or hasn't been created yet, so:
    * Check the carry flag status (as returned by routine &9909). If carry is set then jump to &8FEB to check whether the BASIC statement begins with a special character ('*', '[', '=' or 'EXT keyword', and issue 'Syntax error' [via check end of statement routine &9BA6] if none of these special characters match the character at the start of the statement).
    * Store the BASIC text pointer B offset (in X) back to location &1B.
    * Call routine &9B86 to check whether the next non-space character after the variable name is an '=' character. If it isn't then issue the 'Mistake' error, as the variable assignment is not correct and a variable cannot appear at the start of a statement unless it is being assigned a value.
    * Otherwise, '=' was found sucessfully.
    * Call routine &9854 to add the new variable name to the variable pointer table.
    * If the variable type is a float (location &2C contains #&05) then set X to #&06; otherwise, set X to #&05. This specifies the amount of space to allocate for the variable's value.
    * Call routine &9883 to allocate space for the variable (and initialise it's value to zero/null).
    * Decrement the BASIC text pointer A offset (so that BASIC Text pointer A points to the first character of the variable name) and continue to the LET keyword routine to evaluate the variable name (again!) and assign the value specified after the '=' to the variable.

Disassembly for the Execute next command line / program statement routine

8FAE	169 013	A9 0D	LDA#&0D

8FB0		164 010	A4 0A	LDY &0A
8FB2		136	88	DEY
8FB3		200	C8	INY
8FB4		209 011	D1 0B	CMP (&0B),Y
8FB6		208 251	D0 FB	BNE -5 --> &8FB3
8FB8		032 188 155	20 BC 9B	JSR &9BBC Update BASIC Text pointer A (Add offset value & then reset offset to 1)
8FBB		128 004	80 04	BRA 4 --> &8FC1 Process the next program line
8FBD		201 013	C9 0D	CMP#&0D
8FBF		208 237	D0 ED	BNE -19 --> &8FAE Skip the rest of the line and process the next program line
8FC1		165 012	A5 0C	LDA &0C
8FC3		201 007	C9 07	CMP#&07
8FC5		240 191	F0 BF	BEQ -65 --> &8F86 Read & execute command line input
8FC7		160 001	A0 01	LDY#&01
8FC9		177 011	B1 0B	LDA (&0B),Y
8FCB	0	048 185	30 B9	BMI -71 --> &8F86 Read & execute command line input
8FCD		166 032	A6 20	LDX &20
8FCF		240 010	F0 0A	BEQ 10 --> &8FDB
8FD1	+	133 043	85 2B	STA &2B
8FD3		200	C8	INY
8FD4		177 011	B1 0B	LDA (&0B),Y
8FD6	*	133 042	85 2A	STA &2A
8FD8	K	032 075 156	20 4B 9C	JSR &9C4B Display current line number (IWA) on screen [if TRACE is on]
8FDB		160 004	A0 04	LDY#&04
8FDD		132 010	84 0A	STY &0A
8FDF	,	128 044	80 2C	BRA 44 --> &900D
8FE1		169 003	A9 03	LDA#&03
8FE3	(	133 040	85 28	STA &28
8FE5	L	076 032 137	4C 20 89	JMP &8920 '[' Begin Assembly
8FE8	L	076 147 190	4C 93 BE	JMP &BE93 EXT =
8FEB		164 010	A4 0A	LDY &0A
8FED		136	88	DEY
8FEE		177 011	B1 0B	LDA (&0B),Y
8FF0	*	201 042	C9 2A	CMP#&2A
8FF2		240 176	F0 B0	BEQ -80 --> &8FA4 '*'-Command
8FF4	[	201 091	C9 5B	CMP#&5B
8FF6		240 233	F0 E9	BEQ -23 --> &8FE1

8FF8		201 162	C9 A2	CMP#&A2
8FFA		240 236	F0 EC	BEQ -20 --> &8FE8
8FFC	=	201 061	C9 3D	CMP#&3D
8FFE	`	240 096	F0 60	BEQ 96 --> &9060
9000		198 010	C6 0A	DEC &0A
9002		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
9005		178 011	B2 0B	LDA (&0B)
9007	:	201 058	C9 3A	CMP#&3A
9009		208 178	D0 B2	BNE -78 --> &8FBD Skip the rest of the line (until '<cr>' found) & process the next program line
900B		164 010	A4 0A	LDY &0A
900D		230 010	E6 0A	INC &0A
900F		177 011	B1 0B	LDA (&0B),Y
9011		201 032	C9 20	CMP#&20
9013		240 246	F0 F6	BEQ -10 --> &900B
9015		201 207	C9 CF	CMP#&CF
9017		144 012	90 0C	BCC 12 --> &9025
9019		010	0A	ASL A
901A		170	AA	TAX
901B	ÝM	124 077 135	7C 4D 87	JMP (&874D,X)
901E		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space character pointed to by Ptr A
9021		201 198	C9 C6	CMP#&C6
9023		176 244	B0 F4	BCS -12 --> &9019
9025		166 011	A6 0B	LDX &0B
9027		134 025	86 19	STX &19
9029		166 012	A6 0C	LDX &0C
902B		134 026	86 1A	STX &1A
902D		132 027	84 1B	STY &1B
902F		032 009 153	20 09 99	JSR &9909 Evaluate variable/array name & return the address of the value
9032		208 027	D0 1B	BNE 27 --> &904F Create variable (LET)
9034		176 181	B0 B5	BCS -75 --> &8FEB
9036		134 027	86 1B	STX &1B
9038		032 134 155	20 86 9B	JSR &9B86 Check for '='
903B	T	032 084 152	20 54 98	JSR &9854 Create new variable name in variable pointer table
903E		162 005	A2 05	LDX#&05
9040	,	228 044	E4 2C	CPX &2C
9042		208 001	D0 01	BNE 1 --> &9045

9044		232	E8	INX
9045		032 131 152	20 83 98	JSR &9883 Allocate space for variable
9048		198 010	C6 0A	DEC &0A
904A				...LET keyword...

Disassembly for 9B86 Check for '=' routine

9B86		032 213 142	20 D5 8E	JSR &8ED5 Get next non-space character (PTR B)
9B89	=	201 061	C9 3D	CMP#&3D
9B8B		208 211	D0 D3	BNE -45 --> Mistake error
9B8D	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9BA6 Check for End of Statement

Submitted by Steve Fewell

Routine:9BA6
Name: Check for End of Statement
Starting Address: &9BA6
Entry criteria: &0A, &0B and &0C point to the input line.
Exit: Syntax Error if unexpected input is found at end of a statement.

Description:

This routine is called before a command (e.g. CLS) is executed to check that the BASIC Statement line is terminated correctly, and contains no errors. If all is ok then the routine returns sucessfully and the command is performed.
If errors are found on the input line, then a Syntax Error is generated.

The routine does the following:
Set Y to the offset which contains the next character to check. This will be &0A for the BASIC Text pointer A offset,
or, to &1B for the BASIC text pointer B offset (if called from routine &9B96).
Get next character on the line (at the offset position), skiping any spaces found.
If the next character is ':' or '<return>' or 'ELSE' (token &8B) then the line has been terminated correctly, so update the pointer (&B, &C) and set the pointer offset to 1. Test whether an error has occurred, if no error then return (otherwise goto 9B7D to issue an 'Escape' error).

If the next character on the line is anything else then generate a Syntax Error.

Disassembly for the Check for End of Statement routine

9BA6		164 010	A4 0A	LDY &0A
9BA8		136	88	DEY
9BA9		200	C8	INY
9BAA		177 011	B1 0B	LDA (&0B),Y
9BAC		201 032	C9 20	CMP#&20
9BAE		240 249	F0 F9	BEQ -7 --> &9BA9
9BB0	:	201 058	C9 3A	CMP#&3A
9BB2		240 008	F0 08	BEQ 8 --> &9BBC
9BB4		201 013	C9 0D	CMP#&0D
9BB6		240 004	F0 04	BEQ 4 --> &9BBC
9BB8		201 139	C9 8B	CMP#&8B
9BBA		208 173	D0 AD	BNE -83 --> &9B69 'Syntax error'
9BBC		024	18	CLC
9BBD		152	98	TYA
9BBE	e	101 011	65 0B	ADC &0B
9BC0		133 011	85 0B	STA &0B
9BC2		144 002	90 02	BCC 2 --> &9BC6 Set PTR A Offset to 1 & Check for Escape error condition
9BC4		230 012	E6 0C	INC &0C
9BC6		160 001	A0 01	LDY#&01
9BC8		132 010	84 0A	STY &0A
9BCA	$	036 255	24 FF	BIT &FF
9BCC	0	048 175	30 AF	BMI -81 --> &9B7D Escape error
9BCE	`	096	60	RTS

Disassembly for the Check for End of Statement (PTR B) routine

9B96	164 027	A4 1B	LDY &1B
9B98	128 014	80 0E	BRA 14 --> &9BA8

BASIC IV ROM Routines

Page Last Altered: undefined

8F83 Prompt for command line and execute the entered command(s)

Submitted by Steve Fewell

Description:

If called from &8F83 then a full initialisation will be done before the command line prompt appears.

This will reset the following:

* Program start address
* Stack Pointer
* Clear *EDIT mode
* REPEAT/FOR/GOSUB parameters, return addresses, etc...
* Page &7 user tables (&07F0-&07FF)
* Variable pointer tables (destroys all variables that have been created)

If called from &8F86 then the initialisation will not be done as not required. Any defined variables (etc...) will remain

Reset the BASIC Text pointer A (&0B, &0C) to &0700 (the address of the command line buffer).
Call routine &B2A6 to reset the 'ON ERROR' pointer to BASIC's default error handling routine.

Call OSWRCH with A = #&3E to output the '>' [prompt character] to the screen.
Call routine &BA74 to read the input line (using OSWORD 0) and store the input to the command line buffer (&0700-&07EF). The parameters (line length, minimum and maximum acceptable characters, etc...) for this input as the same as those used during the INPUT command (as both inputs are done by the same routine, (&BA70)).

[&8F97] Set the 6502 stack pointer to #&FF (to clear any FN/PROC subroutines that were in progress when the last command (that was executed by the BASIC Intrepreter) had finished, or when the last program execution had ended).
Call routine &B2A6 (again) to reset the 'ON ERROR' pointer to BASIC's default error handling routine.

Call routine &BAEB to tokenise the command line (&0700-&07EF) and to enter the line into the current BASIC program if a line number was supplied at the start of the command line.
If the Command line contained a program line (which has now been inserted into the current program), then jump back to &8F83 to do a full initialisation (as after a line in the Program has been inserted/changed or removed, any defined variables need to be reset, because the TOP of the program has now been changed - corrupting the variable storage space).

Otherwise, if a direct command (not a program line) has been entered, then call &901E to execute the statement(s) on the Command Line.

Disassembly for the Prompt for command line and execute the entered command(s) routine

8F83		032 172 187	20 AC BB	JSR &BBAC Initialise Page 7 & reset Variable pointers, etc...
8F86		160 007	A0 07	LDY#&07
8F88		132 012	84 0C	STY &0C
8F8A	d	100 011	64 0B	STZ &0B

```
8F8C      032 166 178   20 A6 B2    JSR &B2A6 Reset 'ON ERROR' pointer to BASIC's default error handling routine
8F8F   >  169 062       A9 3E       LDA#&3E
8F91      032 238 255   20 EE FF    JSR &FFEE OSWRCH
8F94   t  032 116 186   20 74 BA    JSR &BA74 Prompt for & get the User's input line (storing it in buffer &0700-&07FF)
8F97      162 255       A2 FF       LDX#&FF
8F99      154           9A          TXS
8F9A      032 166 178   20 A6 B2    JSR &B2A6 Reset 'ON ERROR' pointer to BASIC's default error handling routine
8F9D      032 235 186   20 EB BA    JSR &BAEB Tokenise Command Line and Insert Line into Program (if line number given)
8FA0      176 225       B0 E1       BCS -31 --> &8F83 Initialise & prompt for next command line
8FA2   z  128 122       80 7A       BRA 122 --> &901E Execute the command line
```

BASIC IV ROM Routines

Page Last Altered: undefined

BBAC Initialise Page 7 & reset Variable pointers, etc...

Submitted by Steve Fewell

Description:

Set LOMEM and VARTOP to TOP [end of BASIC program].
Gosub BBCF to do the following:
  Set BASIC PROGRAM START (MSB byte) page (&1D) to PAGE.
  Set BASIC Stack Pointer to HIMEM.
  Clear the top bit of &1F -> LISTO flag (to cancel *EDIT mode).
  Clear the Curent REPEAT level (&24).
  Clear the Current FOR level (&26).
  Clear the Current GOSUB level (&25).
  Clear the BASIC PROGRAM START LSB address (&1C).

Copy the 16-byte table in BASIC ROM locations &BF14 to &BF23 to page 7 (RAM) locations &07F0 to &07FF. This is the Floating-Point routine table, and provides the address for various routines which can be called from an assembly language program.

This table contains the following information (all addresses are stored LSB first):

BASIC ROM Locations	Page &7 location	Value and meaning
BF14-BF15	07F0-07F1	A7B8 = the executable address for FWA = SQR(FWA)
BF16-BF17	07F2-07F3	A5EE = Executable address for FWA = [argp] / FWA
BF18-BF19	07F4-07F5	A6A6 = Executable address for FWA = [argp] * FWA
BF1A-BF1B	07F6-07F7	A68D = Executable address for FWA = [argp] + FWA
BF1C-BF1D	07F8-07F9	ACCA = Executable address for compliment FWA [FWA = -FWA]
BF1E-BF1F	07FA-07FB	A541 = Executable address for Load FWA [FWA = [argp]]
BF20-BF21	07FC-07FD	A519 = Executable address for Store FWA [[argp] = FWA]
BF22	07FE	4A = Start of the zero page location for argp [pointer to Float variable]
BF23	07FF	2E = Start of the 8 byte zero page location of FWA

This page 7 information will be overwritten if a BASIC text (program) line is entered which exceeds &F0 characters. But BASIC refreshes this information often, so that it will always be available to assembly language programs.

Zero the locations &0480 to &04FF. This will overwrite the BASIC variable pointer table. This will, however, preserve the resident Integer (A%, B%, etc...) workspace and the Temporary Floating-Point variable areas.

Disassembly for the Initialise Page 7, etc...

BBAC		165 018	A5 12	LDA &12
BBAE		133 000	85 00	STA &00
BBB0		133 002	85 02	STA &02
BBB2		165 019	A5 13	LDA &13
BBB4		133 001	85 01	STA &01
BBB6		133 003	85 03	STA &03
BBB8		032 207 187	20 CF BB	JSR &BBCF /td>
BBBB		162 016	A2 10	LDX#&10
BBBD		189 019 191	BD 13 BF	LDA &BF13,X
BBC0		157 239 007	9D EF 07	STA &07EF,X
BBC3		202	CA	DEX
BBC4		208 247	D0 F7	BNE -9 --> &BBBD
BBC6		162 128	A2 80	LDX#&80
BBC8		158 127 004	9E 7F 04	STZ &047F,X
BBCB		202	CA	DEX
BBCC		208 250	D0 FA	BNE -6 --> &BBC8
BBCE	`	096	60	RTS
BBCF		165 024	A5 18	LDA &18
BBD1		133 029	85 1D	STA &1D
BBD3		165 006	A5 06	LDA &06
BBD5		133 004	85 04	STA &04
BBD7		165 007	A5 07	LDA &07
BBD9		133 005	85 05	STA &05
BBDB		169 128	A9 80	LDA#&80
BBDD		020 031	14 1F	TRB &1F
BBDF	d$	100 036	64 24	STZ &24
BBE1	d&	100 038	64 26	STZ &26
BBE3	d%	100 037	64 25	STZ &25
BBE5	d	100 028	64 1C	STZ &1C
BBE7	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A519 Pack FWA to Floating-Point Variable

Submitted by Steve Fewell

Routine:apack
Name: Pack FWA to Floating-Point Variable
Starting Address: &A519
Entry criteria: &4A and &4B (the argp) point to the first byte of the address that should be used to store the packed 5-byte Floating-Point variable.
Exit: The FWA value has been converted to 5-byte variable format and copied to the address specified.

Description:

The 1st byte of the variable is set to the FWA exponent.
The 2nd byte of the variable is set to the FWA Sign byte EORed with the FWA Mantissa byte 1.
(this sets the top bit to 1 (if positive number [sign=0]) or 0 (if negative number [sign not 0])
the value is now ANDed with #&80 to clear all bits except for the top bit (which we want to preserve).
the value is lastly EORed with the Mantissa byte 1 to replace bits 0 to 6 with the Mantissa byte 1 (preserving the top bit).
The 3rd byte of the variable is set to the FWA Mantissa byte 2.
The 4th byte of the variable is set to the FWA Mantissa byte 3.
The 5th byte of the variable is set to the FWA Mantissa byte 4.

Disassembly for the Pack FWA to FP Variable routine

A519	0	165 048	A5 30	LDA &30
A51B	J	146 074	92 4A	STA (&4A)
A51D		160 001	A0 01	LDY#&01
A51F	.	165 046	A5 2E	LDA &2E

A521	E1	069 049	45 31	EOR &31
A523	)	041 128	29 80	AND#&80
A525	E1	069 049	45 31	EOR &31
A527	J	145 074	91 4A	STA (&4A),Y
A529	2	165 050	A5 32	LDA &32
A52B		200	C8	INY
A52C	J	145 074	91 4A	STA (&4A),Y
A52E	3	165 051	A5 33	LDA &33
A530		200	C8	INY
A531	J	145 074	91 4A	STA (&4A),Y
A533	4	165 052	A5 34	LDA &34
A535		200	C8	INY
A536	J	145 074	91 4A	STA (&4A),Y
A538	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Reset ON ERROR code pointer

Submitted by Steve Fewell

Routine: ON ERROR OFF
Name: Reset BASIC's error code pointer
Starting Address: &B2A6
Exit: BASIC's error vector (&16, &17) contains the address &B2AF.

Description:
This routine resets the BASIC error vector code address to the address of the default BASIC error routine code (the default address of &B2AF).
If an ON ERROR statement is issued then the error vector will point to the code after the ON ERROR instead of this default.

The BASIC's default error program code is located from &B2AF to &B2C7, and contains the following tokenised BASIC instructions:

Address	Contents (in Hex)	BASIC program Code
B2AF	F6	REPORT
B2B0	3A	:
B2B1	E7	IF
B2B2	9E	ERL
B2B3	F1	PRINT
B2B4	22	"

B2B5	20	<space>
B2B6	61	a
B2B7	74	t
B2B8	20	<space>
B2B9	6C	l
B2BA	69	i
B2BB	6E	n
B2BC	65	e
B2BD	20	<space>
B2BE	22	"
B2BF	3B	;
B2C0	9E	ERL
B2C1	3A	:
B2C2	E0	END
B2C3	8B	ELSE
B2C4	F1	PRINT
B2C5	3A	:
B2C6	E0	END
B2C7	0D	<return>

Therefore, the default BASIC error code is:

REPORT:IFERLPRINT" at line ";ERL:ENDELSEPRINT:END

Disassembly for the Reset ON ERROR Code Pointer routine

B2A6	169 175	A9 AF	LDA#&AF
B2A8	133 022	85 16	STA &16
B2AA	169 178	A9 B2	LDA#&B2
B2AC	133 023	85 17	STA &17
B2AE `	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B8B6 INPUT

Submitted by Steve Fewell

Description:

Check the next non-space character. If it's a '#' then jump to the PRINT# routine [&B847].

If the next character is a 'LINE' keyword token then set the carry flag; otherwise, decrement BASIC Text Pointer A Offset (&0A), as we do not need the next character yet, and clear the carry flag.
Set bit 6 of location &4C to the value of the carry flag [by setting the top bit & shifting right one position].
Set &4D to #&FF (field extraction offset flag, #&FF if not currently extracting fields from the Input String).

[&B8CA] Call routine &9299, which does the following:

* Get the next non-space character.
* Call routine &927A to process any ' [quote], TAB(and SPC special print characters
* If carry is clear on return from routine &927A, then a special function was carried out (either New line, TAB(or SPC), so exit routine.
* If next character is not " [double-quote] then a String is not present, so set the carry flag and exit.
* Otherwise, we need to extract and output the String value, as follows:
* [&92A5] Get next character.
* If next character is <cr> [carriage return] then issue 'Missing "' error [as a closing double quote wasn't present before the end of the statement line.
* If the next character is a double quote and the characer after it is not then clear carry, set BASIC Text Pointer A to BASIC Text Pointer B location and exit, as we have outputted the string value. Otherwise, we need to output a single double quote character (as two double quotes were found inside the string value).
* Call routine &BD98 to output the character.
* Jump back to &92A5 to get next character of the string.

If the carry flag is clear on return from routine &9299, then a special character was found & processed, so keep calling routine &9299 to check for further special operations (i.e. New line ('), TAB(, SPC or String ("")), until the carry flag is set (indicating that no further special characters were found); Set &4D to #&FF & clear carry flag (as special characters terminate a field extraction).

[&B8D9] Shift byte &4C (the LINE-status byte) left 1-position (so that the top bit is set if we are in 'LINE'-mode) (the processor flags are temporarily saved while this is done). This also removes the previous Top-bit setting (special character) of &4C. Shift the carry flag into the top byte of &4C (moving the 'LINE'-mode flag right a position, again). (The carry flag is clear if special characters were not found, or set if special characters were found).
So, now Bit 7 of &4C is clear if special PRINT characters (',",TAB(,SPC) were processed, and Bit 6 is set if were are in 'LINE'-mode.

If the next character is "," or ";", then jump back to &B8CA to check for, and process, and special characters

after the field separator.
[Note: If special characters were found after the last ',' or ';', then a '?'-prompt is not output; also, additional fields cannot be extracted from a previous input entry for any additional variables supplied - instead, the user will be prompted for another entry value]

Next, we should either have a variable or the end of the statement, as we have processed all valid characters that can precede a variable in an INPUT statement, namely: double quote ("), Comma (,), Semi-Colon (;), SPC, TAB(and quote (').
Decrement BASIC Text pointer A offset (so that the variable name starts at the next character, rather than the current one).
Save bytes &4C and &4D to the stack.
Call routine &98AE to evaluate the variable name (and create it if the variable doesn't already exist and isn't a direct memory access (!x, ?x, $x).
If the zero flag is set on return from &98AE, then a variable was not specified, so retrieve the 2-bytes pushed to the stack and jump to &9002 (to continue to the next statement if end of statement was found; otherwise, issue a Syntax error).

Retrieve bytes &4C and &4D from the stack.
Call routine &9275 to Set BASIC Text pointer A offset to BASIC Text pointer B offset.
Store the processor flags to the stack (this stores whether the variable is a String or not).
if bit 6 of &4C is clear (i.e. we are not in LINE mode), and &4D location is not #&FF (i.e. not first field) then jump to &B91F (to extract the next field).
If the top bit of &4C is set (Special characters were not found after the last field break), then output a '?'-character - this is the default user input prompt.

Call routine &BA70 to get the user's input, as follows:

* Set &38 to 6 and zero &37. This is the address to place the user's input entry: &0600 (The SWA).
  [Note: If called from &BA74, the address to place the user's input entry is &0700 (The Command Line)].
* Set &39 to #&EE. This is the maximum input line length.
* Set &3A to #&20 and &3B to #&FF. These are the minimum and maximum characters.
* Set Y to #&00 (parameter block MSB), X to #&37 (parameter block LSB).
* Set A to #&00 (OSWORD call 0 - get input characters from current device).
* Call OSWORD (&FFF1) with A = 0 and the parameter block data in locatons &0037 - &003B.
* If the carry flag is set after OSWORD call (with A = 0), then Escape was pressed, so generate an 'Escape' error; otherwise, all is ok, so zero COUNT (&1E) and exit the &BA70 routine.

Store Y (the length of the input field) in &36 (SWA length field).
Clear the top bit of &4C (the special characters not found flag).

If bit 6 of &4C is clear (i.e. we are not in LINE mode), then fields are separated by commas (etc...), so:

* [B91F] Set &1B (the current SWA offset) to A (this is 0 for the first field, & &4D for other fields).
* Zero &19 and set &1A to #&06 (So that &19-&1A is a Pointer to the SWA value).
* Call routine &ACF8 to Extract the next field from the SWA (& place it in the SWA starting at location &0600)
* Call routine &8EEB to Get the next non-space character and compare with ",".
* If comma not found (then <cr> found [if not <cr> get next char until ',' or <cr>] - end of input entry (SWA)); so, set Y to #&FE (Next statement will increment this to #&FF - indicating that we are no longer extracting fields from the input line).
* Increment Y (SWA offset) [This equals #&FF when we are not extracting fields from the SWA] and store Y in &4D.

Retrieve the processor flags from the stack.

If the variable is an Integer (carry flag clear) then set the variable as follows:

* [&B93B] Call routine &BC43 to store the variable address & type details (&2A,&2B,&2C) to the stack.
* Call ASCII to Numeric conversion routine (ASCNUM, &AB4E) to convert SWA string value to a binary number.
* Call routine &B32B to set the numeric variable to the numeric value (which is either a Floating-Point or Integer value)
* Jump back to &B8CA to check for further variables or special characters on the INPUT statement.

If the variable is a String (carry flag set) then set the variable as follows:

* [&B946] Zero &27 - (Set current result type to String).
* Call routine &90AE to Set the String variable to the SWA value.
* Jump back to &B8CA to check for further variables or special characters on the INPUT statement.

Disassembly for the INPUT routine

B8B2	h	104	68	PLA
B8B3	h	104	68	PLA
B8B4		128 142	80 8E	BRA -114 --> &B844 [JMP &9002] check end of statement & process next program statement
B8B6		032 223 140	20 DF 8C	JSR &8CDF Get next non-space char (PTR A) and compare with '#'
B8B9		240 140	F0 8C	BEQ -116 --> &B847 PRINT#
B8BB		201 134	C9 86	CMP#&86 'LINE'-token
B8BD		240 003	F0 03	BEQ 3 --> &B8C2
B8BF		198 010	C6 0A	DEC &0A
B8C1		024	18	CLC
B8C2	fL	102 076	66 4C	ROR &4C
B8C4	FL	070 076	46 4C	LSR &4C
B8C6		169 255	A9 FF	LDA#&FF
B8C8	M	133 077	85 4D	STA &4D
B8CA		032 153 146	20 99 92	JSR &9299
B8CD		176 010	B0 0A	BCS 10 --> &B8D9
B8CF		032 153 146	20 99 92	JSR &9299
B8D2		144 251	90 FB	BCC -5 --> &B8CF
B8D4		162 255	A2 FF	LDX#&FF
B8D6	M	134 077	86 4D	STX &4D
B8D8		024	18	CLC
B8D9		008	08	PHP
B8DA	L	006 076	06 4C	ASL &4C
B8DC	(	040	28	PLP
B8DD	fL	102 076	66 4C	ROR &4C
B8DF	,	201 044	C9 2C	CMP#&2C ','
B8E1		240 231	F0 E7	BEQ -25 --> &B8CA
B8E3	;	201 059	C9 3B	CMP#&3B ';'
B8E5		240 227	F0 E3	BEQ -29 --> &B8CA
B8E7		198 010	C6 0A	DEC &0A
B8E9	L	165 076	A5 4C	LDA &4C
B8EB	H	072	48	PHA
B8EC	M	165 077	A5 4D	LDA &4D
B8EE	H	072	48	PHA
B8EF		032 174 152	20 AE 98	JSR &98AE Evaluate variable name & create if new variable
B8F2		240 190	F0 BE	BEQ -66 --> &B8B2 exit & process next statement
B8F4	h	104	68	PLA
B8F5	M	133 077	85 4D	STA &4D
B8F7	h	104	68	PLA
B8F8	L	133 076	85 4C	STA &4C

B8FA	u	032 117 146	20 75 92	JSR &9275 PTR A Offset = PTR B offset
B8FD		008	08	PHP
B8FE	$L	036 076	24 4C	BIT &4C
B900	p	112 006	70 06	BVS 6 --> &B908
B902	M	165 077	A5 4D	LDA &4D
B904		201 255	C9 FF	CMP#&FF
B906		208 023	D0 17	BNE 23 --> &B91F
B908	$L	036 076	24 4C	BIT &4C
B90A		016 005	10 05	BPL 5 --> &B911
B90C	?	169 063	A9 3F	LDA#&3F '?'
B90E		032 238 255	20 EE FF	JSR &FFEE OSBYTE
B911	p	032 112 186	20 70 BA	JSR &BA70
B914	6	132 054	84 36	STY &36
B916	L	006 076	06 4C	ASL &4C
B918		024	18	CLC
B919	fL	102 076	66 4C	ROR &4C
B91B	$L	036 076	24 4C	BIT &4C
B91D	p	112 025	70 19	BVS 25 --> &B938
B91F		133 027	85 1B	STA &1B
B921	d	100 025	64 19	STZ &19
B923		169 006	A9 06	LDA#&06
B925		133 026	85 1A	STA &1A
B927		032 248 172	20 F8 AC	JSR &ACF8 Extract next field (PTR B)
B92A		032 235 142	20 EB 8E	JSR &8EEB Get next non-space char (PTR B) & compare with ','
B92D		240 006	F0 06	BEQ 6 --> &B935
B92F		201 013	C9 0D	CMP#&0D
B931		208 247	D0 F7	BNE -9 --> &B92A
B933		160 254	A0 FE	LDY#&FE
B935		200	C8	INY
B936	M	132 077	84 4D	STY &4D
B938	(	040	28	PLP
B939		176 011	B0 0B	BCS 11 --> &B946
B93B	C	032 067 188	20 43 BC	JSR &BC43 Push &2A, &2B & &2C to the Stack
B93E	N	032 078 171	20 4E AB	JSR &AB4E ASCNUM (Convert ASCII String to Numeric value)
B941	+	032 043 179	20 2B B3	JSR &B32B Set numeric variable
B944		128 132	80 84	BRA -124 --> &B8CA
B946	d'	100 039	64 27	STZ &27
B948		032 174 144	20 AE 90	JSR &90AE Set String variable
B94B		128 247	80 F7	BRA -9 --> &B944

9299

9299		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
929C	z	032 122 146	20 7A 92	JSR &927A Check for '"', 'TAB(' or 'SPC'
929F		144 242	90 F2	BCC -14 --> &9293 [RTS]

92A1	"	201 034	C9 22	CMP#&22
92A3		208 237	D0 ED	BNE -19 --> &9292 [SEC : RTS]
92A5		200	C8	INY
92A6		177 025	B1 19	LDA (&19),Y
92A8		201 013	C9 0D	CMP#&0D
92AA		240 232	F0 E8	BEQ -24 --> &9294 'Missing '" error
92AC	"	201 034	C9 22	CMP#&22
92AE		208 009	D0 09	BNE 9 --> &92B9
92B0		200	C8	INY
92B1		132 027	84 1B	STY &1B
92B3		177 025	B1 19	LDA (&19),Y
92B5	"	201 034	C9 22	CMP#&22
92B7		208 177	D0 B1	BNE -79 --> &926A Clear carry flag, Set PTR A offset = PTR B offset & exit
92B9		032 152 189	20 98 BD	JSR &BD98 Output character in A
92BC		128 231	80 E7	BRA -25 --> &92A5

BA70 Prompt for and get the user's input line

BA70		169 006	A9 06	LDA#&06
BA72		128 002	80 02	BRA 2 --> &BA76
BA74		169 007	A9 07	LDA#&07
BA76	d7	100 055	64 37	STZ &37
BA78	8	133 056	85 38	STA &38
BA7A		169 238	A9 EE	LDA#&EE
BA7C	9	133 057	85 39	STA &39
BA7E		169 032	A9 20	LDA#&20
BA80	:	133 058	85 3A	STA &3A
BA82		160 255	A0 FF	LDY#&FF
BA84	;	132 059	84 3B	STY &3B
BA86		200	C8	INY
BA87	7	162 055	A2 37	LDX#&37
BA89		152	98	TYA
BA8A		032 241 255	20 F1 FF	JSR &FFF1 OSWORD
BA8D		144 006	90 06	BCC 6 --> &BA95 Zero COUNT (&1E) and exit routine
BA8F	L}	076 125 155	4C 7D 9B	JMP &9B7D Escape error

BASIC IV ROM Routines

Page Last Altered: undefined

B847 INPUT#

Submitted by Steve Fewell

Description:

Call routine &BA3C to set BASIC Text Pointer B to BASIC Text Pointer A location and check that the next character is a '#'-character. If not then generate 'Missing #' error message. Evaluate the expression after the '#' [Type mismatch error if string value], and set Y to the number found after the '#' character (this is the file channel number). Store the channel number to location &4C.
Call &9275 to set BASIC text pointer A offset to the BASIC text pointer B offset value (to update BASIC Text pointer A to point to the next character on the program line after the channel number.

[&B84F] Get the next non-space character at Text pointer A location. If the next character is not a comma (','), then we are not expecting any more data, so exit (Store Y back to &0A (Text Pointer B offset) and jump to &9002 [if the end of statement was not reached then &9002 will generate a 'Syntax Error']).

Push the channel number (&4C) to the stack.
Call &98AE to Evaluate the variable name at Text pointer A, and create the variable if it doesn't already exist.
If the zero flag is set on return from &98AE, then the text at Text pointer B was not a valid variable name, so generate a Syntax error.
Otherwise, the variable name was valid.
Call &9275 to set BASIC text pointer A offset to the BASIC text pointer B offset value (to update BASIC Text pointer A to point to the next character on the program line after the variable name. This is done because routine &98AE copies Text Pointer A location to Text Pointer B location and works from Text Pointer B.
Pop the channel number from the stack abd set &4C back to the channel number.

Push the flag settings to the stack to preserve the Carry flag value after the &98AE call.
Push the IWA (variable settings) to the BASIC stack (routine &BC26).
Set Y to the channel number (&4C).
Call OSBGET (&FFD7) to get the next Variable Type byte from the data file.
Store the variable type in location &27.

Retrieve the processor flags from the Stack. If the carry flag is clear (meaning that the variable has a numeric value - no '$' at the end) then jump to &B88A to set the Numeric variable.

Otherwise set the String variable as follows:

- If the value type (&27) read from the data file is not zero (Numeric), then generate Type mismatch error as a numeric value cannot be assigned to a String variable.
- Call OSBGET (&FFD7) to get the String length & store the length in &36 (SWA length byte).
- If the string length is not 0 then Call OSBGET (&FFD7) to get each character of the string value (last character first). Store each character in the SWA (&600-&6FF) at the appropriate position (so that &5FF + String length points to the last character of the String value).
- Call &90AB to set the String variable to the SWA value (extending the length of the variable if needbe) [or, if a memory location is being set, i.e. $&1200, then &90AB sets the required memory locations to the SWA value].

&B88A sets the numeric variable as follows:

- If the value type (&27) read from the data file is zero (String), then generate Type mismatch error as

a String value cannot be assigned to a Numeric variable.

If the value type (&27) is an Integer value (&40) then call OSBGET (&FFD7) to get each byte of the Integer value from the data file (MSB first (&2D)). Store the Integer in the IWA.

If the value type (&27) is an Float value (&FF) then call OSBGET (&FFD7) to get each byte of the packed Floating-Point value from the data file (last byte first). Store the packed Float in Temporary Float location &046C-&0470 and call &A539 to load the FWA with the temporary variable's value.

Call routine &BD06 to retrieve the Integer from the BASIC stack to locations &37-&3A.
This restores the variable address and type details.
Call routine &B338 to set the numeric variable. This routine will store the Numeric value (either in the IWA or FWA) to the Numeric variable, and convert the value from Floating-Point to Integer, or vice versa, if the variable is of a different type to the value.

Jump back to &B84F to check for further commas (','), indicating that further values are required to be retrieved from the data file.

Disassembly for the INPUT# routine

B83C	L	076 146 144	4C 92 90	JMP &9092 Type Mismatch error
B83F	Li	076 105 155	4C 69 9B	JMP &9B69 Syntax error
B842		132 010	84 0A	STY &0A
B844	L	076 002 144	4C 02 90	JMP &9002 'Syntax error' if not end of statement; otherwise, execute next statement/program line
B847	<	032 060 186	20 3C BA	JSR &BA3C PTR B=PTR A, Check for '#', Set Y to file channel number (PTR B)
B84A	L	132 076	84 4C	STY &4C Store file channel number
B84C	u	032 117 146	20 75 92	JSR &9275 Set PTR A Offset to PTR B Offset
B84F		032 229 140	20 E5 8C	JSR &8CE5 Get next non-space char (PTR A) and compare with ','
B852		208 238	D0 EE	BNE -18 --> &B842 No comma, so exit as no more data expected
B854	L	165 076	A5 4C	LDA &4C
B856	H	072	48	PHA Store channel number
B857		032 174 152	20 AE 98	JSR &98AE Evaluate variable name & create if new variable
B85A		240 227	F0 E3	BEQ -29 --> &B83F [JMP &9B69 Syntax error]
B85C	u	032 117 146	20 75 92	JSR &9275 Set PTR A Offset to PTR B Offset
B85F	h	104	68	PLA
B860	L	133 076	85 4C	STA &4C
B862		008	08	PHP
B863	&	032 038 188	20 26 BC	JSR &BC26 Push IWA (variable details) to stack
B866	L	164 076	A4 4C	LDY &4C
B868		032 215 255	20 D7 FF	JSR &FFD7 OSBGET [Read variable type from file]
B86B	'	133 039	85 27	STA &27
B86D	(	040	28	PLP
B86E		144 026	90 1A	BCC 26 --> &B88A Numeric variable (no '$')
B870	'	165 039	A5 27	LDA &27
B872		208 200	D0 C8	BNE -56 --> &B83C [JMP &9092 Type mismatch error]
B874		032 215 255	20 D7 FF	JSR &FFD7 OSBGET [Read string length from file]
B877	6	133 054	85 36	STA &36
B879		170	AA	TAX
B87A		240 009	F0 09	BEQ 9 --> &B885
B87C		032 215 255	20 D7 FF	JSR &FFD7 OSBGET [Read next String character from file]
B87F		157 255 005	9D FF 05	STA &05FF,X
B882		202	CA	DEX

B883		208 247	D0 F7	BNE -9 --> &B87C
B885		032 171 144	20 AB 90	JSR &90AB Set String variable
B888		128 197	80 C5	BRA -59 --> &B84F
B88A	'	165 039	A5 27	LDA &27
B88C		240 174	F0 AE	BEQ -82 --> &B83C [JMP &9092 Type mismatch error]
B88E	0	048 012	30 0C	BMI 12 --> &B89C
B890		162 003	A2 03	LDX#&03
B892		032 215 255	20 D7 FF	JSR &FFD7 OSBGET [Get next Integer byte from file]
B895	*	149 042	95 2A	STA &2A,X
B897		202	CA	DEX
B898		016 248	10 F8	BPL -8 --> &B892
B89A		128 014	80 0E	BRA 14 --> &B8AA
B89C		162 004	A2 04	LDX#&04
B89E		032 215 255	20 D7 FF	JSR &FFD7 OSBGET [Get next Float byte from file]
B8A1	l	157 108 004	9D 6C 04	STA &046C,X
B8A4		202	CA	DEX
B8A5		016 247	10 F7	BPL -9 --> &B89E
B8A7	9	032 057 165	20 39 A5	JSR &A539 Load FWA from &046C
B8AA		032 006 189	20 06 BD	JSR &BD06 Retrieve Integer from stack to &37-&3A
B8AD	8	032 056 179	20 38 B3	JSR &B338 Set numeric variable
B8B0		128 157	80 9D	BRA -99 --> &B84F

BASIC IV ROM Routines

Page Last Altered: undefined

9141 PRINT#

Submitted by Steve Fewell

Description:

Copy BASIC Text pointer A values to Text Pointer B.
Check that '#' is the next non-space character - if it isn't then generate a 'Missing #' error.
Note: The next character will always be '#' as this routine is only called from the PRINT routine,
which checks for the '#' character before calling this routine!
Get the Integer value from PTR B (after the '#') and set Y to the LSB of this value.
This is the file channel number for the file that we will write to.
Push channel number to the stack.

If the next non-space character (after the channel number) is not a comma, then we have no
data to write to the file, so exit [if the end of statement was not reached then &9002 will
generate a 'Syntax Error'].

Call the expression handler (&9D3B) to obtain the result of the expression after the comma character.
Store the FWA value (in Floating-Point 5-byte packed format) to temporary Floating-Point location &046C-&0470.
This packs the FWA value (ready to be written to the file). This is done irrespective of whether the expression return
value is a Floating-Point value or not.
Retrieve the channel number (in Y) from the stack.

Store &27 (the returned value type) to the file [1-byte value] using OSBPUT (&FFD4)
[Y = channel handle & A = byte to write].

If the value type (&27) is Integer (positive &27 value) then write the 4-byte IWA value to the file
(using 4 calls to OSBPUT). The integer value is written Most-significant byte (&2D) first.

If the value type (&27) is Float (negative &27 value) then write the 5-byte packed FWA value at
location &046C-&0470 to the file (using 5 calls to OSBPUT). The Float value is written last byte (&0470) first.

If the value type (&27) is String (zero &27 value) then write the SWA value
to the file (using a separate OSBPUT call for each byte). Firstly, the string length is written (&36) [1-byte].
Next, the String value is written (last byte first (&0600+&36-1)-&0600).

After the returned value has been written, go back to &9144 to check if there are any more commas.
If another comma is present (after the expression) then another value is required to be written to the file,
so repeat the processing for the next data value. If no more values are found then goto &9187 to
pop the channel number value from the stack (clean up stack), store back the offset (in Y - set in &8EEB) to &0A
(Pointer A offset location), check for the end of statement (issuing Syntax error if not end of statement)
and start processing the next statement/program line (&9002).

Disassembly for the PRINT# routine

9141	< 032 060 186	20 3C BA	JSR &BA3C	PTR B=PTR A, Check for '#', Set Y to file channel number (PTR B)

9144	Z	090	5A	PHY
9145		032 235 142	20 EB 8E	JSR &8EEB Get next non-space char (PTR B) and compare with ','
9148	=	208 061	D0 3D	BNE 61 --> &9187 No comma found, so exit
914A	;	032 059 157	20 3B 9D	JSR &9D3B Get result of expression
914D		032 017 165	20 11 A5	JSR &A511 Store FWA to &046C & set argp=&046C
9150	z	122	7A	PLY
9151	'	165 039	A5 27	LDA &27
9153		032 212 255	20 D4 FF	JSR &FFD4 OSBPUT [Store variable type to file]
9156		170	AA	TAX
9157		240 027	F0 1B	BEQ 27 --> &9174
9159	0	048 012	30 0C	BMI 12 --> &9167
915B		162 003	A2 03	LDX#&03
915D	*	181 042	B5 2A	LDA &2A,X
915F		032 212 255	20 D4 FF	JSR &FFD4 OSBPUT [Store IWA to file]
9162		202	CA	DEX
9163		016 248	10 F8	BPL -8 --> &915D
9165		128 221	80 DD	BRA -35 --> &9144
9167		162 004	A2 04	LDX#&04
9169	l	189 108 004	BD 6C 04	LDA &046C,X
916C		032 212 255	20 D4 FF	JSR &FFD4 OSBPUT [Store FWA (packed) to file]
916F		202	CA	DEX
9170		016 247	10 F7	BPL -9 --> &9169
9172		128 208	80 D0	BRA -48 --> &9144
9174	6	165 054	A5 36	LDA &36
9176		032 212 255	20 D4 FF	JSR &FFD4 OSBPUT [Store string (SWA) length to file]
9179		170	AA	TAX
917A		240 200	F0 C8	BEQ -56 --> &9144
917C		189 255 005	BD FF 05	LDA &05FF,X
917F		032 212 255	20 D4 FF	JSR &FFD4 OSBPUT [Store SWA to file]
9182		202	CA	DEX
9183		208 247	D0 F7	BNE -9 --> &917C
9185		128 189	80 BD	BRA -67 --> &9144
9187	h	104	68	PLA
9188		132 010	84 0A	STY &0A
918A	L	076 002 144	4C 02 90	JMP &9002 'Syntax error' if not end of statement; otherwise, execute next statement/program line

Copy PTR A to PTR B, check for '#', get file channel number and set Y=channel number

BA3C		198 010	C6 0A	DEC &0A
BA3E		165 010	A5 0A	LDA &0A
BA40		133 027	85 1B	STA &1B
BA42		165 011	A5 0B	LDA &0B
BA44		133 025	85 19	STA &19
BA46		165 012	A5 0C	LDA &0C
BA48		133 026	85 1A	STA &1A
BA4A		032 213 142	20 D5 8E	JSR &8ED5 Get next non-space char (PTR B)
BA4D	#	201 035	C9 23	CMP#&23 check for '#'
BA4F		208 176	D0 B0	BNE -80 --> &BA01 'Missing #' error
BA51		032 180 150	20 B4 96	JSR &96B4 Get integer value (PTR B)
BA54	*	164 042	A4 2A	LDY &2A
BA56		152	98	TYA

BA57 ` 096 60 RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A50D Store FWA to temporary Floating-Point Variable location

Submitted by Steve Fewell

Routine:FWAtoTemp
Name: Store FWA to temporary FP variable location
Starting Address: &A50D or &A511 or &A513
Entry criteria: The FWA contains the value to store.
(if called from &A513, A contains the Temporary variable address (LSB)).
Exit: The FWA has been stored in the temporary variable location.

Description:

If called from &A50D, then the FWA will be saved to Temporary Floating-Point variable address &0476
If called from &A511, then the FWA will be saved to Temporary Floating-Point variable address &046C
If called from &A513, then the FWA will be saved to Temporary Floating-Point variable address &0400 plus A (the LSB address)

Disassembly for the Store FWA to temporary variable location routine

A50D	v	169 118	A9 76	LDA#&76
A50F		128 002	80 02	BRA 2 --> &A513
A511	l	169 108	A9 6C	LDA#&6C
A513	J	133 074	85 4A	STA &4A
A515		169 004	A9 04	LDA#&04
A517	K	133 075	85 4B	STA &4B
A519				... Store FWA to argp address routine

BASIC IV ROM Routines

Page Last Altered: undefined

90AB Set String variable

Submitted by Steve Fewell

Routine:setstringvar
Name: Store string value in the specified Variable
Starting Address: &90AB
Entry criteria: The SWA contains the String value &
the variable address and type details are stored in the Integer value on the BASIC stack.
Exit: The specified variable has been set to the value in the SWA.

Description:

Pop the Integer from the BASIC Stack. Now &2A-&2B will contain the address of the variable's value, and &2C will contain the variable's return type.

If the variable's return type is #&80 then we are not setting a variable, but a direct memory access via a statement such as: LET $&0F55="HELLO"; so, as we are not dealing with a variable we can skip the checking of the variable's existing length and go directly to storing the value.
To store the value of a non-variable String, we first store a carriage return as the next free position in the SWA (as String values must terminate with a carriage return character). If SWA length (&36) is zero then store the carriage return character at the Direct Memory address given and exit; otherwise, copy the SWA value to the Direct Memory address (and subsequent bytes, as necessary) and exit.
Note: The SWA value is stored last character first, and first character last.

Otherwise we are dealing with a String variable (as the return type is probably #&81).
The String variable's address points to a 4-byte parameter block, this parameter block contains the following: A 2-byte pointer to the current value of the String, followed by the maximum size allocated for the String variable value (1 byte), followed by the current size of the String value.

Compare the maximum length assigned for the String Variable value (the 3rd byte in the variable's parameter block) with the length of the SWA (&36).
if the current Maximum length is >&eq; the SWA length then we do not need to allocate more space for the variable, so [&910E] Store the SWA value in variable location.

Otherwise we need to allocate more space, as follows:
Store VARTOP address (the next free Variable storage address) in &2C-&2D, this will be the location of the variable's new String value.
If the new String length is less than 8 and the new string length + 8 is less then #&FF

then set the new String length to #&FF. [This will resolve possible problems with String values more than #&F7 characters long]. Otherwise set the new string length to &36 (SWA length).
Push the new String length to the Stack.

If the current VARTOP value is equal to the Variable value address (first 2 bytes of variable's parameter block) plus the variable length (4th byte of the variable parameter block) then the Variable's current value was the last item added to VARTOP, so
Store zero in location &2D (this will tell the allocate space routine not to change the current Variable value address) and Subtract the old String length from the new String length (in X), so that we only allocate the remaining number of bytes required!

Add the new string length (in X) to VARTOP address (store result in X & Y).
This will be the new VARTOP value, but first check whether this value would overlap with the BASIC Stack, if it would then there is no more variable storage space, so issue a No Room error.
Otherwise, update VARTOP (&02-&03) to the new VARTOP value (in X and Y).

Next, we need to set up the Variable's parameter block with the newly allocated information.
Retrieve the new variable length from the Stack.
Store the new variable length in the Max Size Allocated byte of the variable's parameter block (the 3rd byte).
Check location &2D, if it is zero then we do not need to change the address of the variable's value, as the last location (which was at the top of the BASIC variable Heap) has been extended.
Otherwise, store the new located (in &2C-&2D) as the address of the variable's value in the first 2 bytes of the variable's parameter block.
Now, that we have the required space, continue to &910E to store the SWA value.

[&910E] Store the String SWA value in the variable location
Firstly, store the SWA length (&36) in byte 4 of the String variable's parameter block, this is the 1-byte which represents the current length of the String variable.
If the String is blank (SWA length is 0) then exit.

Store the variable's address (the first 2 bytes of the variable's parameter block) in locations &2C-&2D.
Store the SWA value to the memory location pointed to by &2C-&2D. [Note: The SWA value is stored first character first, last character last].
When we have stored all characters of the SWA's String value then exit, as the variable's value is now complete.

Disassembly for the Set String variable routine

90AB		032 230 188	20 E6 BC	JSR &BCE6 Pop Integer from BASIC Stack
90AE	,	165 044	A5 2C	LDA &2C
90B0		201 128	C9 80	CMP#&80
90B2	x	240 120	F0 78	BEQ 120 --> &912C Set String Variable to value
90B4		160 002	A0 02	LDY#&02
90B6	*	177 042	B1 2A	LDA (&2A),Y
90B8	6	197 054	C5 36	CMP &36
90BA	R	176 082	B0 52	BCS 82 --> &910E
90BC		165 002	A5 02	LDA &02
90BE	,	133 044	85 2C	STA &2C

90C0		165 003	A5 03	LDA &03
90C2	-	133 045	85 2D	STA &2D
90C4	6	165 054	A5 36	LDA &36
90C6		201 008	C9 08	CMP#&08
90C8		144 006	90 06	BCC 6 --> &90D0
90CA	i	105 007	69 07	ADC#&07
90CC		144 002	90 02	BCC 2 --> &90D0
90CE		169 255	A9 FF	LDA#&FF
90D0		024	18	CLC
90D1	H	072	48	PHA
90D2		170	AA	TAX
90D3	*	177 042	B1 2A	LDA (&2A),Y
90D5	r*	114 042	72 2A	ADC (&2A)
90D7	E	069 002	45 02	EOR &02
90D9		208 015	D0 0F	BNE 15 --> &90EA
90DB		136	88	DEY
90DC	q*E	113 042	71 2A	ADC (&2A),Y
90DE		069 003	45 03	EOR &03
90E0		208 008	D0 08	BNE 8 --> &90EA
90E2	-	133 045	85 2D	STA &2D
90E4		138	8A	TXA
90E5		200	C8	INY
90E6	8	056	38	SEC
90E7	*	241 042	F1 2A	SBC (&2A),Y
90E9		170	AA	TAX
90EA		138	8A	TXA
90EB		024	18	CLC
90EC	e	101 002	65 02	ADC &02
90EE		168	A8	TAY
90EF		165 003	A5 03	LDA &03
90F1	i	105 000	69 00	ADC#&00
90F3		170	AA	TAX
90F4		196 004	C4 04	CPY &04
90F6		229 005	E5 05	SBC &05
90F8		176 167	B0 A7	BCS -89 --> &90A1 No Room error
90FA		132 002	84 02	STY &02
90FC		134 003	86 03	STX &03
90FE	h	104	68	PLA
90FF		160 002	A0 02	LDY#&02
9101	*	145 042	91 2A	STA (&2A),Y
9103		136	88	DEY
9104	-	165 045	A5 2D	LDA &2D
9106		240 006	F0 06	BEQ 6 --> &910E
9108	*	145 042	91 2A	STA (&2A),Y

910A	,	165 044	A5 2C	LDA &2C
910C	*	146 042	92 2A	STA (&2A)
910E		160 003	A0 03	LDY#&03
9110	6	165 054	A5 36	LDA &36
9112	*	145 042	91 2A	STA (&2A),Y
9114		240 021	F0 15	BEQ 21 --> &912B
9116		160 001	A0 01	LDY#&01
9118	*	177 042	B1 2A	LDA (&2A),Y
911A	-	133 045	85 2D	STA &2D
911C	*	178 042	B2 2A	LDA (&2A)
911E	,	133 044	85 2C	STA &2C
9120		136	88	DEY
9121		185 000 006	B9 00 06	LDA &0600,Y
9124	,	145 044	91 2C	STA (&2C),Y
9126		200	C8	INY
9127	6	196 054	C4 36	CPY &36
9129		208 246	D0 F6	BNE -10 --> &9121
912B	`	096	60	RTS
912C	+	032 043 190	20 2B BE	JSR &BE2B Store carriage return at end of SWA
912F		192 000	C0 00	CPY#&00
9131		240 011	F0 0B	BEQ 11 --> &913E
9133		185 000 006	B9 00 06	LDA &0600,Y
9136	*	145 042	91 2A	STA (&2A),Y
9138		136	88	DEY
9139		208 248	D0 F8	BNE -8 --> &9133
913B		173 000 006	AD 00 06	LDA &0600
913E	*	146 042	92 2A	STA (&2A)
9140	`	096	60	RTS

Store #&0D (Carriage Return) at end of SWA and set (&37-&38) to point to the SWA

BE25	d7	100 055	64 37	STZ &37
BE27		169 006	A9 06	LDA#&06
BE29	8	133 056	85 38	STA &38
BE2B	6	164 054	A4 36	LDY &36
BE2D		169 013	A9 0D	LDA#&0D
BE2F		153 000 006	99 00 06	STA &0600,Y
BE32	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B32B Set Numeric Variable

Submitted by Steve Fewell

Routine:setnumericvar
Name: Store numerical value in the specified variable
Starting Address: &B32B
Entry criteria: Either the IWA or the FWA
contains the value & the variable address and type details are stored on the Stack (3 bytes).
Exit: The specified variable has been set to the numerical value from the IWA or FWA.

Description:

(if called from &B328 then evaluate the expression value (this will be the value that we will set the variable to).

Temporarily pop the return address from the Stack (that is the address to return to after the end of subroutine RTS statement!).
Pop the variable return type from the stack and store this value in &39.
Pop the Variable value address from the stack and store this address in &37-&38.
Push the return address back onto the stack.
If called from &B338, then the stack manipulation is skipped, as the variable address details and variable type information has already been set (and ae not located on the stack).

If the variable return type is 5 then the variable requires a Floating-Point value, so do the following:

* If the value type (for the current value in the IWA/FWA, in &27) is zero then we have a string value, so issue a Type Mismatch error, as we cannot assign a String to our variable.
* If the value type is positive then we have an Integer value that we need to assign to a Floating-Point variable, so convert the Integer to a Floating-Point value.
* Store the FWA to the variable's Memory address (&B369) to output the FWA value to the address pointed to by &37-&38, and then exit.
  Note: the FWA will be stored in it's 5-byte packed format, as follows:
  Byte 1: The FWA Exponent
  Byte 2: The FWA Sign Bit (MSB) followed by the lower 7-bits of Mantissa Byte 1
  (as the FWA value is normalised, we don't need to store the TOP bit of the FWA Mantissa byte 1 value (as it is always 1), so the FWA sign value replaces this bit).
  Byte 3: The FWA Mantissa Byte 2
  Byte 4: The FWA Mantissa Byte 3
  Byte 5: The FWA Mantissa Byte 4

Otherwise, the variable return type is Integer, so do the following:

* If the value type (for the current value in the IWA/FWA, in &27) is zero then we have a string value, so issue a Type Mismatch error, as we cannot assign a String to our variable.
* If the value type is negative then we have a Floating-Point value that we need to assign to an Integer variable, so convert the Floating-Point value to an Integer.
* Continue to the Store IWA to Memory address routine (&B347) to output the IWA value to the address pointed to by &37-&38, and exit.
  Note The Store IWA to address routine will check the variable type (&39) to see if we need to store a 1-byte or 4-byte value, as if the variable is a direct memory access (e.g. ?&70 = 1), then we must only store 1-byte; all other Integer values are 4-byte variables.

Disassembly for the Set Numerical variable routine

B325	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
B328	;	032 059 157	20 3B 9D	JSR &9D3B Get result of expression
B32B	z	122	7A	PLY
B32C		250	FA	PLX
B32D	h	104	68	PLA
B32E	9	133 057	85 39	STA &39
B330	h	104	68	PLA
B331	8	133 056	85 38	STA &38
B333	h	104	68	PLA
B334	7	133 055	85 37	STA &37
B336		218	DA	PHX
B337	Z	090	5A	PHY
B338	9	165 057	A5 39	LDA &39
B33A		201 005	C9 05	CMP#&05
B33C	"	240 034	F0 22	BEQ 34 --> &B360 Set Float variable
B33E	'	165 039	A5 27	LDA &27
B340		240 227	F0 E3	BEQ -29 --> &B325 issue 'Type mismatch' error
B342		016 003	10 03	BPL 3 --> &B347 Store IWA to address
B344		032 195 150	20 C3 96	JSR &96C3 Convert Float to Integer
B347				... Store IWA to address [iout]

Set Float variable routine

B360	'	165 039	A5 27	LDA &27
B362		240 193	F0 C1	BEQ -63 --> &B325 Issue 'Type mismatch' error
B364	0	048 003	30 03	BMI 3 --> &B369
B366		032 133 129	20 85 81	JSR &8185 Convert Integer to Floating-Point
B369	0	165 048	A5 30	LDA &30
B36B	7	146 055	92 37	STA (&37)
B36D		160 001	A0 01	LDY#&01
B36F	.	165 046	A5 2E	LDA &2E

B371	E1	069 049	45 31	EOR &31
B373	)	041 128	29 80	AND#&80
B375	E1	069 049	45 31	EOR &31
B377	7	145 055	91 37	STA (&37),Y
B379		200	C8	INY
B37A	2	165 050	A5 32	LDA &32
B37C	7	145 055	91 37	STA (&37),Y
B37E		200	C8	INY
B37F	3	165 051	A5 33	LDA &33
B381	7	145 055	91 37	STA (&37),Y
B383		200	C8	INY
B384	4	165 052	A5 34	LDA &34
B386	7	145 055	91 37	STA (&37),Y
B388	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Save Integer to Address

Submitted by Steve Fewell

Routine: iout
Name: Save Integer to Address
Starting Address: &B347
Entry criteria: &37 and &38 (lo, hi) contain the address of the first byte of the memory location to store either the 32-bit Integer from the IWA (4 consecutive bytes required), or the 8-bit Integer from the IWA (1 byte required). &39 must contain zero if an 8-bit result is required, otherwise a 32-bit result is assumed.
Exit: The 1 or 4 byte memory location specified by &37 and &38 contain a copy of the contents of the IWA.

Description:
Saves a copy of the IWA to the memory location specified by &37 and &38. The least significant byte of the IWA (&2A) is copied first. If &39 contains zero, the routine then exists as the 8-bit integer in &2A has been copied to the requested location.

Otherwise, the rest of the Integer in the IWA is copied into the 3 subsequent bytes of the address pointed to by &37 and &38.

Disassembly for the Save Integer to Address routine

B347	*	165 042	A5 2A	LDA &2A
B349	7	146 055	92 37	STA (&37)

```
B34B  9    165 057     A5 39     LDA &39
B34D       240 016     F0 10     BEQ 16 --> &B35F
B34F  +    165 043     A5 2B     LDA &2B
B351       160 001     A0 01     LDY#&01
B353  7    145 055     91 37     STA (&37),Y
B355  ,    165 044     A5 2C     LDA &2C
B357       200         C8        INY
B358  7    145 055     91 37     STA (&37),Y
B35A  -    165 045     A5 2D     LDA &2D
B35C       200         C8        INY
B35D  7    145 055     91 37     STA (&37),Y
B35F  `    096         60        RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

ACF8 Extract next field

Submitted by Steve Fewell

Description:

This routine gets the next field from an input (or data) line. The field value is returned as a string in the SWA

First, get the next non-space character and set Y to point to this character.
If the character is a double quote ("), then we have found a String value, so call routine AD19 to extract the String, place the value of the String (without the surrounding quotes) in the SWA and update BASIC Text Pointer B Offset to point to after the String value.

Keep getting the next character from BASIC Text Pointer B (Starting at Y) and store the character in the next free SWA location (starting at &0600 - uses X as an index), until the character is a carriage return <cr>, or a comma ','.

Decrement Y and X (so that they point to the <cr> or ',' character), store X in &36 (the length of the field excluding the <cr> or ',' character), and store Y in &1B (so that PTR B points to the <cr> or ',' character).

Set A to 00 (as return type is a String), and exit.

Disassembly for the Extract next field routine

ACF8		032 213 142	20 D5 8E	JSR &8ED5 Get next non-space character (PTR B)
ACFB	"	201 034	C9 22	CMP#&22
ACFD		240 026	F0 1A	BEQ 26 --> &AD19 Extract String
ACFF		162 000	A2 00	LDX#&00
AD01		177 025	B1 19	LDA (&19),Y
AD03		157 000 006	9D 00 06	STA &0600,X
AD06		200	C8	INY
AD07		232	E8	INX
AD08		201 013	C9 0D	CMP#&0D
AD0A		240 004	F0 04	BEQ 4 --> &AD10
AD0C	,	201 044	C9 2C	CMP#&2C
AD0E		208 241	D0 F1	BNE -15 --> &AD01
AD10		136	88	DEY

AD11		202	CA	DEX
AD12	6	134 054	86 36	STX &36
AD14		132 027	84 1B	STY &1B
AD16		169 000	A9 00	LDA#&00
AD18	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

AB4E ASCNUM (ASCII String to Binary Number)

Submitted by Steve Fewell

Routine:ascnum
Name: Convert ASCII string to Binary numbers
Starting Address: &AB4E
Entry: &36 = Length of String in SWA. SWA = ASCII Number "String".
Exit: A and &27 contain the conversion type (&40 = Integer; &FF = Float).

Description:

The ASCII string in SWA is converted to either an Integer (IWA)
or a Float (FWA).
The routine places a binary zero at the end of the SWA, and steps &36 by 1).
Push the text pointer B position onto the Hardware stack (PHA).
Read the text at the text pointer B position (offset from &0600).
Skip any spaces at PTR B.
if a '-' is the first character (sign) in the string then: Skip any Spaces.
Call &A2E1 to get the number (result).
Then complement the result (if the carry is set - non-zero)
Set &27 to the Result type and retore PTR B position from stack the return.
Otherwise; If a '+' is the first character found, then skip spaces and get next character
(and ignore the '+'-sign).
Call &A2E1 to get the numeric result.
Store the result type in &27.
restore the PTR B position and return with A=?&27.

Disassembly for the ASCNUM routine

AB3A	h	104	68	PLA
AB3B		133 027	85 1B	STA &1B
AB3D	h	104	68	PLA
AB3E		133 026	85 1A	STA &1A
AB40	h	104	68	PLA
AB41		133 025	85 19	STA &19
AB43	'	165 039	A5 27	LDA &27
AB45	`	096	60	RTS
AB46	-	AB4D		...[VAL routine]...
AB4E	6	166 054	A6 36	LDX &36
AB50		158 000 006	9E 00 06	STZ &0600,X
AB53		165 025	A5 19	LDA &19
AB55	H	072	48	PHA

AB56		165 026	A5 1A	LDA &1A
AB58	H	072	48	PHA
AB59		165 027	A5 1B	LDA &1B
AB5B	H	072	48	PHA
AB5C	d	100 027	64 1B	STZ &1B
AB5E	d	100 025	64 19	STZ &19
AB60		169 006	A9 06	LDA#&06
AB62		133 026	85 1A	STA &1A
AB64		032 213 142	20 D5 8E	JSR &8ED5 Skip Spaces in PTRB
AB67	-	201 045	C9 2D	CMP#&2D
AB69		240 014	F0 0E	BEQ 14 --> &AB79
AB6B	+	201 043	C9 2B	CMP#&2B
AB6D		208 003	D0 03	BNE 3 --> &AB72
AB6F		032 213 142	20 D5 8E	JSR &8ED5 Skip Spaces in PTRB
AB72		198 027	C6 1B	DEC &1B
AB74		032 225 162	20 E1 A2	JSR &A2E1 ASCNUM: Extract Number at PTRB
AB77		128 013	80 0D	BRA 13 --> &AB86
AB79		032 213 142	20 D5 8E	JSR &8ED5 Skip Spaces in PTRB
AB7C		198 027	C6 1B	DEC &1B
AB7E		032 225 162	20 E1 A2	JSR &A2E1 ASCNUM: Extract Number at PTRB
AB81		144 003	90 03	BCC 3 --> &AB86
AB83		032 218 172	20 DA AC	JSR &ACDA Complement Result
AB86	'	133 039	85 27	STA &27
AB88		128 176	80 B0	BRA -80 --> &AB3A

BASIC IV ROM Routines

Page Last Altered: undefined

AB49 VAL

Submitted by Steve Fewell

Description:

Call the Evaluate value routine (&AD36) which will evaluate the next value at BASIC text pointer B location, or the expression at that location if the value begins with an open bracket ("("). If the obtained value is not a String (A is not 0), then the value is already numeric, so generate a type mismatch error.

Otherwise, continue to the ASCNUM (ASCII String to Binary numeric Value conversion) routine (&AB4E). This routine will return a the Binary equivalent of the ASCII value in either the IWA (if Integer) or FWA (if float) storage locations.

Disassembly for the VAL routine

AB46	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
AB49	6	032 054 173	20 36 AD	JSR &AD36 Evaluate Value
AB4C		208 248	D0 F8	BNE -8 --> &AB46
AB4E	6	166 054	A6 36	...ASCNUM (ASCII to numeric) routine

BASIC IV ROM Routines

Page Last Altered: undefined

A2E1 ASCNUM: Extract Number at PTRB

Submitted by Steve Fewell

Routine:ASCNUM:ExtractNum
Name: ASCII TO NUM: Extract Integer/Floating-Point Number at PTRB
Starting Address: &A2E1
Entry criteria: The SWA contains the ASCII number to extract
Exit: Depending of the size of the number, and whether it contains a fractional part or not, either the FWA or IWA will contain the binary conversion
or the number. A=&40 if Integer, or &FF if Floating-point.
Carry is clear if the ASCII text could not be converted to a numeric value.

Description:

Clears bytes &31 to &35 (The FWA Mantissa & rounding bytes).
Clears byte &47 - the 'Decimal Point found' flag. (remains zero if number is Integer)
Clears byte &48 - the Exponent of the extracted number. (remains zero if number is Integer)

If Next Character at PTRB is "." then A31C [*if decimal point has already been found, then A364 (Complete & tidy-up the number value) as the number is complete. Otherwise, increment &47 (to set the decimal point found flag) then goto A315 (to process the next character of the number).*]

If Next Character at PTRB >= ":" or <= "/" then A2DA [*Return 0.00 as no number found.*].
Subtract 48 (47 + 1 as Carry Clear), so A=0 for "0", 1 for "1" ..., 9 for "9".
Store digit found at LSB of FWA (&35, the FWA rounding byte).

Load next character from PTRB in A.
If character >= ":" [non-digit] then A324 [process main body of number], as this checks for "E" before terminating number extraction at an invalid character.]

Subtract 48 (47 + 1 as Carry Clear), so A=0 for "0", 1 for "1" ..., 9 for "9".
If carry clear (underflow occurred, A <= "/" [non-digit]) then A316 [*if char not = "." then A324 (to process main body of number); otherwise, If character is "." then A31C [if decimal point has already been found, then A364 (Complete & tidy-up the number value) as the number is complete. Otherwise, increment &47 (to set the decimal point found flag) then goto A315 (to process the next character of the number).]*]

Temporarily store A in &2E [FWA sign].
Multiplies the number in &35 by 10 and adds ?&2E to the result to form
a 2-digit binary number.
The multiply does this: ?&35=((?&35*4)+?&35)*2) + ?&2E
= ?&35=(?&35*5*2) plus ?&2E.

Get next character, and if it isn't "." then goto A324 [to process the main body of the number]. Otherwise, set decimal found flag and get next character before going to A324 [main body] or finish extraction (A364) if the next character was another ".".

ASCNUM: A324: Process main body of number:
This routine is first entered after the first two digits have been processed, (by A2E1) or after a "." if the decimal point was encountered before that stage. This routine doesn't need to deal with a "no number found" situation, as this has already been dealt with in A2E1.
This routine keeps the Exponent &48 in line with the binary value, and copes with exponential values ("E"). It also copes with much bigger numbers than the 2-digit ones that A2E1 processed.

This routines does the following: If ASCII character is "E" then goto &A35D [to get the exponent value and convert the exponential
number to the completed binary value and exit with the result stored in the FWA]
If the character is not a valid digit (char >= ":" or char <= "/") then goto A364 to complete the number and finish the extraction.
Subtract 48 (47 + 1 as Carry Clear), so A=0 for "0", 1 for "1" ..., 9 for "9".

X = ?&31 (FWA Mantissa byte 1). If this value is less than or equal to 24 then it is ok to add the next digit to the end of the number [so goto &A33E]. Otherwise, the digit is not added [number is too large] and the exponent is incremented (if no decimal point has been found [integer value]). Then goto A315. A315 gets the next character, handles any decimal point character found, or calls A324 again to handle all other characters.

ASCNUM: A33E: Add digit to end of number (and multiply 'old' number by 10):
If the number has a decimal point in it (?&47 not = 0) then decrement &48 (the exponent) as we are adding a digit to the LSB of the value without moving the decimal point from its current position.

Multiply the current number by 10 (using &A279 [multiply FWA Mantissa by 10] routine).
&A279 returns with A unchanged, so we can add the next digit to &35 in order to add the new digit to the end of the current number (now multiplied by 10 to cope with decimal-binary conversion).

If overflow occures, then we need to increment the rest of the bytes of the FWA Mantissa (&34 and then &33 (if incrementing &34 overflowed), etc... to &31 if necessary).
lastly, we need to go back to A315 to get the next charater.
&A315 gets the next character, handles any decimal point character found, or calls &A324 again to handle all other characters.

Disassembly for the ASCNUM: Extract Number at PTRB routine

ASCNUM: return wth Floating-Point value 0.0 [number not found]:

A2DA	032 004 164	20 04 A4	JSR &A404
A2DD	024	18	CLC
A2DE	169 255	A9 FF	LDA#&FF
A2E0 `	096	60	RTS

ASCNUM: Convert String at PTRB to either an Integer/Floating-Point Number:

A2E1	d1	100 049	64 31	STZ &31
A2E3	d2	100 050	64 32	STZ &32
A2E5	d3	100 051	64 33	STZ &33
A2E7	d4	100 052	64 34	STZ &34
A2E9	d5	100 053	64 35	STZ &35
A2EB	dG	100 071	64 47	STZ &47
A2ED	dH	100 072	64 48	STZ &48
A2EF	.	201 046	C9 2E	CMP#&2E
A2F1	)	240 041	F0 29	BEQ 41 --> &A31C
A2F3	:	201 058	C9 3A	CMP#&3A
A2F5		176 227	B0 E3	BCS -29 --> &A2DA
A2F7	/	233 047	E9 2F	SBC#&2F
A2F9	0	048 223	30 DF	BMI -33 --> &A2DA
A2FB	5	133 053	85 35	STA &35
A2FD		200	C8	INY
A2FE		177 025	B1 19	LDA (&19),Y
A300	:	201 058	C9 3A	CMP#&3A
A302		176 032	B0 20	BCS 32 --> &A324
A304	/	233 047	E9 2F	SBC#&2F
A306		144 014	90 0E	BCC 14 --> &A316
A308	.	133 046	85 2E	STA &2E
A30A	5	165 053	A5 35	LDA &35
A30C		010	0A	ASL A
A30D		010	0A	ASL A

A30E	e5	101 053	65 35	ADC &35
A310		010	0A	ASL A
A311	e.	101 046	65 2E	ADC &2E
A313	5	133 053	85 35	STA &35
A315		200	C8	INY
A316		177 025	B1 19	LDA (&19),Y
A318	.	201 046	C9 2E	CMP#&2E
A31A		208 008	D0 08	BNE 8 --> &A324
A31C	G	165 071	A5 47	LDA &47
A31E	D	208 068	D0 44	BNE 68 --> &A364
A320	G	230 071	E6 47	INC &47
A322		128 241	80 F1	BRA -15 --> &A315

ASCNUM: Convert main body of number:

A324	E	201 069	C9 45	CMP#&45
A326	5	240 053	F0 35	BEQ 53 --> &A35D
A328	:	201 058	C9 3A	CMP#&3A
A32A	8	176 056	B0 38	BCS 56 --> &A364
A32C	/	233 047	E9 2F	SBC#&2F
A32E	4	144 052	90 34	BCC 52 --> &A364
A330	1	166 049	A6 31	LDX &31
A332		224 024	E0 18	CPX#&18
A334		144 008	90 08	BCC 8 --> &A33E
A336	G	166 071	A6 47	LDX &47
A338		208 219	D0 DB	BNE -37 --> &A315
A33A	H	230 072	E6 48	INC &48
A33C		128 215	80 D7	BRA -41 --> &A315

ASCNUM: Add digit to end of number (and multiply 'old' number by 10):

A33E	G	166 071	A6 47	LDX &47
A340		240 002	F0 02	BEQ 2 --> &A344
A342	H	198 072	C6 48	DEC &48
A344	y	032 121 162	20 79 A2	JSR &A279
A347	e5	101 053	65 35	ADC &35
A349	5	133 053	85 35	STA &35
A34B		144 200	90 C8	BCC -56 --> &A315
A34D	4	230 052	E6 34	INC &34
A34F		208 196	D0 C4	BNE -60 --> &A315

A351	3	230 051	E6 33	INC &33
A353		208 192	D0 C0	BNE -64 --> &A315
A355	2	230 050	E6 32	INC &32
A357		208 188	D0 BC	BNE -68 --> &A315
A359	1	230 049	E6 31	INC &31
A35B		128 184	80 B8	BRA -72 --> &A315

ASCNUM: Finish zeroing the FWA (to return value 0.00):

A404

BASIC IV ROM Routines

Page Last Altered: undefined

A35D ASCNUM: Handle Exponential values and complete number conversion

Submitted by Steve Fewell

Routine:A35D> **Name**: ASCNUM: Handle Exponential value (decimal to binary conversion) and complete the number conversion
Starting Address: &A35D

Description:

A35D:
Call A3BA to Extract Exponent value into A.
Add current exponent value ?&48 to the extracted exponent, and store back in &48.
Continue to A364 to complete the number conversion.

A364 ASCNUM: Complete the Number Conversion:

Store current PTRB pointer position back in &1B (PTRB Offset).
If exponent is zero and no decimal-point found then goto A399 (to copy Integer to IWA and return with A=&40).

Gosub asign (if zero value in FWA then clear Sign, Exp, etc..., exit with A = sign)
if FWA = 0 then A395 [FWA = 0.0 so exit with C=1 and A=&FF].

Set FWA Exponent to 168 (Default value), FWA Overflow to 0 and FWA Sign to 0 (Positive value).
[Default exponent = 10101000 =&A8 = 2 to the power of 40 as offset from 128.]
gosub 81F7 to Normalise FWA#1 (correcting the exponent value).

If the ?&48 exponent is zero, then it doesn't need to be applied; otherwise:
If exponent in ?&48 is negative then A38B (keep dividing FWA by 10 and incrementing the exponent by 1, until ?&48 is zero. [meaning that we have now applied the base 10 exponent].

If exponent in ?&48 is positive A382 (keep multiplying FWA by 10 and decrementing the

exponent by 1, until ?&48 is zero. [meaning that we have now applied the base 10 exponent].

After applying the exponent to the binary number, gosub A695 (round Mantissa to 4 bytes), and exit with Carry=1 and A=&FF.

A399 ASCNUM: Copy Integer from FWA to IWA if it will fit:

If FWA Mantissa 1 is not 0 or FWA Mantissa byte 2's top bit is set (MSB) then the integer value is too large to fit in the IWA; so go back to the Float routine.

Transfer Integer in FWA to IWA as follows:
- FWA Mantissa byte 2 ==> IWA Byte 4 (MSB)
- FWA Mantissa byte 3 ==> IWA Byte 3
- FWA Mantissa byte 4 ==> IWA Byte 2
- FWA Mantissa byte 5 (rounding) ==> IWA Byte 1 (LSB)

Exit with A=&40 and Carry = 1.

A3BA ASCNUM: Extract Exponent:

Get Next Character (after the 'E').
If '-' then get the Positive value (as described below) and then EOR with #&FF to make it negative and exit with the carry set.

If '+' then get next character (skip the leading '+')

If first digit is invalid, return 0 as the exponend value (in A).
Subtract 48 to get the binary value of the digit - store in &49.

Get next character.
If second digit is invalid return with A = value in ?&49 [1 digit exponent].

Subtract 48 to get the binary value of the digit - store in &42.
Multiply the first digit by 10 (ASL, ASL, Add ?&49, ASL) and then add the second digit to obtain the completed exponent.
Return with exponent in A, and carry set if exponent is negative; otherwise, with carry clear.

Disassembly for the ASCNUM: Handle Exponent Values & complete num conv routine

A35D		032 186 163	20 BA A3	JSR &A3BA ASCNUM: Extract Exponent
A360	eH	101 072	65 48	ADC &48
A362	H	133 072	85 48	STA &48
A364		132 027	84 1B	STY &1B
A366	H	165 072	A5 48	LDA &48

A368	G	005 071	05 47	ORA &47
A36A	-	240 045	F0 2D	BEQ 45 --> &A399 ASCNUM: FWA Integer copied to IWA
A36C		032 242 163	20 F2 A3	JSR &A3F2 asign
A36F	$	240 036	F0 24	BEQ 36 --> &A395
A371		169 168	A9 A8	LDA#&A8
A373	0	133 048	85 30	STA &30
A375	d/	100 047	64 2F	STZ &2F
A377	d.	100 046	64 2E	STZ &2E
A379		032 247 129	20 F7 81	JSR &81F7
A37C	H	165 072	A5 48	LDA &48
A37E	0	048 011	30 0B	BMI 11 --> &A38B
A380		240 016	F0 10	BEQ 16 --> &A392
A382	6	032 054 164	20 36 A4	JSR &A436
A385	H	198 072	C6 48	DEC &48
A387		208 249	D0 F9	BNE -7 --> &A382
A389		128 007	80 07	BRA 7 --> &A392
A38B	x	032 120 164	20 78 A4	JSR &A478
A38E	H	230 072	E6 48	INC &48
A390		208 249	D0 F9	BNE -7 --> &A38B
A392		032 149 166	20 95 A6	JSR &A695
A395	8	056	38	SEC
A396		169 255	A9 FF	LDA#&FF
A398	`	096	60	RTS

ASCNUM: Copy Integer from FWA to IWA if it will fit & exit:

A399	2	165 050	A5 32	LDA &32
A39B	-	133 045	85 2D	STA &2D
A39D	)	041 128	29 80	AND#&80
A39F	1	005 049	05 31	ORA &31
A3A1		208 206	D0 CE	BNE -50 --> &A371
A3A3	5	165 053	A5 35	LDA &35
A3A5	*	133 042	85 2A	STA &2A
A3A7	4	165 052	A5 34	LDA &34
A3A9	+	133 043	85 2B	STA &2B
A3AB	3	165 051	A5 33	LDA &33
A3AD	,	133 044	85 2C	STA &2C
A3AF	@	169 064	A9 40	LDA#&40
A3B1	8	056	38	SEC

A3B2	`	096	60	RTS

ASCNUM: Extract Exponent value:

A3B3		032 197 163	20 C5 A3	JSR &A3C5
A3B6	I	073 255	49 FF	EOR#&FF
A3B8	8	056	38	SEC
A3B9	`	096	60	RTS
A3BA		200	C8	INY
A3BB		177 025	B1 19	LDA (&19),Y
A3BD	-	201 045	C9 2D	CMP#&2D
A3BF		240 242	F0 F2	BEQ -14 --> &A3B3
A3C1	+	201 043	C9 2B	CMP#&2B
A3C3		208 003	D0 03	BNE 3 --> &A3C8
A3C5		200	C8	INY
A3C6		177 025	B1 19	LDA (&19),Y
A3C8	:	201 058	C9 3A	CMP#&3A
A3CA	"	176 034	B0 22	BCS 34 --> &A3EE
A3CC	/	233 047	E9 2F	SBC#&2F
A3CE		144 030	90 1E	BCC 30 --> &A3EE
A3D0	I	133 073	85 49	STA &49
A3D2		200	C8	INY
A3D3		177 025	B1 19	LDA (&19),Y
A3D5	:	201 058	C9 3A	CMP#&3A
A3D7		176 017	B0 11	BCS 17 --> &A3EA
A3D9	/	233 047	E9 2F	SBC#&2F
A3DB		144 013	90 0D	BCC 13 --> &A3EA
A3DD		200	C8	INY
A3DE	B	133 066	85 42	STA &42
A3E0	I	165 073	A5 49	LDA &49
A3E2		010	0A	ASL A
A3E3		010	0A	ASL A
A3E4	eI	101 073	65 49	ADC &49
A3E6		010	0A	ASL A
A3E7	eB	101 066	65 42	ADC &42
A3E9	`	096	60	RTS
A3EA	I	165 073	A5 49	LDA &49
A3EC		024	18	CLC
A3ED	`	096	60	RTS

A3EE	169 000	A9 00	LDA#&00
A3F0	024	18	CLC
A3F1	` 096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Floating-point Sign

Submitted by Steve Fewell

Routine: Asign
Name: Return the sign of the number in the FWA
Starting Address: &A3F2
Entry criteria: None
Exit: A = 0 if the FWA contains zero, A = 1 if the FWA contains a positive number, A = -1 if the FWA contains a negative number.

Description:
Load A with FWA Mantissa Byte 1, OR this value with the other Mantissa bytes (byte 2 to byte 5), If the result is zero, then the whole of the FWA Mantissa is zero, this signifies that the number in the FWA is zero, if this is so, then store zero in the FWA Sign, FWA Exponent and FWA overflow bytes and return with A = 0.

Otherwise, Load A with the FWA Sign Byte, if it isn't zero, then the FWA is negative, and the top bit of the sign byte must be set. So exit with A containing the sign byte (-1).

Otherwise, increment A (to get a result of 1) and return.

Disassembly for the Floating-point Sign routine

A3F2	1	165 049	A5 31	LDA &31
A3F4	2	005 050	05 32	ORA &32

A3F6	3	005 051	05 33	ORA &33
A3F8	4	005 052	05 34	ORA &34
A3FA	5	005 053	05 35	ORA &35
A3FC		240 006	F0 06	BEQ 6 --> &A404
A3FE	.	165 046	A5 2E	LDA &2E
A400		208 008	D0 08	BNE 8 --> &A40A
A402		026	1A	INC A
A403	`	096	60	RTS
A404	d.	100 046	64 2E	STZ &2E
A406	d0	100 048	64 30	STZ &30
A408	d/	100 047	64 2F	STZ &2F
A40A	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A26C (?) / A279 Multiply FWA Mantissa by 10

Submitted by Steve Fewell

Routine:FWAMantissaMult10
Name: Floating-Point Division (FWA=FWB/FWA)
Starting Address: &A279
Entry criteria: The FWA contains the number to multiply by 10.
Exit: FWA Mantissa has been multiplied by 10.

Description:
A26C (Not used by any routines so far described - context not fully understood)

A = FWA Mantissa byte 1. Divide by 8 [move top 4 bits to bottom 4 bits]
gosub A2CE :- Convert A to ASCII & put ASCII value back at the end of string (&0600)
A = ?&F0.
Test & reset bits in &31 :- Set top 4 bits of &31 to zero (to reduce the value by 48)
thus converting the digit from ASCII to decimal.

A279 : Multiply the FWA's Mantissa by 10.
Temporarily backup A on the stack.
X = ?&34 - FWA Mantissa bit 4
Push ?&31 [FWA Man 1] to Stack
Push ?&32 [FWA Man 2] to Stack
Push ?&33 [FWA Man 3] to Stack

A = ?&35 [FWA Rounding]
A = A * 2 (Transferring the overflow & effect of the multiplication through to
&34, &33, &32, &31 (the rest of the mantissa).
A = A * 2 again (Transferring the overflow & effect of the multiplication through to
&34, &33, &32, &31 (the rest of the mantissa).

Add ?&35, store result in &35.
Add original ?&34 [stored in X] to &34
Add original ?&33 [stored on stack] to &33
Add original ?&32 [stored on stack] to &32
Add original ?&31 [stored on stack] to &31
Multiply &35, &34, &33, &32, &31 (the FWA mantissa) by 2

This in effect does Mantissa = ((Mantissa * 2) + Mantissa) * 2
Which is: Mantissa = (Mantissa * 5) * 2 or Mantissa = Mantissa * 10.

Disassembly for the FWA Mantissa multiplied by 10 routine

A26C	1	165 049	A5 31	LDA &31
A26E	J	074	4A	LSR A
A26F	J	074	4A	LSR A
A270	J	074	4A	LSR A
A271	J	074	4A	LSR A
A272		032 206 162	20 CE A2	JSR &A2CE
A275		169 240	A9 F0	LDA#&F0
A277	1	020 049	14 31	TRB &31
A279	H	072	48	PHA
A27A	4	166 052	A6 34	LDX &34
A27C	1	165 049	A5 31	LDA &31
A27E	H	072	48	PHA
A27F	2	165 050	A5 32	LDA &32
A281	H	072	48	PHA
A282	3	165 051	A5 33	LDA &33
A284	H	072	48	PHA
A285	5	165 053	A5 35	LDA &35
A287		010	0A	ASL A
A288	&4	038 052	26 34	ROL &34
A28A	&3	038 051	26 33	ROL &33
A28C	&2	038 050	26 32	ROL &32
A28E	&1	038 049	26 31	ROL &31
A290		010	0A	ASL A
A291	&4	038 052	26 34	ROL &34
A293	&3	038 051	26 33	ROL &33
A295	&2	038 050	26 32	ROL &32
A297	&1	038 049	26 31	ROL &31

A299	e5	101 053	65 35	ADC &35
A29B	5	133 053	85 35	STA &35
A29D		138	8A	TXA
A29E	e4	101 052	65 34	ADC &34
A2A0	4	133 052	85 34	STA &34
A2A2	h	104	68	PLA
A2A3	e3	101 051	65 33	ADC &33
A2A5	3	133 051	85 33	STA &33
A2A7	h	104	68	PLA
A2A8	e2	101 050	65 32	ADC &32
A2AA	2	133 050	85 32	STA &32
A2AC	h	104	68	PLA
A2AD	e1	101 049	65 31	ADC &31
A2AF	5	006 053	06 35	ASL &35
A2B1	&4	038 052	26 34	ROL &34
A2B3	&3	038 051	26 33	ROL &33
A2B5	&2	038 050	26 32	ROL &32
A2B7	*	042	2A	ROL A
A2B8	1	133 049	85 31	STA &31
A2BA	h	104	68	PLA
A2BB	`	096	60	RTS

BASIC IV ROM Routines

Disassembly A000 to B000

A000	&9	038 057	26 39	ROL &39
A002	&:	038 058	26 3A	ROL &3A
A004	&;	038 059	26 3B	ROL &3B
A006	&<	038 060	26 3C	ROL &3C
A008	$7	036 055	24 37	BIT &37
A00A		008	08	PHP
A00B	9	162 057	A2 39	LDX#&39
A00D		128 165	80 A5	BRA -91 --> &9FB4
A00F	&	032 038 188	20 26 BC	JSR &BC26
A012	6	032 054 173	20 36 AD	JSR &AD36
A015	H	072	48	PHA
A016		164 027	A4 1B	LDY &1B
A018		230 027	E6 1B	INC &1B
A01A		177 025	B1 19	LDA (&19),Y
A01C		201 032	C9 20	CMP#&20

A01E		240 246	F0 F6	BEQ -10 --> &A016
A020		170	AA	TAX
A021	h	104	68	PLA
A022	^	224 094	E0 5E	CPX#&5E
A024		240 001	F0 01	BEQ 1 --> &A027
A026	`	096	60	RTS
A027		168	A8	TAY
A028		032 221 150	20 DD 96	JSR &96DD
A02B		032 250 187	20 FA BB	JSR &BBFA
A02E		032 218 150	20 DA 96	JSR &96DA
A031	0	165 048	A5 30	LDA &30
A033		201 135	C9 87	CMP#&87
A035	B	176 066	B0 42	BCS 66 --> &A079
A037		032 224 130	20 E0 82	JSR &82E0
A03A		208 013	D0 0D	BNE 13 --> &A049
A03C		032 232 187	20 E8 BB	JSR &BBE8
A03F	A	032 065 165	20 41 A5	JSR &A541
A042	I	165 073	A5 49	LDA &49
A044		032 190 165	20 BE A5	JSR &A5BE
A047	,	128 044	80 2C	BRA 44 --> &A075
A049		032 013 165	20 0D A5	JSR &A50D
A04C		165 004	A5 04	LDA &04
A04E	J	133 074	85 4A	STA &4A
A050		165 005	A5 05	LDA &05
A052	K	133 075	85 4B	STA &4B
A054	A	032 065 165	20 41 A5	JSR &A541
A057	I	165 073	A5 49	LDA &49
A059		032 190 165	20 BE A5	JSR &A5BE
A05C	q	169 113	A9 71	LDA#&71
A05E		032 019 165	20 13 A5	JSR &A513
A061		032 232 187	20 E8 BB	JSR &BBE8
A064	A	032 065 165	20 41 A5	JSR &A541
A067	I	032 073 167	20 49 A7	JSR &A749
A06A		032 159 169	20 9F A9	JSR &A99F
A06D		032 226 169	20 E2 A9	JSR &A9E2

A070	q	169 113	A9 71	LDA#&71
A072		032 161 169	20 A1 A9	JSR &A9A1
A075		169 255	A9 FF	LDA#&FF
A077		128 156	80 9C	BRA -100 --> &A015
A079		032 013 165	20 0D A5	JSR &A50D
A07C		032 216 165	20 D8 A5	JSR &A5D8
A07F		128 219	80 DB	BRA -37 --> &A05C
A081		169 000	A9 00	LDA#&00
A083		128 002	80 02	BRA 2 --> &A087
A085		169 005	A9 05	LDA#&05
A087		133 020	85 14	STA &14
A089		162 004	A2 04	LDX#&04
A08B	t?	116 063	74 3F	STZ &3F,X
A08D	8	056	38	SEC
A08E	*	165 042	A5 2A	LDA &2A
A090	&	253 038 128	FD 26 80	SBC &8026,X
A093		168	A8	TAY
A094	+	165 043	A5 2B	LDA &2B
A096	!	253 033 128	FD 21 80	SBC &8021,X
A099		144 008	90 08	BCC 8 --> &A0A3
A09B	+	133 043	85 2B	STA &2B
A09D	*	132 042	84 2A	STY &2A
A09F	?	246 063	F6 3F	INC &3F,X
A0A1		128 235	80 EB	BRA -21 --> &A08E
A0A3		202	CA	DEX
A0A4		016 229	10 E5	BPL -27 --> &A08B
A0A6		162 005	A2 05	LDX#&05
A0A8		202	CA	DEX
A0A9		240 004	F0 04	BEQ 4 --> &A0AF
A0AB	?	181 063	B5 3F	LDA &3F,X
A0AD		240 249	F0 F9	BEQ -7 --> &A0A8
A0AF	7	134 055	86 37	STX &37
A0B1		165 020	A5 14	LDA &14
A0B3		240 010	F0 0A	BEQ 10 --> &A0BF
A0B5	7	229 055	E5 37	SBC &37

A0B7		240 006	F0 06	BEQ 6 --> &A0BF
A0B9		170	AA	TAX
A0BA		032 191 189	20 BF BD	JSR &BDBF
A0BD	7	166 055	A6 37	LDX &37
A0BF	?	181 063	B5 3F	LDA &3F,X
A0C1	0	009 048	09 30	ORA#&30
A0C3		032 148 189	20 94 BD	JSR &BD94
A0C6		202	CA	DEX
A0C7		016 246	10 F6	BPL -10 --> &A0BF
A0C9	`	096	60	RTS
A0CA		152	98	TYA
A0CB		016 003	10 03	BPL 3 --> &A0D0
A0CD		032 195 150	20 C3 96	JSR &96C3
A0D0		162 000	A2 00	LDX#&00
A0D2		160 000	A0 00	LDY#&00
A0D4	*	185 042 000	B9 2A 00	LDA &002A,Y
A0D7	H	072	48	PHA
A0D8	)	041 015	29 0F	AND#&0F
A0DA	?	149 063	95 3F	STA &3F,X
A0DC	h	104	68	PLA
A0DD	J	074	4A	LSR A
A0DE	J	074	4A	LSR A
A0DF	J	074	4A	LSR A
A0E0	J	074	4A	LSR A
A0E1		232	E8	INX
A0E2	?	149 063	95 3F	STA &3F,X
A0E4		232	E8	INX
A0E5		200	C8	INY
A0E6		192 004	C0 04	CPY#&04
A0E8		208 234	D0 EA	BNE -22 --> &A0D4
A0EA		202	CA	DEX
A0EB		240 004	F0 04	BEQ 4 --> &A0F1
A0ED	?	181 063	B5 3F	LDA &3F,X
A0EF		240 249	F0 F9	BEQ -7 --> &A0EA
A0F1	?	181 063	B5 3F	LDA &3F,X

A0F3		201 010	C9 0A	CMP#&0A
A0F5		144 002	90 02	BCC 2 --> &A0F9
A0F7	i	105 006	69 06	ADC#&06
A0F9	i0	105 048	69 30	ADC#&30
A0FB		032 208 162	20 D0 A2	JSR &A2D0
A0FE		202	CA	DEX
A0FF		016 240	10 F0	BPL -16 --> &A0F1
A101	`	096	60	RTS
A102		016 007	10 07	BPL 7 --> &A10B
A104	-	169 045	A9 2D	LDA#&2D
A106	d.	100 046	64 2E	STZ &2E
A108		032 208 162	20 D0 A2	JSR &A2D0
A10B	0	165 048	A5 30	LDA &30
A10D		201 129	C9 81	CMP#&81
A10F	K	176 075	B0 4B	BCS 75 --> &A15C
A111	6	032 054 164	20 36 A4	JSR &A436
A114	H	198 072	C6 48	DEC &48
A116		128 243	80 F3	BRA -13 --> &A10B
A118		174 002 004	AE 02 04	LDX &0402
A11B		224 003	E0 03	CPX#&03
A11D		144 002	90 02	BCC 2 --> &A121
A11F		162 000	A2 00	LDX#&00
A121	7	134 055	86 37	STX &37
A123		173 001 004	AD 01 04	LDA &0401
A126		240 006	F0 06	BEQ 6 --> &A12E
A128		201 010	C9 0A	CMP#&0A
A12A		176 006	B0 06	BCS 6 --> &A132
A12C		128 006	80 06	BRA 6 --> &A134
A12E		224 002	E0 02	CPX#&02
A130		240 002	F0 02	BEQ 2 --> &A134
A132		169 010	A9 0A	LDA#&0A
A134	8	133 056	85 38	STA &38
A136	M	133 077	85 4D	STA &4D
A138	d6	100 054	64 36	STZ &36
A13A	dH	100 072	64 48	STZ &48

A13C	$	036 021	24 15	BIT &15
A13E	0	048 138	30 8A	BMI -118 --> &A0CA
A140		152	98	TYA
A141	0	048 003	30 03	BMI 3 --> &A146
A143		032 133 129	20 85 81	JSR &8185
A146		032 242 163	20 F2 A3	JSR &A3F2
A149		208 183	D0 B7	BNE -73 --> &A102
A14B	7	165 055	A5 37	LDA &37
A14D		208 005	D0 05	BNE 5 --> &A154
A14F	0	169 048	A9 30	LDA#&30
A151	L	076 208 162	4C D0 A2	JMP &A2D0
A154	L	076 208 161	4C D0 A1	JMP &A1D0
A157		032 216 165	20 D8 A5	JSR &A5D8
A15A		128 015	80 0F	BRA 15 --> &A16B
A15C		201 132	C9 84	CMP#&84
A15E		144 015	90 0F	BCC 15 --> &A16F
A160		208 006	D0 06	BNE 6 --> &A168
A162	1	165 049	A5 31	LDA &31
A164		201 160	C9 A0	CMP#&A0
A166		144 007	90 07	BCC 7 --> &A16F
A168	x	032 120 164	20 78 A4	JSR &A478
A16B	H	230 072	E6 48	INC &48
A16D		128 156	80 9C	BRA -100 --> &A10B
A16F	5	165 053	A5 35	LDA &35
A171	'	133 039	85 27	STA &27
A173		032 017 165	20 11 A5	JSR &A511
A176	M	165 077	A5 4D	LDA &4D
A178	8	133 056	85 38	STA &38
A17A	7	166 055	A6 37	LDX &37
A17C		224 002	E0 02	CPX#&02
A17E		208 016	D0 10	BNE 16 --> &A190
A180	eH	101 072	65 48	ADC &48
A182	0P	048 080	30 50	BMI 80 --> &A1D4
A184	8	133 056	85 38	STA &38
A186		201 011	C9 0B	CMP#&0B

A188		144 006	90 06	BCC 6 --> &A190
A18A		169 010	A9 0A	LDA#&0A
A18C	8	133 056	85 38	STA &38
A18E	d7	100 055	64 37	STZ &37
A190		032 184 166	20 B8 A6	JSR &A6B8
A193		169 160	A9 A0	LDA#&A0
A195	1	133 049	85 31	STA &31
A197		169 131	A9 83	LDA#&83
A199	0	133 048	85 30	STA &30
A19B	8	166 056	A6 38	LDX &38
A19D		240 006	F0 06	BEQ 6 --> &A1A5
A19F	x	032 120 164	20 78 A4	JSR &A478
A1A2		202	CA	DEX
A1A3		208 250	D0 FA	BNE -6 --> &A19F
A1A5		032 146 165	20 92 A5	JSR &A592
A1A8		032 224 164	20 E0 A4	JSR &A4E0
A1AB	'	165 039	A5 27	LDA &27
A1AD	A	133 065	85 41	STA &41
A1AF	h	032 104 131	20 68 83	JSR &8368
A1B2	0	165 048	A5 30	LDA &30
A1B4		201 132	C9 84	CMP#&84
A1B6		176 014	B0 0E	BCS 14 --> &A1C6
A1B8	f1	102 049	66 31	ROR &31
A1BA	f2	102 050	66 32	ROR &32
A1BC	f3	102 051	66 33	ROR &33
A1BE	f4	102 052	66 34	ROR &34
A1C0	f5	102 053	66 35	ROR &35
A1C2	0	230 048	E6 30	INC &30
A1C4		208 236	D0 EC	BNE -20 --> &A1B2
A1C6	1	165 049	A5 31	LDA &31
A1C8		201 160	C9 A0	CMP#&A0
A1CA		176 139	B0 8B	BCS -117 --> &A157
A1CC	8	165 056	A5 38	LDA &38
A1CE		208 014	D0 0E	BNE 14 --> &A1DE
A1D0		201 001	C9 01	CMP#&01

A1D2	A	240 065	F0 41	BEQ 65 --> &A215
A1D4		032 180 166	20 B4 A6	JSR &A6B4
A1D7	dH	100 072	64 48	STZ &48
A1D9	M	165 077	A5 4D	LDA &4D
A1DB		026	1A	INC A
A1DC	8	133 056	85 38	STA &38
A1DE		169 001	A9 01	LDA#&01
A1E0	7	197 055	C5 37	CMP &37
A1E2	1	240 049	F0 31	BEQ 49 --> &A215
A1E4	H	164 072	A4 48	LDY &48
A1E6	0	048 010	30 0A	BMI 10 --> &A1F2
A1E8	8	196 056	C4 38	CPY &38
A1EA	)	176 041	B0 29	BCS 41 --> &A215
A1EC	dH	100 072	64 48	STZ &48
A1EE		200	C8	INY
A1EF		152	98	TYA
A1F0	#	208 035	D0 23	BNE 35 --> &A215
A1F2	7	165 055	A5 37	LDA &37
A1F4		201 002	C9 02	CMP#&02
A1F6		240 006	F0 06	BEQ 6 --> &A1FE
A1F8		169 001	A9 01	LDA#&01
A1FA		192 255	C0 FF	CPY#&FF
A1FC		208 023	D0 17	BNE 23 --> &A215
A1FE	0	169 048	A9 30	LDA#&30
A200		032 208 162	20 D0 A2	JSR &A2D0
A203	.	169 046	A9 2E	LDA#&2E
A205		032 208 162	20 D0 A2	JSR &A2D0
A208	0	169 048	A9 30	LDA#&30
A20A	H	230 072	E6 48	INC &48
A20C		240 005	F0 05	BEQ 5 --> &A213
A20E		032 208 162	20 D0 A2	JSR &A2D0
A211		128 247	80 F7	BRA -9 --> &A20A
A213		169 128	A9 80	LDA#&80
A215	M	133 077	85 4D	STA &4D
A217	l	032 108 162	20 6C A2	JSR &A26C

A21A	M	198 077	C6 4D	DEC &4D
A21C		208 005	D0 05	BNE 5 --> &A223
A21E	.	169 046	A9 2E	LDA#&2E
A220		032 208 162	20 D0 A2	JSR &A2D0
A223	8	198 056	C6 38	DEC &38
A225		208 240	D0 F0	BNE -16 --> &A217
A227	7	164 055	A4 37	LDY &37
A229		136	88	DEY
A22A		240 024	F0 18	BEQ 24 --> &A244
A22C		136	88	DEY
A22D		240 017	F0 11	BEQ 17 --> &A240
A22F	6	164 054	A4 36	LDY &36
A231		136	88	DEY
A232		185 000 006	B9 00 06	LDA &0600,Y
A235	0	201 048	C9 30	CMP#&30
A237		240 248	F0 F8	BEQ -8 --> &A231
A239	.	201 046	C9 2E	CMP#&2E
A23B		240 001	F0 01	BEQ 1 --> &A23E
A23D		200	C8	INY
A23E	6	132 054	84 36	STY &36
A240	H	165 072	A5 48	LDA &48
A242	'	240 039	F0 27	BEQ 39 --> &A26B
A244	E	169 069	A9 45	LDA#&45
A246		032 208 162	20 D0 A2	JSR &A2D0
A249	H	165 072	A5 48	LDA &48
A24B		016 010	10 0A	BPL 10 --> &A257
A24D	-	169 045	A9 2D	LDA#&2D
A24F		032 208 162	20 D0 A2	JSR &A2D0
A252	8	056	38	SEC
A253		169 000	A9 00	LDA#&00
A255	H	229 072	E5 48	SBC &48
A257		032 188 162	20 BC A2	JSR &A2BC
A25A	7	165 055	A5 37	LDA &37
A25C		240 013	F0 0D	BEQ 13 --> &A26B
A25E		169 032	A9 20	LDA#&20

A260	H	164 072	A4 48	LDY &48
A262	0	048 003	30 03	BMI 3 --> &A267
A264		032 208 162	20 D0 A2	JSR &A2D0
A267		224 000	E0 00	CPX#&00
A269	e	240 101	F0 65	BEQ 101 --> &A2D0
A26B	`	096	60	RTS
A26C	1	165 049	A5 31	LDA &31
A26E	J	074	4A	LSR A
A26F	J	074	4A	LSR A
A270	J	074	4A	LSR A
A271	J	074	4A	LSR A
A272		032 206 162	20 CE A2	JSR &A2CE
A275		169 240	A9 F0	LDA#&F0
A277	1	020 049	14 31	TRB &31
A279	H	072	48	PHA
A27A	4	166 052	A6 34	LDX &34
A27C	1	165 049	A5 31	LDA &31
A27E	H	072	48	PHA
A27F	2	165 050	A5 32	LDA &32
A281	H	072	48	PHA
A282	3	165 051	A5 33	LDA &33
A284	H	072	48	PHA
A285	5	165 053	A5 35	LDA &35
A287		010	0A	ASL A
A288	&4	038 052	26 34	ROL &34
A28A	&3	038 051	26 33	ROL &33
A28C	&2	038 050	26 32	ROL &32
A28E	&1	038 049	26 31	ROL &31
A290		010	0A	ASL A
A291	&4	038 052	26 34	ROL &34
A293	&3	038 051	26 33	ROL &33
A295	&2	038 050	26 32	ROL &32
A297	&1	038 049	26 31	ROL &31
A299	e5	101 053	65 35	ADC &35
A29B	5	133 053	85 35	STA &35

A29D		138	8A	TXA
A29E	e4	101 052	65 34	ADC &34
A2A0	4	133 052	85 34	STA &34
A2A2	h	104	68	PLA
A2A3	e3	101 051	65 33	ADC &33
A2A5	3	133 051	85 33	STA &33
A2A7	h	104	68	PLA
A2A8	e2	101 050	65 32	ADC &32
A2AA	2	133 050	85 32	STA &32
A2AC	h	104	68	PLA
A2AD	e1	101 049	65 31	ADC &31
A2AF	5	006 053	06 35	ASL &35
A2B1	&4	038 052	26 34	ROL &34
A2B3	&3	038 051	26 33	ROL &33
A2B5	&2	038 050	26 32	ROL &32
A2B7	*	042	2A	ROL A
A2B8	1	133 049	85 31	STA &31
A2BA	h	104	68	PLA
A2BB	`	096	60	RTS
A2BC		162 255	A2 FF	LDX#&FF
A2BE	8	056	38	SEC
A2BF		232	E8	INX
A2C0		233 010	E9 0A	SBC#&0A
A2C2		176 251	B0 FB	BCS -5 --> &A2BF
A2C4	i	105 010	69 0A	ADC#&0A
A2C6	H	072	48	PHA
A2C7		138	8A	TXA
A2C8		240 003	F0 03	BEQ 3 --> &A2CD
A2CA		032 206 162	20 CE A2	JSR &A2CE
A2CD	h	104	68	PLA
A2CE	0	009 048	09 30	ORA#&30
A2D0		218	DA	PHX
A2D1	6	166 054	A6 36	LDX &36
A2D3		157 000 006	9D 00 06	STA &0600,X
A2D6		250	FA	PLX

A2D7	6	230 054	E6 36	INC &36
A2D9	`	096	60	RTS
A2DA		032 004 164	20 04 A4	JSR &A404
A2DD		024	18	CLC
A2DE		169 255	A9 FF	LDA#&FF
A2E0	`	096	60	RTS
A2E1	d1	100 049	64 31	STZ &31
A2E3	d2	100 050	64 32	STZ &32
A2E5	d3	100 051	64 33	STZ &33
A2E7	d4	100 052	64 34	STZ &34
A2E9	d5	100 053	64 35	STZ &35
A2EB	dG	100 071	64 47	STZ &47
A2ED	dH	100 072	64 48	STZ &48
A2EF	.	201 046	C9 2E	CMP#&2E
A2F1	)	240 041	F0 29	BEQ 41 --> &A31C
A2F3	:	201 058	C9 3A	CMP#&3A
A2F5		176 227	B0 E3	BCS -29 --> &A2DA
A2F7	/	233 047	E9 2F	SBC#&2F
A2F9	0	048 223	30 DF	BMI -33 --> &A2DA
A2FB	5	133 053	85 35	STA &35
A2FD		200	C8	INY
A2FE		177 025	B1 19	LDA (&19),Y
A300	:	201 058	C9 3A	CMP#&3A
A302		176 032	B0 20	BCS 32 --> &A324
A304	/	233 047	E9 2F	SBC#&2F
A306		144 014	90 0E	BCC 14 --> &A316
A308	.	133 046	85 2E	STA &2E
A30A	5	165 053	A5 35	LDA &35
A30C		010	0A	ASL A
A30D		010	0A	ASL A
A30E	e5	101 053	65 35	ADC &35
A310		010	0A	ASL A
A311	e.	101 046	65 2E	ADC &2E
A313	5	133 053	85 35	STA &35
A315		200	C8	INY

A316		177 025	B1 19	LDA (&19),Y
A318	.	201 046	C9 2E	CMP#&2E
A31A		208 008	D0 08	BNE 8 --> &A324
A31C	G	165 071	A5 47	LDA &47
A31E	D	208 068	D0 44	BNE 68 --> &A364
A320	G	230 071	E6 47	INC &47
A322		128 241	80 F1	BRA -15 --> &A315
A324	E	201 069	C9 45	CMP#&45
A326	5	240 053	F0 35	BEQ 53 --> &A35D
A328	:	201 058	C9 3A	CMP#&3A
A32A	8	176 056	B0 38	BCS 56 --> &A364
A32C	/	233 047	E9 2F	SBC#&2F
A32E	4	144 052	90 34	BCC 52 --> &A364
A330	1	166 049	A6 31	LDX &31
A332		224 024	E0 18	CPX#&18
A334		144 008	90 08	BCC 8 --> &A33E
A336	G	166 071	A6 47	LDX &47
A338		208 219	D0 DB	BNE -37 --> &A315
A33A	H	230 072	E6 48	INC &48
A33C		128 215	80 D7	BRA -41 --> &A315
A33E	G	166 071	A6 47	LDX &47
A340		240 002	F0 02	BEQ 2 --> &A344
A342	H	198 072	C6 48	DEC &48
A344	y	032 121 162	20 79 A2	JSR &A279
A347	e5	101 053	65 35	ADC &35
A349	5	133 053	85 35	STA &35
A34B		144 200	90 C8	BCC -56 --> &A315
A34D	4	230 052	E6 34	INC &34
A34F		208 196	D0 C4	BNE -60 --> &A315
A351	3	230 051	E6 33	INC &33
A353		208 192	D0 C0	BNE -64 --> &A315
A355	2	230 050	E6 32	INC &32
A357		208 188	D0 BC	BNE -68 --> &A315
A359	1	230 049	E6 31	INC &31
A35B		128 184	80 B8	BRA -72 --> &A315

A35D		032 186 163	20 BA A3	JSR &A3BA
A360	eH	101 072	65 48	ADC &48
A362	H	133 072	85 48	STA &48
A364		132 027	84 1B	STY &1B
A366	H	165 072	A5 48	LDA &48
A368	G	005 071	05 47	ORA &47
A36A	-	240 045	F0 2D	BEQ 45 --> &A399
A36C		032 242 163	20 F2 A3	JSR &A3F2
A36F	$	240 036	F0 24	BEQ 36 --> &A395
A371		169 168	A9 A8	LDA#&A8
A373	0	133 048	85 30	STA &30
A375	d/	100 047	64 2F	STZ &2F
A377	d.	100 046	64 2E	STZ &2E
A379		032 247 129	20 F7 81	JSR &81F7
A37C	H	165 072	A5 48	LDA &48
A37E	0	048 011	30 0B	BMI 11 --> &A38B
A380		240 016	F0 10	BEQ 16 --> &A392
A382	6	032 054 164	20 36 A4	JSR &A436
A385	H	198 072	C6 48	DEC &48
A387		208 249	D0 F9	BNE -7 --> &A382
A389		128 007	80 07	BRA 7 --> &A392
A38B	x	032 120 164	20 78 A4	JSR &A478
A38E	H	230 072	E6 48	INC &48
A390		208 249	D0 F9	BNE -7 --> &A38B
A392		032 149 166	20 95 A6	JSR &A695
A395	8	056	38	SEC
A396		169 255	A9 FF	LDA#&FF
A398	`	096	60	RTS
A399	2	165 050	A5 32	LDA &32
A39B	-	133 045	85 2D	STA &2D
A39D	)	041 128	29 80	AND#&80
A39F	1	005 049	05 31	ORA &31
A3A1		208 206	D0 CE	BNE -50 --> &A371
A3A3	5	165 053	A5 35	LDA &35
A3A5	*	133 042	85 2A	STA &2A

A3A7	4	165 052	A5 34	LDA &34
A3A9	+	133 043	85 2B	STA &2B
A3AB	3	165 051	A5 33	LDA &33
A3AD	,	133 044	85 2C	STA &2C
A3AF	@	169 064	A9 40	LDA#&40
A3B1	8	056	38	SEC
A3B2	`	096	60	RTS
A3B3		032 197 163	20 C5 A3	JSR &A3C5
A3B6	I	073 255	49 FF	EOR#&FF
A3B8	8	056	38	SEC
A3B9	`	096	60	RTS
A3BA		200	C8	INY
A3BB		177 025	B1 19	LDA (&19),Y
A3BD	-	201 045	C9 2D	CMP#&2D
A3BF		240 242	F0 F2	BEQ -14 --> &A3B3
A3C1	+	201 043	C9 2B	CMP#&2B
A3C3		208 003	D0 03	BNE 3 --> &A3C8
A3C5		200	C8	INY
A3C6		177 025	B1 19	LDA (&19),Y
A3C8	:	201 058	C9 3A	CMP#&3A
A3CA	"	176 034	B0 22	BCS 34 --> &A3EE
A3CC	/	233 047	E9 2F	SBC#&2F
A3CE		144 030	90 1E	BCC 30 --> &A3EE
A3D0	I	133 073	85 49	STA &49
A3D2		200	C8	INY
A3D3		177 025	B1 19	LDA (&19),Y
A3D5	:	201 058	C9 3A	CMP#&3A
A3D7		176 017	B0 11	BCS 17 --> &A3EA
A3D9	/	233 047	E9 2F	SBC#&2F
A3DB		144 013	90 0D	BCC 13 --> &A3EA
A3DD		200	C8	INY
A3DE	B	133 066	85 42	STA &42
A3E0	I	165 073	A5 49	LDA &49
A3E2		010	0A	ASL A
A3E3		010	0A	ASL A

A3E4	eI	101 073	65 49	ADC &49
A3E6		010	0A	ASL A
A3E7	eB	101 066	65 42	ADC &42
A3E9	`	096	60	RTS
A3EA	I	165 073	A5 49	LDA &49
A3EC		024	18	CLC
A3ED	`	096	60	RTS
A3EE		169 000	A9 00	LDA#&00
A3F0		024	18	CLC
A3F1	`	096	60	RTS
A3F2	1	165 049	A5 31	LDA &31
A3F4	2	005 050	05 32	ORA &32
A3F6	3	005 051	05 33	ORA &33
A3F8	4	005 052	05 34	ORA &34
A3FA	5	005 053	05 35	ORA &35
A3FC		240 006	F0 06	BEQ 6 --> &A404
A3FE	.	165 046	A5 2E	LDA &2E
A400		208 008	D0 08	BNE 8 --> &A40A
A402		026	1A	INC A
A403	`	096	60	RTS
A404	d.	100 046	64 2E	STZ &2E
A406	d0	100 048	64 30	STZ &30
A408	d/	100 047	64 2F	STZ &2F
A40A	`	096	60	RTS
A40B	.	165 046	A5 2E	LDA &2E
A40D	;	133 059	85 3B	STA &3B
A40F	0	165 048	A5 30	LDA &30
A411	<	133 060	85 3C	STA &3C
A413	1	165 049	A5 31	LDA &31
A415	=	133 061	85 3D	STA &3D
A417	2	165 050	A5 32	LDA &32
A419	>	133 062	85 3E	STA &3E
A41B	3	165 051	A5 33	LDA &33
A41D	?	133 063	85 3F	STA &3F
A41F	4	165 052	A5 34	LDA &34

A421	@	133 064	85 40	STA &40
A423	5	165 053	A5 35	LDA &35
A425	A	133 065	85 41	STA &41
A427	`	096	60	RTS
A428		032 011 164	20 0B A4	JSR &A40B
A42B	F=	070 061	46 3D	LSR &3D
A42D	f>	102 062	66 3E	ROR &3E
A42F	f?	102 063	66 3F	ROR &3F
A431	f@	102 064	66 40	ROR &40
A433	fA	102 065	66 41	ROR &41
A435	`	096	60	RTS
A436		024	18	CLC
A437	0	165 048	A5 30	LDA &30
A439	i	105 003	69 03	ADC#&03
A43B	0	133 048	85 30	STA &30
A43D		144 002	90 02	BCC 2 --> &A441
A43F	/	230 047	E6 2F	INC &2F
A441	(	032 040 164	20 28 A4	JSR &A428
A444	+	032 043 164	20 2B A4	JSR &A42B
A447	5	165 053	A5 35	LDA &35
A449	eA	101 065	65 41	ADC &41
A44B	5	133 053	85 35	STA &35
A44D	4	165 052	A5 34	LDA &34
A44F	e@	101 064	65 40	ADC &40
A451	4	133 052	85 34	STA &34
A453	3	165 051	A5 33	LDA &33
A455	e?	101 063	65 3F	ADC &3F
A457	3	133 051	85 33	STA &33
A459	2	165 050	A5 32	LDA &32
A45B	e>	101 062	65 3E	ADC &3E
A45D	2	133 050	85 32	STA &32
A45F	1	165 049	A5 31	LDA &31
A461	e=	101 061	65 3D	ADC &3D
A463	1	133 049	85 31	STA &31
A465		144 016	90 10	BCC 16 --> &A477

A467	f1	102 049	66 31	ROR &31
A469	f2	102 050	66 32	ROR &32
A46B	f3	102 051	66 33	ROR &33
A46D	f4	102 052	66 34	ROR &34
A46F	f5	102 053	66 35	ROR &35
A471	0	230 048	E6 30	INC &30
A473		208 002	D0 02	BNE 2 --> &A477
A475	/	230 047	E6 2F	INC &2F
A477	`	096	60	RTS
A478	8	056	38	SEC
A479	0	165 048	A5 30	LDA &30
A47B		233 004	E9 04	SBC#&04
A47D	0	133 048	85 30	STA &30
A47F		176 002	B0 02	BCS 2 --> &A483
A481	/	198 047	C6 2F	DEC &2F
A483	(	032 040 164	20 28 A4	JSR &A428
A486	G	032 071 164	20 47 A4	JSR &A447
A489	(	032 040 164	20 28 A4	JSR &A428
A48C	+	032 043 164	20 2B A4	JSR &A42B
A48F	+	032 043 164	20 2B A4	JSR &A42B
A492	+	032 043 164	20 2B A4	JSR &A42B
A495	G	032 071 164	20 47 A4	JSR &A447
A498	d=	100 061	64 3D	STZ &3D
A49A	1	165 049	A5 31	LDA &31
A49C	>	133 062	85 3E	STA &3E
A49E	2	165 050	A5 32	LDA &32
A4A0	?	133 063	85 3F	STA &3F
A4A2	3	165 051	A5 33	LDA &33
A4A4	@	133 064	85 40	STA &40
A4A6	4	165 052	A5 34	LDA &34
A4A8	A	133 065	85 41	STA &41
A4AA	5	165 053	A5 35	LDA &35
A4AC	*	042	2A	ROL A
A4AD	G	032 071 164	20 47 A4	JSR &A447
A4B0	d>	100 062	64 3E	STZ &3E

A4B2	1	165 049	A5 31	LDA &31
A4B4	?	133 063	85 3F	STA &3F
A4B6	2	165 050	A5 32	LDA &32
A4B8	@	133 064	85 40	STA &40
A4BA	3	165 051	A5 33	LDA &33
A4BC	A	133 065	85 41	STA &41
A4BE	4	165 052	A5 34	LDA &34
A4C0	*	042	2A	ROL A
A4C1	G	032 071 164	20 47 A4	JSR &A447
A4C4	2	165 050	A5 32	LDA &32
A4C6	*	042	2A	ROL A
A4C7	1	165 049	A5 31	LDA &31
A4C9	e5	101 053	65 35	ADC &35
A4CB	5	133 053	85 35	STA &35
A4CD		144 016	90 10	BCC 16 --> &A4DF
A4CF	4	230 052	E6 34	INC &34
A4D1		208 012	D0 0C	BNE 12 --> &A4DF
A4D3	3	230 051	E6 33	INC &33
A4D5		208 008	D0 08	BNE 8 --> &A4DF
A4D7	2	230 050	E6 32	INC &32
A4D9		208 004	D0 04	BNE 4 --> &A4DF
A4DB	1	230 049	E6 31	INC &31
A4DD		240 136	F0 88	BEQ -120 --> &A467
A4DF	`	096	60	RTS
A4E0	dA	100 065	64 41	STZ &41
A4E2		160 004	A0 04	LDY#&04
A4E4	J	177 074	B1 4A	LDA (&4A),Y
A4E6	@	133 064	85 40	STA &40
A4E8		136	88	DEY
A4E9	J	177 074	B1 4A	LDA (&4A),Y
A4EB	?	133 063	85 3F	STA &3F
A4ED		136	88	DEY
A4EE	J	177 074	B1 4A	LDA (&4A),Y
A4F0	>	133 062	85 3E	STA &3E
A4F2		136	88	DEY

A4F3	J	177 074	B1 4A	LDA (&4A),Y
A4F5	;	133 059	85 3B	STA &3B
A4F7		168	A8	TAY
A4F8	J	178 074	B2 4A	LDA (&4A)
A4FA	<	133 060	85 3C	STA &3C
A4FC		208 009	D0 09	BNE 9 --> &A507
A4FE		152	98	TYA
A4FF	>	005 062	05 3E	ORA &3E
A501	?	005 063	05 3F	ORA &3F
A503	@	005 064	05 40	ORA &40
A505		240 003	F0 03	BEQ 3 --> &A50A
A507		152	98	TYA
A508		009 128	09 80	ORA#&80
A50A	=	133 061	85 3D	STA &3D
A50C	`	096	60	RTS
A50D	v	169 118	A9 76	LDA#&76
A50F		128 002	80 02	BRA 2 --> &A513
A511	l	169 108	A9 6C	LDA#&6C
A513	J	133 074	85 4A	STA &4A
A515		169 004	A9 04	LDA#&04
A517	K	133 075	85 4B	STA &4B
A519	0	165 048	A5 30	LDA &30
A51B	J	146 074	92 4A	STA (&4A)
A51D		160 001	A0 01	LDY#&01
A51F	.	165 046	A5 2E	LDA &2E
A521	E1	069 049	45 31	EOR &31
A523	)	041 128	29 80	AND#&80
A525	E1	069 049	45 31	EOR &31
A527	J	145 074	91 4A	STA (&4A),Y
A529	2	165 050	A5 32	LDA &32
A52B		200	C8	INY
A52C	J	145 074	91 4A	STA (&4A),Y
A52E	3	165 051	A5 33	LDA &33
A530		200	C8	INY
A531	J	145 074	91 4A	STA (&4A),Y

A533	4	165 052	A5 34	LDA &34
A535		200	C8	INY
A536	J	145 074	91 4A	STA (&4A),Y
A538	`	096	60	RTS
A539	l	169 108	A9 6C	LDA#&6C
A53B	J	133 074	85 4A	STA &4A
A53D		169 004	A9 04	LDA#&04
A53F	K	133 075	85 4B	STA &4B
A541	d5	100 053	64 35	STZ &35
A543	d/	100 047	64 2F	STZ &2F
A545		160 004	A0 04	LDY#&04
A547	J	177 074	B1 4A	LDA (&4A),Y
A549	4	133 052	85 34	STA &34
A54B		136	88	DEY
A54C	J	177 074	B1 4A	LDA (&4A),Y
A54E	3	133 051	85 33	STA &33
A550		136	88	DEY
A551	J	177 074	B1 4A	LDA (&4A),Y
A553	2	133 050	85 32	STA &32
A555		136	88	DEY
A556	J	177 074	B1 4A	LDA (&4A),Y
A558	.	133 046	85 2E	STA &2E
A55A		168	A8	TAY
A55B	J	178 074	B2 4A	LDA (&4A)
A55D	0	133 048	85 30	STA &30
A55F		208 009	D0 09	BNE 9 --> &A56A
A561		152	98	TYA
A562	2	005 050	05 32	ORA &32
A564	3	005 051	05 33	ORA &33
A566	4	005 052	05 34	ORA &34
A568		240 003	F0 03	BEQ 3 --> &A56D
A56A		152	98	TYA
A56B		009 128	09 80	ORA#&80
A56D	1	133 049	85 31	STA &31
A56F	`	096	60	RTS

A570	d;	100 059	64 3B	STZ &3B
A572	d<	100 060	64 3C	STZ &3C
A574	d=	100 061	64 3D	STZ &3D
A576	d>	100 062	64 3E	STZ &3E
A578	d?	100 063	64 3F	STZ &3F
A57A	d@	100 064	64 40	STZ &40
A57C	dA	100 065	64 41	STZ &41
A57E	`	096	60	RTS
A57F		024	18	CLC
A580	L	165 076	A5 4C	LDA &4C
A582	i	105 005	69 05	ADC#&05
A584	L	133 076	85 4C	STA &4C
A586	J	133 074	85 4A	STA &4A
A588	`	096	60	RTS
A589	.	169 046	A9 2E	LDA#&2E
A58B	J	133 074	85 4A	STA &4A
A58D		169 191	A9 BF	LDA#&BF
A58F	K	133 075	85 4B	STA &4B
A591	`	096	60	RTS
A592	l	169 108	A9 6C	LDA#&6C
A594	J	133 074	85 4A	STA &4A
A596		169 004	A9 04	LDA#&04
A598	K	133 075	85 4B	STA &4B
A59A	`	096	60	RTS
A59B	:	032 058 169	20 3A A9	JSR &A93A
A59E	{	169 123	A9 7B	LDA#&7B
A5A0		032 019 165	20 13 A5	JSR &A513
A5A3		032 023 169	20 17 A9	JSR &A917
A5A6	v	169 118	A9 76	LDA#&76
A5A8		032 019 165	20 13 A5	JSR &A513
A5AB	{	169 123	A9 7B	LDA#&7B
A5AD	;	032 059 165	20 3B A5	JSR &A53B
A5B0		032 021 169	20 15 A9	JSR &A915
A5B3	v	169 118	A9 76	LDA#&76
A5B5		032 148 165	20 94 A5	JSR &A594

A5B8		032 238 165	20 EE A5	JSR &A5EE
A5BB		169 255	A9 FF	LDA#&FF
A5BD	`	096	60	RTS
A5BE		170	AA	TAX
A5BF		016 008	10 08	BPL 8 --> &A5C9
A5C1	:	058	3A	DEC A
A5C2	I	073 255	49 FF	EOR#&FF
A5C4	H	072	48	PHA
A5C5		032 233 165	20 E9 A5	JSR &A5E9
A5C8		250	FA	PLX
A5C9		240 013	F0 0D	BEQ 13 --> &A5D8
A5CB		032 017 165	20 11 A5	JSR &A511
A5CE		202	CA	DEX
A5CF		240 006	F0 06	BEQ 6 --> &A5D7
A5D1		032 166 166	20 A6 A6	JSR &A6A6
A5D4		202	CA	DEX
A5D5		208 250	D0 FA	BNE -6 --> &A5D1
A5D7	`	096	60	RTS
A5D8		169 128	A9 80	LDA#&80
A5DA	1	133 049	85 31	STA &31
A5DC		026	1A	INC A
A5DD	0	133 048	85 30	STA &30
A5DF	L	076 184 166	4C B8 A6	JMP &A6B8
A5E2	Lr	076 114 129	4C 72 81	JMP &8172
A5E5		002	02	xxx Invalid Code
A5E6		008	08	PHP
A5E7		008	08	PHP
A5E8		008	08	PHP
A5E9		169 146	A9 92	LDA#&92
A5EB		032 139 165	20 8B A5	JSR &A58B
A5EE	1	165 049	A5 31	LDA &31
A5F0		240 240	F0 F0	BEQ -16 --> &A5E2
A5F2		032 224 164	20 E0 A4	JSR &A4E0
A5F5		208 003	D0 03	BNE 3 --> &A5FA
A5F7	L	076 180 166	4C B4 A6	JMP &A6B4

A5FA	;	165 059	A5 3B	LDA &3B
A5FC	E.	069 046	45 2E	EOR &2E
A5FE	.	133 046	85 2E	STA &2E
A600	8	056	38	SEC
A601	<	165 060	A5 3C	LDA &3C
A603	i	105 129	69 81	ADC#&81
A605	&/	038 047	26 2F	ROL &2F
A607	0	229 048	E5 30	SBC &30
A609		176 002	B0 02	BCS 2 --> &A60D
A60B	/	198 047	C6 2F	DEC &2F
A60D	0	133 048	85 30	STA &30
A60F		160 004	A0 04	LDY#&04
A611	<	132 060	84 3C	STY &3C
A613	=	165 061	A5 3D	LDA &3D
A615		162 008	A2 08	LDX#&08
A617		128 009	80 09	BRA 9 --> &A622
A619	C	150 067	96 43	STX &43,Y
A61B		190 229 165	BE E5 A5	LDX &A5E5,Y
A61E	<	132 060	84 3C	STY &3C
A620		176 022	B0 16	BCS 22 --> &A638
A622	1	197 049	C5 31	CMP &31
A624		208 016	D0 10	BNE 16 --> &A636
A626	>	164 062	A4 3E	LDY &3E
A628	2	196 050	C4 32	CPY &32
A62A		208 010	D0 0A	BNE 10 --> &A636
A62C	?	164 063	A4 3F	LDY &3F
A62E	3	196 051	C4 33	CPY &33
A630		208 004	D0 04	BNE 4 --> &A636
A632	@	164 064	A4 40	LDY &40
A634	4	196 052	C4 34	CPY &34
A636		144 023	90 17	BCC 23 --> &A64F
A638		168	A8	TAY
A639	@	165 064	A5 40	LDA &40
A63B	4	229 052	E5 34	SBC &34
A63D	@	133 064	85 40	STA &40

```
A63F    ?    165 063      A5 3F      LDA &3F
A641    3    229 051      E5 33      SBC &33
A643    ?    133 063      85 3F      STA &3F
A645    >    165 062      A5 3E      LDA &3E
A647    2    229 050      E5 32      SBC &32
A649    >    133 062      85 3E      STA &3E
A64B         152          98         TYA
A64C    1    229 049      E5 31      SBC &31
A64E    8    056          38         SEC
A64F    &;   038 059      26 3B      ROL &3B
A651    @    006 064      06 40      ASL &40
A653    &?   038 063      26 3F      ROL &3F
A655    &>   038 062      26 3E      ROL &3E
A657    *    042          2A         ROL A
A658         202          CA         DEX
A659         208 197      D0 C5      BNE -59 --> &A620
A65B    ;    166 059      A6 3B      LDX &3B
A65D    <    164 060      A4 3C      LDY &3C
A65F         136          88         DEY
A660         016 183      10 B7      BPL -73 --> &A619
A662    >    005 062      05 3E      ORA &3E
A664    ?    005 063      05 3F      ORA &3F
A666    @    005 064      05 40      ORA &40
A668         240 001      F0 01      BEQ 1 --> &A66B
A66A    8    056          38         SEC
A66B         138          8A         TXA
A66C    j    106          6A         ROR A
A66D    j    106          6A         ROR A
A66E    j    106          6A         ROR A
A66F    )    041 224      29 E0      AND#&E0
A671    5    133 053      85 35      STA &35
A673    C    165 067      A5 43      LDA &43
A675    4    133 052      85 34      STA &34
A677    D    165 068      A5 44      LDA &44
A679    3    133 051      85 33      STA &33
```

A67B	E	165 069	A5 45	LDA &45
A67D	2	133 050	85 32	STA &32
A67F	F	165 070	A5 46	LDA &46
A681	1	133 049	85 31	STA &31
A683	0	048 016	30 10	BMI 16 --> &A695
A685	*	032 042 130	20 2A 82	JSR &822A
A688		128 011	80 0B	BRA 11 --> &A695
A68A		032 202 172	20 CA AC	JSR &ACCA
A68D		032 224 164	20 E0 A4	JSR &A4E0
A690	2	240 050	F0 32	BEQ 50 --> &A6C4
A692	h	032 104 131	20 68 83	JSR &8368
A695	5	165 053	A5 35	LDA &35
A697		201 128	C9 80	CMP#&80
A699		144 019	90 13	BCC 19 --> &A6AE
A69B		240 014	F0 0E	BEQ 14 --> &A6AB
A69D	4	230 052	E6 34	INC &34
A69F		208 013	D0 0D	BNE 13 --> &A6AE
A6A1		032 211 164	20 D3 A4	JSR &A4D3
A6A4		128 008	80 08	BRA 8 --> &A6AE
A6A6		032 207 166	20 CF A6	JSR &A6CF
A6A9		128 234	80 EA	BRA -22 --> &A695
A6AB	*	042	2A	ROL A
A6AC	4	004 052	04 34	TSB &34
A6AE	/	165 047	A5 2F	LDA &2F
A6B0		240 016	F0 10	BEQ 16 --> &A6C2
A6B2		016 017	10 11	BPL 17 --> &A6C5
A6B4	d0	100 048	64 30	STZ &30
A6B6	d1	100 049	64 31	STZ &31
A6B8	d.	100 046	64 2E	STZ &2E
A6BA	d/	100 047	64 2F	STZ &2F
A6BC	d2	100 050	64 32	STZ &32
A6BE	d3	100 051	64 33	STZ &33
A6C0	d4	100 052	64 34	STZ &34
A6C2	d5	100 053	64 35	STZ &35
A6C4	`	096	60	RTS

A6C5		000	00	BRK
A6C6		020	14	EQUB &14
A6C7	T	084	54	xxx Invalid Code
A6C8	o	111	6F	xxx Invalid Code
A6C9	o	111	6F	xxx Invalid Code
A6CA	bi	032 098 105	20 62 69	JSR &6962
A6CD	g	103	67	xxx Invalid Code
A6CE		000	00	EQUB &00
A6CF	1	165 049	A5 31	LDA &31
A6D1		240 241	F0 F1	BEQ -15 --> &A6C4
A6D3		032 224 164	20 E0 A4	JSR &A4E0
A6D6		240 220	F0 DC	BEQ -36 --> &A6B4
A6D8		024	18	CLC
A6D9	0	165 048	A5 30	LDA &30
A6DB	e<	101 060	65 3C	ADC &3C
A6DD	&/	038 047	26 2F	ROL &2F
A6DF		233 127	E9 7F	SBC#&7F
A6E1	0	133 048	85 30	STA &30
A6E3		176 002	B0 02	BCS 2 --> &A6E7
A6E5	/	198 047	C6 2F	DEC &2F
A6E7	.	165 046	A5 2E	LDA &2E
A6E9	E;	069 059	45 3B	EOR &3B
A6EB	.	133 046	85 2E	STA &2E
A6ED		218	DA	PHX
A6EE		162 248	A2 F8	LDX#&F8
A6F0		160 004	A0 04	LDY#&04
A6F2	9	181 057	B5 39	LDA &39,X
A6F4	t9	116 057	74 39	STZ &39,X
A6F6	A	153 065 000	99 41 00	STA &0041,Y
A6F9		232	E8	INX
A6FA		136	88	DEY
A6FB		208 245	D0 F5	BNE -11 --> &A6F2
A6FD	d<	100 060	64 3C	STZ &3C
A6FF	d;	100 059	64 3B	STZ &3B
A701	d:	100 058	64 3A	STZ &3A

A703	0	128 048	80 30	BRA 48 --> &A735
A705		218	DA	PHX
A706	F=	070 061	46 3D	LSR &3D
A708	f>	102 062	66 3E	ROR &3E
A70A	f?	102 063	66 3F	ROR &3F
A70C	f@	102 064	66 40	ROR &40
A70E	fA	102 065	66 41	ROR &41
A710	F	022 070	16 46	ASL &46,X
A712		144 029	90 1D	BCC 29 --> &A731
A714		024	18	CLC
A715		152	98	TYA
A716	uB	117 066	75 42	ADC &42,X
A718		168	A8	TAY
A719	4	165 052	A5 34	LDA &34
A71B	uA	117 065	75 41	ADC &41,X
A71D	4	133 052	85 34	STA &34
A71F	3	165 051	A5 33	LDA &33
A721	u@	117 064	75 40	ADC &40,X
A723	3	133 051	85 33	STA &33
A725	2	165 050	A5 32	LDA &32
A727	u?	117 063	75 3F	ADC &3F,X
A729	2	133 050	85 32	STA &32
A72B	1	165 049	A5 31	LDA &31
A72D	u>	117 062	75 3E	ADC &3E,X
A72F	1	133 049	85 31	STA &31
A731		232	E8	INX
A732	0	048 220	30 DC	BMI -36 --> &A710
A734		250	FA	PLX
A735	F	181 070	B5 46	LDA &46,X
A737		208 204	D0 CC	BNE -52 --> &A705
A739		232	E8	INX
A73A	0	048 249	30 F9	BMI -7 --> &A735
A73C		250	FA	PLX
A73D	5	132 053	84 35	STY &35
A73F	1	165 049	A5 31	LDA &31

A741	0	048 129	30 81	BMI -127 --> &A6C4
A743	L	076 251 129	4C FB 81	JMP &81FB
A746		032 218 150	20 DA 96	JSR &96DA
A749		032 242 163	20 F2 A3	JSR &A3F2
A74C		240 002	F0 02	BEQ 2 --> &A750
A74E		016 022	10 16	BPL 22 --> &A766
A750		000	00	BRK
A751		022	16	EQUB &16
A752	L	076	4C	xxx Invalid Code
A753	o	111	6F	xxx Invalid Code
A754	g	103	67	xxx Invalid Code
A755	ra	032 114 097	20 72 61	JSR &6172
A758	nge	110 103 101	6E 67 65	ROR &6567
A75B		000	00	BRK
A75C		021	15	EQUB &15
A75D	-	045	2D	xxx Invalid Code
A75E	ve	118 101	76 65	ROR &65,X
A760	ro	032 114 111	20 72 6F	JSR &6F72
A763	o	111	6F	xxx Invalid Code
A764	t	116	74	xxx Invalid Code
A765		000	00	EQUB &00
A766	v	032 118 165	20 76 A5	JSR &A576
A769		160 128	A0 80	LDY#&80
A76B	;	132 059	84 3B	STY &3B
A76D	=	132 061	84 3D	STY &3D
A76F		200	C8	INY
A770	<	132 060	84 3C	STY &3C
A772	0	166 048	A6 30	LDX &30
A774		240 006	F0 06	BEQ 6 --> &A77C
A776	1	165 049	A5 31	LDA &31
A778		201 181	C9 B5	CMP#&B5
A77A		144 002	90 02	BCC 2 --> &A77E
A77C		232	E8	INX
A77D		136	88	DEY
A77E		218	DA	PHX

A77F	0	132 048	84 30	STY &30
A781		032 146 166	20 92 A6	JSR &A692
A784	{	169 123	A9 7B	LDA#&7B
A786		032 019 165	20 13 A5	JSR &A513
A789	Q	162 081	A2 51	LDX#&51
A78B	o	169 111	A9 6F	LDA#&6F
A78D		160 002	A0 02	LDY#&02
A78F	a	032 097 168	20 61 A8	JSR &A861
A792	{	169 123	A9 7B	LDA#&7B
A794		032 161 169	20 A1 A9	JSR &A9A1
A797		032 166 166	20 A6 A6	JSR &A6A6
A79A		032 141 166	20 8D A6	JSR &A68D
A79D		032 017 165	20 11 A5	JSR &A511
A7A0	h	104	68	PLA
A7A1	8	056	38	SEC
A7A2		233 129	E9 81	SBC#&81
A7A4		032 213 129	20 D5 81	JSR &81D5
A7A7	L	169 076	A9 4C	LDA#&4C
A7A9		032 212 169	20 D4 A9	JSR &A9D4
A7AC		032 146 165	20 92 A5	JSR &A592
A7AF		032 141 166	20 8D A6	JSR &A68D
A7B2		169 255	A9 FF	LDA#&FF
A7B4	`	096	60	RTS
A7B5		032 218 150	20 DA 96	JSR &96DA
A7B8		032 242 163	20 F2 A3	JSR &A3F2
A7BB		240 245	F0 F5	BEQ -11 --> &A7B2
A7BD	0	048 156	30 9C	BMI -100 --> &A75B
A7BF	0	165 048	A5 30	LDA &30
A7C1	J	074	4A	LSR A
A7C2		008	08	PHP
A7C3	iA	105 065	69 41	ADC#&41
A7C5	0	133 048	85 30	STA &30
A7C7	(	040	28	PLP
A7C8		144 010	90 0A	BCC 10 --> &A7D4
A7CA	F1	070 049	46 31	LSR &31

A7CC	f2	102 050	66 32	ROR &32
A7CE	f3	102 051	66 33	ROR &33
A7D0	f4	102 052	66 34	ROR &34
A7D2	f5	102 053	66 35	ROR &35
A7D4	p	032 112 165	20 70 A5	JSR &A570
A7D7	dC	100 067	64 43	STZ &43
A7D9	dD	100 068	64 44	STZ &44
A7DB	dE	100 069	64 45	STZ &45
A7DD	dF	100 070	64 46	STZ &46
A7DF	@	169 064	A9 40	LDA#&40
A7E1	=	133 061	85 3D	STA &3D
A7E3	B	133 066	85 42	STA &42
A7E5		162 251	A2 FB	LDX#&FB
A7E7		160 016	A0 10	LDY#&10
A7E9	8	056	38	SEC
A7EA	1	165 049	A5 31	LDA &31
A7EC	@	233 064	E9 40	SBC#&40
A7EE	1	133 049	85 31	STA &31
A7F0		152	98	TYA
A7F1	UB	085 066	55 42	EOR &42,X
A7F3	G	149 071	95 47	STA &47,X
A7F5	1	165 049	A5 31	LDA &31
A7F7	B	197 066	C5 42	CMP &42
A7F9		208 013	D0 0D	BNE 13 --> &A808
A7FB		218	DA	PHX
A7FC		162 252	A2 FC	LDX#&FC
A7FE	6	181 054	B5 36	LDA &36,X
A800	G	213 071	D5 47	CMP &47,X
A802		208 003	D0 03	BNE 3 --> &A807
A804		232	E8	INX
A805		208 247	D0 F7	BNE -9 --> &A7FE
A807		250	FA	PLX
A808	)	144 041	90 29	BCC 41 --> &A833
A80A	5	165 053	A5 35	LDA &35
A80C	F	229 070	E5 46	SBC &46

A80E	5	133 053	85 35	STA &35
A810	4	165 052	A5 34	LDA &34
A812	E	229 069	E5 45	SBC &45
A814	4	133 052	85 34	STA &34
A816	3	165 051	A5 33	LDA &33
A818	D	229 068	E5 44	SBC &44
A81A	3	133 051	85 33	STA &33
A81C	2	165 050	A5 32	LDA &32
A81E	C	229 067	E5 43	SBC &43
A820	2	133 050	85 32	STA &32
A822	1	165 049	A5 31	LDA &31
A824	B	229 066	E5 42	SBC &42
A826	1	133 049	85 31	STA &31
A828		152	98	TYA
A829		010	0A	ASL A
A82A		144 011	90 0B	BCC 11 --> &A837
A82C		026	1A	INC A
A82D	UA	085 065	55 41	EOR &41,X
A82F	A	149 065	95 41	STA &41,X
A831	F	149 070	95 46	STA &46,X
A833	B	181 066	B5 42	LDA &42,X
A835		128 004	80 04	BRA 4 --> &A83B
A837	UB	085 066	55 42	EOR &42,X
A839	B	149 066	95 42	STA &42,X
A83B	G	149 071	95 47	STA &47,X
A83D	5	006 053	06 35	ASL &35
A83F	&4	038 052	26 34	ROL &34
A841	&3	038 051	26 33	ROL &33
A843	&2	038 050	26 32	ROL &32
A845	&1	038 049	26 31	ROL &31
A847		152	98	TYA
A848	J	074	4A	LSR A
A849		168	A8	TAY
A84A		144 164	90 A4	BCC -92 --> &A7F0
A84C		160 128	A0 80	LDY#&80

A84E		232	E8	INX
A84F		208 159	D0 9F	BNE -97 --> &A7F0
A851	S	032 083 131	20 53 83	JSR &8353
A854	1	165 049	A5 31	LDA &31
A856	0	048 003	30 03	BMI 3 --> &A85B
A858		032 251 129	20 FB 81	JSR &81FB
A85B		032 149 166	20 95 A6	JSR &A695
A85E		169 255	A9 FF	LDA#&FF
A860	`	096	60	RTS
A861	G	132 071	84 47	STY &47
A863	L	134 076	86 4C	STX &4C
A865	0	166 048	A6 30	LDX &30
A867	@	224 064	E0 40	CPX#&40
A869	+	144 043	90 2B	BCC 43 --> &A896
A86B		032 233 165	20 E9 A5	JSR &A5E9
A86E		032 017 165	20 11 A5	JSR &A511
A871	L	165 076	A5 4C	LDA &4C
A873		032 139 165	20 8B A5	JSR &A58B
A876		032 141 166	20 8D A6	JSR &A68D
A879		032 134 168	20 86 A8	JSR &A886
A87C		032 146 165	20 92 A5	JSR &A592
A87F		032 141 166	20 8D A6	JSR &A68D
A882	G	198 071	C6 47	DEC &47
A884		208 243	D0 F3	BNE -13 --> &A879
A886		169 191	A9 BF	LDA#&BF
A888	K	133 075	85 4B	STA &4B
A88A		032 127 165	20 7F A5	JSR &A57F
A88D		032 238 165	20 EE A5	JSR &A5EE
A890		032 127 165	20 7F A5	JSR &A57F
A893	L	076 141 166	4C 8D A6	JMP &A68D
A896		032 139 165	20 8B A5	JSR &A58B
A899	LA	076 065 165	4C 41 A5	JMP &A541
A89C		032 161 168	20 A1 A8	JSR &A8A1
A89F	@	128 064	80 40	BRA 64 --> &A8E1
A8A1		032 218 150	20 DA 96	JSR &96DA

A8A4	.	165 046	A5 2E	LDA &2E
A8A6		016 007	10 07	BPL 7 --> &A8AF
A8A8	d.	100 046	64 2E	STZ &2E
A8AA		032 175 168	20 AF A8	JSR &A8AF
A8AD	#	128 035	80 23	BRA 35 --> &A8D2
A8AF		032 013 165	20 0D A5	JSR &A50D
A8B2	)	032 041 169	20 29 A9	JSR &A929
A8B5	1	165 049	A5 31	LDA &31
A8B7		240 005	F0 05	BEQ 5 --> &A8BE
A8B9		032 179 165	20 B3 A5	JSR &A5B3
A8BC		128 008	80 08	BRA 8 --> &A8C6
A8BE	.	169 046	A9 2E	LDA#&2E
A8C0	L	076 150 168	4C 96 A8	JMP &A896
A8C3		032 218 150	20 DA 96	JSR &96DA
A8C6		032 242 163	20 F2 A3	JSR &A3F2
A8C9	[	240 091	F0 5B	BEQ 91 --> &A926
A8CB		016 008	10 08	BPL 8 --> &A8D5
A8CD	d.	100 046	64 2E	STZ &2E
A8CF		032 213 168	20 D5 A8	JSR &A8D5
A8D2	.	133 046	85 2E	STA &2E
A8D4	`	096	60	RTS
A8D5	0	165 048	A5 30	LDA &30
A8D7		201 129	C9 81	CMP#&81
A8D9		144 015	90 0F	BCC 15 --> &A8EA
A8DB		032 233 165	20 E9 A5	JSR &A5E9
A8DE		032 234 168	20 EA A8	JSR &A8EA
A8E1		032 137 165	20 89 A5	JSR &A589
A8E4		032 138 166	20 8A A6	JSR &A68A
A8E7		169 255	A9 FF	LDA#&FF
A8E9	`	096	60	RTS
A8EA	0	165 048	A5 30	LDA &30
A8EC	s	201 115	C9 73	CMP#&73
A8EE	6	144 054	90 36	BCC 54 --> &A926
A8F0		032 013 165	20 0D A5	JSR &A50D
A8F3	v	032 118 165	20 76 A5	JSR &A576

```
A8F6          169 128          A9 80          LDA#&80
A8F8    <     133 060          85 3C          STA &3C
A8FA    =     133 061          85 3D          STA &3D
A8FC    ;     133 059          85 3B          STA &3B
A8FE          032 146 166      20 92 A6       JSR &A692
A901          162 151          A2 97          LDX#&97
A903          169 201          A9 C9          LDA#&C9
A905          160 004          A0 04          LDY#&04
A907    a     032 097 168      20 61 A8       JSR &A861
A90A    L     076 159 169      4C 9F A9       JMP &A99F
A90D          024              18             CLC
A90E          008              08             PHP
A90F    :     032 058 169      20 3A A9       JSR &A93A
A912    (     040              28             PLP
A913          144 002          90 02          BCC 2 --> &A917
A915    I     230 073          E6 49          INC &49
A917    I     165 073          A5 49          LDA &49
A919          137 002          89 02          BIT#&02
A91B          240 006          F0 06          BEQ 6 --> &A923
A91D    #     032 035 169      20 23 A9       JSR &A923
A920    L     076 202 172      4C CA AC       JMP &ACCA
A923    J     074              4A             LSR A
A924          176 003          B0 03          BCS 3 --> &A929
A926          169 255          A9 FF          LDA#&FF
A928    `     096              60             RTS
A929          032 017 165      20 11 A5       JSR &A511
A92C          032 166 166      20 A6 A6       JSR &A6A6
A92F          169 146          A9 92          LDA#&92
A931          032 139 165      20 8B A5       JSR &A58B
A934          032 138 166      20 8A A6       JSR &A68A
A937    L     076 184 167      4C B8 A7       JMP &A7B8
A93A          032 218 150      20 DA 96       JSR &96DA
A93D    0     165 048          A5 30          LDA &30
A93F          201 152          C9 98          CMP#&98
A941    j     176 106          B0 6A          BCS 106 --> &A9AD
```

A943		032 017 165	20 11 A5	JSR &A511
A946		032 137 165	20 89 A5	JSR &A589
A949		032 224 164	20 E0 A4	JSR &A4E0
A94C	.	165 046	A5 2E	LDA &2E
A94E	;	133 059	85 3B	STA &3B
A950	<	198 060	C6 3C	DEC &3C
A952		032 146 166	20 92 A6	JSR &A692
A955	3	169 051	A9 33	LDA#&33
A957		032 212 169	20 D4 A9	JSR &A9D4
A95A		032 195 150	20 C3 96	JSR &96C3
A95D	I	133 073	85 49	STA &49
A95F	+	005 043	05 2B	ORA &2B
A961	,	005 044	05 2C	ORA &2C
A963	(	240 040	F0 28	BEQ 40 --> &A98D
A965		032 137 129	20 89 81	JSR &8189
A968	q	169 113	A9 71	LDA#&71
A96A		032 019 165	20 13 A5	JSR &A513
A96D	$	169 036	A9 24	LDA#&24
A96F		032 212 169	20 D4 A9	JSR &A9D4
A972		032 146 165	20 92 A5	JSR &A592
A975		032 141 166	20 8D A6	JSR &A68D
A978		032 025 165	20 19 A5	JSR &A519
A97B	q	169 113	A9 71	LDA#&71
A97D	;	032 059 165	20 3B A5	JSR &A53B
A980	)	169 041	A9 29	LDA#&29
A982		032 212 169	20 D4 A9	JSR &A9D4
A985		032 146 165	20 92 A5	JSR &A592
A988		032 141 166	20 8D A6	JSR &A68D
A98B		128 003	80 03	BRA 3 --> &A990
A98D	9	032 057 165	20 39 A5	JSR &A539
A990		032 013 165	20 0D A5	JSR &A50D
A993		032 166 166	20 A6 A6	JSR &A6A6
A996	t	162 116	A2 74	LDX#&74
A998		169 146	A9 92	LDA#&92
A99A		160 002	A0 02	LDY#&02

A99C	a	032 097 168	20 61 A8	JSR &A861
A99F	v	169 118	A9 76	LDA#&76
A9A1		160 004	A0 04	LDY#&04
A9A3	K	132 075	84 4B	STY &4B
A9A5	J	133 074	85 4A	STA &4A
A9A7		032 166 166	20 A6 A6	JSR &A6A6
A9AA		169 255	A9 FF	LDA#&FF
A9AC	`	096	60	RTS
A9AD		000	00	BRK
A9AE		023	17	EQUB &17
A9AF	Ac	065 099	41 63	EOR (&63,X)
A9B1	c	099	63	xxx Invalid Code
A9B2	ur	117 114	75 72	ADC &72,X
A9B4	acy	097 099 121	61 63 79	ADC (&7963,X)
A9B7	lo	032 108 111	20 6C 6F	JSR &6F6C
A9BA	s	115	73	xxx Invalid Code
A9BB	t	116	74	xxx Invalid Code
A9BC		000	00	BRK
A9BD		024	18	EQUB &18
A9BE	Ex	069 120	45 78	EOR &78
A9C0	p	112 032	70 20	BVS 32 --> &A9E2
A9C2	ra	114 097	72 61	ADC (&61)
A9C4	nge	110 103 101	6E 67 65	ROR &6567
A9C7		000	00	EQUB &00
A9C8		032 218 150	20 DA 96	JSR &96DA
A9CB	8	169 056	A9 38	LDA#&38
A9CD		128 005	80 05	BRA 5 --> &A9D4
A9CF	F	032 070 167	20 46 A7	JSR &A746
A9D2	B	169 066	A9 42	LDA#&42
A9D4		160 191	A0 BF	LDY#&BF
A9D6		128 203	80 CB	BRA -53 --> &A9A3
A9D8		032 218 150	20 DA 96	JSR &96DA
A9DB	=	169 061	A9 3D	LDA#&3D
A9DD		128 245	80 F5	BRA -11 --> &A9D4
A9DF		032 218 150	20 DA 96	JSR &96DA

A9E2	0	165 048	A5 30	LDA &30
A9E4		201 135	C9 87	CMP#&87
A9E6		144 015	90 0F	BCC 15 --> &A9F7
A9E8		208 006	D0 06	BNE 6 --> &A9F0
A9EA	1	164 049	A4 31	LDY &31
A9EC		192 179	C0 B3	CPY#&B3
A9EE		144 007	90 07	BCC 7 --> &A9F7
A9F0	.	165 046	A5 2E	LDA &2E
A9F2		016 200	10 C8	BPL -56 --> &A9BC
A9F4	L	076 180 166	4C B4 A6	JMP &A6B4
A9F7		032 224 130	20 E0 82	JSR &82E0
A9FA		162 206	A2 CE	LDX#&CE
A9FC		169 246	A9 F6	LDA#&F6
A9FE		160 003	A0 03	LDY#&03
AA00	a	032 097 168	20 61 A8	JSR &A861
AA03		032 013 165	20 0D A5	JSR &A50D
AA06	G	169 071	A9 47	LDA#&47
AA08		032 150 168	20 96 A8	JSR &A896
AA0B	I	165 073	A5 49	LDA &49
AA0D		032 190 165	20 BE A5	JSR &A5BE
AA10		128 141	80 8D	BRA -115 --> &A99F
AA12		032 180 150	20 B4 96	JSR &96B4
AA15		169 129	A9 81	LDA#&81
AA17	*	166 042	A6 2A	LDX &2A
AA19	+	164 043	A4 2B	LDY &2B
AA1B	L	076 244 255	4C F4 FF	JMP &FFF4
AA1E		032 030 131	20 1E 83	JSR &831E
AA21	d.	100 046	64 2E	STZ &2E
AA23	d/	100 047	64 2F	STZ &2F
AA25	d5	100 053	64 35	STZ &35
AA27		169 128	A9 80	LDA#&80
AA29	0	133 048	85 30	STA &30
AA2B		160 000	A0 00	LDY#&00
AA2D		162 003	A2 03	LDX#&03
AA2F	Y	089 013 000	59 0D 00	EOR &000D,Y

AA32	1	149 049	95 31	STA &31,X
AA34		200	C8	INY
AA35		202	CA	DEX
AA36		016 247	10 F7	BPL -9 --> &AA2F
AA38	LT	076 084 168	4C 54 A8	JMP &A854
AA3B		230 027	E6 1B	INC &1B
AA3D		032 167 150	20 A7 96	JSR &96A7
AA40	-	165 045	A5 2D	LDA &2D
AA42	0%	048 037	30 25	BMI 37 --> &AA69
AA44	,	005 044	05 2C	ORA &2C
AA46	+	005 043	05 2B	ORA &2B
AA48		208 008	D0 08	BNE 8 --> &AA52
AA4A	*	165 042	A5 2A	LDA &2A
AA4C		240 211	F0 D3	BEQ -45 --> &AA21
AA4E		201 001	C9 01	CMP#&01
AA50		240 204	F0 CC	BEQ -52 --> &AA1E
AA52		032 133 129	20 85 81	JSR &8185
AA55		032 250 187	20 FA BB	JSR &BBFA
AA58		032 030 170	20 1E AA	JSR &AA1E
AA5B		032 232 187	20 E8 BB	JSR &BBE8
AA5E		032 207 166	20 CF A6	JSR &A6CF
AA61		032 195 150	20 C3 96	JSR &96C3
AA64		032 239 190	20 EF BE	JSR &BEEF
AA67	'	128 039	80 27	BRA 39 --> &AA90
AA69		162 013	A2 0D	LDX#&0D
AA6B		032 198 189	20 C6 BD	JSR &BDC6
AA6E	@	169 064	A9 40	LDA#&40
AA70		133 017	85 11	STA &11
AA72	`	096	60	RTS
AA73		164 027	A4 1B	LDY &1B
AA75		177 025	B1 19	LDA (&19),Y
AA77	(	201 040	C9 28	CMP#&28
AA79		240 192	F0 C0	BEQ -64 --> &AA3B
AA7B		032 030 131	20 1E 83	JSR &831E
AA7E		162 013	A2 0D	LDX#&0D

AA80		181 000	B5 00	LDA &00,X
AA82	*	133 042	85 2A	STA &2A
AA84		181 001	B5 01	LDA &01,X
AA86	+	133 043	85 2B	STA &2B
AA88		181 002	B5 02	LDA &02,X
AA8A	,	133 044	85 2C	STA &2C
AA8C		181 003	B5 03	LDA &03,X
AA8E	-	133 045	85 2D	STA &2D
AA90	@	169 064	A9 40	LDA#&40
AA92	`	096	60	RTS
AA93		032 180 150	20 B4 96	JSR &96B4
AA96		162 003	A2 03	LDX#&03
AA98	*	181 042	B5 2A	LDA &2A,X
AA9A	I	073 255	49 FF	EOR#&FF
AA9C	*	149 042	95 2A	STA &2A,X
AA9E		202	CA	DEX
AA9F		016 247	10 F7	BPL -9 --> &AA98
AAA1		128 237	80 ED	BRA -19 --> &AA90
AAA3		032 188 170	20 BC AA	JSR &AABC
AAA6	*	134 042	86 2A	STX &2A
AAA8	`	096	60	RTS
AAA9		032 180 150	20 B4 96	JSR &96B4
AAAC		032 004 147	20 04 93	JSR &9304
AAAF	*	133 042	85 2A	STA &2A
AAB1	+	134 043	86 2B	STX &2B
AAB3	,	132 044	84 2C	STY &2C
AAB5		008	08	PHP
AAB6	h	104	68	PLA
AAB7	-	133 045	85 2D	STA &2D
AAB9		216	D8	CLD
AABA		128 212	80 D4	BRA -44 --> &AA90
AABC		169 134	A9 86	LDA#&86
AABE		032 244 255	20 F4 FF	JSR &FFF4
AAC1		152	98	TYA
AAC2	L	076 024 174	4C 18 AE	JMP &AE18

AAC5		169 002	A9 02	LDA#&02
AAC7		128 002	80 02	BRA 2 --> &AACB
AAC9		169 000	A9 00	LDA#&00
AACB	H	072	48	PHA
AACC	J	032 074 186	20 4A BA	JSR &BA4A
AACF	*	162 042	A2 2A	LDX#&2A
AAD1	h	104	68	PLA
AAD2		032 218 255	20 DA FF	JSR &FFDA
AAD5		128 185	80 B9	BRA -71 --> &AA90
AAD7	J	032 074 186	20 4A BA	JSR &BA4A
AADA		032 215 255	20 D7 FF	JSR &FFD7
AADD		128 227	80 E3	BRA -29 --> &AAC2
AADF	@	169 064	A9 40	LDA#&40
AAE1		128 006	80 06	BRA 6 --> &AAE9
AAE3		169 128	A9 80	LDA#&80
AAE5		128 002	80 02	BRA 2 --> &AAE9
AAE7		169 192	A9 C0	LDA#&C0
AAE9	H	072	48	PHA
AAEA	6	032 054 173	20 36 AD	JSR &AD36
AAED		208 013	D0 0D	BNE 13 --> &AAFC
AAEF	+	032 043 190	20 2B BE	JSR &BE2B
AAF2		162 000	A2 00	LDX#&00
AAF4		160 006	A0 06	LDY#&06
AAF6	h	104	68	PLA
AAF7		032 206 255	20 CE FF	JSR &FFCE
AAFA		128 198	80 C6	BRA -58 --> &AAC2
AAFC	L	076 146 144	4C 92 90	JMP &9092
AAFF		032 190 168	20 BE A8	JSR &A8BE
AB02	0	230 048	E6 30	INC &30
AB04	`	096	60	RTS
AB05	6	032 054 173	20 36 AD	JSR &AD36
AB08	<	208 060	D0 3C	BNE 60 --> &AB46
AB0A	6	230 054	E6 36	INC &36
AB0C	6	164 054	A4 36	LDY &36
AB0E		169 013	A9 0D	LDA#&0D

AB10		153 255 005	99 FF 05	STA &05FF,Y
AB13	Q	032 081 188	20 51 BC	JSR &BC51
AB16		165 025	A5 19	LDA &19
AB18	H	072	48	PHA
AB19		165 026	A5 1A	LDA &1A
AB1B	H	072	48	PHA
AB1C		165 027	A5 1B	LDA &1B
AB1E	H	072	48	PHA
AB1F		164 004	A4 04	LDY &04
AB21		166 005	A6 05	LDX &05
AB23		200	C8	INY
AB24		132 025	84 19	STY &19
AB26	7	132 055	84 37	STY &37
AB28		208 001	D0 01	BNE 1 --> &AB2B
AB2A		232	E8	INX
AB2B		134 026	86 1A	STX &1A
AB2D	8	134 056	86 38	STX &38
AB2F		032 031 142	20 1F 8E	JSR &8E1F
AB32	d	100 027	64 1B	STZ &1B
AB34	;	032 059 157	20 3B 9D	JSR &9D3B
AB37		032 225 188	20 E1 BC	JSR &BCE1
AB3A	h	104	68	PLA
AB3B		133 027	85 1B	STA &1B
AB3D	h	104	68	PLA
AB3E		133 026	85 1A	STA &1A
AB40	h	104	68	PLA
AB41		133 025	85 19	STA &19
AB43	'	165 039	A5 27	LDA &27
AB45	`	096	60	RTS
AB46	L	076 146 144	4C 92 90	JMP &9092
AB49	6	032 054 173	20 36 AD	JSR &AD36
AB4C		208 248	D0 F8	BNE -8 --> &AB46
AB4E	6	166 054	A6 36	LDX &36
AB50		158 000 006	9E 00 06	STZ &0600,X
AB53		165 025	A5 19	LDA &19

AB55	H	072	48	PHA
AB56		165 026	A5 1A	LDA &1A
AB58	H	072	48	PHA
AB59		165 027	A5 1B	LDA &1B
AB5B	H	072	48	PHA
AB5C	d	100 027	64 1B	STZ &1B
AB5E	d	100 025	64 19	STZ &19
AB60		169 006	A9 06	LDA#&06
AB62		133 026	85 1A	STA &1A
AB64		032 213 142	20 D5 8E	JSR &8ED5
AB67	-	201 045	C9 2D	CMP#&2D
AB69		240 014	F0 0E	BEQ 14 --> &AB79
AB6B	+	201 043	C9 2B	CMP#&2B
AB6D		208 003	D0 03	BNE 3 --> &AB72
AB6F		032 213 142	20 D5 8E	JSR &8ED5
AB72		198 027	C6 1B	DEC &1B
AB74		032 225 162	20 E1 A2	JSR &A2E1
AB77		128 013	80 0D	BRA 13 --> &AB86
AB79		032 213 142	20 D5 8E	JSR &8ED5
AB7C		198 027	C6 1B	DEC &1B
AB7E		032 225 162	20 E1 A2	JSR &A2E1
AB81		144 003	90 03	BCC 3 --> &AB86
AB83		032 218 172	20 DA AC	JSR &ACDA
AB86	'	133 039	85 27	STA &27
AB88		128 176	80 B0	BRA -80 --> &AB3A
AB8A	6	032 054 173	20 36 AD	JSR &AD36
AB8D	=	240 061	F0 3D	BEQ 61 --> &ABCC
AB8F	!	016 033	10 21	BPL 33 --> &ABB2
AB91	.	165 046	A5 2E	LDA &2E
AB93		008	08	PHP
AB94	u	032 117 130	20 75 82	JSR &8275
AB97	(	040	28	PLP
AB98		016 019	10 13	BPL 19 --> &ABAD
AB9A	=	165 061	A5 3D	LDA &3D
AB9C	>	005 062	05 3E	ORA &3E

AB9E	?	005 063	05 3F	ORA &3F
ABA0	@	005 064	05 40	ORA &40
ABA2		240 009	F0 09	BEQ 9 --> &ABAD
ABA4		032 200 130	20 C8 82	JSR &82C8
ABA7		032 013 131	20 0D 83	JSR &830D
ABAA		032 200 130	20 C8 82	JSR &82C8
ABAD		032 198 150	20 C6 96	JSR &96C6
ABB0	@	169 064	A9 40	LDA#&40
ABB2	`	096	60	RTS
ABB3	6	032 054 173	20 36 AD	JSR &AD36
ABB6		208 020	D0 14	BNE 20 --> &ABCC
ABB8	6	165 054	A5 36	LDA &36
ABBA		240 031	F0 1F	BEQ 31 --> &ABDB
ABBC		173 000 006	AD 00 06	LDA &0600
ABBF	L	076 024 174	4C 18 AE	JMP &AE18
ABC2		032 018 170	20 12 AA	JSR &AA12
ABC5		152	98	TYA
ABC6		208 019	D0 13	BNE 19 --> &ABDB
ABC8		138	8A	TXA
ABC9	L	076 026 174	4C 1A AE	JMP &AE1A
ABCC	L	076 146 144	4C 92 90	JMP &9092
ABCF	J	032 074 186	20 4A BA	JSR &BA4A
ABD2		170	AA	TAX
ABD3		169 127	A9 7F	LDA#&7F
ABD5		032 244 255	20 F4 FF	JSR &FFF4
ABD8		138	8A	TXA
ABD9		240 002	F0 02	BEQ 2 --> &ABDD
ABDB		162 255	A2 FF	LDX#&FF
ABDD	*	134 042	86 2A	STX &2A
ABDF	+	134 043	86 2B	STX &2B
ABE1	,	134 044	86 2C	STX &2C
ABE3	-	134 045	86 2D	STX &2D
ABE5	@	169 064	A9 40	LDA#&40
ABE7	`	096	60	RTS
ABE8		162 000	A2 00	LDX#&00

ABEA		128 241	80 F1	BRA -15 --> &ABDD
ABEC		032 242 163	20 F2 A3	JSR &A3F2
ABEF		240 247	F0 F7	BEQ -9 --> &ABE8
ABF1		016 023	10 17	BPL 23 --> &AC0A
ABF3		128 230	80 E6	BRA -26 --> &ABDB
ABF5	6	032 054 173	20 36 AD	JSR &AD36
ABF8		240 210	F0 D2	BEQ -46 --> &ABCC
ABFA	0	048 240	30 F0	BMI -16 --> &ABEC
ABFC	-	165 045	A5 2D	LDA &2D
ABFE	,	005 044	05 2C	ORA &2C
AC00	+	005 043	05 2B	ORA &2B
AC02	*	005 042	05 2A	ORA &2A
AC04		240 223	F0 DF	BEQ -33 --> &ABE5
AC06	-	165 045	A5 2D	LDA &2D
AC08	0	048 209	30 D1	BMI -47 --> &ABDB
AC0A		169 001	A9 01	LDA#&01
AC0C		128 177	80 B1	BRA -79 --> &ABBF
AC0E		032 175 150	20 AF 96	JSR &96AF
AC11	&	032 038 188	20 26 BC	JSR &BC26
AC14		032 241 142	20 F1 8E	JSR &8EF1
AC17		032 167 150	20 A7 96	JSR &96A7
AC1A	*	165 042	A5 2A	LDA &2A
AC1C	H	072	48	PHA
AC1D	+	166 043	A6 2B	LDX &2B
AC1F		032 230 188	20 E6 BC	JSR &BCE6
AC22	-	134 045	86 2D	STX &2D
AC24	h	104	68	PLA
AC25	,	133 044	85 2C	STA &2C
AC27		160 000	A0 00	LDY#&00
AC29	*	162 042	A2 2A	LDX#&2A
AC2B		169 009	A9 09	LDA#&09
AC2D		032 241 255	20 F1 FF	JSR &FFF1
AC30	.	165 046	A5 2E	LDA &2E
AC32	0	048 167	30 A7	BMI -89 --> &ABDB
AC34		128 214	80 D6	BRA -42 --> &AC0C

AC36	;	032 059 157	20 3B 9D	JSR &9D3B
AC39		208 145	D0 91	BNE -111 --> &ABCC
AC3B	,	224 044	E0 2C	CPX#&2C
AC3D		208 024	D0 18	BNE 24 --> &AC57
AC3F		230 027	E6 1B	INC &1B
AC41	Q	032 081 188	20 51 BC	JSR &BC51
AC44	;	032 059 157	20 3B 9D	JSR &9D3B
AC47		208 131	D0 83	BNE -125 --> &ABCC
AC49		169 001	A9 01	LDA#&01
AC4B	*	133 042	85 2A	STA &2A
AC4D		230 027	E6 1B	INC &1B
AC4F	)	224 041	E0 29	CPX#&29
AC51		240 013	F0 0D	BEQ 13 --> &AC60
AC53	,	224 044	E0 2C	CPX#&2C
AC55		240 003	F0 03	BEQ 3 --> &AC5A
AC57	L	076 246 142	4C F6 8E	JMP &8EF6
AC5A		032 164 150	20 A4 96	JSR &96A4
AC5D		032 210 188	20 D2 BC	JSR &BCD2
AC60	*	166 042	A6 2A	LDX &2A
AC62		208 002	D0 02	BNE 2 --> &AC66
AC64		162 001	A2 01	LDX#&01
AC66	*	134 042	86 2A	STX &2A
AC68		138	8A	TXA
AC69		202	CA	DEX
AC6A	-	134 045	86 2D	STX &2D
AC6C		024	18	CLC
AC6D	e	101 004	65 04	ADC &04
AC6F	7	133 055	85 37	STA &37
AC71		169 000	A9 00	LDA#&00
AC73	e	101 005	65 05	ADC &05
AC75	8	133 056	85 38	STA &38
AC77		178 004	B2 04	LDA (&04)
AC79	8	056	38	SEC
AC7A	-	229 045	E5 2D	SBC &2D
AC7C	!	144 033	90 21	BCC 33 --> &AC9F

AC7E	6	229 054	E5 36	SBC &36
AC80		144 029	90 1D	BCC 29 --> &AC9F
AC82	i	105 000	69 00	ADC#&00
AC84	+	133 043	85 2B	STA &2B
AC86		032 225 188	20 E1 BC	JSR &BCE1
AC89		160 000	A0 00	LDY#&00
AC8B	6	166 054	A6 36	LDX &36
AC8D		240 011	F0 0B	BEQ 11 --> &AC9A
AC8F	7	177 055	B1 37	LDA (&37),Y
AC91		217 000 006	D9 00 06	CMP &0600,Y
AC94		208 016	D0 10	BNE 16 --> &ACA6
AC96		200	C8	INY
AC97		202	CA	DEX
AC98		208 245	D0 F5	BNE -11 --> &AC8F
AC9A	*	165 042	A5 2A	LDA &2A
AC9C	L	076 024 174	4C 18 AE	JMP &AE18
AC9F		032 225 188	20 E1 BC	JSR &BCE1
ACA2		169 000	A9 00	LDA#&00
ACA4		128 246	80 F6	BRA -10 --> &AC9C
ACA6	*	230 042	E6 2A	INC &2A
ACA8	+	198 043	C6 2B	DEC &2B
ACAA		240 246	F0 F6	BEQ -10 --> &ACA2
ACAC	7	230 055	E6 37	INC &37
ACAE		208 217	D0 D9	BNE -39 --> &AC89
ACB0	8	230 056	E6 38	INC &38
ACB2		128 213	80 D5	BRA -43 --> &AC89
ACB4	L	076 146 144	4C 92 90	JMP &9092
ACB7	6	032 054 173	20 36 AD	JSR &AD36
ACBA		240 248	F0 F8	BEQ -8 --> &ACB4
ACBC	0	048 006	30 06	BMI 6 --> &ACC4
ACBE	$-	036 045	24 2D	BIT &2D
ACC0	0	048 028	30 1C	BMI 28 --> &ACDE
ACC2	1	128 049	80 31	BRA 49 --> &ACF5
ACC4	d.	100 046	64 2E	STZ &2E
ACC6	`	096	60	RTS

ACC7		032 138 166	20 8A A6	JSR &A68A
ACCA	1	165 049	A5 31	LDA &31
ACCC		240 006	F0 06	BEQ 6 --> &ACD4
ACCE	.	165 046	A5 2E	LDA &2E
ACD0	I	073 128	49 80	EOR#&80
ACD2	.	133 046	85 2E	STA &2E
ACD4		169 255	A9 FF	LDA#&FF
ACD6	`	096	60	RTS
ACD7	L	032 076 173	20 4C AD	JSR &AD4C
ACDA		240 216	F0 D8	BEQ -40 --> &ACB4
ACDC	0	048 236	30 EC	BMI -20 --> &ACCA
ACDE	8	056	38	SEC
ACDF		169 000	A9 00	LDA#&00
ACE1		168	A8	TAY
ACE2	*	229 042	E5 2A	SBC &2A
ACE4	*	133 042	85 2A	STA &2A
ACE6		152	98	TYA
ACE7	+	229 043	E5 2B	SBC &2B
ACE9	+	133 043	85 2B	STA &2B
ACEB		152	98	TYA
ACEC	,	229 044	E5 2C	SBC &2C
ACEE	,	133 044	85 2C	STA &2C
ACF0		152	98	TYA
ACF1	-	229 045	E5 2D	SBC &2D
ACF3	-	133 045	85 2D	STA &2D
ACF5	@	169 064	A9 40	LDA#&40
ACF7	`	096	60	RTS
ACF8		032 213 142	20 D5 8E	JSR &8ED5
ACFB	"	201 034	C9 22	CMP#&22
ACFD		240 026	F0 1A	BEQ 26 --> &AD19
ACFF		162 000	A2 00	LDX#&00
AD01		177 025	B1 19	LDA (&19),Y
AD03		157 000 006	9D 00 06	STA &0600,X
AD06		200	C8	INY
AD07		232	E8	INX

AD08		201 013	C9 0D	CMP#&0D
AD0A		240 004	F0 04	BEQ 4 --> &AD10
AD0C	,	201 044	C9 2C	CMP#&2C
AD0E		208 241	D0 F1	BNE -15 --> &AD01
AD10		136	88	DEY
AD11		202	CA	DEX
AD12	6	134 054	86 36	STX &36
AD14		132 027	84 1B	STY &1B
AD16		169 000	A9 00	LDA#&00
AD18	`	096	60	RTS
AD19		162 000	A2 00	LDX#&00
AD1B		200	C8	INY
AD1C		177 025	B1 19	LDA (&19),Y
AD1E		201 013	C9 0D	CMP#&0D
AD20		240 017	F0 11	BEQ 17 --> &AD33
AD22		157 000 006	9D 00 06	STA &0600,X
AD25		200	C8	INY
AD26		232	E8	INX
AD27	"	201 034	C9 22	CMP#&22
AD29		208 241	D0 F1	BNE -15 --> &AD1C
AD2B		177 025	B1 19	LDA (&19),Y
AD2D	"	201 034	C9 22	CMP#&22
AD2F		240 234	F0 EA	BEQ -22 --> &AD1B
AD31		208 222	D0 DE	BNE -34 --> &AD11
AD33	L	076 148 146	4C 94 92	JMP &9294
AD36		164 027	A4 1B	LDY &1B
AD38		230 027	E6 1B	INC &1B
AD3A		177 025	B1 19	LDA (&19),Y
AD3C		201 032	C9 20	CMP#&20
AD3E		240 246	F0 F6	BEQ -10 --> &AD36
AD40	-	201 045	C9 2D	CMP#&2D
AD42		240 147	F0 93	BEQ -109 --> &ACD7
AD44	"	201 034	C9 22	CMP#&22
AD46		240 209	F0 D1	BEQ -47 --> &AD19
AD48	+	201 043	C9 2B	CMP#&2B

AD4A		208 003	D0 03	BNE 3 --> &AD4F
AD4C		032 213 142	20 D5 8E	JSR &8ED5
AD4F		201 142	C9 8E	CMP#&8E
AD51		144 007	90 07	BCC 7 --> &AD5A
AD53		201 198	C9 C6	CMP#&C6
AD55	5	176 053	B0 35	BCS 53 --> &AD8C
AD57	L	076 025 144	4C 19 90	JMP &9019
AD5A	?	201 063	C9 3F	CMP#&3F
AD5C		176 012	B0 0C	BCS 12 --> &AD6A
AD5E	.	201 046	C9 2E	CMP#&2E
AD60		176 018	B0 12	BCS 18 --> &AD74
AD62	&	201 038	C9 26	CMP#&26
AD64	Q	240 081	F0 51	BEQ 81 --> &ADB7
AD66	(	201 040	C9 28	CMP#&28
AD68	B	240 066	F0 42	BEQ 66 --> &ADAC
AD6A		198 027	C6 1B	DEC &1B
AD6C		032 009 153	20 09 99	JSR &9909
AD6F		240 009	F0 09	BEQ 9 --> &AD7A
AD71	L	076 160 177	4C A0 B1	JMP &B1A0
AD74		032 225 162	20 E1 A2	JSR &A2E1
AD77		144 019	90 13	BCC 19 --> &AD8C
AD79	`	096	60	RTS
AD7A	(	165 040	A5 28	LDA &28
AD7C	)	041 002	29 02	AND#&02
AD7E		208 012	D0 0C	BNE 12 --> &AD8C
AD80		176 010	B0 0A	BCS 10 --> &AD8C
AD82		134 027	86 1B	STX &1B
AD84	@	173 064 004	AD 40 04	LDA &0440
AD87	A	172 065 004	AC 41 04	LDY &0441
AD8A	k	128 107	80 6B	BRA 107 --> &ADF7
AD8C		000	00	BRK
AD8D		026	1A	EQUB &1A
AD8E	No	078 111 032	4E 6F 20	LSR &206F
AD91	s	115	73	xxx Invalid Code
AD92	uc	117 099	75 63	ADC &63,X

AD94	h	104	68	PLA
AD95	va	032 118 097	20 76 61	JSR &6176
AD98	ri	114 105	72 69	ADC (&69)
AD9A	abl	097 098 108	61 62 6C	ADC (&6C62,X)
AD9D	e	101	65	xxx Invalid Code
AD9E		000	00	BRK
AD9F		027	1B	EQUB &1B
ADA0	)	141 041 000	8D 29 00	STA &0029
ADA3	Ba	028 066 097	1C 42 61	TRB &6142
ADA6	d	100 032	64 20	STZ &20
ADA8	H	072	48	PHA
ADA9	ex	101 120	65 78	ADC &78
ADAB		000	00	EQUB &00
ADAC	;	032 059 157	20 3B 9D	JSR &9D3B
ADAF		230 027	E6 1B	INC &1B
ADB1	)	224 041	E0 29	CPX#&29
ADB3		208 233	D0 E9	BNE -23 --> &AD9E
ADB5		168	A8	TAY
ADB6	`	096	60	RTS
ADB7		032 232 171	20 E8 AB	JSR &ABE8
ADBA		200	C8	INY
ADBB		177 025	B1 19	LDA (&19),Y
ADBD	0	201 048	C9 30	CMP#&30
ADBF	#	144 035	90 23	BCC 35 --> &ADE4
ADC1	:	201 058	C9 3A	CMP#&3A
ADC3		144 010	90 0A	BCC 10 --> &ADCF
ADC5	7	233 055	E9 37	SBC#&37
ADC7		201 010	C9 0A	CMP#&0A
ADC9		144 025	90 19	BCC 25 --> &ADE4
ADCB		201 016	C9 10	CMP#&10
ADCD		176 021	B0 15	BCS 21 --> &ADE4
ADCF		010	0A	ASL A
ADD0		010	0A	ASL A
ADD1		010	0A	ASL A
ADD2		010	0A	ASL A

ADD3		162 003	A2 03	LDX#&03
ADD5		010	0A	ASL A
ADD6	&*	038 042	26 2A	ROL &2A
ADD8	&+	038 043	26 2B	ROL &2B
ADDA	&,	038 044	26 2C	ROL &2C
ADDC	&-	038 045	26 2D	ROL &2D
ADDE		202	CA	DEX
ADDF		016 244	10 F4	BPL -12 --> &ADD5
ADE1		200	C8	INY
ADE2		208 215	D0 D7	BNE -41 --> &ADBB
ADE4		138	8A	TXA
ADE5		016 187	10 BB	BPL -69 --> &ADA2
ADE7		132 027	84 1B	STY &1B
ADE9	@	169 064	A9 40	LDA#&40
ADEB	`	096	60	RTS
ADEC		032 180 150	20 B4 96	JSR &96B4
ADEF	*	166 042	A6 2A	LDX &2A
ADF1		169 128	A9 80	LDA#&80
ADF3		032 244 255	20 F4 FF	JSR &FFF4
ADF6		138	8A	TXA
ADF7	!	128 033	80 21	BRA 33 --> &AE1A
ADF9		200	C8	INY
ADFA		177 025	B1 19	LDA (&19),Y
ADFC	P	201 080	C9 50	CMP#&50
ADFE		208 140	D0 8C	BNE -116 --> &AD8C
AE00		230 027	E6 1B	INC &1B
AE02		165 018	A5 12	LDA &12
AE04		164 019	A4 13	LDY &13
AE06		128 018	80 12	BRA 18 --> &AE1A
AE08		164 024	A4 18	LDY &18
AE0A		169 000	A9 00	LDA#&00
AE0C		128 012	80 0C	BRA 12 --> &AE1A
AE0E	L	076 146 144	4C 92 90	JMP &9092
AE11	6	032 054 173	20 36 AD	JSR &AD36
AE14		208 248	D0 F8	BNE -8 --> &AE0E

```
AE16    6    165 054        A5 36       LDA &36
AE18         160 000        A0 00       LDY#&00
AE1A    *    133 042        85 2A       STA &2A
AE1C    +    132 043        84 2B       STY &2B
AE1E    d,   100 044        64 2C       STZ &2C
AE20    d-   100 045        64 2D       STZ &2D
AE22    @    169 064        A9 40       LDA#&40
AE24    `    096            60          RTS
AE25         165 030        A5 1E       LDA &1E
AE27         128 239        80 EF       BRA -17 --> &AE18
AE29         165 000        A5 00       LDA &00
AE2B         164 001        A4 01       LDY &01
AE2D         128 235        80 EB       BRA -21 --> &AE1A
AE2F         165 006        A5 06       LDA &06
AE31         164 007        A4 07       LDY &07
AE33         128 229        80 E5       BRA -27 --> &AE1A
AE35         164 009        A4 09       LDY &09
AE37         165 008        A5 08       LDA &08
AE39         128 223        80 DF       BRA -33 --> &AE1A
AE3B         178 253        B2 FD       LDA (&FD)
AE3D         128 217        80 D9       BRA -39 --> &AE18
AE3F         032 224 255    20 E0 FF    JSR &FFE0
AE42         128 212        80 D4       BRA -44 --> &AE18
AE44         200            C8          INY
AE45         177 025        B1 19       LDA (&19),Y
AE47    $    201 036        C9 24       CMP#&24
AE49         240 012        F0 0C       BEQ 12 --> &AE57
AE4B    *    162 042        A2 2A       LDX#&2A
AE4D         160 000        A0 00       LDY#&00
AE4F         169 001        A9 01       LDA#&01
AE51         032 241 255    20 F1 FF    JSR &FFF1
AE54    @    169 064        A9 40       LDA#&40
AE56    `    096            60          RTS
AE57         230 027        E6 1B       INC &1B
AE59         169 014        A9 0E       LDA#&0E
```

AE5B		162 000	A2 00	LDX#&00
AE5D		160 006	A0 06	LDY#&06
AE5F		156 000 006	9C 00 06	STZ &0600
AE62		032 241 255	20 F1 FF	JSR &FFF1
AE65		169 024	A9 18	LDA#&18
AE67	&	128 038	80 26	BRA 38 --> &AE8F
AE69		032 224 255	20 E0 FF	JSR &FFE0
AE6C		141 000 006	8D 00 06	STA &0600
AE6F		169 001	A9 01	LDA#&01
AE71		128 028	80 1C	BRA 28 --> &AE8F
AE73		024	18	CLC
AE74		008	08	PHP
AE75	;	032 059 157	20 3B 9D	JSR &9D3B
AE78	E	208 069	D0 45	BNE 69 --> &AEBF
AE7A	,	224 044	E0 2C	CPX#&2C
AE7C	D	208 068	D0 44	BNE 68 --> &AEC2
AE7E		230 027	E6 1B	INC &1B
AE80		032 164 150	20 A4 96	JSR &96A4
AE83		032 210 188	20 D2 BC	JSR &BCD2
AE86	(	040	28	PLP
AE87		176 011	B0 0B	BCS 11 --> &AE94
AE89	*	165 042	A5 2A	LDA &2A
AE8B	6	197 054	C5 36	CMP &36
AE8D		176 002	B0 02	BCS 2 --> &AE91
AE8F	6	133 054	85 36	STA &36
AE91		169 000	A9 00	LDA#&00
AE93	`	096	60	RTS
AE94	6	165 054	A5 36	LDA &36
AE96	*	229 042	E5 2A	SBC &2A
AE98		144 247	90 F7	BCC -9 --> &AE91
AE9A		240 247	F0 F7	BEQ -9 --> &AE93
AE9C		170	AA	TAX
AE9D	*	165 042	A5 2A	LDA &2A
AE9F	6	133 054	85 36	STA &36
AEA1		240 240	F0 F0	BEQ -16 --> &AE93

AEA3		160 000	A0 00	LDY#&00
AEA5		189 000 006	BD 00 06	LDA &0600,X
AEA8		153 000 006	99 00 06	STA &0600,Y
AEAB		232	E8	INX
AEAC		200	C8	INY
AEAD	*	198 042	C6 2A	DEC &2A
AEAF		208 244	D0 F4	BNE -12 --> &AEA5
AEB1		128 222	80 DE	BRA -34 --> &AE91
AEB3		032 018 170	20 12 AA	JSR &AA12
AEB6		138	8A	TXA
AEB7		192 000	C0 00	CPY#&00
AEB9		240 177	F0 B1	BEQ -79 --> &AE6C
AEBB		169 000	A9 00	LDA#&00
AEBD		128 208	80 D0	BRA -48 --> &AE8F
AEBF	L	076 146 144	4C 92 90	JMP &9092
AEC2	L	076 246 142	4C F6 8E	JMP &8EF6
AEC5	;	032 059 157	20 3B 9D	JSR &9D3B
AEC8		208 245	D0 F5	BNE -11 --> &AEBF
AECA	,	224 044	E0 2C	CPX#&2C
AECC		208 244	D0 F4	BNE -12 --> &AEC2
AECE	Q	032 081 188	20 51 BC	JSR &BC51
AED1		230 027	E6 1B	INC &1B
AED3		032 175 150	20 AF 96	JSR &96AF
AED6	*	165 042	A5 2A	LDA &2A
AED8	H	072	48	PHA
AED9		169 255	A9 FF	LDA#&FF
AEDB	*	133 042	85 2A	STA &2A
AEDD		230 027	E6 1B	INC &1B
AEDF	)	224 041	E0 29	CPX#&29
AEE1		240 007	F0 07	BEQ 7 --> &AEEA
AEE3	,	224 044	E0 2C	CPX#&2C
AEE5		208 219	D0 DB	BNE -37 --> &AEC2
AEE7		032 167 150	20 A7 96	JSR &96A7
AEEA		032 210 188	20 D2 BC	JSR &BCD2
AEED	h	104	68	PLA

AEEE		168	A8	TAY
AEEF		024	18	CLC
AEF0		240 006	F0 06	BEQ 6 --> &AEF8
AEF2	6	229 054	E5 36	SBC &36
AEF4		176 197	B0 C5	BCS -59 --> &AEBB
AEF6		136	88	DEY
AEF7		152	98	TYA
AEF8	,	133 044	85 2C	STA &2C
AEFA		170	AA	TAX
AEFB		160 000	A0 00	LDY#&00
AEFD	6	165 054	A5 36	LDA &36
AEFF	8	056	38	SEC
AF00	,	229 044	E5 2C	SBC &2C
AF02	*	197 042	C5 2A	CMP &2A
AF04		176 002	B0 02	BCS 2 --> &AF08
AF06	*	133 042	85 2A	STA &2A
AF08	*	165 042	A5 2A	LDA &2A
AF0A		240 175	F0 AF	BEQ -81 --> &AEBB
AF0C		189 000 006	BD 00 06	LDA &0600,X
AF0F		153 000 006	99 00 06	STA &0600,Y
AF12		200	C8	INY
AF13		232	E8	INX
AF14	*	196 042	C4 2A	CPY &2A
AF16		208 244	D0 F4	BNE -12 --> &AF0C
AF18	6	132 054	84 36	STY &36
AF1A	^	128 094	80 5E	BRA 94 --> &AF7A
AF1C		032 213 142	20 D5 8E	JSR &8ED5
AF1F		160 255	A0 FF	LDY#&FF
AF21	~	201 126	C9 7E	CMP#&7E
AF23		240 004	F0 04	BEQ 4 --> &AF29
AF25		160 000	A0 00	LDY#&00
AF27		198 027	C6 1B	DEC &1B
AF29	Z	090	5A	PHY
AF2A	6	032 054 173	20 36 AD	JSR &AD36
AF2D		240 021	F0 15	BEQ 21 --> &AF44

AF2F		168	A8	TAY
AF30	h	104	68	PLA
AF31		133 021	85 15	STA &15
AF33		173 003 004	AD 03 04	LDA &0403
AF36		208 007	D0 07	BNE 7 --> &AF3F
AF38	7	133 055	85 37	STA &37
AF3A	2	032 050 161	20 32 A1	JSR &A132
AF3D	;	128 059	80 3B	BRA 59 --> &AF7A
AF3F		032 024 161	20 18 A1	JSR &A118
AF42	6	128 054	80 36	BRA 54 --> &AF7A
AF44	L	076 146 144	4C 92 90	JMP &9092
AF47		032 175 150	20 AF 96	JSR &96AF
AF4A	&	032 038 188	20 26 BC	JSR &BC26
AF4D		032 241 142	20 F1 8E	JSR &8EF1
AF50		032 172 173	20 AC AD	JSR &ADAC
AF53		208 239	D0 EF	BNE -17 --> &AF44
AF55		032 230 188	20 E6 BC	JSR &BCE6
AF58	6	164 054	A4 36	LDY &36
AF5A		240 030	F0 1E	BEQ 30 --> &AF7A
AF5C	*	165 042	A5 2A	LDA &2A
AF5E		240 029	F0 1D	BEQ 29 --> &AF7D
AF60	*	198 042	C6 2A	DEC &2A
AF62		240 022	F0 16	BEQ 22 --> &AF7A
AF64		162 000	A2 00	LDX#&00
AF66		189 000 006	BD 00 06	LDA &0600,X
AF69		153 000 006	99 00 06	STA &0600,Y
AF6C		232	E8	INX
AF6D		200	C8	INY
AF6E		240 016	F0 10	BEQ 16 --> &AF80
AF70	6	228 054	E4 36	CPX &36
AF72		144 242	90 F2	BCC -14 --> &AF66
AF74	*	198 042	C6 2A	DEC &2A
AF76		208 236	D0 EC	BNE -20 --> &AF64
AF78	6	132 054	84 36	STY &36
AF7A		169 000	A9 00	LDA#&00

AF7C	`	096	60	RTS
AF7D	6	133 054	85 36	STA &36
AF7F	`	096	60	RTS
AF80	L	076 016 158	4C 10 9E	JMP &9E10
AF83	h	104	68	PLA
AF84		133 012	85 0C	STA &0C
AF86	h	104	68	PLA
AF87		133 011	85 0B	STA &0B
AF89		000	00	BRK
AF8A		029	1D	EQUB &1D
AF8B	No	078 111	4E 6F	xxx Invalid Code
AF8D	su	032 115 117	20 73 75	JSR &7573
AF90	c	099	63	xxx Invalid Code
AF91	h	104	68	PLA
AF92	/	032 164 047	20 A4 2F	JSR &2FA4
AF95		242	F2	xxx Invalid Code
AF96		000	00	EQUB &00
AF97		165 024	A5 18	LDA &18
AF99		133 012	85 0C	STA &0C
AF9B	d	100 011	64 0B	STZ &0B
AF9D		160 001	A0 01	LDY#&01
AF9F		177 011	B1 0B	LDA (&0B),Y
AFA1	0	048 224	30 E0	BMI -32 --> &AF83
AFA3		160 003	A0 03	LDY#&03
AFA5		200	C8	INY
AFA6		177 011	B1 0B	LDA (&0B),Y
AFA8		201 032	C9 20	CMP#&20
AFAA		240 249	F0 F9	BEQ -7 --> &AFA5
AFAC		201 221	C9 DD	CMP#&DD
AFAE		240 015	F0 0F	BEQ 15 --> &AFBF
AFB0		160 003	A0 03	LDY#&03
AFB2		177 011	B1 0B	LDA (&0B),Y
AFB4		024	18	CLC
AFB5	e	101 011	65 0B	ADC &0B
AFB7		133 011	85 0B	STA &0B

AFB9		144 226	90 E2	BCC -30 --> &AF9D
AFBB		230 012	E6 0C	INC &0C
AFBD		128 222	80 DE	BRA -34 --> &AF9D
AFBF		200	C8	INY
AFC0		132 010	84 0A	STY &0A
AFC2		032 224 142	20 E0 8E	JSR &8EE0
AFC5		152	98	TYA
AFC6		170	AA	TAX
AFC7		024	18	CLC
AFC8	e	101 011	65 0B	ADC &0B
AFCA		164 012	A4 0C	LDY &0C
AFCC		144 002	90 02	BCC 2 --> &AFD0
AFCE		200	C8	INY
AFCF		024	18	CLC
AFD0		233 000	E9 00	SBC#&00
AFD2	<	133 060	85 3C	STA &3C
AFD4		152	98	TYA
AFD5		233 000	E9 00	SBC#&00
AFD7	=	133 061	85 3D	STA &3D
AFD9		160 001	A0 01	LDY#&01
AFDB		232	E8	INX
AFDC	<	177 060	B1 3C	LDA (&3C),Y
AFDE	7	209 055	D1 37	CMP (&37),Y
AFE0		208 206	D0 CE	BNE -50 --> &AFB0
AFE2		200	C8	INY
AFE3	9	196 057	C4 39	CPY &39
AFE5		208 244	D0 F4	BNE -12 --> &AFDB
AFE7	<	177 060	B1 3C	LDA (&3C),Y
AFE9		032 132 141	20 84 8D	JSR &8D84
AFEC		176 194	B0 C2	BCS -62 --> &AFB0
AFEE		138	8A	TXA
AFEF		168	A8	TAY
AFF0		032 188 155	20 BC 9B	JSR &9BBC
AFF3	E	032 069 152	20 45 98	JSR &9845
AFF6		162 001	A2 01	LDX#&01

AFF8	032 131 152	20 83 98	JSR &9883
AFFB	165 011	A5 0B	LDA &0B
AFFD	146 002	92 02	STA (&02)
AFFF	160 001	A0 01	LDY#&01
B001	165 012	A5 0C	LDA &0C

BASIC IV ROM Routines

Page Last Altered: undefined

918D PRINT

Submitted by Steve Fewell

Description:

Get the next non-space character from BASIC Text Pointer A. If the character is a '#' (hash), then jump to the PRINT# routine (&9141); otherwise, decrement &0A (Text Pointer A offset).

[&91AB]: Start a new Output field (&14 is reset and so is &15)
Clear carry flag (as we are outputting numbers in decimal - not hexadecimal).
Store &0400 (the LSB byte of @%) [Print width] in location &14 [PRINT BYTES] (which specifies the number of bytes in each output field).
[&91B1] Move the carry flag (specified output Hexadecimal (if set) or Decimal (if clear) for numeric values) to the top bit of location &15 (PRINT FLAG).

Get the next non-space character from BASIC Text Pointer A (&0B,&0C plus offset &0A).
If the next character is ':', then we have an empty PRINT field/statement, so call &BA92 to Output a new line and goto &9000 to process the next statement.
If the next character is '<cr>' (carriage return), then we have an empty PRINT field/statement, so call &BA92 to Output a new line and goto &9000 to process the next statement.
If the next character is 'ELSE'-token, then we have an empty PRINT field/statement, so call &BA92 to Output a new line & goto &9000 to process the next statement.

[&91DB]: If the next character is '~' [tilde], then jump back to &91B1 to set the top bit of &15 to 1 (print numeric values in hexadecimal notation) and check again for end of statement (newline if so).

If the next character is ',' then jump to &9196 to complete the output field, as follows:

* If &0400 (Width of print field) is 0 then we do not need to pad the field, so jump to &91AB to start a new output field.
* If COUNT is zero (we have just began a new line) then do not pad (as nothing has been output to the new line yet), so jump to &91AB to start a new output field.
* Set A to COUNT (&1E).
* Subtract &0400 (field width) from A (COUNT).
* If the subtraction did not underflow and the result > 0 then jump to the previous step to subtract again.
* If the subtraction did not underflow and the result = 0 then jump to &91AB to start a new output field.
* Keep incrementing Y, calling &BD92 to output a space for each incrementation, until Y = 00.
* Now the field has been padded with the required number of spaces.
* Continue to &91AB to start a new output field.

If the next character is ';' then jump to &91C8 to end the current field, as follows:

* Zero number of bytes in print field (&14)
* Zero Print [Decimal/Hexadecimal] Flag (&15)
* Get the next non-space charater pointed to by BASIC Text pointer A.
* If the character is ':' then jump to &9000 (to process the next statement) without outputting a new line.
* If the character is <cr> (carriage-return) then jump to &9000 (to process the next statement) without outputting a new line.
* If the character is 'ELSE-token' then jump to &9000 (to process the next stetement) without outputting a new line.
* If not end of statement then continue with &91DB to check for '~', ',' or ';' again.

[&91E7] Not end of statement and not tilde, comma or semicolon - check ', TAB & SPC

Call routine &927A to check for specific characters and PRINT options, this routine checks the following:

* Copy BASIC Text pointer A details to BASIC Text Pointer B.
* If character = """ [quote], then jump to routine &9267 to output a new line.
* If character = '**TAB(**'-token then jump to routine &9241 to go to the requested screen position.
* If character = '**SPC**'-token then jump to routine &925B to output the requested number of spaces.
* Otherwise, set carry flag and exit.

If carry is clear on return from routine &927A, then a special function was carried out (either New line, TAB(or SPC), so jump back to &91B3 to check for the end of statement, tilde, comma, semicolon and further New line ("""), TAB(and SPC commands before continuing.

[&91EC] Get result of the expression to output
Store &14 and &15 to the stack, decrement BASIC Text pointer B offset, and call &9D3B to get the result of the expression at BASIC Text Pointer B.
Retrieve the &14 and &15 values from the stack.
Update BASIC Text Pointer A Offset to the BASIC Text Pointer B Offset value.

Check the variable type of the expression result (Y). If the variable type is not String (0) then convert as follows:

* Call routine &A118 (NUMASC) to convert the Numeric result value into a String value and place the result in the SWA. This routine will convert in Hexadecimal-format if the Print flag (&15) is negative.
* Load A with the width of the PRINT field (&14).
* Subtract from this the length of the ASCII string (&36).
* If the SWA String value is less than the PRINT field width then:
* Set Y to A (the difference in width).
* call routine &BD92 for Y number of times, to pad the field with leading spaces until it is the required width. This is to make sure that numerical values line up.
* Now the SWA output string (along with any spaces that have already been output) is greater than or equal to the required field width.

[&9217] Output the value of the expression result
If the SWA length (&36) is zero then there is nothing to output, so jump back to &91B3 to check for end of statement or to continue with the next PRINT field (if not end of statement).

Starting at the first character of the SWA, continue setting A to the next character and calling routine &BD98 to output that character; until all &36 characters of the SWA have been output.

Jump back to &91B3 to check for end of statement or to continue with the next PRINT field (if not end of statement).

Disassembly for the PRINT routine

918D	032 223 140	20 DF 8C	JSR &8CDF Get next non-space char (PTR A) and compare with '#'
9190	240 175	F0 AF	BEQ -81 --> &9141 PRINT#
9192	198 010	C6 0A	DEC &0A
9194	128 021	80 15	BRA 21 --> &91AB Start new output field
9196	173 000 004	AD 00 04	LDA &0400
9199	240 016	F0 10	BEQ 16 --> &91AB Start new output field
919B	165 030	A5 1E	LDA &1E
919D	240 012	F0 0C	BEQ 12 --> &91AB Start new output field
919F	237 000 004	ED 00 04	SBC &0400
91A2	176 249	B0 F9	BCS -7 --> &919D
91A4	168	A8	TAY
91A5	032 146 189	20 92 BD	JSR &BD92 Print a space to the screen
91A8	200	C8	INY

```
91A9       208 250        D0 FA       BNE -6 --> &91A5
91AB       024            18          CLC
91AC       173 000 004    AD 00 04    LDA &0400
91AF       133 020        85 14       STA &14
91B1   f   102 021        66 15       ROR &15
91B3       032 224 142    20 E0 8E    JSR &8EE0 Get next non-space character (PTR A)
91B6   :   201 058        C9 3A       CMP#&3A
91B8       240 008        F0 08       BEQ 8 --> &91C2
91BA       201 013        C9 0D       CMP#&0D
91BC       240 004        F0 04       BEQ 4 --> &91C2
91BE       201 139        C9 8B       CMP#&8B
91C0       208 025        D0 19       BNE 25 --> &91DB
91C2       032 146 186    20 92 BA    JSR &BA92 Output new line & Zero COUNT
91C5   L   076 000 144    4C 00 90    JMP &9000 End of PRINT. Check end of Statement & continue to process next program statement
91C8   d   100 020        64 14       STZ &14
91CA   d   100 021        64 15       STZ &15
91CC       032 224 142    20 E0 8E    JSR &8EE0 Get next non-space character (PTR A)
91CF   :   201 058        C9 3A       CMP#&3A
91D1       240 242        F0 F2       BEQ -14 --> &91C5
91D3       201 013        C9 0D       CMP#&0D
91D5       240 238        F0 EE       BEQ -18 --> &91C5
91D7       201 139        C9 8B       CMP#&8B
91D9       240 234        F0 EA       BEQ -22 --> &91C5
91DB   ~   201 126        C9 7E       CMP#&7E
91DD       240 210        F0 D2       BEQ -46 --> &91B1
91DF   ,   201 044        C9 2C       CMP#&2C
91E1       240 179        F0 B3       BEQ -77 --> &9196 Comma (pad field)
91E3   ;   201 059        C9 3B       CMP#&3B
91E5       240 225        F0 E1       BEQ -31 --> &91C8 Semicolon (check for end of PRINT statement)
91E7   z   032 122 146    20 7A 92    JSR &927A Check for """, 'TAB(' or 'SPC'
91EA       144 199        90 C7       BCC -57 --> &91B3
91EC       165 020        A5 14       LDA &14
91EE   H   072            48          PHA
91EF       165 021        A5 15       LDA &15
91F1   H   072            48          PHA
91F2       198 027        C6 1B       DEC &1B
91F4   ;   032 059 157    20 3B 9D    JSR &9D3B Get the result of the expression at PTR B
91F7   h   104            68          PLA
91F8       133 021        85 15       STA &15
91FA   h   104            68          PLA
91FB       133 020        85 14       STA &14
91FD       165 027        A5 1B       LDA &1B
91FF       133 010        85 0A       STA &0A
9201       152            98          TYA
9202       240 019        F0 13       BEQ 19 --> &9217 Print output string
9204       032 024 161    20 18 A1    JSR &A118 NUMASC (Convert Numeric value to ASCII String)
9207       165 020        A5 14       LDA &14
9209   8   056            38          SEC
920A   6   229 054        E5 36       SBC &36
```

920C		144 009	90 09	BCC 9 --> &9217 Print Output string
920E		240 007	F0 07	BEQ 7 --> &9217 Print output String
9210		168	A8	TAY
9211		032 146 189	20 92 BD	JSR &BD92 Output a space character
9214		136	88	DEY
9215		208 250	D0 FA	BNE -6 --> &9211
9217	6	165 054	A5 36	LDA &36
9219		240 152	F0 98	BEQ -104 --> &91B3
921B		160 000	A0 00	LDY#&00
921D		185 000 006	B9 00 06	LDA &0600,Y
9220		032 152 189	20 98 BD	JSR &BD98 Output character in A
9223		200	C8	INY
9224	6	196 054	C4 36	CPY &36
9226		208 245	D0 F5	BNE -11 --> &921D
9228		128 137	80 89	BRA -119 --> &91B3

927A Check special print character (', TAB(and SPC)

927A		166 011	A6 0B	LDX &0B
927C		134 025	86 19	STX &19
927E		166 012	A6 0C	LDX &0C
9280		134 026	86 1A	STX &1A
9282		166 010	A6 0A	LDX &0A
9284		134 027	86 1B	STX &1B
9286	'	201 039	C9 27	CMP#&27 char = (')
9288		240 221	F0 DD	BEQ -35 --> &9267 (PRINT) quote (')
928A		201 138	C9 8A	CMP#&8A char = TAB(
928C		240 179	F0 B3	BEQ -77 --> &9241 (PRINT) TAB(
928E		201 137	C9 89	CMP#&89 char = SPC
9290		240 201	F0 C9	BEQ -55 --> &925B (PRINT) SPC
9292	8	056	38	SEC
9293	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

BD92 Output character to screen

Submitted by Steve Fewell

Description:

There are various routines which output to the screen. Here is the main one used when printing to the screen.

If called from &BD8F then routine &BD6C is executed before contining to &BD92.

&BD6C outputs the 2-digit Hexadecimal number (stored in A), as follows:

Store A to the Stack.
Divide A by 16 (to move the top 4-bits of the byte to the lower half, and clearing the top 4-bits).
Call routine &BD77 to output the upper Hex digit as follows:

* Add #&06 to A if A is greater than or equal to &0A.
* Add #&30 to A (to turn A from a binary Hex digit to an ASCII character).
* Store A to the stack.
* If WIDTH (&23) is less than COUNT (&1E) then call &BA92 to output a new line.
* Retrieve A from the stack.
* Increment COUNT (&1E).
* Jump to the Operating System's Write Character vector (&020E) to output the ASCII character in A.

Retrieve A from the Stack (now A = the original value of A when the routine was entered).
AND A with #&0F to clear the top 4-bits (as the Hex character these represent has already been output).
Continue to routine &BD77, which outputs the lower Hex digit as follows:

* Add #&06 to A if A is greater than or equal to &0A.
* Add #&30 to A (to turn A from a binary Hex digit to an ASCII character).
* Store A to the stack.
* If WIDTH (&23) is less than COUNT (&1E) then call &BA92 to output a new line.
* Retrieve A from the stack.
* Increment COUNT (&1E).
* Jump to the Operating System's Write Character vector (&020E) to output the ASCII character in A.

If called from &BD92 then Set A to #&20 (Space character). This forces the character to output to be a space.

&BD94 Overview. This routine outputs the character with correct handling of the following:

* The COUNT variable
* New line when COUNT exceeds the WIDTH value
* Redirection of output (to memory) when *EDIT-mode is active
* New line when character to output is &0D ('<cr>')

If called from &BD94 then Check whether &1F (LISTO Flag) is negative (indicating '*EDIT' mode), if it is then

goto &BDA2; otherwise, con tinue to &BD98. In '*EDIT' mode any output to the screen is instead redirected to memory so that '*EDIT' can pickup the text (program listing) and allow the user to edit it. If we are not in '*EDIT' mode then continue to &BD98. &BDA2 stores the ASCII code in A to VARTOP (the memory location pointed to by (&02, &03)), and then increments VARTOP LSB (&02). If the VARTOP LSB (&02) doesn't overflow then exit; otherwise, increment &03 also, and do the following check (storing A temporarily to the stack while it is done): If VARTOP MSB byte (&03) = HIMEM MSB byte (&07), goto &BD31 to call the BASIC Initialise routine (to reset, VARTOP, LOMEM, HIMEM, etc...) & generate a 'No Room' error. Otherwise, exit the output routine (as we are not out of memory for '*EDIT' data).

If called from &BD98 and the ASCII character to output (in A) is #&0D (Carriage return) then send the character to the screen (JSR &FFEE), zero COUNT (&1E) and exit.

Otherwise (&BD7F):

* Store the character on the stack.
* If WIDTH (&23) is less than COUNT (&1E) then call &BA92 to output a new line.
* Retrieve A from the stack.
* Increment COUNT (&1E).
* Jump to the Operating System's Write Character vector (&020E) to output the ASCII character in A to the screen.
* Exit the output routine.

BDB3 routine

If called from &BDB3 then the following is done:

Clear carry flag.
AND A with &1F to clear the top 3 bits.
If A now contains zero then exit.
If X is negative (top bit set) then exit.
Multiply X by 2 (shift the bits left 1 position).
If X is now zero then exit.
continue to &BDBF to output X number of spaces.

[&BDBF] Output X number of Spaces, as follows:

[&BDBF] Call routine &BD92 to output a space.
Decrement X.
if X is still positive then jump back to &BDBF.
Exit.
Note: Line breaks are automatically made as needed (as X can be a maximum of 255).

Disassembly for the Output Character to screen routine

BD6C	H	072	48	PHA
BD6D	J	074	4A	LSR A
BD6E	J	074	4A	LSR A
BD6F	J	074	4A	LSR A
BD70	J	074	4A	LSR A
BD71	w	032 119 189	20 77 BD	JSR &BD77
BD74	h	104	68	PLA
BD75	)	041 015	29 0F	AND#&0F
BD77		201 010	C9 0A	CMP#&0A
BD79		144 002	90 02	BCC 2 --> &BD7D
BD7B	i	105 006	69 06	ADC#&06
BD7D	i0	105 048	69 30	ADC#&30
BD7F	H	072	48	PHA

BD80	#	165 035	A5 23	LDA &23
BD82		197 030	C5 1E	CMP &1E
BD84		176 003	B0 03	BCS 3 --> &BD89
BD86		032 146 186	20 92 BA	JSR &BA92
BD89	h	104	68	PLA
BD8A		230 030	E6 1E	INC &1E
BD8C	l	108 014 002	6C 0E 02	JMP (&020E) WRCHV (OS Write Character vector)
BD8F	l	032 108 189	20 6C BD	JSR &BD6C
BD92		169 032	A9 20	LDA#&20
BD94	$	036 031	24 1F	BIT &1F
BD96	0	048 010	30 0A	BMI 10 --> &BDA2
BD98		201 013	C9 0D	CMP#&0D
BD9A		208 227	D0 E3	BNE -29 --> &BD7F
BD9C		032 238 255	20 EE FF	JSR &FFEE OSWRCH (Output the <cr> character to the screen)
BD9F	L	076 149 186	4C 95 BA	JMP &BA95 Zero COUNT (&1E) and exit routine
BDA2		146 002	92 02	STA (&02)
BDA4		230 002	E6 02	INC &02
BDA6		208 029	D0 1D	BNE 29 --> &BDC5
BDA8		230 003	E6 03	INC &03
BDAA	H	072	48	PHA
BDAB		165 003	A5 03	LDA &03
BDAD	E	069 007	45 07	EOR &07
BDAF		240 128	F0 80	BEQ -128 --> &BD31 Initialise page &7 etc... and generate 'No room' error
BDB1	h	104	68	PLA
BDB2	`	096	60	RTS
BDB3		024	18	CLC
BDB4	037 031	25 1F	AND &1F	
BDB6		240 013	F0 0D	BEQ 13 --> &BDC5
BDB8		138	8A	TXA
BDB9	0	048 010	30 0A	BMI 10 --> &BDC5
BDBB	*	042	2A	ROL A
BDBC		170	AA	TAX
BDBD		240 006	F0 06	BEQ 6 --> &BDC5
BDBF		032 146 189	20 92 BD	JSR &BD92 Output a Space
BDC2		202	CA	DEX
BDC3		208 250	D0 FA	BNE -6 --> &BDBF
BDC5	`	096	60	RTS

Initialise page &7 (etc...) and generate 'No room' error

BD31		032 172 187	20 AC BB	JSR &BBAC Initialise Page 7 & reset Variable pointers, etc...
BD34	L	076 161 144	4C A1 90	JMP &90A1 No room error

BASIC IV ROM Routines

Page Last Altered: undefined

BA92 Start new output line

Submitted by Steve Fewell

Description:

Call &FFE7 to output new line character(s), (meaning that further display will start on the next line).
Zero the COUNT flag (&1E) as now we are at the start of a new line and exit.

Disassembly for the Start new output line routine

BA92		032 231 255	20 E7 FF	JSR &FFE7 OSNEWLINE - Output new line character
BA95	d	100 030	64 1E	STZ &1E [Zero COUNT]
BA97	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A118 NUMASC (Numeric value to ASCII String conversion)

Submitted by Steve Fewell

Routine:numasc
Name: NUMASC (Numeric value to ASCII String conversion)
Starting Address: &A118
Entry criteria: Either the IWA or the FWA contains the numeric value to convert into ASCII (for printing to the screen, etc...).
&15 is negative if the value should be converted to ASCII with a Hexadecimal representation of the value.
Y is negative if the value is a Floating-Point value (stored in the FWA), or positive
if the value is an Integer value (stored in the IWA).
Exit: The SWA contains the ASCII result.

Description:

A118 - A137: validate the format & number of digits to print
The format that the number will be converted to and the number of digits to print
is specified by variable @%. This variable consists of 4 bytes with the following meanings:
Byte 4: Conains a non-zero value if STR$ function should use @% Number conversion format (default value 0).

Byte 3: Contains one of the following:
- 0 - for General format (e.g. nnn.nnn), Byte 2 contains max digits printed
- 1 - for Exponential format (e.g. n.nnnnnEnn), Byte 2 contains number of digits printed
- 2 - for Fixed format (e.g. nnn.nnnnn), Byte 2 contains numberof digits after decimal point to print

The default value for byte 3 is 0 (general format).

Byte 2: Has the following meanings depending on Format type:
- If General format, contains number of digits to print before Exponential format needs to be used.
- If Exponential format, contains maximum number of significant digits to be printed after the decimal point.
- If Fixed format, contains exact number of digits required to follow the decimal point.

The default value for byte 2 is 9 (print 9 decimals).

Byte 1: Print field width (0-255) (when using the PRINT statement with commas).
Only bytes 3 and 2 are used by the NUMASC routine. The default value for byte 1 is 10.

This routine sets &37 to a copy of byte 3 and &4D to a copy of byte 2.
If the Format is not between 0 to 2, then &37 (format) is set to 0.
If the Number of decimals is not between 0 to 10, then &4D (decimals) is set to 10.
If the Format is not 2 (Fixed) and the number of decimals is 0 then set &4D (decimals)

to 10.
If this routine is called from location &A132, then the @% format settings will not be used.

A138 - A154: Initialise & Convert numeric value to Float & handle 0.00 valuet
Sets &36 to zero (SWA length), as we are starting with a blank string.
Set &48 to zero (this is the Exponential value (the value printed after the 'E')).
If &15 is negative then convert the numeric value to an ASCII string in Hexadecimal notation (&A0CA).
If the numeric value is an Integer (Y is positive, most likely #&40, which means Integer) then convert the Integer value to a Float (storing the result in the FWA). This is so that the routine only needs to handle Floating-Point values (i.e. one format of storage).
Call routine &A3F2 to obtain the sign of the Floating-Point value.
If the sign is 0 then the value is 0.00, so if the format is 1 (exponential) or 2 (Fixed) then jump to &A1D0 to print the value 0.00 in that format; Otherwise, set A to "0" and jump to &A2D0 to add a single 0 to the SWA, and exit.
If the value is not 0.00 then jump to &A102 to check the value.

A0CA - A101: Convert Numeric value to ASCII String in Hexadecimal notation
If the Numeric value is a Floating-Point value (Y is negative) then call &96C3 to convert the Floating-Point value to Integer and put the result in the IWA.
This is because only the Integer portion of Hexadecimal values is converted.
Additionally, the Hexadecimal conversion for negative values is in two's complement format, I.e. -4 in BASIC's Hexadecimal format ('~') is &FFFFFFFC.

Set X and Y to zero. Y is the current byte offset and X is the next free location for the Hex values.

Loop for each byte of the IWA value. For each byte do the following:
* Get the next byte of the Integer value (LSB first) and push this value to the stack.
* Clear the top half (top 4 bits of the value). This leaves the lower Hex digit of the 2-digit byte (4 bits per digit).
* Store the Hex digit (lower half of byte) in the next location (&3F-&46) and increment X.
* Pop the byte from the Stack. Divide this value by 8 (to get rid of the lower 4-bits of the value) and move the top 4-bits to the lower half of the byte.
* Store the Hex digit (top half of byte) in the next location (&3F-&46) and increment X.
* Increment Y (byte offset) to point to the next byte in the IWA.
* If the byte offset (Y) is less than 4 then loop again.

Now, locations &3F to &46 contain the 8 Hex digit values for the Integer value in the IWA.

Now, loop for each of the Hex dgits (starting at the last one, &46, which is the most significant digit).
For each digit, check if it is zero. If it is zero (and it isn't the first digit) then skip it, as we do not want leading zeros in the ASCII string value.

When we have found the first non-zero digit, or the last digit (if all other digits are zero), then we can begin to add the digits to the SWA (ASCII string value).
To add a digit, add #&30 to the value (plus an extra #&07 if the digit is 10-15 ['A'-'F']).
Now, we have the ASCII value for the Hex digit, so call routine &A2D0 to add the ASCII character to the end of the existing SWA string value.
Next, decrement X and repeat the adding digit to SWA process for the other digits.
When X is negative then we have finished as the last digit added to the SWA value was the least significant digit of the hexadecimal value.

Finally exit as we have finished the ASCNUM conversion.

A102 - A10A: Handle '-' for negative Floating-Point values
If the Floating-Point value is negative (the FWA sign routine returned with the negative flag set)

so add a leading '-' to the SWA (ASCII String) output value [via routine &A2D0].

A10B - A117: Multiply Floating-Point value until it is >= 1
Test the FWA exponent, if the exponent is more than or equal to #&81 then jump to &A15C to check that the Floating-Point value is less than 1.
If the FWA exponent is less than #&81 then the FWA value is less than 1 so multiply the numeric value by 10 [via routine &A436] and decrement &48 (the Exponent value).
Keep repeating this until the FWA exponent is >=&81 (i.e. we have a value with an Integer part between 1 and 9.99999999 - this representation will aid the conversion to ASCII procedure).
When the value is within the required range then jump to &A15C.

A15C - A16E: Divide Floating-Point value until it is <= 9.9999999999
Test the FWA exponent, if the exponent is less then #&84 then jump to &A16F as the FWA value is within the required range (less than 10).
If the FWA exponent is equal to #&84 then check the FWA Mantissa byte 1 (MSB) value.
If the FWA Mantissa byte 1 value is less ten &A0 then the FWA contains a value less than 10, so jump to &A16F to continue the conversion procedure.
Otherwise the FWA value is >&eq;10, so divide the numeric value by 10 [via routine &A478] and increment &48 (the Exponent value).
Jump back to &A10B to check the FWA value again.
Keep repeating this until the FWA contains a value between 1 and 9.99999999, and then jump to &A16F to continue the conversion to ASCII procedure.

A16F - A1CF: Check that the value can be printed in the requested format
At this stage the Floating-Point value is between 1.0 and 9.99999999999, and &48 contains the base 10 exponent value (so that the original value can be determined).

Temporarily store the FWA rounding byte (&35) into location &27 and store the FWA value as a packed Floating-Point variable at location &046C-&0470.
The rounding byte needs to be stored separately as it is not included in the packed variable.

If the requested format is Fixed-Point then check the following:

* Add the number of digits required to follow the decimal point (&4D) to the exponent (&48).
* If the result is negative then the Floating-Point value is too small to fix to the number of decimal places suggested, so set the FWA to 0.00 & zero the exponent (&48) [by jumping to &A1D4].
* If the result is less than 11 then the value will fit in the display field (without the need to display the 'E' value (as in the exponential format)).
* Otherwise, if the result is more than 11 then the value's exponent is too large to display in Fixed-point format without including the exponent value as part of the output, so set the number of digits (&38) to 10 and set the format to 0 (general), so that the 'E' part of the value will be output.

[A190] Set the FWA to 5.0 and divide this value by 10 * The number of digits to output.
I.e. if the number of digits to output is 9 then set FWA to 0.000000005.
Unpack the temporary Floating-Point variable stored at &046C-&0470 (i.e. the numeric value that we are outputting) to the FWB, and restore the
Rounding byte (that we temporarily stored to &27).
Add the FWB number to the FWA value. This will round up the last significant digit (that we are outputting) if the entire numeric value cannot be outputted (to the required number of decimals) and the next (non-outputted) digit is >&eq;5.

If the FWA exponent is less than #&84 then keep dividing the FWA mantissa by 2 and incrementing the FWA exponent until the FWA exponent >&eq;#&84. This will make sure that the FWA value is >&eq;1 if the calculations above caused the FWA value to drop below 1.0.
If the FWA exponent is more than #&A0 then the calculations above caused the FWA value to be rounded

to more than 10.0, so jump to &A157 to set the FWA value to 1.0 and increment the decimal exponent (&48) by 1 to result in an actual FWA value of 10.0. This solves the problems caused by the rounding.

If the number of digits is 0 then [&A1D4] set the FWA value to 0.0 and the decimal exponent (&48) to 0, as there are no digits required to be output.
Otherwise, jump to &A1DE to output the value to the requested number of digits in the requested format.

A1D0 - A1DD: Set Floating-Point value to 0.00 if A is not 1
If the number format requested (A) is 0 (general) or 2 (Fixed) then the FWA value to print is 0.00, so set the FWA to 0.00 and [&A1D4] zero &48 (the decimal Exponent); next, increment the number of decimals to print (&38) [from 0 to 1], as 0 decimals were originally requested.

A1DE - A2CF: Convert Floating-Point value to ASCII in required format
If the requested number format is 1 (Exponential format) then [&A215] Set the position of the decimal point to 1 (&4D) - as in exponential format, the decimal point always appears after the first digit; andÿ jump to &A217 to convert the Floating-Point value to ASCII.

Otherwise (format is General or Fixed-Point).
If the exponent value (&48) is positive and greater than or equal to the number of decimals to print (&38), then the value will be converted in Exponential-format (as it is too large to be converted without the 'E'xponent), so [&A215] Set the position of the decimal point to 1 (&4D) - as in exponential format, the decimal point always appears after the first digit; & jump to &A217 to convert the Floating-Point value to ASCII.

If the exponent value (&48) is positive and less than the number of decimals to print (&38), then the value can be converted without the exponential value, so [&A215] Set the position of the decimal point (&4D) to the exponential value (&48) plus 1, so that the decimal point will occur in the correct potition (i.e. value 1E3 will print the decimal point after the 4th digit - as the decimal value is '1000.00'), clear &48 - the exponential value & jump to &A217 to convert the Floating-Point value to ASCII.

[&A1F2] If the requested format is General and the exponential value is not &FF, then the Floating-Point value contains a negative exponent that needs to be converted in exponential format. (i.e. the Floating-Point value is not a single fractional digit, i.e. 0.3), so [&A215] Set the position of the decimal point to 1 (&4D) and jump to &A217 to convert the Floating-Point value to ASCII in exponential format. This is because BASIC always converts a general format number to exponential format if it is smaller than or equal to 0.01.

Otherwise, [&A1FE] The Floating-Point value can be converted in Fixed-Point format
Firstly, add the characters "0." to the output string (SWA), as we are converting a fractional value.
Next, Keep incrementing &48 (the exponent value), until the exponent &48 is zero, for each loop add a zero "0" to the output string (SWA). This adds the required number of leading zeros to the output value.
Set &4D to #&80. &4D is the number of digits to output before the decimal point (".") is output.
The value &80 means that routine section &A217-&A2CF will not output a decimal point (as it has already been output). Now, continue with the next routine section - &A217.

A217 - A2CF: Convert the Floating-Point value to ASCII string (in SWA)
Call &A26C to add the top digit of the Floating-Point (FWA) Mantissa value to the output string and then remove the top digit and multiply the rest of the FWA mantissa by 10 (so that the new top digit is the next digit).

&A26C does the following:

1) Load FWA Mantissa byte 1 and divide it by 8 (to place the top 4 bits in the lower 4 bits - also clearing the top 4 bits.
2) Call &A2CE to ORA the 4 lower bits (from step 1) with #&30 (this adds &30 to the digit value, and thus converts the digit from a numeric value to ASCII value (offset from 48)) and continue to &A2D0 to add the digit to the output string.

3) Set the top 4 bits of the FWA mantissa to zero (this removes the digit that we have just output from the FWA mantissa value).
4) Store result from step 3, and FWA Mantissa bytes &31 to &33 to the stack.
Set X to &34 and A to &35.
5) Multiply FWA Mantissa (A and Mantissa bytes &34 to &31) by 4
6) Add the FWA Mantissa values (now, multiplied by 4) to the previous FWA mantissa values (before the multiplication, stored in &35, X and the other 3 bytes from the Stack).
7) Now the FWA mantissa has been multiplied by 5 (4 plus 1), so multiply the FWA mantissa again, this time by 2, in order to achieve the desired result of multipling the FWA value by 10.
8) Exit with A = the lower 4 bits of &31 (before the multiplication by 10 occurred).

Next, decrement &4D (the number of digits to output before the decimal point value).
If &4D has reached zero then add a decimal point, ".", to the output string.

Decrement &38, the number of decinals to print, and if this value hasn't reached zero then there are still further digits to print, so jump back to &A217 to output the next digit.

Now, all required digits have been added to the output string.

If the output format is 0 (general) then strip any trailing zeros and decimal points, as follows:
1) Set Y to &36; i.e. the next free character in the output string.
2) Decrement Y (to point to the next character from the end of the output value).
3) If the current last character (&0600, Y) is zero then go back to step 2 (to skip all trailing zeros)
4) If the current last character is a decimal point then skip this also.
5) Now Y points to the actual length of the value, excluding any training zeros and decimal points, so store Y in &36 to set the new length of the output string (SWA).

If the output format is 2 (fixed-point) or 0 (general) and the Exponent (&48) is zero then exit.
Otherwise, there is an exponent value to output.

The exponent value is outputted as follows:
1) Add "E" to the output string value.
2) If &48 is negative then add "-" to the output string and subtract the exponent (&48) from zero to obtain the positive exponent value (reverse the two's compliment).
3) [&A2BF] Keep subtracting 10 from the exponent value and incrementing X until the exponent value is less than 10. Now X contains the ten's part (first digit) of the exponent value.
4) A now contains the number of units, the second digit of the exponent value. Store this value on the stack.
5) If the number of ten's (X) (the first digit) of the exponent value is not zero then set A = X and call &A2D0 to add the digit to the end of the output string.
6) Pop the number of units (second digit) in the exponent value from the stack, and call &A2D0 to add the digit to the end of the output string.
7) If the output format (&37) is 0 (general) then exit.
Otherwise, we may need to add spaces to the end of the output value, to ensure that the exponent value lines up with other printed numerical values.
8) If the exponent value (&48) is positive then add a space to the end of the output string (to make up for the "-" that we didn't need to add!).
9) If X is zero (the exponent was 1 digit long, no ten's value) then add a space to the end of the output string (to make up for the extra digit that we did not need to add!).

Exit from the routine as the SWA now contains the original numerical value (from the FWA) converted to ASCII format.
Note: this routine does not preserve the original value of the FWA.

A2D0 - A2D9: Add character to the end of the current ASCII value
Temporarily store X on the stack.

Store the ASCII character in A in location &0600 + &36 (the end of the current SWA value).
Retrieve X from the stack.
Increment &36, so that the SWA length is now correctly incrememted by 1 character and return.

Examples:

Example 1:
Value = 1000 (FWA Exponent is &8A, FWA Mantissa is &FA)
Output format = 0 (General format).
The NUMASC routine will process this value as follows:
[&A10B] The FWA exponent is > &83, so divide the FWA by 10 and increment the decimal exponent.
Now, FWA = 100 (Exponent &87, Mantissa &C8). &48 = 1.
The FWA exponent is > &83, so divide the FWA by 10 and increment the decimal exponent.
Now, FWA = 10 (Exponent &84, Mantissa &A0). &48 = 2.
The FWA exponent is > &83, so divide the FWA by 10 and increment the decimal exponent.
Now, FWA = 1 (Exponent &81, Mantissa &80). &48 = 3.

[&A16F] ?&38 = ?&4D (9).
Output format is not 2 so goto &A190.
Decimals (?&38) is 9, so the roundup value is 0.000000005.
Add roundup value to numeric value = 1.000000005 (Exponent = &81, Mantissa = &8000000ABC).
FWA Exponent is < &84, so divide FWA by 2 & increment exponent; FWA now = 1.000000005 (Exp: &82, Mantissa: &400000055E)
FWA Exponent is < &84, so divide FWA by 2 & increment exponent; FWA now = 1.000000005 (Exp: &83, Mantissa: &20000002AF)
FWA Exponent is < &84, so divide FWA by 2 & increment exponent; FWA now = 1.000000005 (Exp: &84, Mantissa: &1000000157)

[&A1C6] FWA Mantissa byte 1 is less than #&A0, so okay, continue to &A1DE.
[&A1DE] Format is not scientific and Exponent is not negative or greater than &48 (the number of digits to print, so the value can be outputted without the exponent, so set &48 to zero.
The exponent is not #&FF so jump to &A215.
[&A215] Set ?&4D to decimal point position (A) = position 4 in the output string.
[&A217] Call &A26C (output top digit "1", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&0000000D66)
Decrement &4D to 3; &4D is not zero, so [&A223] Decrement &38 to 8; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&00000085FC)
Decrement &4D to 2; &4D is not zero, so [&A223] Decrement &38 to 7; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&0000053BD8)
Decrement &4D to 1; &4D is not zero, so [&A223] Decrement &38 to 6; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&0000345670)
Decrement &4D to 0; &4D is zero, so [&A21E] Add "." to the output string.
Decrement &38 to 5; (no decrement for the "." as it isn't a significant digit!). Jump to &A217 as &38 not zero.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&00020B6060)
Decrement &4D to &FF; &4D is not zero, so [&A223] Decrement &38 to 4; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&001471C3C0)
Decrement &4D to &FE; &4D is not zero, so [&A223] Decrement &38 to 3; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&00CC71A580)
Decrement &4D to &FD; &4D is not zero, so [&A223] Decrement &38 to 2; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&07FC707700)
Decrement &4D to &FC; &4D is not zero, so [&A223] Decrement &38 to 1; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&4FDC64A600)
Decrement &4D to &FB; &4D is not zero, so [&A223] Decrement &38 to 0; &38 is now zero so goto &A227.

[&A227] Now the output string is contained in SWA locations &600-&609, which contain the value "1000.00000".
Strip any trailing "0" and "."-characters from the value, now the value is "1000".
The Exponent value (&48) is zero, meaning that there is no exponent to print, so exit.

Example 2:
Value = 28.96301767 (FWA Exponent is &85, FWA Mantissa is &E7B4429C00)
Output format = 1 (Exponential format).
The NUMASC routine will process this value as follows:
[&A10B] The FWA exponent is > &83, so divide the FWA by 10 and increment the decimal exponent.
Now, FWA = 2.896301767 (Exponent &82, Mantissa &B95D021666). &48 = 1.

[&A16F] ?&38 = ?&4D (9).
Output format is not 2 so goto &A190.
Decimals (?&38) is 9, so the roundup value is 0.000000005.
Add roundup value to numeric value = 2.896301772 (Exponent = &82, Mantissa = &B95D021BC4).
FWA Exponent is < &84, so divide FWA by 2 & increment exponent; FWA now = 2.896301772 (Exp: &83, Mantissa: &5CAE810DE2)
FWA Exponent is < &84, so divide FWA by 2 & increment exponent; FWA now = 2.896301772 (Exp: &84, Mantissa: &2E574086F1)

[&A1C6] FWA Mantissa byte 1 is less than #&A0, so okay, continue to &A1DE.
[&A1DE] Format is scientific so jump to &A215
[&A215] Set ?&4D to decimal point position (A) = position 1 in the output string (as we are outputting in exponential format).
[&A217] Call &A26C (output top digit "2", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&8F6885456A)
Decrement &4D to 0; &4D is zero, so [&A21E] Add "." to the output string.
Decrement &38 to 8; (no decrement for the "." as it isn't a significant digit!). Jump to &A217 as &38 not zero.
[&A217] Call &A26C (output top digit "8", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&9A1534B624)
Decrement &4D to &FF; &4D is not zero, so [&A223] Decrement &38 to 7; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "9", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&64D40F1D68)
Decrement &4D to &FE; &4D is not zero, so [&A223] Decrement &38 to 6; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "6", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&3048972610)
Decrement &4D to &FD; &4D is not zero, so [&A223] Decrement &38 to 5; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "3", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&02D5E77CA0)
Decrement &4D to &FC; &4D is not zero, so [&A223] Decrement &38 to 4; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&1C5B0ADE40)
Decrement &4D to &FB; &4D is not zero, so [&A223] Decrement &38 to 3; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "1", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&7B8E6CAE80)
Decrement &4D to &FA; &4D is not zero, so [&A223] Decrement &38 to 2; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "7", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&73903ED100)
Decrement &4D to &F9; &4D is not zero, so [&A223] Decrement &38 to 1; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "7", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&23A2742A00)
Decrement &4D to &F8; &4D is not zero, so [&A223] Decrement &38 to 0; &38 is now zero so goto &A244.

[&A244] Now the output string is contained in SWA locations &600-&609, which contain the value "2.89630177".
The output format is exponential, so add "E" to the output string (for the exponent).
The Exponent value (&48) is positive (1), so no minus sign required.
Call &A2BC to add the exponent to the output string - "1" is added.
The exponent was not negative, so add a space to the end of the output string.
The exponent was only 1-digit long, so add a space to the end of the output string.
The SWA output string value is now: "2.89630177E1 " so exit as complete.

Example 3:

Value = 3.1417321E-5 (FWA Exponent is &72, FWA Mantissa is &8F55347000)
Output format = 2 (Fixed-Point format).
The NUMASC routine will process this value as follows:
[&A10B] The FWA exponent is < &81, so multiply the FWA by 10 and increment the decimal exponent.
Now, FWA = 0.0003417321 (Exponent &75, Mantissa &B32A818C). &48 = &FF.
The FWA exponent is < &81, so multiply the FWA by 10 and increment the decimal exponent.
Now, FWA = 0.003417321 (Exponent &78, Mantissa &EFF521EF). &48 = &FE.
The FWA exponent is < &81, so multiply the FWA by 10 and increment the decimal exponent.
Now, FWA = 0.03417321 (Exponent &7C, Mantissa &8BF93536). &48 = &FD.
The FWA exponent is < &81, so multiply the FWA by 10 and increment the decimal exponent.
Now, FWA = 0.3417321 (Exponent &7F, Mantissa &AEF78283). &48 = &FC.
The FWA exponent is < &81, so multiply the FWA by 10 and increment the decimal exponent.
Now, FWA = 3.417321 (Exponent &82, Mantissa &DAB5632366). &48 = &FB.

[&A16F] ?&38 = ?&4D (9).
Output format is 2 (fixed) so do the following:
Add &48 (decimal exponent) to &38 (decimals to output) = &FB + 9 + 1 (carry) = &05.
The result (&05) is < #&0B, so Set &38 (digits to output) to 5 (the result), zero the exponent (&48),
and goto &A190.
[&A190] Decimals is 9, so the roundup value is 0.000000005.
Add roundup value to numeric value = 3.417321005 (Exponent = &82, Mantissa = &DAB56328C4).
FWA Exponent is < &84, so divide FWA by 2 & increment exponent; FWA now = 3.417321005 (Exp: &83, Mantissa: &6D5AB19462)
FWA Exponent is < &84, so divide FWA by 2 & increment exponent; FWA now = 3.417321005 (Exp: &84, Mantissa: &36AD58CA31)

[&A1C6] FWA Mantissa byte 1 is less than #&A0, so okay, continue to &A1E4.
[&A1DE] The exponent is negative and format is 2 (Fixed), so goto &A1FE.
[&A1FE] Add "0." to the output string (SWA).
Increment the Exponent value (&48), and keep adding "0" to the output string until &48 is 0.
Now the output string is "0.0000".
[&A213] Set ?&4D to &80 (as we do not need to print a decimal point now).
[&A217] Call &A26C (output top digit "3", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&42C577E5EA)
Decrement &4D to &7F; &4D is not zero, so [&A223] Decrement &38 to 4; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "4", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&1BB6AEFB24)
Decrement &4D to &7E; &4D is not zero, so [&A223] Decrement &38 to 3; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "1", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&7522D5CF68)
Decrement &4D to &7D; &4D is not zero, so [&A223] Decrement &38 to 2; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "7", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&335C5A1A10)
Decrement &4D to &7C; &4D is not zero, so [&A223] Decrement &38 to 1; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "3", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&219B8504A0)
Decrement &4D to &7B; &4D is not zero, so [&A223] Decrement &38 to 0; &38 is now zero so goto &A240.
[&A240] The exponent (&48) is zero so exit with SWA = "0.000034173".

Example 4:
value = 8.92381E-10 (FWA Exponent is &62, FWA Mantissa is &F54BBAF5)
Output format = 0 (General format).
The NUMASC routine will process this value as follows:
[&A10B] While the FWA exponent is < &81, so multiply the FWA by 10 and increment the decimal exponent.
Now, FWA = 8.92381 (Exponent &84, Mantissa &8EC7ECFEB7). &48 = &F6 (-10).

[&A16F] ?&38 = ?&4D (9).
Output format is not 2 so goto &A190.
Decimals (?&38) is 9, so the roundup value is 0.000000005.

Add roundup value to numeric value = 8.923810005 (Exponent = &84, Mantissa = &8EC7ED000E).

[&A1C6] FWA Mantissa byte 1 is less than #&A0, so okay, continue to &A1E4.
[&A1E4] The Exponent is negative, so [&A1F4] check whether the exponent is = #&FF
as a single zero before the first non-zero digit after the decimal point is the threshold for the
general output format. Our value does not meet this threshold, so store 1 in &4D (output in Exponential format).

[&A217] Call &A26C (output top digit "8", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&93CF42008C)
Decrement &4D to 0; &4D is zero, so [&A21E] Add "." to the output string.
Decrement &38 to 8; (no decrement for the "." as it isn't a significant digit!). Jump to &A217 as &38 not zero.
[&A217] Call &A26C (output top digit "9", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&2618940578)
Decrement &4D to &FF; &4D is not zero, so [&A223] Decrement &38 to 7; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "2", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&3CF5C836B0)
Decrement &4D to &FE; &4D is not zero, so [&A223] Decrement &38 to 6; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "3", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&8199D222E0)
Decrement &4D to &FD; &4D is not zero, so [&A223] Decrement &38 to 5; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "8", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&1002355CC0)
Decrement &4D to &FC; &4D is not zero, so [&A223] Decrement &38 to 4; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "1", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&0016159F80)
Decrement &4D to &FB; &4D is not zero, so [&A223] Decrement &38 to 3; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&00DCD83B00)
Decrement &4D to &FA; &4D is not zero, so [&A223] Decrement &38 to 2; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&08A0724E00)
Decrement &4D to &F9; &4D is not zero, so [&A223] Decrement &38 to 1; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&5644770C00)
Decrement &4D to &F8; &4D is not zero, so [&A223] Decrement &38 to 0; &38 is now zero so goto &A244.

[&A244] Now the output string is contained in SWA locations &600-&609, which contain the value "8.92381000".
The output format is exponential, so add "E" to the output string (for the exponent).
The Exponent value (&48) is negative (-10), so add a minus sign ("-") to the output string.
Call &A2BC to add the exponent to the output string - "10" is added.
The exponent is subtracted from 0 in order to obtain its positive value.
The SWA output string value is now: "8.92381000E-10" so exit as complete.

Example 5:
Value = 0.8 (FWA Exponent is &80, FWA Mantissa is &CCCCCCCD00)
Output format = 2 (Fixed-Point format).
The NUMASC routine will process this value as follows:
[&A10B] The FWA exponent is < &81, so multiply the FWA by 10 and increment the decimal exponent.
Now, FWA = 8.0 (Exponent &84, Mantissa &8000000020). &48 = &FF.

[&A16F] ?&38 = ?&4D (9).
Output format is 2 (fixed) so do the following:
Add &48 (decimal exponent) to &38 (decimals to output) = &FF + 9 + 1 (carry) = &09.
The result (&09) is < #&0B, so Set &38 (digits to output) to 9 (the result), zero the exponent (&48),
and goto &A190.
[&A190] Decimals is 9, so the roundup value is 0.000000005.
Add roundup value to numeric value = 8.000000005 (Exponent = &84, Mantissa = &8000000177).

[&A1C6] FWA Mantissa byte 1 is less than #&A0, so okay, continue to &A1E4.
[&A1E4] The exponent is negative and format is 2 (Fixed), so goto &A1FE.
[&A1FE] Add "0." to the output string (SWA).
Increment the Exponent value (&48), the exponent value is now 0, so there are no further "0"'s to add after the
decimal point. Now the output string is "0.".
[&A213] Set ?&4D to &80 (as we do not need to print a decimal point now).

[&A217] Call &A26C (output top digit "8", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&0000000EA6)
Decrement &4D to &7F; &4D is not zero, so [&A223] Decrement &38 to 8; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&000000927C)
Decrement &4D to &7E; &4D is not zero, so [&A223] Decrement &38 to 7; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&000005B8D8)
Decrement &4D to &7D; &4D is not zero, so [&A223] Decrement &38 to 6; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&0000393870)
Decrement &4D to &7C; &4D is not zero, so [&A223] Decrement &38 to 5; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&00023C3460)
Decrement &4D to &7B; &4D is not zero, so [&A223] Decrement &38 to 4; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&00165A0BC0)
Decrement &4D to &7A; &4D is not zero, so [&A223] Decrement &38 to 3; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&00DF847580)
Decrement &4D to &79; &4D is not zero, so [&A223] Decrement &38 to 2; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&08BB2C9700)
Decrement &4D to &78; &4D is not zero, so [&A223] Decrement &38 to 1; &38 isn't zero so goto &A217.
[&A217] Call &A26C (output top digit "0", & multiply FWA mantissa by 10; now FWA=(Exp=&84, Mantissa=&574FBDE600)
Decrement &4D to &77; &4D is not zero, so [&A223] Decrement &38 to 0; &38 is now zero so goto &A240.
[&A240] The exponent (&48) is zero so exit with SWA = "0.800000000". Note: In Fixed-Point format the trailing zeros are not stripped.

Disassembly for the NUMASC (Numeric value to ASCII String conversion) routine

Convert numeric value to ASCII (in Hexadecimal notation)

A0CA		152	98	TYA
A0CB		016 003	10 03	BPL 3 --> &A0D0
A0CD		032 195 150	20 C3 96	JSR &96C3 Convert Float to Integer
A0D0		162 000	A2 00	LDX#&00
A0D2		160 000	A0 00	LDY#&00
A0D4	*	185 042 000	B9 2A 00	LDA &002A,Y
A0D7	H	072	48	PHA
A0D8	)	041 015	29 0F	AND#&0F
A0DA	?	149 063	95 3F	STA &3F,X
A0DC	h	104	68	PLA
A0DD	J	074	4A	LSR A
A0DE	J	074	4A	LSR A
A0DF	J	074	4A	LSR A
A0E0	J	074	4A	LSR A
A0E1		232	E8	INX
A0E2	?	149 063	95 3F	STA &3F,X
A0E4		232	E8	INX
A0E5		200	C8	INY
A0E6		192 004	C0 04	CPY#&04
A0E8		208 234	D0 EA	BNE -22 --> &A0D4
A0EA		202	CA	DEX
A0EB		240 004	F0 04	BEQ 4 --> &A0F1
A0ED	?	181 063	B5 3F	LDA &3F,X
A0EF		240 249	F0 F9	BEQ -7 --> &A0EA

A0F1	?	181 063	B5 3F	LDA &3F,X
A0F3		201 010	C9 0A	CMP#&0A
A0F5		144 002	90 02	BCC 2 --> &A0F9
A0F7	i	105 006	69 06	ADC#&06
A0F9	i0	105 048	69 30	ADC#&30
A0FB		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A0FE		202	CA	DEX
A0FF		016 240	10 F0	BPL -16 --> &A0F1
A101	`	096	60	RTS

Multiply FWA value until number is greater than or equal to 1

A102		016 007	10 07	BPL 7 --> &A10B
A104	-	169 045	A9 2D	LDA#&2D
A106	d.	100 046	64 2E	STZ &2E
A108		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A10B	0	165 048	A5 30	LDA &30
A10D		201 129	C9 81	CMP#&81
A10F	K	176 075	B0 4B	BCS 75 --> &A15C
A111	6	032 054 164	20 36 A4	JSR &A436 Floating-point multiply by 10 [FWA=FWA*10]
A114	H	198 072	C6 48	DEC &48
A116		128 243	80 F3	BRA -13 --> &A10B

NUMASC : Check conversion formats & conv IWA to FWA & handle zero

A118		174 002 004	AE 02 04	LDX &0402
A11B		224 003	E0 03	CPX#&03
A11D		144 002	90 02	BCC 2 --> &A121
A11F		162 000	A2 00	LDX#&00
A121	7	134 055	86 37	STX &37
A123		173 001 004	AD 01 04	LDA &0401
A126		240 006	F0 06	BEQ 6 --> &A12E
A128		201 010	C9 0A	CMP#&0A
A12A		176 006	B0 06	BCS 6 --> &A132
A12C		128 006	80 06	BRA 6 --> &A134
A12E		224 002	E0 02	CPX#&02
A130		240 002	F0 02	BEQ 2 --> &A134
A132		169 010	A9 0A	LDA#&0A
A134	8	133 056	85 38	STA &38
A136	M	133 077	85 4D	STA &4D
A138	d6	100 054	64 36	STZ &36
A13A	dH	100 072	64 48	STZ &48
A13C	$	036 021	24 15	BIT &15
A13E	0	048 138	30 8A	BMI -118 --> &A0CA
A140		152	98	TYA
A141	0	048 003	30 03	BMI 3 --> &A146
A143		032 133 129	20 85 81	JSR &8185 Convert Integer to Float

A146		032 242 163	20 F2 A3	JSR &A3F2 Obtain Sign of the FWA Floating-Point value
A149		208 183	D0 B7	BNE -73 --> &A102
A14B	7	165 055	A5 37	LDA &37
A14D		208 005	D0 05	BNE 5 --> &A154
A14F	0	169 048	A9 30	LDA#&30
A151	L	076 208 162	4C D0 A2	JMP &A2D0 Add ASCII character to existing output string (SWA)
A154	L	076 208 161	4C D0 A1	JMP &A1D0

Divide FWA value until number is less than 1

A157		032 216 165	20 D8 A5	JSR &A5D8 Set FWA to 1.0
A15A		128 015	80 0F	BRA 15 --> &A16B
A15C		201 132	C9 84	CMP#&84
A15E		144 015	90 0F	BCC 15 --> &A16F
A160		208 006	D0 06	BNE 6 --> &A168
A162	1	165 049	A5 31	LDA &31
A164		201 160	C9 A0	CMP#&A0
A166		144 007	90 07	BCC 7 --> &A16F
A168	x	032 120 164	20 78 A4	JSR &A478 Divide Floating-Point value by 10 [FWA=FWA/10]
A16B	H	230 072	E6 48	INC &48
A16D		128 156	80 9C	BRA -100 --> &A10B

Check the number can be converted to required format

A16F	5	165 053	A5 35	LDA &35
A171	'	133 039	85 27	STA &27
A173		032 017 165	20 11 A5	JSR &A511 Store FWA to &046C & set argp=&046C
A176	M	165 077	A5 4D	LDA &4D
A178	8	133 056	85 38	STA &38
A17A	7	166 055	A6 37	LDX &37
A17C		224 002	E0 02	CPX#&02
A17E		208 016	D0 10	BNE 16 --> &A190
A180	eH	101 072	65 48	ADC &48
A182	0P	048 080	30 50	BMI 80 --> &A1D4
A184	8	133 056	85 38	STA &38
A186		201 011	C9 0B	CMP#&0B
A188		144 006	90 06	BCC 6 --> &A190
A18A		169 010	A9 0A	LDA#&0A
A18C	8	133 056	85 38	STA &38
A18E	d7	100 055	64 37	STZ &37
A190		032 184 166	20 B8 A6	JSR &A6B8 Clear FWA (except Exp/Mantissa1)
A193		169 160	A9 A0	LDA#&A0
A195	1	133 049	85 31	STA &31
A197		169 131	A9 83	LDA#&83
A199	0	133 048	85 30	STA &30
A19B	8	166 056	A6 38	LDX &38

A19D		240 006	F0 06	BEQ 6 --> &A1A5
A19F	x	032 120 164	20 78 A4	JSR &A478 Divide Floating-Point value by 10 [FWA=FWA/10]
A1A2		202	CA	DEX
A1A3		208 250	D0 FA	BNE -6 --> &A19F
A1A5		032 146 165	20 92 A5	JSR &A592 Set argp to &046C
A1A8		032 224 164	20 E0 A4	JSR &A4E0 Unpack (&4A, &4B) variable to FWB
A1AB	'	165 039	A5 27	LDA &27
A1AD	A	133 065	85 41	STA &41
A1AF	h	032 104 131	20 68 83	JSR &8368 Floating-Point Addition
A1B2	0	165 048	A5 30	LDA &30
A1B4		201 132	C9 84	CMP#&84
A1B6		176 014	B0 0E	BCS 14 --> &A1C6
A1B8	f1	102 049	66 31	ROR &31
A1BA	f2	102 050	66 32	ROR &32
A1BC	f3	102 051	66 33	ROR &33
A1BE	f4	102 052	66 34	ROR &34
A1C0	f5	102 053	66 35	ROR &35
A1C2	0	230 048	E6 30	INC &30
A1C4		208 236	D0 EC	BNE -20 --> &A1B2
A1C6	1	165 049	A5 31	LDA &31
A1C8		201 160	C9 A0	CMP#&A0
A1CA		176 139	B0 8B	BCS -117 --> &A157
A1CC	8	165 056	A5 38	LDA &38
A1CE		208 014	D0 0E	BNE 14 --> &A1DE

If A is not 1 then set the FWA value to 0.00

A1D0		201 001	C9 01	CMP#&01	
A1D2	A	240 065	F0 41	BEQ 65 --> &A215	
A1D4		032 180 166	20 B4 A6	JSR &A6B4	JSR &A6B4 Clear FWA
A1D7	dH	100 072	64 48	STZ &48	
A1D9	M	165 077	A5 4D	LDA &4D	
A1DB		026	1A	INC A	
A1DC	8	133 056	85 38	STA &38	

Covert the numeric to the required format

A1DE		169 001	A9 01	LDA#&01
A1E0	7	197 055	C5 37	CMP &37
A1E2	1	240 049	F0 31	BEQ 49 --> &A215
A1E4	H	164 072	A4 48	LDY &48
A1E6	0	048 010	30 0A	BMI 10 --> &A1F2
A1E8	8	196 056	C4 38	CPY &38
A1EA	)	176 041	B0 29	BCS 41 --> &A215
A1EC	dH	100 072	64 48	STZ &48
A1EE		200	C8	INY

A1EF		152	98	TYA
A1F0	#	208 035	D0 23	BNE 35 --> &A215
A1F2	7	165 055	A5 37	LDA &37
A1F4		201 002	C9 02	CMP#&02
A1F6		240 006	F0 06	BEQ 6 --> &A1FE
A1F8		169 001	A9 01	LDA#&01
A1FA		192 255	C0 FF	CPY#&FF
A1FC		208 023	D0 17	BNE 23 --> &A215
A1FE	0	169 048	A9 30	LDA#&30
A200		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A203	.	169 046	A9 2E	LDA#&2E
A205		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A208	0	169 048	A9 30	LDA#&30
A20A	H	230 072	E6 48	INC &48
A20C		240 005	F0 05	BEQ 5 --> &A213
A20E		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A211		128 247	80 F7	BRA -9 --> &A20A
A213		169 128	A9 80	LDA#&80
A215	M	133 077	85 4D	STA &4D
A217	l	032 108 162	20 6C A2	JSR &A26C
A21A	M	198 077	C6 4D	DEC &4D
A21C		208 005	D0 05	BNE 5 --> &A223
A21E	.	169 046	A9 2E	LDA#&2E
A220		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A223	8	198 056	C6 38	DEC &38
A225		208 240	D0 F0	BNE -16 --> &A217
A227	7	164 055	A4 37	LDY &37
A229		136	88	DEY
A22A		240 024	F0 18	BEQ 24 --> &A244
A22C		136	88	DEY
A22D		240 017	F0 11	BEQ 17 --> &A240
A22F	6	164 054	A4 36	LDY &36
A231		136	88	DEY
A232		185 000 006	B9 00 06	LDA &0600,Y
A235	0	201 048	C9 30	CMP#&30
A237		240 248	F0 F8	BEQ -8 --> &A231
A239	.	201 046	C9 2E	CMP#&2E
A23B		240 001	F0 01	BEQ 1 --> &A23E
A23D		200	C8	INY
A23E	6	132 054	84 36	STY &36
A240	H	165 072	A5 48	LDA &48
A242	'	240 039	F0 27	BEQ 39 --> &A26B
A244	E	169 069	A9 45	LDA#&45

A246		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A249	H	165 072	A5 48	LDA &48
A24B		016 010	10 0A	BPL 10 --> &A257
A24D	-	169 045	A9 2D	LDA#&2D
A24F		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A252	8	056	38	SEC
A253		169 000	A9 00	LDA#&00
A255	H	229 072	E5 48	SBC &48
A257		032 188 162	20 BC A2	JSR &A2BC Add exponent to the output string
A25A	7	165 055	A5 37	LDA &37
A25C		240 013	F0 0D	BEQ 13 --> &A26B
A25E		169 032	A9 20	LDA#&20
A260	H	164 072	A4 48	LDY &48
A262	0	048 003	30 03	BMI 3 --> &A267
A264		032 208 162	20 D0 A2	JSR &A2D0 Add ASCII character to existing output string (SWA)
A267		224 000	E0 00	CPX#&00
A269	e	240 101	F0 65	BEQ 101 --> &A2D0 Add ASCII character to existing output string (SWA)
A26B	`	096	60	RTS
A26C	1	165 049	A5 31	LDA &31
A26E	J	074	4A	LSR A
A26F	J	074	4A	LSR A
A270	J	074	4A	LSR A
A271	J	074	4A	LSR A
A272		032 206 162	20 CE A2	JSR &A2CE
A275		169 240	A9 F0	LDA#&F0
A277	1	020 049	14 31	TRB &31
A279	H	072	48	PHA
A27A	4	166 052	A6 34	LDX &34
A27C	1	165 049	A5 31	LDA &31
A27E	H	072	48	PHA
A27F	2	165 050	A5 32	LDA &32
A281	H	072	48	PHA
A282	3	165 051	A5 33	LDA &33
A284	H	072	48	PHA
A285	5	165 053	A5 35	LDA &35
A287		010	0A	ASL A
A288	&4	038 052	26 34	ROL &34
A28A	&3	038 051	26 33	ROL &33
A28C	&2	038 050	26 32	ROL &32
A28E	&1	038 049	26 31	ROL &31
A290		010	0A	ASL A
A291	&4	038 052	26 34	ROL &34
A293	&3	038 051	26 33	ROL &33
A295	&2	038 050	26 32	ROL &32

A297	&1	038 049	26 31	ROL &31
A299	e5	101 053	65 35	ADC &35
A29B	5	133 053	85 35	STA &35
A29D		138	8A	TXA
A29E	e4	101 052	65 34	ADC &34
A2A0	4	133 052	85 34	STA &34
A2A2	h	104	68	PLA
A2A3	e3	101 051	65 33	ADC &33
A2A5	3	133 051	85 33	STA &33
A2A7	h	104	68	PLA
A2A8	e2	101 050	65 32	ADC &32
A2AA	2	133 050	85 32	STA &32
A2AC	h	104	68	PLA
A2AD	e1	101 049	65 31	ADC &31
A2AF	5	006 053	06 35	ASL &35
A2B1	&4	038 052	26 34	ROL &34
A2B3	&3	038 051	26 33	ROL &33
A2B5	&2	038 050	26 32	ROL &32
A2B7	*	042	2A	ROL A
A2B8	1	133 049	85 31	STA &31
A2BA	h	104	68	PLA
A2BB	`	096	60	RTS
A2BC		162 255	A2 FF	LDX#&FF
A2BE	8	056	38	SEC
A2BF		232	E8	INX
A2C0		233 010	E9 0A	SBC#&0A
A2C2		176 251	B0 FB	BCS -5 --> &A2BF
A2C4	i	105 010	69 0A	ADC#&0A
A2C6	H	072	48	PHA
A2C7		138	8A	TXA
A2C8		240 003	F0 03	BEQ 3 --> &A2CD
A2CA		032 206 162	20 CE A2	JSR &A2CE
A2CD	h	104	68	PLA
A2CE	0	009 048	09 30	ORA#&30

Add ASCII character to existing output string (SWA)

A2D0		218	DA	PHX
A2D1	6	166 054	A6 36	LDX &36
A2D3		157 000 006	9D 00 06	STA &0600,X
A2D6		250	FA	PLX
A2D7	6	230 054	E6 36	INC &36
A2D9	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A5BE Raise the FWA to the Power of the Integer Value in A

Submitted by Steve Fewell

Description:

If the Integer power (A) is negative then decrement if and EOR it with #&FF to reverse the two's complement and obtain the absolute value of the power. Then call the Floating-Point Reciple routine (&A5E9) to set FWA = 1/FWA. Now the FWA can be raised to the positive power value to give the required result.

If the Integer power is 0, then set FWA = 1.0 and exit.

Store the FWA to Temporary Floating-Point variable location &046C.
If the power = 1 then exit (the FWA already contains the result, as raising to the power of 1 just returns the same value).

Reduce the power value by 1.
Keep multiplying the FWA by the original copy of the FWA (&046C) (FWA = &046C * FWA) and reducing the power value until the power value is zero.
Exit, as the FWA now contains the correct result of FWA^A.

Note: It's strange that there is no quicker way to perform this calculation!

Disassembly for the Raise FWA to Power of 'A' value routine

A5BE		170	AA	TAX
A5BF		016 008	10 08	BPL 8 --> &A5C9
A5C1	:	058	3A	DEC A
A5C2	I	073 255	49 FF	EOR#&FF
A5C4	H	072	48	PHA
A5C5		032 233 165	20 E9 A5	JSR &A5E9 Floating-Point Reciple (FWA=1/FWA)
A5C8		250	FA	PLX
A5C9		240 013	F0 0D	BEQ 13 --> &A5D8 Set FWA = 1.0
A5CB		032 017 165	20 11 A5	JSR &A511 Store FWA to &046C & set argp=&046C
A5CE		202	CA	DEX

A5CF		240 006	F0 06	BEQ 6 --> &A5D7
A5D1		032 166 166	20 A6 A6	JSR &A6A6 Floating-Point multiplication (FWA=argp*FWA)
A5D4		202	CA	DEX
A5D5		208 250	D0 FA	BNE -6 --> &A5D1
A5D7	`	096	60	RTS
A5D8		169 128	A9 80	LDA#&80
A5DA	1	133 049	85 31	STA &31
A5DC		026	1A	INC A
A5DD	0	133 048	85 30	STA &30
A5DF	L	076 184 166	4C B8 A6	JMP &A6B8 Clear FWA (except Exp/Mantissa1)

BASIC IV ROM Routines

Page Last Altered: undefined

A5E9 Floating-Point Reciple (FWA=1/FWA)

Submitted by Steve Fewell

Routine:areciple
Name: Floating-Point Reciple
Starting Address: &A5E9
Entry criteria: The FWA Contains a Floating-Point value.
Exit: The FWA contains the value of FWA = 1 / FWA.

Description:

Set argp to point to the Floating-Point Constant at &BF92 which contains the value 1.0.
This value is stored in the Floating-Point Constant table.
Jump to the Floating-Point divide routine to set the FWA to: FWA = argp / FWA.
The FWA will now contain the reciple value of the previous value in the FWA.

Disassembly for the Floating-Point Reciple routine

A5E9	169 146	A9 92	LDA#&92
A5EB	032 139 165	20 8B A5	JSR &A58B
A5EE			...Floating-Point Divide

Disassembly for the A58B Set argp to &BF00 + A

A589	.	169 046	A9 2E	LDA#&2E
A58B	J	133 074	85 4A	STA &4A
A58D		169 191	A9 BF	LDA#&BF

A58F	K	133 075	85 4B	STA &4B
A591	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

BF24 Table of Floating Point Constants

Submitted by Steve Fewell

Description:

This is a start of the table of Mathematical Constants that BASIC uses.
Each Floating Point Constant is stored as a 5-byte floating point variable.
That is where the 1st Byte = Exponent and the next 4 bytes = the Mantissa value (MSB first).

It's contents are as follows:

Start Address	Exponent	Mantissa 1	Mantissa 2	Mantissa 3	Mantissa 4	Value	Description
BF24	81	C9	10	00	00	-1.57080078	
BF29	6F	15	77	7A	61	0.00000445445511	
BF2E	81	49	0F	DA	A2	1.57079633	Pi/2
BF33	80	22	F9	83	6E	0.636619772	2/Pi
BF38	7B	0E	FA	35	12	0.0174532925	Degree (= Pi / 180)
BF3D	86	65	2E	E0	D3	57.2957795	Radan (= 1 / (Pi / 180) [i.e.: 1 / Degree])
BF42	7F	5E	5B	D8	AA	0.434294482	1/Log(10)
BF47	82	2D	F8	54	58	2.71828183	exp(1) [or just e]
BF4C	80	31	72	17	F8	0.693147181	Log(2)
BF51	80	0B	D7	50	29	0.546254168	LN continued-fraction expanion series - adder 1
BF56	7C	D2	7C	86	05	-0.0513882861	LN continued-fraction expanion series - divisor 1
BF5B	80	15	52	B6	36	0.583293331	LN continued-fraction expanion series - adder 2
BF60	7C	99	98	36	04	-0.0374986753	LN continued-fraction expanion series - divisor 2

BF65	80	40	00	01	10	0.750000063	LN continued-fraction expanion series - adder 3
BF6A	7F	2A	AA	AA	E3	0.33333334	1/3 -> LN continued-fraction expanion series - divisor 3
BF6F	7F	FF	FF	FF	FF	-0.5	-1/2 -> LN continued-fraction expanion series - adder 4
BF74	7A	C3	1E	18	BE	-0.0119090311	SIN/COS continued-fraction expanion series - adder 1
BF79	73	61	71	55	2D	0.000107499459	SIN/COS continued-fraction expanion series - divisor 1
BF7E	7B	8C	9B	91	88	-0.0171640246	SIN/COS continued-fraction expanion series - adder 2
BF83	77	2B	A4	C4	53	0.0013095369	SIN/COS continued-fraction expanion series - divisor 2
BF88	7C	4C	CC	CA	B7	0.0499999922	SIN/COS continued-fraction expanion series - adder 3
BF8D	7E	AA	AA	AA	A6	-0.166666666	-1/6 -> SIN/COS continued-fraction expanion series -divisor 3
BF92	81	00	00	00	00	1	SIN/COS continued-fraction expanion series - adder 4
BF97	7D	A3	F2	EF	44	-0.0800532048	ATN continued-fraction expanion series - adder 1
BF9C	7E	1F	01	A1	4D	0.155279656	ATN continued-fraction expanion series - divisor 1
BFA1	7F	61	6D	F4	3F	0.440292008	ATN continued-fraction expanion series - adder 2
BFA6	7E	5C	91	23	AC	0.215397413	ATN continued-fraction expanion series - divisor 2
BFAB	7E	76	B8	8D	1A	0.240938382	ATN continued-fraction expanion series - adder 3
BFB0	7D	1D	3E	AB	2C	0.0767796872	ATN continued-fraction expanion series - divisor 3
BFB5	81	09	41	81	D2	1.07231162	ATN continued-fraction expanion series - adder 4
BFBA	80	74	DF	BD	20	0.956538983	ATN continued-fraction expanion series - divisor 4
BFBF	80	83	8B	1F	B5	-0.513841612	ATN continued-fraction expanion series - adder 5
BFC4	7F	82	59	AD	AB	-0.254590442	ATN continued-fraction expanion series - divisor 5
BFC9	80	6D	63	38	2C	0.927295218	2*arctan(1/2) [or arccos(sqrt(9/25))] -> ATN continued-fraction expanion series - adder 6
BFCE	7D	11	D4	B1	D1	0.071206464	EXP continued-fraction expanion series - adder 1
BFD3	79	68	BC	4F	59	0.00710252642	EXP continued-fraction expanion series - divisor 1
BFD8	75	05	2C	9E	39	0.000254009799	EXP continued-fraction expanion series - adder 2
BFDD	7B	08	88	3B	A6	0.0166665235	EXP continued-fraction expanion series - divisor 2
BFE2	6C	31	CF	D1	8C	0.000000662400541	EXP continued-fraction expanion series - adder 3
BFE7	7D	2A	AA	AA	89	0.0833333324	EXP continued-fraction expanion series - divisor 3
BFEC	7F	FF	FF	FF	E8	-0.499999997	EXP continued-fraction expanion series - adder 4
BFF1	81	00	00	00	00	1	EXP continued-fraction expanion series - divisor 4
BFF6	81	00	00	00	00	1	EXP continued-fraction expanion series - adder 5
Other Constants:							
----	81	80	00	00	00	-1	Calc in LN (A746)

BASIC IV ROM Routines

Page Last Altered: undefined

A861 Evaluate continued-fraction expansions Series

Submitted by Steve Fewell

Routine:EvaluateSeries
Name: Evaluate continued-fraction expansions Series
Starting Address: &A861
Entry criteria: The FWA contains the value to apply the Number Series to.
Y contains the number of cycles to evaluate.
X contains the LSB address of the first Floating-Point constant (in the FP constant table: &BF24) to use in the series calculation. This value and subsequent values will be used.
A contains the LSB address of the Floating-Point constant value to set the FWA to if the number in the FWA is too small to evaluate.
Exit: The FWA contains the result of the calculation.

Description:

Set &47 to the number of cycles that we need to evaluate (Y).
Set &4C to the LSB address of the first Floating-Point constant to use (X).
All Floating-Point constant values used are located in the Floating-Point constant table.

If the exponent of the FWA is less than &40, then the value is too small to evaluate, so set the FWA to the default value specified (The Floating-Point constant located at &BF00 + A) and exit. Note: This will also trap a 0.0 value, and thus avoid a Division by zero error when the reciple value is calculated.

Obtain the reciple of the FWA value (FWA = 1/FWA).
Store the reciple value in the temporary Floating-Point variable located at &046C.

Set argp to the address of the first Floating-Point constant in the series calculation: that is &BF00 + the LSB value in &4C.
Add the first Floating-Point constant value to the Reciple value in the FWA (FWA=argp+FWA).

For each cycle do the following: [i.e. if the cycle (&47) is 2 then repeat twice]
 * Update argp to point to the address of the next Floating-Point constant (i.e. add 5 to the LSB address and update &4C to the new LSB address. The MSB address is always #&BF). Divide this Floating-Point constant value by the FWA (FWA = argp/FWA).
 * Update argp to point to the address of the next Floating-Point constant (i.e. add 5 to the LSB address and update &4C to the new LSB address. The MSB address is always #&BF). Add this Floating-Point constant value to the FWA (FWA = argp+FWA).
 * Add the Temporary Floating-Point variable at &046C (the reciple of the original value) to the FWA (FWA = argp + FWA).
 * Decrement the cycle number (&47).

Lastly perform a division and addition calculation with the next two floating-point constants, as follows:
Update argp to point to the address of the next Floating-Point constant (i.e. add 5 to the LSB address and update &4C to the new LSB address. The MSB address is always #&BF).
Divide this Floating-Point constant value by the FWA (FWA = argp/FWA).
Update argp to point to the address of the next Floating-Point constant (i.e. add 5 to the LSB address and update &4C to the new LSB address. The MSB address is always #&BF).
Add this Floating-Point constant value to the FWA (FWA = argp+FWA).

The FWA now contains the required result.

Disassembly for the Evaluate continued-fraction expansions series routine

A861	G	132 071	84 47	STY &47
A863	L	134 076	86 4C	STX &4C
A865	0	166 048	A6 30	LDX &30
A867	@	224 064	E0 40	CPX#&40
A869	+	144 043	90 2B	BCC 43 --> &A896 Set FWA = Default value
A86B		032 233 165	20 E9 A5	JSR &A5E9 Floating-Point Reciple (FWA=1/FWA)
A86E		032 017 165	20 11 A5	JSR &A511 Store FWA to Temp address &046C
A871	L	165 076	A5 4C	LDA &4C
A873		032 139 165	20 8B A5	JSR &A58B Set argp to &BF00 + A
A876		032 141 166	20 8D A6	JSR &A68D Floating-Point Addition
A879		032 134 168	20 86 A8	JSR &A886
A87C		032 146 165	20 92 A5	JSR &A592 Set argp to &046C
A87F		032 141 166	20 8D A6	JSR &A68D Floating-Point Addition
A882	G	198 071	C6 47	DEC &47
A884		208 243	D0 F3	BNE -13 --> &A879

A886		169 191	A9 BF	LDA#&BF
A888	K	133 075	85 4B	STA &4B
A88A		032 127 165	20 7F A5	JSR &A57F Set &4C to address of next FP constant
A88D		032 238 165	20 EE A5	JSR &A5EE Floating-Point Division
A890		032 127 165	20 7F A5	JSR &A57F Set &4C to address of next FP constant
A893	L	076 141 166	4C 8D A6	JMP &A68D Floating-Point Addition
A896		032 139 165	20 8B A5	JSR &A58B Set argp to &BF00 + A
A899	LA	076 065 165	4C 41 A5	JMP &A541 Load FWA with Floating-Point variable

A57F - Update &4C to point to the LSB address of the next FP variable

A57F		024	18	CLC
A580	L	165 076	A5 4C	LDA &4C
A582	i	105 005	69 05	ADC#&05
A584	L	133 076	85 4C	STA &4C
A586	J	133 074	85 4A	STA &4A
A588	`	096	60	RTS

A592 - Set argp to &046C

A592	l	169 108	A9 6C	LDA#&6C
A594	J	133 074	85 4A	STA &4A
A596		169 004	A9 04	LDA#&04
A598	K	133 075	85 4B	STA &4B
A59A	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9267 (PRINT) quote (')

Submitted by Steve Fewell

Description:

This routine is called when a quote character (') is found within a PRINT statement (e.g. "PRINT '").
Call &BA92 to output a new line (OSNEWLINE, &FFE7) and set COUNT (&1E) to zero.
Clear the carry flag, update BASIC Text pointer A to the value of BASIC Text pointer B and exit.

Disassembly for the (PRINT) quote (') routine

9267	032 146 186	20 92 BA	JSR &BA92 Output a new line
926A	024	18	CLC
926B	128 008	80 08	BRA 8 --> &9275 PTR A offset = PTR B offset and exit

---

BASIC IV ROM Routines

Page Last Altered: undefined

9241 (PRINT) TAB(

Submitted by Steve Fewell

Description:

Get the Integer result of the expression after the 'TAB(' keyword.
If the result is a Float then convert it to an Integer, if the result is a string then generate 'Type mismatch' error.

Get the next non-space character, if it is a "," then the TAB function call has two arguments so jump to &922D to do the following:

* Push the first argument (&2A) to the Stack.
* Get the Integer result of the expression after the ','. If the result is a Floating-Point value then convert to an Integer, if the result is a string then generate 'Type mismatch' error.
* If the next non-space character is not ')' then generate 'Missing)' error.
* Send #&1F to the Operating System output character routine (&FFEE) to start a VDU 31 command.
* Pop the first argument from the stack and send it to the Operating System output character routine (&FFEE).
* Call routine &9840 to send &2A (the second argument) to the Operating System output character vector (&020E).
* Now, the cursor should be in the required position (as VDU 31 command with 2 arguments moves the cursor to the specified location).
* Jump to &926A: (Clear carry, Set PTR A Offset to PTR B Offset & exit (back to the PRINT routine).

Otherwise, the TAB function has only one argument so continue with &9249 and do the following:

* Check whether next character is a ')', if it isn't then generate 'Missing ,' error.
* Subtract COUNT (&1E) from the 1-byte Integer result (&2A).
* If the subtraction result is zero then jump to &926A: (Clear carry, Set &0A (PTR A Offset) to &1B (PTR B Offset) and exit (back to the PRINT routine).
* If the subtraction result was positive (carry flag still set) then set X to the subtraction result and jump to &9262 to output X number of spaces (and then Clear carry, Set &0A (PTR A Offset) to &1B (PTR B Offset), and exit (back to the PRINT routine).
* If the subtraction result was negative (carry flag is clear) then call &BA92 to output a new line and set COUNT (&1E) to zero and jump to &925E to Set X to &2A and output X number of spaces (and then Clear carry, Set &0A (PTR A Offset) to &1B (PTR B Offset), and exit (back to the PRINT routine).

Disassembly for the (PRINT) TAB(routine

```
922A  L   076 246 142   4C F6 8E   JMP &8EF6 'Missing ,' error
```

922D	*	165 042	A5 2A	LDA &2A
922F	H	072	48	PHA
9230		032 167 150	20 A7 96	JSR &96A7 Extract Integer result of expression and check for closing bracket
9233		169 031	A9 1F	LDA#&1F
9235		032 238 255	20 EE FF	JSR &FFEE OSWRCH (Write character)
9238	h	104	68	PLA
9239		032 238 255	20 EE FF	JSR &FFEE OSWRCH (Write character)
923C	@	032 064 152	20 40 98	JSR &9840 Send ?&2A to the OS output vector
923F	)	128 041	80 29	BRA 41 --> &926A Clear carry flag, Set PTR A offset = PTR B offset and exit
9241		032 175 150	20 AF 96	JSR &96AF Get Integer result of expression
9244		032 235 142	20 EB 8E	JSR &8EEB Get next non-space character (PTR B) and compare with ','
9247		240 228	F0 E4	BEQ -28 --> &922D
9249	)	201 041	C9 29	CMP#&29 ')'
924B		208 221	D0 DD	BNE -35 --> &922A
924D	*	165 042	A5 2A	LDA &2A
924F		229 030	E5 1E	SBC &1E (COUNT)
9251		240 023	F0 17	BEQ 23 --> &926A Clear carry flag, Set PTR A offset = PTR B offset and exit
9253		170	AA	TAX
9254		176 012	B0 0C	BCS 12 --> &9262 Output X number of spaces, clr carry, PTR A=PTR B & exit
9256		032 146 186	20 92 BA	JSR &BA92 Start new output line
9259		128 003	80 03	BRA 3 --> &925E Output ?&2A number of spaces, clr carry, PTR A=PTR B & exit

9840 Send character in ?&2A to the Output vector

9840	*	165 042	A5 2A	LDA &2A
9842	l	108 014 002	6C 0E 02	JMP (&020E) OS Output character vector (WRCHV)

BASIC IV ROM Routines

Page Last Altered: undefined

96A7 Extract Integer result of expression and check for closing bracket

Submitted by Steve Fewell

Description:

This routine calls &ADAC to evaluate the expression contained at the current text pointer location and check the next character found after the expression (returned in X). If this character is not a closing bracket ')', then &ADAC generates a "Missing)" error.

exit via routine &96BF (which checks the Return Value Type - converting a Float value to an Integer and generating Type Mismatch (if the result is a String)).

Disassembly for the Extract Integer result of expression and check for Closing bracket routine

96A7	032 172 173	20 AC AD	JSR &ADAC Get result of expression and check for closing bracket
96AA	128 019	80 13	BRA 19 --> Check if Integer (Convert if Float or error if String)

BASIC IV ROM Routines

Page Last Altered: undefined

ADAC Evaluate expression and check for closing bracket

Submitted by Steve Fewell

Description:

This routine calls &9D3B to evaluate the expression contained at the current text pointer location.
This will either return a String in the SWA, an Integer in the IWA or a Floating-Point value in the FWA.
Next, it will check the next character found after the expression (returned in X).
If this character was not a closing bracket ')', then generate a "Missing)" error.
Otherwise, all is correct, so increment the text pointer (&1B),
set Y = Return Value Type (A) and exit the routine.

Disassembly for the Evaluate expression and check for Closing bracket routine

ADAC	;	032 059 157	20 3B 9D	JSR &9D3B Evaluate expression
ADAF		230 027	E6 1B	INC &1B
ADB1	)	224 041	E0 29	CPX#&29
ADB3		208 233	D0 E9	BNE -23 --> &AD9E Missing) error
ADB5		168	A8	TAY
ADB6	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

96AF Get Integer result of expression

Submitted by Steve Fewell

Description:

[If called from &96AC, this routine will check for a comma ',' (issuing a "Missing ," error if one is not found), before continuing with the rest of this routine].

This routine calls the Expression Handler routine (&9D3B)
to obtain the value of the expression on the command line.
If the value is a String, then a Type Mismatch error will be generated.
If this value is a Float then it will be converted to an Integer.
Exit with the Integer value.

Disassembly for the Get Integer result of expression

96AC	032 241 142	20 F1 8E	JSR &8EF1 Check for ',' and issue error if not found
96AF	; 032 059 157	20 3B 9D	JSR &9D3B Expression Handler
96B2	128 011	80 0B	BRA 11 --> &96BF Check if Integer & convert if float or error if String

Disassembly for the Check ',' and issue error if not found routine

8EF1	032 235 142	20 EB 8E	JSR &8EEB Get next non-space char (PTR B) & compare with ','
8EF4	240 244	F0 F4	BEQ -12 --> &8EEA [RTS]
8EF6			... 'Missing ,' error

BASIC IV ROM Routines

Page Last Altered: undefined

925B (PRINT) SPC

Submitted by Steve Fewell

Description:

Call &96B4 to get the Integer value after the SPC keyword.
If &2A is zero (number of spaces to output is zero) then Clear carry, Set &0A (PTR A Offset) to &1B (PTR B Offset) and exit (back to the PRINT routine).
Load X with &2A (number of spaces to output) and call routine &BDBF to output X number of spaces.
Clear carry, Set &0A (PTR A Offset) to &1B (PTR B Offset) and exit (back to the PRINT routine).

Disassembly for the (PRINT) SPC routine

925B		032 180 150	20 B4 96	JSR &96B4 Get Integer value at PTR B (error if String)
925E	*	166 042	A6 2A	LDX &2A
9260		240 008	F0 08	BEQ 8 --> &926A Clear carry, set PTR A offset = PTR B offset and exit
9262		032 191 189	20 BF BD	JSR &BDBF Output X number of Spaces
9265		128 003	80 03	BRA 3 --> &926A Clear carry, set PTR A = PTR B and exit

---

BASIC IV ROM Routines

Page Last Altered: undefined

BAEB Tokenise Command Line and Insert Line into Program

Submitted by Steve Fewell

Routine:Tokenise
Name: Tokenise Command Line (and insert into program if line number found)
Starting Address: &BAEB
Entry criteria: &0700-&07EF (the command line input location) contains the program text to tokenise
Exit: C = 0 if the command is a direct command and not part of a program - this inputt is requried to be executed immediately.
C = 1 if a program line was entered (and inserted into the program) - not required to be executed immediately.

Description:

Set &28 (OPT flag) to #&FF.
Set &3C to #&FF (this tells the &8DB2 routine to Check for a Line Number at the start of the line).
Call routine &BBCF to initialise the Program start address (&1C, &1D), Stack pointer & REPEAT/FOR/GOSUB levels and also reset the top bit of the LISTO flag (#&1F, to cancel *EDIT mode).
Note: variables are NOT reset by this initialisation.

Set pointer (&37, &38) to point to BASIC Text Pointer A (i.e. the Command Line input text).
Zero &3B (to initialise routine &8DB2 to be in 'Start of Statement' mode).
Zero BASIC Text pointer A offset (as we will start processing from the beginning of the command line text).

Call routine &8DB2 to tokenise the command line text pointed to by (&37, &38).

Call routine &9B1E to set the IWA to the Line Number (which is now tokenised) at the location pointed to by BASIC Text Pointer A (i.e. the start of the command line input), and set Y to point to the next character after the Line Number.

If the carry flag is not set then a Line Number was not present, so exit as we do not need to alter the current BASIC program at all (no lines to insert, change or delete).

[&BB08] If the LISTO flag (&1F) is not 0, then skip any multiple leading spaces after the line number - meaning that a maximum of 1 space character will be kept after the line number. This enables helps LIST command with its formatting. Additionally, if no spaces are specified, then none will be stored.

Set location &3B to Y - that is the offset (from &0700) to the start of the program line (after the Line Number), with any extra leading spaces ignored (if the LISTO flag is not 0).

Call routine &BA98 to remove the line number (specified in the IWA) from the program (if it is found).
Set location &3C to 7 and Y to 0.
Now pointer (&3B, &3C) points to the first character (after the Line Number) of the tokenised program line, and Y contains the offset (from this pointer) of the current character that we are working with.

If the start of the program line (pointed to by (&3B, &3C)) contains the '<cr>' character then exit as there is no line to insert - and we have deleted the specified line from the program (as per the user's request).

[&BB26] Search for the next '<cr>' character and set Y to the length of the program line (excluding the Line Number and the '<cr>' character.
Reduce Y (if necessary) so that it points to the last non-space character on the line. This is done in order to skip any trailing spaces on the command line.

Store a '<cr>' character at location (&37, &38 + offset Y+1) [i.e. at the end of the program line].
Add 4 to Y (to take into account the 4-byte tokenised Line Number [1 byte for Line Number token and 3 for the Line Number value]).
Store Y (the line length) in location &3F.

Set &39-&3A to TOP.
Update TOP (by adding Y tm the TOP value via routine &BE04).
Set (&37, &38) to the new TOP value.

Set Y to 0 (Y is 1 on return from &BE04).
If HIMEM (&06, &07) is less than the new TOP value then there is not enough space for the program line to be stored, so:

* Call &BBAC to check that the program can be read correctly (and to set the correct TOP value).
  [at this stage the new/changed Line will not be present in the program at all].
* generate a 'LINE space' error.

[&BB6B] Copy bytes from the old TOP location (&39, &3A) to the new TOP location (&37, &38), starting with the last character in the program as follows:

* Copy character from (&39, &3A) + Y to location (&37, &38) + Y
* If Y is 0 then decrement old TOP MSB byte (&3A) and new TOP MSB byte (&38).

* [&BB76] decrement Y
* Set A (LSB byte) and X (MSB byte) to the old TOP value plus Y.
* [&BB7F] if (&3D, &3E) [which is a pointer to the program line that is greater than or equal to the line that we are inserting, as set by routine &BA98] is less than the old TOP value + Y (stored in: A (LSB byte), X (MSB byte)) then jump back to &BB6B to copy the next byte from the old TOP location to the new TOP location.

Otherwise, it must be equal to the location of the program line after the one we want to insert (as this program line has just been copied to the new TOP location, so now we have space for the new program line to be inserted.

Store the program line number MSB (from the IWA byte &2B) in the first byte of the reserved program space pointed to by (&3D-&3E).
Store the program line number LSB (from the IWA byte &2A) in the second byte of the reserved program space pointed to by (&3D-&3E).
Store the line length (from location &3F) in the third byte of the reserved program space pointed to by (&3D-&3E).

Add 4 (Y + 1) to the (&3D-&3E) pointer.
Copy the new BASIC program (from location (&3B, &3C), i.e. the command line) to the new location (&3D, &3E).
(The program line is terminated by the '<cr>' character).
Exit with Carry flag set.
Note: the lines before the inserted line do not need to be moved, only the lines between the inserted line and the old TOP location.

Disassembly for the 'Tokenise Command Line and Insert Line into Program' routine

BAEB		162 255	A2 FF	LDX#&FF
BAED	(	134 040	86 28	STX &28
BAEF	<	134 060	86 3C	STX &3C
BAF1		032 207 187	20 CF BB	JSR &BBCF Initialise Program start address, Stack pointer, *EDIT mode & REPEAT/FOR/GOSUB levels
BAF4		165 011	A5 0B	LDA &0B
BAF6	7	133 055	85 37	STA &37
BAF8		165 012	A5 0C	LDA &0C
BAFA	8	133 056	85 38	STA &38
BAFC	d;	100 059	64 3B	STZ &3B
BAFE	d	100 010	64 0A	STZ &0A
BB00		032 178 141	20 B2 8D	JSR &8DB2 Tokenise Command Line
BB03		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
BB06		144 226	90 E2	BCC -30 --> &BAEA [RTS]

```
BB08       165 031       A5 1F      LDA &1F
BB0A       240 009       F0 09      BEQ 9 --> &BB15
BB0C       185 000 007   B9 00 07   LDA &0700,Y
BB0F       200           C8         INY
BB10       201 032       C9 20      CMP#&20
BB12       240 248       F0 F8      BEQ -8 --> &BB0C
BB14       136           88         DEY
BB15   ;   132 059       84 3B      STY &3B
BB17       032 152 186   20 98 BA   JSR &BA98 Remove Line Number (specified in IWA) from Program
BB1A       160 007       A0 07      LDY#&07
BB1C   <   132 060       84 3C      STY &3C
BB1E       160 000       A0 00      LDY#&00
BB20       169 013       A9 0D      LDA#&0D
BB22   ;   210 059       D2 3B      CMP (&3B)
BB24       240 196       F0 C4      BEQ -60 --> &BAEA [RTS]
BB26       200           C8         INY
BB27   ;   209 059       D1 3B      CMP (&3B),Y
BB29       208 251       D0 FB      BNE -5 --> &BB26
BB2B       169 032       A9 20      LDA#&20
BB2D       136           88         DEY
BB2E       240 004       F0 04      BEQ 4 --> &BB34
BB30   ;   209 059       D1 3B      CMP (&3B),Y
BB32       240 249       F0 F9      BEQ -7 --> &BB2D
BB34       200           C8         INY
BB35       169 013       A9 0D      LDA#&0D
BB37   ;   145 059       91 3B      STA (&3B),Y
BB39       200           C8         INY
BB3A       200           C8         INY
BB3B       200           C8         INY
BB3C       200           C8         INY
BB3D   ?   132 063       84 3F      STY &3F
BB3F       165 018       A5 12      LDA &12
BB41   9   133 057       85 39      STA &39
BB43       165 019       A5 13      LDA &13
```

```
BB45  :  133 058      85 3A     STA &3A
BB47     032 004 190  20 04 BE  JSR &BE04 Update TOP (add Y to TOP value, set Y to 1 & exit)
BB4A  7  133 055      85 37     STA &37
BB4C     165 019      A5 13     LDA &13
BB4E  8  133 056      85 38     STA &38
BB50     136          88        DEY
BB51     165 006      A5 06     LDA &06
BB53     197 018      C5 12     CMP &12
BB55     165 007      A5 07     LDA &07
BB57     229 019      E5 13     SBC &13
BB59     176 016      B0 10     BCS 16 --> &BB6B
BB5B     032 229 189  20 E5 BD  JSR &BDE5 Check program can be read correctly ('Bad program' message if not)
BB5E     032 172 187  20 AC BB  JSR &BBAC Initialise Page 7 & reset Variable pointers, etc...
BB61                            ... 'LINE space' error ...
BB6B  9  177 057      B1 39     LDA (&39),Y
BB6D  7  145 055      91 37     STA (&37),Y
BB6F     152          98        TYA
BB70     208 004      D0 04     BNE 4 --> &BB76
BB72  :  198 058      C6 3A     DEC &3A
BB74  8  198 056      C6 38     DEC &38
BB76     136          88        DEY
BB77     152          98        TYA
BB78  e9 101 057      65 39     ADC &39
BB7A  :  166 058      A6 3A     LDX &3A
BB7C     144 001      90 01     BCC 1 --> &BB7F
BB7E     232          E8        INX
BB7F  =  197 061      C5 3D     CMP &3D
BB81     138          8A        TXA
BB82  >  229 062      E5 3E     SBC &3E
BB84     176 229      B0 E5     BCS -27 --> &BB6B
BB86     160 001      A0 01     LDY#&01
BB88  +  165 043      A5 2B     LDA &2B
BB8A  =  145 061      91 3D     STA (&3D),Y
```

```
BB8C      200         C8        INY
BB8D  *   165 042     A5 2A     LDA &2A
BB8F  =   145 061     91 3D     STA (&3D),Y
BB91      200         C8        INY
BB92  ?   165 063     A5 3F     LDA &3F
BB94  =   145 061     91 3D     STA (&3D),Y
BB96  8   056         38        SEC
BB97      152         98        TYA
BB98  e=  101 061     65 3D     ADC &3D
BB9A  =   133 061     85 3D     STA &3D
BB9C      144 002     90 02     BCC 2 --> &BBA0
BB9E  >   230 062     E6 3E     INC &3E
BBA0      160 255     A0 FF     LDY#&FF
BBA2      200         C8        INY
BBA3  ;   177 059     B1 3B     LDA (&3B),Y
BBA5  =   145 061     91 3D     STA (&3D),Y
BBA7      201 013     C9 0D     CMP#&0D
BBA9      208 247     D0 F7     BNE -9 --> &BBA2
BBAB  `   096         60        RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

8DB2 Tokenise Command Line

Submitted by Steve Fewell

Description:

Variables:

&3B = Zero if at the start of a Statement or #&FF if in the middle of a Statement.
&3C = #&FF is a Line Number is expected at the current location.
[On entry &3B is set to 0 (start of statement) and &3C is set to #&FF (Line Number expected).
as soon as a non-line number character is found, location &3C is set to 0, as a Line Number is no longer expected].

During the tokenisation process, the characters on the command line are checked in a specific order in order to ensure that the Line is tokenised correctly.

Each character is checked against each character, and tokenised when the character is found, in the order specified by this list:

<cr> (carriage return) - End of line
<sp> (space) - separator
& - Hex Number literal
" - String literal
: - Statement separator
, - Comma (field separator)
* - Operating System call
* - Multiplication Operator
'.' (decimal point) - Numeric literal
Digit - Line Number
Digit or '.' (decimal point) - Numeric literal
Characters less then 'A' - Other symbol characters (skipped)
Characters greater then of equal to "X" - non Keyword start characters (skipped)
Characters between "A" and "W" - Possible Keywords

The following is a description of the routine in more detail.
[&8DAF] Increment the (&37, &38) pointer.
[&8DB2] Get the next character pointed to by (&37, &38).
If the character is '<cr>' [carriage-return] then the end of the line has been reached, so exit as tokenisation is complete.

If the character is a space then skip all spaces by jumping back to &8DAF until we find a non-space character.

If the character is '&', then the Hex number is skipped (as is doesn't need to be tokenised) as follows:
* Keep Incrementing and reading the next character from (&37, &38) until the next character is not '0'-'9' or 'A' to 'F' (i.e. a Hex digit).

* If the first character after the Hex number is less than 'A' then jump back to &8DB2 to check whether this character (to see if it is '<cr>' or '&'.
* If the first character after the Hex number is greater than 'F' then continue to &8DD0 (as it cannot be ' or '&'!.

[&8DD0] If the character is a quote ("), then check the next character. If the next character is also a quote then the string is complete, so jump to &8DAF to continue with the next character after the "" (this method also works when quote characters are doubled-up within a String literal, as we are only skipping the string literal - and not bothered about obtaining its actual value).
Skip any non quote (") characters in the string literal but if a <cr> (carriage return) is found then exit as the line has been tokenised. When a quote character is found, jump back to &8DAF to tokenise the rest of the line.

[&8DE0] (the character is not a quote). If the character is ':' then increment the pointer to the next character (in order to skip the ':' character), zero location &3B to indicate that we are at the start of a statement, zero location &3C to indicate that a Line Number is no longer expected & jump to &8DB2 to check the next character.

[&8DED] If the character is a comma then jump to &8DAF (skip comma as it doesn't need to be tokenised).

[&8DF1] If the character is '*' (star), then check location &3B. If location &3B is 0 then we are at the start of the statement, so the '*' is an Operating System call (not a multiplication operator). As the whole command line after the '*' will be passed to the Operating System on execution, the rest of the line should not be tokenised - so exit from the tokenisation routine.
Otherwise, the '*' is a multiplication operator, so store #&FF in location &3B (to indicate that we are no longer at the start of a statement - note: this should already be the case anyway), zero location &3C as we are not expecting a line number, and then jump to &8DAF to check the next character (no further tokenisation of the '*' required).

[&8E01] If the character is a '.' [Decimal point] then a numeric literal has been found, so [&8E13] keep checking the character pointed to by (&37, &38). If the character at (&37, &38) is "." or a digit('0' to '9') then increment the (&37, &38) pointer and jump back to &8E13 to check the next character. This ensures that the numeric literal is ignored, as it does not need to be tokenised.
When a non-digit (and non-'.') character is found, Store #&FF in location &3B (meaning that we are not at the start of a statement), Zero &3C (as a line number is not expected) and jump back to &8DB2 to check the new character for tokenisation.

[&8E05] If the character is a digit ['0' to '9'] and location &3C is not 0 (i.e. &FF) then a line number is expected and we have found the start of a line number, so call routine &8D04 to tokenise the line number.
If carry is set on return from &8D04, then there was a error and the numeric value was not a valid Line number, so continue to &8E13 to treat the value as a numeric literal instead of a line number. Otherwise, if carry is clear, the Line Number was tokenised correctly, so jump back to &8DAF to check the next character on the command line.

[&8E05] If the character is a digit ['0' to '9'] and location &3C is 0 then we have found a numeric literal (it cannot be a line number as a line number is not expected at this time), so [&8E13] keep checking the character pointed to by (&37, &38). If the character at (&37, &38) is "." or a digit('0' to '9') then increment the (&37, &38) pointer and jump back to &8E13 to check the next character. This ensures that the numeric literal is ignored, as it does not need to be tokenised.
When a non-digit (and non-'.') character is found, Store #&FF in location &3B (meaning that we are not at the start of a statement), Zero &3C (as a line number is not expected) and jump back to &8DB2 to check the new character for tokenisation.

[&8E36] If the character is less than 'A' then the character does not need to be tokenised, so set location &3B to #&FF (to indicate that we are not at the start of a statement), zero location &3C (to indicate that a Line Number is not expected at this point (as we are no longer at the start of the line) and jump to &8DAF to check the next character on the command line.

[&8E3A] If the character is greater than or equal to 'X' then it cannot be the start of a Keyword (as no BASIC keywords begin with 'X', 'Y' or 'Z') so jump to &8E25 to check for a variable name.

[&8E3E] The character is between 'A' and 'W', so check whether it is the start of a BASIC Keyword, as follows:
* Set pointer (&39, &3A) to point to &8456, which is the address of the beginning of the BASIC Keyword table within the BASIC rom.

* [&8E46] Compare the character (in A) with the first charater of the next BASIC Keyword at (&39, &3A).
* If the character is less than that of the start character of the next BASIC keyword then no tokenisation is required as the character belongs to a variable name and not a BASIC Keyword, so jump to &8E2A to skip the rest of the characters in the variable name, store #&FF in location &3B (meaning that we are not at the start of a statement), zero location &3C (as a line number is not expected) and jump back to &8DB2 to check the new character (the next character after the variable name) for tokenisation.
* If the character (in A) is not equal to the first character of the next Keyword then check the next character in the Keyword table the jump to &8E5D to advance to the next Keyword in the Keyword table.
* [&8E4E] Otherwise, increment the character index (Y - the position of the current character within the Keyword).
* If the next character is negative (>= #&80 - i.e. a Token value) then all the characters from the current word on the command line (pointed to by (&37, &38)) match with the Keyword, so goto &8E84 to tokenise the keyword. Otherwise, compare the next character of the Keyword with that of the next character on the Command Line. If the next character matches then jump to &8E4E to check the next character.
* If the characters do not match and the character is a dot '.', then jump to &8E68 to advance (&39,&3A) to the next token value and jump to &8E84 to tokenise (replace the Keyword on the command line with that token).
* [&8E5D] Advance (&39,&3A) to point to the token value (the byte that is >= #&80 and located directly after the keyword).
* If the token value is &FE (WIDTH, the last token in the BASIC Keyword table) then we have reached the end of the token table, so jump to &8E2A to skip the rest of the characters in the variable name, store #&FF in location &3B (meaning that we are not at the start of a statement), zero location &3C (as a line number is not expected) and jump back to &8DB2 to check the new character (the next character after the variable name) for tokenisation.
* Otherwise, jump to &8E75 to advance to the next Keyword in the Keyword table and then jump to &8E46 to compare the character with the next BASIC Keyword.

[&8E25] Check for a variable name:
Call routine &8D84 to check whether character is valid within a Variable name (letter, '_', or digit).
If the character is not a valid variable name character then the character does not need to be tokenised, so set location &3B to #&FF (to indicate that we are not at the start of a statement), zero location &3C (to indicate that a Line Number is not expected at this point (as we are no longer at the start of the line) and jump to &8DAF to check the next character on the command line.
Otherwise (valid variable name character found), so [&8E2A] keep checking the character pointed to by (&37, &38).
If the character at (&37, &38) is a valid variable name character ('A'-'Z', '_' or '0'-'9') then increment the (&37, &38) pointer and jump back to &8E2A to check the next character. This ensures that the entire variable name is ignored, as it does not need to be tokenised.
When a variable name character is found (we have reached the end of the variable name), Store #&FF in location &3B (meaning that we are not at the start of a statement), Zero &3C (as a line number is not expected) and jump back to &8DB2 to check the new character for tokenisation.

[&8E84] Tokenise the Keyword:
Set X to the value in A. Now, X = the token value of the Keyword that was matched against the text at BASIC Text Pointer A, and Y is the offset for the token value in the BASIC Keyword table (pointed to by (&39, &3A).
Store the flag for the BASIC Keyword (from the BASIC Keyword table - (&39, &3A) + Y + 1) in location &3D.
This flag specifies certain attributes of that particular Keyword.

If bit 0 of the flag (meaning 'Don't tokenise if Keyword is followed by an alphabetic character') is set then:
Load the next character from the command prompt location (&37, &38) call &8D84 to check whether the character is valid for a variable name (i.e. it's a digit, a letter or '_'). If it is a valid variable name character then do not tokenise the Keyword (as, in this context, it is not a Keyword, but a variable name) and jump to &8E2A to skip the rest of the characters in the variable name, store #&FF in location &3B (meaning that we are not at the start of a statement), zero location &3C (as a line number is not expected) and jump back to &8DB2 to check the new character (the next character after the variable name) for tokenisation.

[&8E95] Set A to the Token Value (in X).

If bit 6 of the &3D flag (meaning 'Pseudo Variable - where the keyword can be on either side of an assignment, i.e. PAGE= and =PAGE') is set then:

If location &3B is 0 (we are at the start of a statement, i.e. 'PAGE=') add #&40 to the token value, as Pseudo variable Keywords at the start of a statement are being assigned to, and so have a different token value.

[&8EA0] Decrement Y (to point to the last character of the Keyword).
Call routine &8CEB to replace the ASCII Keyword with the (1-byte) token value.

If bit 1 of the Keyword flag (meaning 'Go into middle of statement mode' - i.e. Keywords IF & LET) is set then:
Set location &3B to value #&FF and zero byte &3C. This sets the tokenise routine to middle of statement mode, and clears the 'Line Number expected' byte.

If bit 2 of the Keyword flag (meaning 'Go into Start of Statement mode' - i.e. Keywords THEN & FOR) is set then:
Clear location &3B (to tell the tokenise routine to go into start of statement mode) and clear location &3C (as a line number is no longer expected).

If bit 3 of the Keyword flag (meaning 'The Keyword is FN or PROC' - so don't tokenise the subroutine name) is set then:
Push A (Flag) to the stack. Skip any alphabetic characters (including digits and '_' characters) on the program line after the 'FN' or 'PROC' token and then, after the name has been skipped (a non-variable name character is found), retrieve A back (the keyword flag) from the stack again.

If bit 4 of the Keyword flag (meaning 'Tokenise a Line Number next' - i.e. Keywords GOTO, GOSUB, ELSE, THEN) is set then:
Set location &3C to #&FF (i.e. tell the tokenise routine to expect a Line Number next - however, if no line number is found next then this flag is ignored).

If bit 5 of the Keyword flag (meaning 'Don't tokenise the rest of the line' - i.e. Keywords REM and DATA) is set then:
exit from the tokenise line routine (so that the rest of the line is not tokenised) - and should instead be ignored as no more keywords are valid on this line after a REM or DATA keyword.

Disassembly for the Tokenise Command Line routine

8DAF		032 162 141	20 A2 8D	JSR &8DA2 Increment (&37, &38) pointer
8DB2	7	178 055	B2 37	LDA (&37)
8DB4		201 013	C9 0D	CMP#&0D
8DB6	'	240 039	F0 27	BEQ 39 --> &8DDF [RTS (exit when &0D char found)]
8DB8		201 032	C9 20	CMP#&20
8DBA		240 243	F0 F3	BEQ -13 --> &8DAF
8DBC	&	201 038	C9 26	CMP#&26
8DBE		208 016	D0 10	BNE 16 --> &8DD0
8DC0		032 169 141	20 A9 8D	JSR &8DA9 Increment and read character at (&37, &38) pointer
8DC3		032 148 141	20 94 8D	JSR &8D94 Check for numeric digit [Line Number]
8DC6		176 248	B0 F8	BCS -8 --> &8DC0
8DC8	A	201 065	C9 41	CMP#&41
8DCA		144 230	90 E6	BCC -26 --> &8DB2 Continue to Tokenise
8DCC	G	201 071	C9 47	CMP#&47
8DCE		144 240	90 F0	BCC -16 --> &8DC0
8DD0	"	201 034	C9 22	CMP#&22
8DD2		208 012	D0 0C	BNE 12 --> &8DE0
8DD4		032 169 141	20 A9 8D	JSR &8DA9 Increment and read character at (&37, &38) pointer
8DD7	"	201 034	C9 22	CMP#&22
8DD9		240 212	F0 D4	BEQ -44 --> &8DAF
8DDB		201 013	C9 0D	CMP#&0D

```
8DDD       208 245       D0 F5       BNE -11 --> &8DD4
8DDF  `    096           60          RTS
8DE0  :    201 058       C9 3A       CMP#&3A
8DE2       208 009       D0 09       BNE 9 --> &8DED
8DE4       032 162 141   20 A2 8D    JSR &8DA2 Increment (&37, &38) pointer
8DE7  d;   100 059       64 3B       STZ &3B
8DE9  d<   100 060       64 3C       STZ &3C
8DEB       128 197       80 C5       BRA -59 --> &8DB2 Continue to Tokenise
8DED  ,    201 044       C9 2C       CMP#&2C
8DEF       240 190       F0 BE       BEQ -66 --> &8DAF
8DF1  *    201 042       C9 2A       CMP#&2A
8DF3       208 012       D0 0C       BNE 12 --> &8E01
8DF5  ;    165 059       A5 3B       LDA &3B
8DF7       240 230       F0 E6       BEQ -26 --> &8DDF [RTS (as '*' Star command, don't tokenise line)]
8DF9       162 255       A2 FF       LDX#&FF
8DFB  ;    134 059       86 3B       STX &3B
8DFD  d<   100 060       64 3C       STZ &3C
8DFF       128 174       80 AE       BRA -82 --> &8DAF
8E01  .    201 046       C9 2E       CMP#&2E
8E03       240 014       F0 0E       BEQ 14 --> &8E13
8E05       032 148 141   20 94 8D    JSR &8D94 Check for numeric digit [Line Number]
8E08  ,    144 044       90 2C       BCC 44 --> &8E36
8E0A  <    166 060       A6 3C       LDX &3C
8E0C       240 005       F0 05       BEQ 5 --> &8E13
8E0E       032 004 141   20 04 8D    JSR &8D04 Tokenise Line Number
8E11       144 156       90 9C       BCC -100 --> &8DAF
8E13  7    178 055       B2 37       LDA (&37)
8E15       032 155 141   20 9B 8D    JSR &8D9B If character is not "." then check for Digit (Carry is set if found)
8E18       144 005       90 05       BCC 5 --> &8E1F
8E1A       032 162 141   20 A2 8D    JSR &8DA2 Increment (&37, &38) pointer
8E1D       128 244       80 F4       BRA -12 --> &8E13
8E1F       162 255       A2 FF       LDX#&FF
8E21  ;    134 059       86 3B       STX &3B
8E23       128 196       80 C4       BRA -60 --> &8DE9
8E25       032 132 141   20 84 8D    JSR &8D84 Check whether character is valid within a Variable name (letter, '_', or digit)
8E28       144 207       90 CF       BCC -49 --> &8DF9
8E2A  7    178 055       B2 37       LDA (&37)
8E2C       032 132 141   20 84 8D    JSR &8D84 Check whether character is valid within a Variable name (letter, '_', or digit)
8E2F       144 238       90 EE       BCC -18 --> &8E1F
8E31       032 162 141   20 A2 8D    JSR &8DA2 Increment (&37, &38) pointer
8E34       128 244       80 F4       BRA -12 --> &8E2A
8E36  A    201 065       C9 41       CMP#&41
8E38       144 191       90 BF       BCC -65 --> &8DF9
8E3A  X    201 088       C9 58       CMP#&58
8E3C       176 231       B0 E7       BCS -25 --> &8E25
8E3E  V    162 086       A2 56       LDX#&56
```

```
8E40  9   134 057      86 39     STX &39
8E42      162 132      A2 84     LDX#&84
8E44  :   134 058      86 3A     STX &3A
8E46      160 000      A0 00     LDY#&00
8E48  9   210 057      D2 39     CMP (&39)
8E4A      144 222      90 DE     BCC -34 --> &8E2A
8E4C      208 015      D0 0F     BNE 15 --> &8E5D
8E4E      200          C8        INY
8E4F  9   177 057      B1 39     LDA (&39),Y
8E51  01  048 049      30 31     BMI 49 --> &8E84
8E53  7   209 055      D1 37     CMP (&37),Y
8E55      240 247      F0 F7     BEQ -9 --> &8E4E
8E57  7   177 055      B1 37     LDA (&37),Y
8E59  .   201 046      C9 2E     CMP#&2E
8E5B      240 011      F0 0B     BEQ 11 --> &8E68
8E5D      200          C8        INY
8E5E  9   177 057      B1 39     LDA (&39),Y
8E60      016 251      10 FB     BPL -5 --> &8E5D
8E62      201 254      C9 FE     CMP#&FE
8E64      208 015      D0 0F     BNE 15 --> &8E75
8E66      176 194      B0 C2     BCS -62 --> &8E2A
8E68      200          C8        INY
8E69  9   177 057      B1 39     LDA (&39),Y
8E6B  0   048 023      30 17     BMI 23 --> &8E84
8E6D  9   230 057      E6 39     INC &39
8E6F      208 248      D0 F8     BNE -8 --> &8E69
8E71  :   230 058      E6 3A     INC &3A
8E73      128 244      80 F4     BRA -12 --> &8E69
8E75  8   056          38        SEC
8E76      200          C8        INY
8E77      152          98        TYA
8E78  e9  101 057      65 39     ADC &39
8E7A  9   133 057      85 39     STA &39
8E7C      144 002      90 02     BCC 2 --> &8E80
8E7E  :   230 058      E6 3A     INC &3A
8E80  7   178 055      B2 37     LDA (&37)
8E82      128 194      80 C2     BRA -62 --> &8E46
8E84      170          AA        TAX
8E85      200          C8        INY
8E86  9   177 057      B1 39     LDA (&39),Y
8E88  =   133 061      85 3D     STA &3D
8E8A      136          88        DEY
8E8B  J   074          4A        LSR A
8E8C      144 007      90 07     BCC 7 --> &8E95
8E8E  7   177 055      B1 37     LDA (&37),Y
8E90      032 132 141  20 84 8D  JSR &8D84 Check whether character is valid within a Variable name (letter, '_', or digit)
8E93      176 149      B0 95     BCS -107 --> &8E2A
```

8E95		138	8A	TXA
8E96	$=	036 061	24 3D	BIT &3D
8E98	P	080 006	50 06	BVC 6 --> &8EA0
8E9A	;	166 059	A6 3B	LDX &3B
8E9C		208 002	D0 02	BNE 2 --> &8EA0
8E9E	i@	105 064	69 40	ADC#&40
8EA0		136	88	DEY
8EA1		032 235 140	20 EB 8C	JSR &8CEB Replace untokenised value with token
8EA4		162 255	A2 FF	LDX#&FF
8EA6	=	165 061	A5 3D	LDA &3D
8EA8	J	074	4A	LSR A
8EA9	J	074	4A	LSR A
8EAA		144 004	90 04	BCC 4 --> &8EB0
8EAC	;	134 059	86 3B	STX &3B
8EAE	d<	100 060	64 3C	STZ &3C
8EB0	J	074	4A	LSR A
8EB1		144 004	90 04	BCC 4 --> &8EB7
8EB3	d;	100 059	64 3B	STZ &3B
8EB5	d<	100 060	64 3C	STZ &3C
8EB7	J	074	4A	LSR A
8EB8		144 016	90 10	BCC 16 --> &8ECA
8EBA	H	072	48	PHA
8EBB		160 001	A0 01	LDY#&01
8EBD	7	177 055	B1 37	LDA (&37),Y
8EBF		032 132 141	20 84 8D	JSR &8D84 Check whether character is valid within a Variable name (letter, '_', or digit)
8EC2		144 005	90 05	BCC 5 --> &8EC9
8EC4		032 162 141	20 A2 8D	JSR &8DA2 Increment (&37, &38) pointer
8EC7		128 244	80 F4	BRA -12 --> &8EBD
8EC9	h	104	68	PLA
8ECA	J	074	4A	LSR A
8ECB		144 002	90 02	BCC 2 --> &8ECF
8ECD	<	134 060	86 3C	STX &3C
8ECF	J	074	4A	LSR A
8ED0		176 013	B0 0D	BCS 13 --> &8EDF
8ED2	L	076 175 141	4C AF 8D	JMP &8DAF Keep tokenising until end of line found

If character is not "." then check for digit (Line Number) (carry set if found)

8D9B	.	201 046	C9 2E	CMP#&2E
8D9D		208 245	D0 F5	BNE -11 --> &8D94 Check for numeric digit [Line Number]
8D9F	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8456 BASIC Keyword List

Submitted by Steve Fewell

Description:

The Keywords are listed one after the other. Each keyword contains the following parts
A) The Keyword itself
B) A 1-byte token value for the keyword (this value is always a negative value (>=&80) this allows BASIC to easily determine when a Keyword stops and the token begins
(as to save space, the keywords are not stored as BASIC strings (which terminate with a binary 0)).
C) A 1-byte flag value (this flag describes certain attibutes of the keyword and is used by BASIC to syntax check the program and determine whether the keyword has been used correctly or not.

The Keywords are listed in the order that BASIC evaluates keyword abbreviations.
I.e. AND is listed before ABS, so BASIC will interpret "A." as AND (not ABS).

The 1-byte flag associated with each Keyword has the following meanings depending on which bit is set (a keyword may have a flag with many bits set at the same time):

- Bit 0 - Conditional tokenisation (don't tokenise if followed by an alphabetic character).
  This flag indicates that the keyword can safely be used as the start of a variable name (without being tokenised as a keyword).
- Bit 1 - Go into "middle of Statement" mode.
- Bit 2 - Go into "Start of Statement" mode.
- Bit 3 - FN/PROC keyword - don't tokenise the name of the subroutine.
- Bit 4 - Start tokenising a line number now (after a GOTO, etc...).
- Bit 5 - Don't tokenise rest of line (REM,DATA keyword, etc...)
- Bit 6 - Pseudo variable flag - add &40 to token if fund at the start of a statement.
  I.e. this will apply to "PAGE=x" but not to "x=PAGE".
- Bit 7 - Unused.

This values will be described at a later date, when the relevant routines are described.

This list is used for tokenising and de-tokenising (in REPORT/LIST) a program.

Note that the open bracket of many functions is included as part of the token,
so that when abreviating the function name (i.e. LE.) you must not include the open bracket.

Address of first char of Keyword	Keyword	Token	Flag (byte 7 first)
&8456	AND	&80	00000000
&845B	ABS	&94	00000000
&8460	ACS	&95	00000000
&8465	ADVAL	&96	00000000
&846C	ASC	&97	00000000
&8471	ASN	&98	00000000
&8476	ATN	&99	00000000
&847B	AUTO	&C6	00010000
&8481	BGET	&9A	00000001
&8487	BPUT	&D5	00000011
&848D	COLOUR	&FB	00000010
&8495	CALL	&D6	00000010
&849B	CHAIN	&D7	00000010
&84A2	CHR$	&BD	00000000
&84A8	CLEAR	&D8	00000001
&84AF	CLOSE	&D9	00000011
&84B6	CLG	&DA	00000001
&84BB	CLS	&DB	00000001
&84C0	COS	&9B	00000000
&84C5	COUNT	&9C	00000001
&84CC	COLOR	&FB	00000010
&84D3	DATA	&DC	00100000
&84D9	DEG	&9D	00000000
&84DE	DEF	&DD	00000000
&84E3	DELETE	&C7	00010000
&84EB	DIV	&81	00000000
&84F0	DIM	&DE	00000010
&84F5	DRAW	&DF	00000010
&84FB	ENDPROC	&E1	00000001
&8504	END	&E0	00000001
&8509	ENVELOPE	&E2	00000010

&8513	ELSE	&8B	00010100
&8519	EVAL	&A0	00000000
&851F	ERL	&9E	00000001
&8524	ERROR	&85	00000100
&852B	EOF	&C5	00000001
&8530	EOR	&82	00000000
&8535	ERR	&9F	00000001
&853A	EXP	&A1	00000000
&853F	EXT	&A2	00000001
&8544	EDIT	&CE	00010000
&854A	FOR	&E3	00000010
&854F	FALSE	&A3	00000001
&8556	FN	&A4	00001000
&855A	GOTO	&E5	00010010
&8560	GET$	&BE	00000000
&8566	GET	&A5	00000000
&856B	GOSUB	&E4	00010010
&8572	GCOL	&E6	00000010
&8578	HIMEM	&93	01000011
&857F	INPUT	&E8	00000010
&8586	IF	&E7	00000010
&858A	INKEY$	&BF	00000000
&8592	INKEY	&A6	00000000
&8599	INT	&A8	00000000
&859E	INSTR(	&A7	00000000
&85A6	LIST	&C9	00010000
&85AC	LINE	&86	00000000
&85B2	LOAD	&C8	00000010
&85B8	LOMEM	&92	01000011
&85BF	LOCAL	&EA	00000010
&85C6	LEFT$(	&C0	00000000
&85CE	LEN	&A9	00000000
&85D3	LET	&E9	00000100
&85D8	LOG	&AB	00000000
&85DD	LN	&AA	00000000
&85E1	MID$(	&C1	00000000
&85E8	MODE	&EB	00000010
&85EE	MOD	&83	00000000

&85F3	MOVE	&EC	00000010
&85F9	NEXT	&ED	00000010
&85FF	NEW	&CA	00000001
&8604	NOT	&AC	00000000
&8609	OLD	&CB	00000001
&860E	ON	&EE	00000010
&8612	OFF	&87	00000000
&8617	OR	&84	00000000
&861B	OPENIN	&8E	00000000
&8623	OPENOUT	&AE	00000000
&862C	OPENUP	&AD	00000000
&8634	OSCLI	&FF	00000010
&863B	PRINT	&F1	00000010
&8642	PAGE	&90	01000011
&8648	PTR	&8F	01000011
&864D	PI	&AF	00000001
&8651	PLOT	&F0	00000010
&8657	POINT(	&B0	00000000
&865F	PROC	&F2	00001010
&8665	POS	&B1	00000001
&866A	RETURN	&F8	00000001
&8672	REPEAT	&F5	00000000
&867A	REPORT	&F6	00000001
&8682	READ	&F3	00000010
&8688	REM	&F4	00100000
&868D	RUN	&F9	00000001
&8692	RAD	&B2	00000000
&8697	RESTORE	&F7	00010010
&86A0	RIGHT$(	&C2	00000000
&86A9	RND	&B3	00000001
&86AE	RENUMBER	&CC	00010000
&86B8	STEP	&88	00000000
&86BE	SAVE	&CD	00000010
&86C4	SGN	&B4	00000000
&86C9	SIN	&B5	00000000
&86CE	SQR	&B6	00000000
&86D3	SPC	&89	00000000
&86D8	STR$	&C3	00000000

&86DE	STRING$(	&C4	00000000
&86E8	SOUND	&D4	00000010
&86EF	STOP	&FA	00000001
&86F5	TAN	&B7	00000000
&86FA	THEN	&8C	00010100
&8700	TO	&B8	00000000
&8704	TAB(	&8A	00000000
&870A	TRACE	&FC	00010010
&8711	TIME	&91	01000011
&8717	TRUE	&B9	00000001
&871D	UNTIL	&FD	00000010
&8724	USR	&BA	00000000
&8729	VDU	&EF	00000010
&872E	VAL	&BB	00000000
&8733	VPOS	&BC	00000001
&8739	WIDTH	&FE	00000010

End of Tokeniser list, the following list is a continuation of the above for LISTing/REPORT commands (It lists the Pseudo Variables with the +&40 added.

&8740	PAGE	&D0	00000000
&8746	PTR	&CF	00000000
&874B	TIME	&D1	00000000
&8751	LOMEM	&D2	00000000
&8758	HIMEM	&D3	00000000

The following token is only used by the REPORT command.

&875F	Missing [+ ASCII 32 (Space)]	&8D	00000000

BASIC IV ROM Routines

Page Last Altered: undefined

8DA0 Increment [and read char at] Pointer (&37, &38)

Submitted by Steve Fewell

Description:

If called from 8DA0, The character pointed to by (&37, &38) pointer is retrieved (in A), and the pointer is then incremented. The character (prior to incrementing) is returned in A.
If called from 8DA2, then the Pointer (&37, &38) is incremented.
If called from 8DA9, The pointer at (&37, &38) is incremented, and the next character is retrieved from the pointer location and returned in A.

Disassembly for the Increment [& read] pointer (&37, &38) routine

8DA0	7	178 055	B2 37	LDA (&37)
8DA2	7	230 055	E6 37	INC &37
8DA4	9	208 057	D0 39	BNE 57 --> &8DDF [RTS]
8DA6	8	230 056	E6 38	INC &38
8DA8	`	096	60	RTS
8DA9		032 162 141	20 A2 8D	JSR &8DA2
8DAC	7	178 055	B2 37	LDA (&37)
8DAE	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8D84 Check for Variable name character or digit (in A)

Submitted by Steve Fewell

Entry: A contains an ASCII character code.
Exit: The carry flag is 1 if the character in A is a valid variable name letter ('A'-'Z', '_', '£', 'a'-'z') or a digit ('0'-'9'). The carry flag is 0 otherwise.

Description:

If the character (in A) is greater than or equal to 123 (&7B) then exit with Carry = 0 [char > 'z'].
If the character is greater than or equal to 95 (&5F) then exit with Carry = 1 [char = '_' or '£' or 'a'-'z'].
If the character is greater than or equal to 91 (&5B) then exit with Carry = 0 [char = '[', '\', ']' or '^'].
If the character is greater than or equal to 65 (&41) then exit with Carry = 1 [char = 'A'-'Z'].
If the character is greater than or equal to 58 (&3A) then exit with Carry = 0 [char = ':', ';', '<', '=', '>', ?' or '@'].
[&8D94] If the character is greater than or equal to 48 (&30) then exit with Carry = 1 [char = '0'-'9'].
Otherwise (if character is less than 48 (&30) then exit with Carry = 0.

Disassembly for the Check for Variable name character or digit (in A) routine

8D84	{	201 123	C9 7B	CMP#&7B
8D86		176 250	B0 FA	BCS -6 --> &8D82 [CLC : RTS]
8D88	_	201 095	C9 5F	CMP#&5F
8D8A		176 014	B0 0E	BCS 14 --> &8D9A
8D8C	[	201 091	C9 5B	CMP#&5B
8D8E		176 242	B0 F2	BCS -14 --> &8D82 [CLC : RTS]
8D90	A	201 065	C9 41	CMP#&41
8D92		176 006	B0 06	BCS 6 --> &8D9A
8D94	:	201 058	C9 3A	CMP#&3A
8D96		176 234	B0 EA	BCS -22 --> &8D82 [CLC : RTS]
8D98	0	201 048	C9 30	CMP#&30
8D9A	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8D04 Tokenise Line Number

Submitted by Steve Fewell

Routine:tokeniseLineNumber
Name: Tokenise Line Number
Starting Address: &8D04
Entry criteria: A contains the first digit of the line number.
Pointer (&37, &38) points to the address in the program of the first digit of the Program Line Number.
Exit: The line number in the Program Line is tokenised.
Carry flag is set if error occurred (Line number too large).

Description:
Store the 4 lowest bits (bits 0, 1, 2 & 3) of the first digit (in A) in location &3D.
Set X to zero (the upper digit of the 2-byte binary line number) Set Y to 1 (so the next character read will be the character after the first digit of the line number).

[&8D0C] Read next character from the (&37, &38) pointer.
If the next character is not a digit then go to &8D42 to store the line number token and exit.

Store the 4 lowest bits (bits 0, 1, 2 & 3) of the new digit on the stack.
Store X in location &3E.

Multiply the Line Number (in locations &3D (LSB) abd &3E (MSB) by 10, by:
*1) Storing the value of location &3D and &3E.
*2) Multiplying the Line Number value (in locations &3D-&3E) by 4
*3) Adding the stored &3D and &3E value to the result from step *2).
*4) Multiplying the Line Number value (in locations &3D-&3E [note: actually A at this stage] by 2.
If the Line Number MSB (&3E) becomes negative (or overflows) at any time during this multiply operation, the routine will be exited with Carry set (as the line number is too large).

Next, retrieve the encoded next digit value from the Stack and add this digit to the Line Number (&3D-&3E).
This adds the digit to the binary line number.
If X (working copy of &3E) is negative then jump to &8D3F (exit with carry flag set), as the line number is too large.
Otherwise, jump back to &8D0C to process any further digits.

&8D42 Store Line Number Token & exit:

Decrement Y offset (so that Y points to the end of the ASCII Line Number on the command line).
Set A to #&8D (The Token for Line Number).

Call routine &8CEB to replace the ASCII Line Number currently present in the Program line with #&8D.

Next, Copy the program text pointer (&37, &38) to locations (&39 - &3A).
This is a pointer to the #&8D token value on the program line.
Increment Pointer (&37, &38) by 3 (to point to the location after the 3-byte tokenised line number.
by the 3-digit encoded line number value).

Copy program line text from location pointed to by (&39,&3A) to the (&37, &38) location (last byte first)
[routine &8CEB set Y to point to the end of the command line].
The first byte is correctly set to #&8D, so don't copy this byte.
This moves the command line text (after the program line number lcoation) along 3 bytes - to make space for the tokenised line number.

Set Y to 3 (deal with the last byte of the tokenised line Number first).
Set A to X (the MSB of the Line Number).
OR A value with #&40 (01000000) to set bit 6 of the Byte, then store this result in the 3rd byte of the tokenised line number (i.e. the 3rd byte after the #&8D token value).

Set A to &3D value (LSB of Line Number). AND this value with #&3F (00111111) to clear the top 2 bits of the value and OR with #&40 (01000000) to set bit 6. Finally, store the result in the 2nd byte of the Tokenised Line number (i.e. the 2nd byte after the #&8D token value).

Next, clear all but the top two bits of the &3D (LSB of Line Number) value.
AND the MSB of the Line Number with #&C0 (11000000) to clear all but the top 2 bits. Shift the top two bits of the MSB Line Number value along 2 positions and set the top 2 bits to the top two bits of the &3D (LSB of Line Number) value. EOR this result with #&54 (01010100).
Store the result as the 1st byte of the tokenised Line Number.

The resulting tokenised value contains 3 bytes which each contain an encoded value between &40 and &7F.
This ensures that tokenised Line Numbers cannot be confused with BASIC tokenised keyword values (which are all >= &80), or with the quote character (").
This is important, otherwise BASIC might interpret the Line Number value as a String or as another BASIC keyword.

Examples:

Example1: Line Number = 10
[&8D04] A=&31. AND #&0F with A (1st digit), result=&01 (store in &3D).
[&8D14] A=&30. AND #&0F with A (2nd digit), result=&00. &3E & X = &00.
[&8D1E] Now: A=&02 and &3E=&00.
[&8D23] Now: A=&04 and &3E=&00.
[&8D27] Now: &3D and A = &05 [&8D2E] Now: A=&00, after addition, A=&00. &3D=&0A, A=&00.
[&8D39] Now: X=&00, A=&00 and &3D=&0A.

This 2-byte value is tokenised as follows:
(X = &00, &3D = &0A)
3rd byte of tokenised value = &40 (X (&00) OR &40).
2nd byte of tokenised value = &4A (A (&0A) OR &40).
1st byte of tokenised value = &54 (&00 EOR &54).

Example2: Line Number = 12345
[&8D04] A=&31. AND #&0F with A (1st digit), result=&01 (store in &3D).
[&8D14] A=&32. AND #&0F with A (2nd digit), result=&02. &3E & X = &00.
[&8D1E] Now: A=&02 and &3E=&00.
[&8D27] Now: A = &04, &3E = &00 and &3D = &05. [&8D2E] Now: A=&00, after addition, A=&00. &3D=&0A, A=&00.
[&8D39] Now: X=&00, A=&0C and &3D=&0C.

[&8D14] A=&33. AND #&0F with A (3rd digit), result=&03. &3E & X = &00.
[&8D1E] Now: A=&0C (= &18 after addition) and &3E=&00.
[&8D27] Now: A = &30, &3E = &00 and &3D = &3C. [&8D2E] Now: A=&00, after addition, A=&00. &3D=&78, A=&00.
[&8D39] Now: X=&00, &3D=&7B.
[&8D14] A=&34. AND #&0F with A (4th digit), result=&04. &3E & X = &00.
[&8D1E] Now: A=&7B (= &F6 after addition) and &3E=&00.
[&8D27] Now: A = &EC, &3E = &01 and &3D = &67. [&8D2E] Now: A=&00, after addition, A=&02. &3D=&CE, A=&04.
[&8D39] Now: X=&04 and &3D=&D2.
[&8D14] A=&35. AND #&0F with A (5th digit), result=&05. &3E & X = &04.
[&8D1E] Now: A=&D2 (= &A4 after addition) and &3E=&09.
[&8D27] Now: A = &48, &3E = &13 and &3D = &1A. [&8D2E] Now: A=&04, after addition, A=&18. &3D=&34, A=&30.
[&8D39] Now: X=&30 and &3D=&39.

This 2-byte value is tokenised as follows:
(X = &30, &3D = &39)
3rd byte of tokenised value = &70 (X (&40) OR &30).
2nd byte of tokenised value = &79 (A (&40) OR &39).
1st byte of tokenised value = &54 (&00 EOR &54).

Example3: Line Number = 333
[&8D04] A=&33. AND #&0F with A (1st digit), result=&03 (store in &3D).
[&8D14] A=&33. AND #&0F with A (2nd digit), result=&03. &3E & X = &00.
[&8D1E] Now: A=&06 and &3E=&00.
[&8D27] Now: A = &0C, &3E = &00 and &3D = &0F. [&8D2E] Now: A=&00, after addition, A=&00. &3D=&1E, A=&00.
[&8D39] Now: X=&00 and &3D=&21.
[&8D14] A=&33. AND #&0F with A (3rd digit), result=&03. &3E & X = &00.
[&8D1E] Now: A=&42 and &3E=&00.
[&8D27] Now: A = &84, &3E = &00 and &3D = &A5. [&8D2E] Now: A=&00, after addition, A=&00. &3D=&4A, A=&01.
[&8D39] Now: X=&01 and &3D=&4D.
This 2-byte value is tokenised as follows:
(X = &01, &3D = &4D)
3rd byte of tokenised value = &41 (X (&01) OR &40).
2nd byte of tokenised value = &4D (A (&0D) OR &40).
1st byte of tokenised value = &44 (&10 EOR &54) [(A (&01) AND &C0 => &00) ORA &3D (top 2 bits) = &40;
divide by 4 = &10 EOR &54 ==> &44].

Disassembly for the Tokenise Line Number routine

8D04	)	041 015	29 0F	AND#&0F
8D06	=	133 061	85 3D	STA &3D
8D08		162 000	A2 00	LDX#&00
8D0A		160 000	A0 00	LDY#&00
8D0C		200	C8	INY
8D0D	7	177 055	B1 37	LDA (&37),Y
8D0F		032 148 141	20 94 8D	JSR &8D94 Check for numeric digit
8D12	.	144 046	90 2E	BCC 46 --> &8D42 (no more numeric digits present)
8D14	)	041 015	29 0F	AND#&0F
8D16	H	072	48	PHA
8D17	>	134 062	86 3E	STX &3E
8D19	=	165 061	A5 3D	LDA &3D

8D1B		010	0A	ASL A
8D1C	&>	038 062	26 3E	ROL &3E
8D1E	0	048 031	30 1F	BMI 31 --> &8D3F
8D20		010	0A	ASL A
8D21	&>	038 062	26 3E	ROL &3E
8D23	0	048 026	30 1A	BMI 26 --> &8D3F
8D25	e=	101 061	65 3D	ADC &3D
8D27	=	133 061	85 3D	STA &3D
8D29		138	8A	TXA
8D2A	e>	101 062	65 3E	ADC &3E
8D2C	=	006 061	06 3D	ASL &3D
8D2E	*	042	2A	ROL A
8D2F	0	048 014	30 0E	BMI 14 --> &8D3F
8D31		176 012	B0 0C	BCS 12 --> &8D3F
8D33		170	AA	TAX
8D34	h	104	68	PLA
8D35	e=	101 061	65 3D	ADC &3D
8D37	=	133 061	85 3D	STA &3D
8D39		144 209	90 D1	BCC -47 --> &8D0C
8D3B		232	E8	INX
8D3C		016 206	10 CE	BPL -50 --> &8D0C
8D3E	H	072	48	PHA
8D3F	h	104	68	PLA
8D40	8	056	38	SEC
8D41	`	096	60	RTS
8D42		136	88	DEY
8D43		169 141	A9 8D	LDA#&8D
8D45		032 235 140	20 EB 8C	JSR &8CEB Replace untokenised value with token
8D48	7	165 055	A5 37	LDA &37
8D4A	9	133 057	85 39	STA &39
8D4C	8	165 056	A5 38	LDA &38
8D4E	:	133 058	85 3A	STA &3A
8D50		032 162 141	20 A2 8D	JSR &8DA2 Increment Pointer (&37, &38)
8D53		032 162 141	20 A2 8D	JSR &8DA2 Increment Pointer (&37, &38)
8D56		032 162 141	20 A2 8D	JSR &8DA2 Increment Pointer (&37, &38)
8D59	9	177 057	B1 39	LDA (&39),Y
8D5B	7	145 055	91 37	STA (&37),Y
8D5D		136	88	DEY
8D5E		208 249	D0 F9	BNE -7 --> &8D59
8D60		160 003	A0 03	LDY#&03
8D62		138	8A	TXA
8D63	@	009 064	09 40	ORA#&40
8D65	9	145 057	91 39	STA (&39),Y
8D67		136	88	DEY

8D68	=	165 061	A5 3D	LDA &3D
8D6A	)?	041 063	29 3F	AND#&3F
8D6C	@	009 064	09 40	ORA#&40
8D6E	9	145 057	91 39	STA (&39),Y
8D70		136	88	DEY
8D71	?	169 063	A9 3F	LDA#&3F
8D73	=	020 061	14 3D	TRB &3D
8D75		138	8A	TXA
8D76	)	041 192	29 C0	AND#&C0
8D78	J	074	4A	LSR A
8D79	J	074	4A	LSR A
8D7A	=	005 061	05 3D	ORA &3D
8D7C	J	074	4A	LSR A
8D7D	J	074	4A	LSR A
8D7E	IT	073 084	49 54	EOR#&54
8D80	9	145 057	91 39	STA (&39),Y
8D82		024	18	CLC
8D83	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8CEB Replace untokenised ASCII value with a 1-byte token

Submitted by Steve Fewell

Routine:ASCIItoToken
Name: Replace untokenised ASCII value with a 1-byte token
Starting Address: &8CEB
Entry criteria: &37 and &38 point to the start of the ASCII Keyword in the program text.
A contains the token value to replace the ASCII text with.
Y contains the offset to the end character of the Keyword.
Exit: Y = offset pointer to the end of the program command line.

Description:

Store A (the token value) as the first byte of the ASCII Keyword (the location pointed to by (&37, &38)).
Set (&39, &3A) to point to (&37, &38) + Y.
So that (&37, &38) point to the start of the Keyword and (&39, &3A) point to the end of the Keyword.
Copy each character from ((&39, &3A) + 1) to ((&37, &38) + 1) until a <cr> (carriage return character)
is encountered. After a <cr> has been copied, the routine ends, as the rest of the program line has been moved sucessfully to be positioned directly after the token.

Disassembly for the 'Replace untokenised ASCII value with a 1-byte token' routine

8CEB	7	146 055	92 37	STA (&37)
8CED		024	18	CLC
8CEE		152	98	TYA
8CEF	e7	101 055	65 37	ADC &37
8CF1	9	133 057	85 39	STA &39
8CF3		160 000	A0 00	LDY#&00
8CF5		152	98	TYA
8CF6	e8	101 056	65 38	ADC &38
8CF8	:	133 058	85 3A	STA &3A
8CFA		200	C8	INY
8CFB	9	177 057	B1 39	LDA (&39),Y
8CFD	7	145 055	91 37	STA (&37),Y

8CFF	201 013	C9 0D	CMP#&0D
8D01	208 247	D0 F7	BNE -9 --> &8CFA
8D03 `	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9B1C Detokenise the requested Line Number and Set IWA to the Line Number value

Submitted by Steve Fewell

Starting Address: &9B1C (or 9B1E if &0A doesn't need incrementing)
Entry criteria: &0A and &0B (the PTR A) point to the position on the program line where a Line Number is expected.
Exit: The IWA contains the detokenised value of the Line Number.
Carry Flag = 1 if a line number was found (and its value is in the IWA).
Carry Flag = 0 if a tokenised line number was not found.
Y contains the offset of the next character after the line number (or the next non-space character if no Line Number was found.

Description:

If called from &9B1C, increment the PTR A Offset.
Get the next non-space character from Basic text Pointer A (including the offset (&0A)).

If the next character is not #&8D (Line Number token) then goto &9B44 (exit with Carry = 0) as there is no Line Number to detokenise.

Get the next (1st) character from PTR A (the character after the &8D token).
Multiply the ASCII code by 4 (to clear the lower 2-bits, ready to be merged with the next byte).
Store the result in X, to backup current value of the first ASCII code.
AND the ASCII code by #&C0 (1100 0000) to clear all but the top 2 bits.
EOR the ASCII code with the next (2nd) character from PTR A.
Store the result (of the merged first and second bytes) in &2A.
A = X (i.e. the backup value of the Line Number's first ASCII code).
Multiply A by 4 (to clear the top 2-bits).
EOR A with the next (3rd) character from PTR A.
Store the result in &2B.
So, basically, the detokenised line number LSB is the 2nd byte of the line number EORed with bytes 4 and 5 (positioned in bytes 6 and 7 of the byte) of the 1st byte; and the detokenised line number MSB is the 3rd byte of the line number EORed with bytes 2 and 3 (positioned in bytes 6 and 7 of the byte) of the 1st byte.

Update PTR A offset to point to the next character after the line number.
Set carry and exit.

Examples:

Example 1: [Line Number = 10]

Tokenised Line value = &8D &54 &4A &40
A = &54. Multiply by 4, now A = &50. X = 50.
AND A with #&C0 -> A = &40.
EOR A with next character [&4A EOR &40] -> A = &0A (Store in &2A).
A = X (&50). Multiply by 4, now A = &40.
EOR A with next character [&40 EOR &40] -> A = &00 (Store in &2B).
Finished. &2A-&2B contains &000A (LSB first) = 10 in decimal.

Example 2: [Line Number = 1816]

Tokenised Line value = &8D &54 &58 &47
A = &54. Multiply by 4, now A = &50. X = 50.
AND A with #&C0 -> A = &40.
EOR A with next character [&58 EOR &40] -> A = &18 (Store in &2A).
A = X (&50). Multiply by 4, now A = &40.
EOR A with next character [&47 EOR &40] -> A = &07 (Store in &2B).
Finished. &2A-&2B contains &0718 (LSB first) = 1816 in decimal.

Example 3: [Line Number = 1000]

Tokenised Line value = &8D &64 &68 &43
A = &64. Multiply by 4, now A = &90. X = 90.
AND A with #&C0 -> A = &80.
EOR A with next character [&68 EOR &80] -> A = &E8 (Store in &2A).
A = X (&90). Multiply by 4, now A = &40.
EOR A with next character [&43 EOR &40] -> A = &03 (Store in &2B).
Finished. &2A-&2B contains &03E8 (LSB first) = 1000 in decimal.

Example 4: [Line Number = 121]

Tokenised Line value = &8D &44 &79 &40
A = &44. Multiply by 4, now A = &10. X = 10.
AND A with #&C0 -> A = &00.
EOR A with next character [&79 EOR &00] -> A = &79 (Store in &2A).
A = X (&10). Multiply by 4, now A = &40.
EOR A with next character [&40 EOR &40] -> A = &00 (Store in &2B).
Finished. &2A-&2B contains &0079 (LSB first) = 121 in decimal.

Example 5: [Line Number = 32766]

Tokenised Line value = &8D &60 &7E &7F
A = &60. Multiply by 4, now A = &80. X = 80.
AND A with #&C0 -> A = &80.
EOR A with next character [&7E EOR &80] -> A = &FE (Store in &2A).
A = X (&80). Multiply by 4, now A = &00.
EOR A with next character [&7F EOR &00] -> A = &7F (Store in &2B).
Finished. &2A-&2B contains &7FFE (LSB first) = 32766 in decimal.

Disassembly for the Detokenise the requested Line Number and Set IWA to the Line Number value routine

9B1C	230 010	E6 0A	INC &0A
9B1E	164 010	A4 0A	LDY &0A
9B20	177 011	B1 0B	LDA (&0B),Y

9B22		201 032	C9 20	CMP#&20
9B24		240 246	F0 F6	BEQ -10 --> &9B1C
9B26		201 141	C9 8D	CMP#&8D
9B28		208 026	D0 1A	BNE 26 --> &9B44
9B2A		200	C8	INY
9B2B		177 011	B1 0B	LDA (&0B),Y
9B2D		010	0A	ASL A
9B2E		010	0A	ASL A
9B2F		170	AA	TAX
9B30	)	041 192	29 C0	AND#&C0
9B32		200	C8	INY
9B33	Q	081 011	51 0B	EOR (&0B),Y
9B35	*	133 042	85 2A	STA &2A
9B37		138	8A	TXA
9B38		010	0A	ASL A
9B39		010	0A	ASL A
9B3A		200	C8	INY
9B3B	Q	081 011	51 0B	EOR (&0B),Y
9B3D	+	133 043	85 2B	STA &2B
9B3F		200	C8	INY
9B40		132 010	84 0A	STY &0A
9B42	8	056	38	SEC
9B43	`	096	60	RTS
9B44		024	18	CLC
9B45	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

BA98 Remove the Line Number (specified in the IWA) from the program

Submitted by Steve Fewell

Routine:RemoveLine
Name: Remove Line Number from Program
Starting Address: &BA98
Entry criteria: The IWA contains the Line number to remove (in ASCII Text format).
Exit: The specified line number has been removed from the program.

Description:

Call routine &80CD to find the first program line that is greater than or equal to the line numeric number value in the IWA.
If the line number in the IWA does not exist in the program (Carry flag is clear) then exit, as no line to remove.

Routine &80CD returns with pointer (&3D, &3E) set to point to the start of the program line (specified in the IWA).
Set TOP and pointer (&37, &38) to point to the current program text for the line we want to remove - i.e. the value of the pointer (&3D, &3E).

Read the 3rd byte from location (&37, &38), this contains the length of the program line.
Add the length of the program line to the (&37, &38) pointer.
Now: TOP points to the location of the line to remove and pointer (&37, &38) points to the next line after the line we wish to remove.

[&BAB6] Move the next line number (pointed to by (&37, &38)) to the TOP location (overwriting the current line in the program (that is the Line that TOP originally pointed to) as follows:

Set Y to 0
[&BAB8] Keep doing the following:
* Copy the next character from the ((&37, &38) + Y) location to the (TOP + Y) location
* [&BAD3] increment Y (if Y overflowed then also increment &38 and TOP MSB).
Until the next character is <cr> [end of program line]

[&BAC0] Increment Y (to account for the <cr> byte that was copied). If Y overflowed then also increment &38 and TOP MSB.
[&BAC7] Copy the next character from ((&37, &38) + Y) pointer to (TOP + Y) location.
If the character (which should be the first byte of the next line number) is negative (i.e. #&FF, then end of the program has been found, so update TOP (Add Y+1 to TOP) and exit as the whole program (from the line after the IWA line number) has been copied to its new location.

Otherwise (end of program not found yet, so need to move next line), increment Y (to account for the copied byte).

If Y overflowed then also increment &38 and TOP MSB.
Copy the next character (Line number byte 2) from ((&37, &38) + Y) pointer to (TOP + Y) location.
Increment Y (to account for the copied byte). If Y overflowed then also increment &38 and TOP MSB.
Copy the next character from ((&37, &38) + Y) pointer to (TOP + Y) location.
Increment Y (to account for the copied byte). If Y overflowed then also increment &38 and TOP MSB.
Jump back to &BAB8 to copy the rest of the line, and continue checking for more program lines.

Disassembly for the Remove Line Number from Program routine

BA98		032 205 128	20 CD 80	JSR &80CD Search for Program Line >= the Line Number in the IWA
BA9B	M	144 077	90 4D	BCC 77 --> &BAEA [RTS]
BA9D	=	165 061	A5 3D	LDA &3D
BA9F	7	133 055	85 37	STA &37
BAA1		133 018	85 12	STA &12
BAA3	>	165 062	A5 3E	LDA &3E
BAA5	8	133 056	85 38	STA &38
BAA7		133 019	85 13	STA &13
BAA9		160 003	A0 03	LDY#&03
BAAB	7	177 055	B1 37	LDA (&37),Y
BAAD		024	18	CLC
BAAE	e7	101 055	65 37	ADC &37
BAB0	7	133 055	85 37	STA &37
BAB2		144 002	90 02	BCC 2 --> &BAB6
BAB4	8	230 056	E6 38	INC &38
BAB6		160 000	A0 00	LDY#&00
BAB8	7	177 055	B1 37	LDA (&37),Y
BABA		145 018	91 12	STA (&12),Y
BABC		201 013	C9 0D	CMP#&0D
BABE		208 019	D0 13	BNE 19 --> &BAD3
BAC0		200	C8	INY
BAC1		208 004	D0 04	BNE 4 --> &BAC7
BAC3	8	230 056	E6 38	INC &38
BAC5		230 019	E6 13	INC &13
BAC7	7	177 055	B1 37	LDA (&37),Y
BAC9		145 018	91 12	STA (&12),Y
BACB	0	048 015	30 0F	BMI 15 --> &BADC
BACD		032 223 186	20 DF BA	JSR &BADF
BAD0		032 223 186	20 DF BA	JSR &BADF
BAD3		200	C8	INY
BAD4		208 226	D0 E2	BNE -30 --> &BAB8
BAD6	8	230 056	E6 38	INC &38
BAD8		230 019	E6 13	INC &13
BADA		128 220	80 DC	BRA -36 --> &BAB8
BADC	L	076 005 190	4C 05 BE	JMP &BE05 Update TOP (TOP value = Y (+1 if carry flag set), set Y to 1 & exit)
BADF		200	C8	INY

```
BAE0       208 004     D0 04       BNE 4 --> &BAE6
BAE2       230 019     E6 13       INC &13
BAE4   8   230 056     E6 38       INC &38
BAE6   7   177 055     B1 37       LDA (&37),Y
BAE8       145 018     91 12       STA (&12),Y
BAEA   `   096         60          RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

80CD Search for Program Line

Submitted by Steve Fewell

Routine:searchprogline
Name: Search for Program Line (in IWA)
Starting Address: &80CD
Entry criteria: The IWA contains the 2-byte Program line number to search for (&2A-&2B).
Exit: &3D-&3E contain a pointer to the Program Line that is greater than or equal to the Line Number in the IWA. C (carry flag) = 1 and Y = 2 if the Line number in the IWA was found (equal). C = 0 and Y = 2 if the Line number wasn't found and &3D-&3E point to the next existing line after the Line Number contained in the IWA>.

Description:

Set &3D-&3E to PAGE (by zeroing &3D and setting &3E to &18).
[&80D3] Compare line number MSB [the value pointed to by (&3D-&3E) + 1] with &2B (MSB of the Line Number to search for).

If the Line Number MSB at &3D-&3E is less than the Line Number MSB in the IWA (&2B) then:

* [&80DB] Update (&3D-&3E) pointer to the next Line Number by adding the Number of Bytes in the current Line [the value pointed to by (&3D-&3E) + 3] to the (&3D-&3E) pointer.
  Note: Program Line numbers are stored as follows [1st Byte = Line Number LSB, 2nd Byte = Line Number MSB, 3rd Byte = Number of Bytes in Line].
* Jump back to &80D3 to Compare the next Line Number.

[&80E9] If the MSB Line Number bytes were not equal then (no exact line number match) exit with C = 0 and Y = 2.
Compare line number LSB [the value pointed to by (&3D-&3E) + 2] with &2A (LSB of the Line Number to search for).
If the current line number is less than the one we are searching for (in the IWA) then jump back to &80DB to update the pointer (&3D-&3E) to point to the next Line Number in the Program, and compare the next line number with the IWA MSB Byte again.

Now we have found a line number that is >= the Line Number in the IWA (or reached the end of the program [#&FF marker]).
If the Line Number is equal to the IWA line, then exit with C = 1; Otherwise, exit with Y = 2 and C = 0.

Disassembly for the Search for Program Line routine

80CD	d=	100 061	64 3D	STZ &3D
80CF		165 024	A5 18	LDA &18
80D1	>	133 062	85 3E	STA &3E
80D3		160 001	A0 01	LDY#&01
80D5	=	177 061	B1 3D	LDA (&3D),Y
80D7	+	197 043	C5 2B	CMP &2B
80D9		176 014	B0 0E	BCS 14 --> &80E9
80DB		160 003	A0 03	LDY#&03
80DD	=	177 061	B1 3D	LDA (&3D),Y
80DF	e=	101 061	65 3D	ADC &3D
80E1	=	133 061	85 3D	STA &3D
80E3		144 238	90 EE	BCC -18 --> &80D3
80E5	>	230 062	E6 3E	INC &3E
80E7		128 234	80 EA	BRA -22 --> &80D3
80E9		208 010	D0 0A	BNE 10 --> &80F5
80EB		200	C8	INY
80EC	=	177 061	B1 3D	LDA (&3D),Y
80EE	*	197 042	C5 2A	CMP &2A
80F0		144 233	90 E9	BCC -23 --> &80DB
80F2		208 001	D0 01	BNE 1 --> &80F5
80F4	`	096	60	RTS
80F5		160 002	A0 02	LDY#&02
80F7		024	18	CLC
80F8	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

BDE5 Check program can be read correctly

Submitted by Steve Fewell

Description:

Set TOP (&12-&13) to PAGE (&18) [The LSB byte of TOP is zero as PAGE does not store an LSB byte].

[&BDED] If the first byte of the program line is not #&0D then show the 'Bad program' message [described below].

If the second byte of the program line is negative then we have reached the end of the program (#&FF value), so add 2 to the TOP value to account for the 2-byte null line value and exit with Y = 1. The program was checked correctly!

If the fourth byte of the program line (line length) is #&00 then the program is corrupt, as this is an invalid line length, so show the 'Bad program' message [described below - &BE11].

[&BDFE] Otherwise, the program line is ok, so Add the line length to TOP and set Y back to 1 (offset for next line); then, jump back to &BDED to check the next program line.

[&BE11] Show 'Bad program' message
This routine is called if the program check located an error or corruption in the program, meaning that the program is not correct. The text 'Bad program' will be shown on the screen - this is only a screen message and not a forced error that stops the current operation and jumps to a specified error routine - specified by the BBC's Break [BRK] vector.

This routine call routine &BECF.
Routine &BECF does the following:
* Retrieves the address of the position to return to (after the subroutine finishes) from the stack.
* Store this address in (&37, &38). Now, (&37,. &38) point to the first character after the &BECF call.
* Keep on incrementing (&37, &38), reading the character from this location and, if the character is positive (less than #&80), then:
    -> Show the character on the screen [if the characetr is #&0D then a new line will be shown instead]
    -> Jump back to increment (&37, &38) and process the next character.
* If the character is negative (greater than or equal to #&80) then there are no more characters to display, so jump to the current location pointed to by (&37, &38) -> this will be the location directly after the text message.
[The first character after the text message is usually #&EA (The NOP Mnenomic), which represents the end of the text, as its value is negative (>=&80].

In the case of the 'check program can be read correctly' routine, the program code after the Text message is:
NOP
JMP &8F86

The NOP command does nothing (but as it has a value >= #&80, it represents the end of the message).
The JMP command jumps to &8F86 which returns to the BASIC command line to read and execute the user's next command line instruction.

Note: This routine does not need to end with an RTS, as the stack has already been tidied up by the removal of the return address bytes.

Disassembly for the Check program can be read correctly routine

BDE5		165 024	A5 18	LDA &18
BDE7		133 019	85 13	STA &13
BDE9	d	100 018	64 12	STZ &12
BDEB		160 001	A0 01	LDY#&01
BDED		178 018	B2 12	LDA (&12)
BDEF		201 013	C9 0D	CMP#&0D
BDF1		208 030	D0 1E	BNE 30 --> &BE11
BDF3		177 018	B1 12	LDA (&12),Y
BDF5	0	048 012	30 0C	BMI 12 --> &BE03
BDF7		160 003	A0 03	LDY#&03
BDF9		177 018	B1 12	LDA (&12),Y
BDFB		240 020	F0 14	BEQ 20 --> &BE11
BDFD		024	18	CLC
BDFE		032 006 190	20 06 BE	JSR &BE06
BE01		128 234	80 EA	BRA -22 --> &BDED
BE03		200	C8	INY
BE04		024	18	CLC
BE05		152	98	TYA
BE06	e	101 018	65 12	ADC &12
BE08		133 018	85 12	STA &12
BE0A		144 002	90 02	BCC 2 --> &BE0E
BE0C		230 019	E6 13	INC &13
BE0E		160 001	A0 01	LDY#&01
BE10	`	096	60	RTS
BE11		032 207 190	20 CF BE	JSR &BECF Print following text as a warning message & jump to address specified
BE14	Bad program	013 066 097 100 032 112 114 111 103 114 097 109 013	0D 42 61 64 20 70 72 6F 67 72 61 6D 0D	EQUS chr$(13) + "Bad program" + chr$(13)
BE21		234	EA	NOP
BE22	L	076 134 143	4C 86 8F	JMP &8F86 Read & execute command line input

Output text warning message and jump to the address specified

BECF	h	104	68	PLA
BED0	7	133 055	85 37	STA &37
BED2	h	104	68	PLA
BED3	8	133 056	85 38	STA &38
BED5		128 003	80 03	BRA 3 --> &BEDA
BED7		032 227 255	20 E3 FF	JSR &FFE3 OSASCI (OSWRCH - or OSNEWL (if A = #&0D))
BEDA		032 169 141	20 A9 8D	JSR &8DA9 Increment and read character at (&37, &38) pointer
BEDD		016 248	10 F8	BPL -8 --> &BED7

BEDF l7 108 055 000 6C 37 00 JMP (&0037)

BASIC IV ROM Routines

Page Last Altered: undefined

9C4B Display Current Line Number on screen [for TRACE]

Submitted by Steve Fewell

Entry criteria: The IWA contains the current line number.

Description:

This routine is called if TRACE is on.
Check whether the Line number in the IWA (&2A-&2B) is greater than the maximum TRACE line (&21-&22), if it is then exit.

Otherwise output the '[' (open square bracket) character, which is the Start of TRACE Line Number identifier character.
call routine &A081 to output the Line Number (in the IWA) to the screen.
Output the ']' (close square bracket) character, which is the End of TRACE Line Number identifier character.

Call routine &BD92 to output a space character and exit.

Disassembly for the Display Current Line Number on screen [TRACE] routine

9C4B	*	165 042	A5 2A	LDA &2A
9C4D	!	197 033	C5 21	CMP &21
9C4F	+	165 043	A5 2B	LDA &2B
9C51	"	229 034	E5 22	SBC &22
9C53		176 172	B0 AC	BCS -84 --> &9C01 [RTS]
9C55	[	169 091	A9 5B	LDA#&5B
9C57		032 152 189	20 98 BD	JSR &BD98 Output character in A
9C5A		032 129 160	20 81 A0	JSR &A081 Print Line Number on screen
9C5D	]	169 093	A9 5D	LDA#&5D
9C5F		032 152 189	20 98 BD	JSR &BD98 Output character in A
9C62	L	076 146 189	4C 92 BD	JMP &BD92 Output a space character

---

BASIC IV ROM Routines

Page Last Altered: undefined

A085 Print Line Number on screen

Submitted by Steve Fewell

Routine:Print Line Number
Name: Print Line number (&2A-&2B) on the screen
Starting Address: &A085
Entry criteria: &2A and &2B (the IWA) contain an untokenised Line Number.

Description:

This routine uses bytes stored at &8021-&802A, which contain the five 16-bit values: 1, 10, 100, 1000 and 10000.
These values are used to print Line Numbers (which are a maximum of 5 decimal digits wide).
The tens conversion table (&8021-&802A) consists of the following:

High byte Location	Value	Low Byte Location	Value	Description
8021	&00	8026	&01	The high and low byte of the value &0001 (1 in decimal)
8022	&00	8027	&0A	The high and low byte of the value &000A (10 in decimal)
8023	&00	8028	&64	The high and low byte of the value &0064 (100 in decimal)
8024	&03	8029	&E8	The high and low byte of the value &03E8 (1000 in decimal)
8025	&27	802A	&10	The high and low byte of the value &2710 (10000 in decimal)

Firstly, this routine sets the number of bytes in the Print output field (location &14).
If this routine is entered at location &A085 (i.e. from LIST) then the Print output field will be set to 5-digits wide - so that all Line Numbers are positioned to 5 characters wide & therefore line up, as the following example shows:

```
    1
    5
   10
   70
  110
 2000
 2010
11100
```

If this routine is entered at location &A081 (i.e. from TRACE line printout) then the Print output field will be set to 0 (no fixed width) - so that all Line Numbers are positioned to 0 characters wide (and will take up as many characters as

needed), the following is an example of this output width, using a selection of different Line Numbers:

```
1
5
10
70
110
2000
2010
11100
```

Locations &3F-&43 are used to store the units/tens values (digit counts) as follows:
- Location &3F is the number of 1's units in the Line Number (i.e. the value of the digit in the unit position).
- Location &40 is the number of 10's units in the Line Number.
- Location &41 is the number of 100's units in the Line Number.
- Location &42 is the number of 1000's units in the Line Number.
- Location &43 is the number of 10000's units in the Line Number.

Set X to #&04 (as the number of values to process is 5). The largest value (10000) will be processed first, followed by the second largest (1000), and then the next largest (100), and then 10 and finally 1.

[&A08B] Zero the digit count for the current search value. The search value is the value pointed to by the value in X. I.e. when X is 4, the search value's digit count store is location &3F + #&04 = &43, the search value MSB byte is located at location &8021 + #&04 = &8025, and the search value LSB byte is located at location &8026 + #&04 = &802A.
E.g. So, when X is 3 we will search for the value 1000, and location &42 will be zeroed at the start of the search; or when X is 1 we will search for the value 10, and location &40 will be zeroed at the start of the search.

Set the carry flag to begin a subtraction. Note: the carry flag is not reset during subtractions of the same Search Value, as the carry is carried over between subtractions of the same Search Value (where X is unchanged).
[&A08E] Subtract the current Search Number low byte (&8026 + X) from the low byte of the Line Number in the IWA (&2A). Store the result in Y.
Subtract the current Search Number high byte (&8021 + X) from the high byte of the Line Number in the IWA (&2B). Store the result in A.
If the subtraction did not underflow (carry is set), then the search value was sucessfully subtracted; so store the new Line Number value (after the subtraction -> Y = low byte, A = high byte) in the IWA (locatons &2A, &2B respecively), increment the Digit Counter (&3F + X) and jump back to &A08E to try another subtraction of the Search value.

Otherwise [&A0A3], the Search Value could not be subtracted, so we need to move on to the next (lower) search value).
Decrement X (to move down to the next lower Search value).
If X is positive (>=0) then jump back to &A08B to check whether we can subject this Search value from the IWA value.

Otherwise, the search has been completed, and locations &3F-&43 should now contain the separate digits of the line Number - ready to print.

Find the first non-zero digit Set X to 5 (possible number of bytes to output).
[&A0A8] decrement X (and if X is zero then jump to the output routine - &A0AF) and check whether the Digit Counter value at location &3F + X is zero. if it is then jump back to &A0A8 to find the first non-zero digit.

Output the Line Number value to the screen Now X points to the first non-zero byte (or the first byte if the Line Number is zero), where X=4 points to the digit at
location &43 and X=0 points to the digit at location &3F.
Store X in location &37 (this is a pointer to the first digit required to be output).

Print any required leading spaces, as follows:

If the Output field width value (location &14) is not zero, then subtract &37 (X) from the &14 value, the result is the number of spaces required to be output on the line before the first non-zero digit is output.
If the number of spaces required is not zero then set X to the number of spaces required and call routine &BDBF output X number of spaces.

Set X to the first digit to output pointer (from location &37).
Print each digit of the line number to the screen (starting at location &3F + X and finishing at location &3F, reducing X by 1 after each digit has been printed.
To print each digit, the decimal value is ORA-ed with #&30 to add the ASCII offset value (64).
Exit after all required digits have been output.

Disassembly for the Print Line Number on screen routine

A081		169 000	A9 00	LDA#&00
A083		128 002	80 02	BRA 2 --> &A087
A085		169 005	A9 05	LDA#&05
A087		133 020	85 14	STA &14
A089		162 004	A2 04	LDX#&04
A08B	t?	116 063	74 3F	STZ &3F,X
A08D	8	056	38	SEC
A08E	*	165 042	A5 2A	LDA &2A
A090	&	253 038 128	FD 26 80	SBC &8026,X
A093		168	A8	TAY
A094	+	165 043	A5 2B	LDA &2B
A096	!	253 033 128	FD 21 80	SBC &8021,X
A099		144 008	90 08	BCC 8 --> &A0A3
A09B	+	133 043	85 2B	STA &2B
A09D	*	132 042	84 2A	STY &2A
A09F	?	246 063	F6 3F	INC &3F,X
A0A1		128 235	80 EB	BRA -21 --> &A08E
A0A3		202	CA	DEX
A0A4		016 229	10 E5	BPL -27 --> &A08B
A0A6		162 005	A2 05	LDX#&05
A0A8		202	CA	DEX
A0A9		240 004	F0 04	BEQ 4 --> &A0AF
A0AB	?	181 063	B5 3F	LDA &3F,X
A0AD		240 249	F0 F9	BEQ -7 --> &A0A8
A0AF	7	134 055	86 37	STX &37
A0B1		165 020	A5 14	LDA &14
A0B3		240 010	F0 0A	BEQ 10 --> &A0BF
A0B5	7	229 055	E5 37	SBC &37
A0B7		240 006	F0 06	BEQ 6 --> &A0BF
A0B9		170	AA	TAX
A0BA		032 191 189	20 BF BD	JSR &BDBF Output X number of Spaces
A0BD	7	166 055	A6 37	LDX &37
A0BF	?	181 063	B5 3F	LDA &3F,X

A0C1	0	009 048	09 30	ORA#&30
A0C3		032 148 189	20 94 BD	JSR &BD94 Output character to the screen
A0C6		202	CA	DEX
A0C7		016 246	10 F6	BPL -10 --> &A0BF
A0C9	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8920 '[' Begin Assembly

Submitted by Steve Fewell

Description:

Get the next non-space character.
If the character is ']' (end assembly character) then [&891A] Decrement A (to #&FF), set the OPT flag (location &28) to A (#&FF) and jump to &900B to process the next BASIC language statement.

Call &9BBC to add Y (the BASIC Text pointer A offset) to the BASIC text pointer address (&0A-&0B) and reset the PTR A offset to 1; and then check whether an ESCAPE keypress condition has occurred, and process it if it has. Decrement the PTR A offset to 0.

Call routine &89EB to analyse, validate and assemble the current Assembly Statement (pointed to by BASIC text pointer A).

Decrement the PTR A offset so that BASIC text pointer A points to the Statement terminator character.
If bit 0 of the OPT flag (&28) is clear then no listing is required, so skip the listing routine and jump to &89AE.

[&8936] Produce a listing for the Assembled statement
The line listing will have the following layout:
EXEC B1 B2 B3 LABEL---- TEXT

Where:
* EXEC is the execution address (4 hex digits)
* B1 is the first 6502 Machine Code byte (usually the Instruction Opcode (unless an EQU-command is used)) (2 hex digits) (or 2 spaces if the statement doesn't assemble into any actual Machine Code bytes).
* B2 is the second 6502 Machine Code byte (or 2 spaces if no second byte is present) (2 hex digits)
* B3 is the third 6502 Machine Code byte (or 2 spaces if no third byte is present) (2 hex digits)
* LABEL is any label (e.g. '.xxx') that preceeded the assembly statement (minimum of 9 characters wide - right padded with spaces, but the LABEL field will be longer if the label name is more than 9 characters wide).

* TEXT is the actual unassembled text (e.g. 'LDA#SIN(2.345)') (this field has no width limit).

Note that the EQUS and EQUD assembly mnemonics may use more than 3 bytes, and as such, the bytes will be listed in groups of 3, each group being displayed on a new line. The execution address will only be output on the first line. The label and text will be shown on the last line.

Set location &3F to the value of COUNT (&1E) + 5. This stores what the position on the current output line would be after the execution address (below) has been printed.

The (EXEC) Execution address (&37-&38) is output as follows:
* Set A to the Execution address MSB (from location &38) and call routine &BD6C to output the 2-digit value (in A).
* Set A to the Execution address LSB (from location &37) and call routine &BD8F to output the 2-digit value (in A) followed by a space.

Set X to #&FC.
Set location &38 to the length of the Assembly Mnemonic (in number of bytes); as follows:
* Set &38 to the value of &39 (number of bytes used by Assembly Statement (which is usually 1, 2, 3, 4 or &FF))
* If &39 is a negaive value then a the Mnemonic was a String value ('EQUS'), so set the length (&38) to the SWA length (&36).

If &38 is not 0 then the Assembly statement contains some bytes (B1, B2, B3, etc...), which are displayed as follows:
Set Y (physical location byte number) to 0.
[&8954] Increment X
If X has reached 0 (i.e. after 3 bytes have been displayed) then output a new line followed by &3F number of spaces
(This positions the position on the new line to line up with the first byte (after address) of the previous line and reset X to #&FD (so that another 3 bytes can be output).
Load the next byte located at the Physical location [(&3A-&3B) + Y] and call routine &BD8F to output the 2-digit hex value of the byte (in A) followed by a space
Increment Y to point to the next byte
Decrement &38 (bytes to output)
If &38 is not 0 then jump back to &8954 to output the next byte value

[&896B] Now each hex byte of the assembled statement has been output (with 3 bytes output per line).
Set Y to X (i.e. the number of spare printable bytes (is byte is printed as 2 hex digits followed by a space) on the screen).

If Y is not &FF then we need to output some spaces, as not all 3 byte positions (B1, B2, B3) contained a value, so:
* [&896D] Increment Y
* If Y is not 0 then output 3 spaces and jump back to &896D

[&8977] Set X to #&0A (minimum length of label + 1 space).
Set A to the first byte of the Statement line (from BASIC Text pointer A (&0B,&0C)).
[Note: Y is initially zero].

If A is '.' then the label is output as follows:

* [&897F] Call routine &BD37 to output the character/token in A
* Decrement the length of the label field (X)
* If X is 0 then set X to 1 (as will must output 1 space after a label name)
* Increment Y (the current byte offset at the BASIC Text Pointer A address)
* Set A to the next byte from the BASIC Text Pointer A address (with the byte offset in Y)
* If Y is not equal &4E (the offset to the end of the label name in BASIC Text Pointer A, as stored by the assembly routine (&89EB)) then jump back to &897F to output the next character of the label name.

[&898E] Output X number of spaces (if a label was printed then X will be any used spaces in the 9-character label field, or 1 if the label exceeded 8 characters in width; otherwise X will be 10).

Decrement Y (to point to the first character after the label name - which must be a space character).
[&8992] Skip any spaces after the label name by: incrementing Y until the character at the BASIC Text pointer A location plus Y offset, is not the same as the character in A (which will be a space).

[&8997] Output the statement text as follows:
* Set A to the character at BASIC Text Pointer A plus offset (Y)

* If the character is ':' then:
  # If Y is less that the BASIC Text Pointer A offset (location &0A), i.e. the position that routine &89EB stopped processing, then we are not at the end of the statement (i.e. the statement is a statement like 'LDA#ASC(":")' which contains a ':' prior to the end of statement) then jump to &89A1 to output the character and then continue outputting the statement text.
  # Otherwise, Output a new line (line break) and jump to &89AE to Check and skip the end of the statement/program line.

* If the character is '<cr>' then output a new line and jump to &89AE to check and skip the end of the program line
* [&89A1] call routine &BD37 to output the character
* Increment Y (BASIC Text Pointer A offset)
* Jump back to &8997 to output the next character of the statement text

[&89AE] Check and skip end of statement/program line
Set Y to the BASIC Text pointer A offset value (location &0A), i.e. the position that routine &89EB stopped processing; and decrement Y (to point to the last character, which should either be ':' or '<cr>').

[&89B1] Load the next character at the BASIC Text pointer A location + the Y offset value.
If the character is not ':' or '<cr>' then jump back to &89B1 to check the nexct character. This will skip any program comment that is located between the end of the Assembly statement and the end of statement marker (note that any such comments [which do not have to be preceeded by a '\' character!] will still be output in the assembly listing, but they will not be processed - as all characters are skipped until a ':' or '<cr>' character is found).
This means that problems could occur in such cases as follows:
    'CMP#ASC("9") \ Was ASC(":")'
which will not work, as BASIC will assume that the ':' is an end of statement marker and start processing '")' as the next assembly statement - resulting in a 'Syntax error'.

Call routine &9BA8 to check for the end of statement (and add Y to the BASIC Text Pointer A address).
If the character pointed to by BASIC text pointer A (i.e. the last character of the statement) is ':' then jump to &8920 to assemble the next statement on the program line.

Otherwise, the character is '<cr>', so check the BASIC Text pointer A MSB address (&0C). If the MSB address of the statement text is &07, then we are not currently executing a program, instead we are executing a command line command, so jump to &8F86 to return to the command line prompt.
Otherwise, we are executing a program, so call routine &9BDE to Check the next program line, returning to the command line prompt if no more lines were found, and to skip and display (if TRACE is on) the Line Number.
Jump back to &8920 to assemble the statement on this new program line, or to exit if a ']' character is found.

Disassembly for the '[' Begin Assembly routine

891A	:	058	3A	DEC A
891B	(	133 040	85 28	STA &28
891D	L	076 011 144	4C 0B 90	JMP &900B Process next BASIC program statement
8920		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space character pointed to by Ptr A
8923	I]	073 093	49 5D	EOR#&5D
8925		240 243	F0 F3	BEQ -13 --> &891A
8927		032 188 155	20 BC 9B	JSR &9BBC Add Y to Text pointer A & then reset offset to 1) & check for Escape condition
892A		198 010	C6 0A	DEC &0A
892C		032 235 137	20 EB 89	JSR &89EB Assemble Assembly instruction/statement
892F		198 010	C6 0A	DEC &0A
8931	(	165 040	A5 28	LDA &28
8933	J	074	4A	LSR A
8934	x	144 120	90 78	BCC 120 --> &89AE
8936		165 030	A5 1E	LDA &1E
8938	i	105 004	69 04	ADC#&04
893A	?	133 063	85 3F	STA &3F
893C	8	165 056	A5 38	LDA &38
893E	l	032 108 189	20 6C BD	JSR &BD6C Output the 2-digit Hexadecimal number stored in A
8941	7	165 055	A5 37	LDA &37
8943		032 143 189	20 8F BD	JSR &BD8F Output the 2-digit Hexadecimal number stored in A followed by a space
8946		162 252	A2 FC	LDX#&FC
8948	9	164 057	A4 39	LDY &39
894A		016 002	10 02	BPL 2 --> &894E
894C	6	164 054	A4 36	LDY &36

```
894E  8  132 056      84 38     STY &38
8950     240 025      F0 19     BEQ 25 --> &896B
8952     160 000      A0 00     LDY#&00
8954     232          E8        INX
8955     208 010      D0 0A     BNE 10 --> &8961
8957     032 146 186  20 92 BA  JSR &BA92 Output a new line
895A  ?  166 063      A6 3F     LDX &3F
895C     032 191 189  20 BF BD  JSR &BDBF Output X number of Spaces
895F     162 253      A2 FD     LDX#&FD
8961  :  177 058      B1 3A     LDA (&3A),Y
8963     032 143 189  20 8F BD  JSR &BD8F Output the 2-digit Hexadecimal number stored in A followed by a space
8966     200          C8        INY
8967  8  198 056      C6 38     DEC &38
8969     208 233      D0 E9     BNE -23 --> &8954
896B     138          8A        TXA
896C     168          A8        TAY
896D     200          C8        INY
896E     240 007      F0 07     BEQ 7 --> &8977
8970     162 003      A2 03     LDX#&03
8972     032 191 189  20 BF BD  JSR &BDBF Output X number of Spaces
8975     128 246      80 F6     BRA -10 --> &896D
8977     162 010      A2 0A     LDX#&0A
8979     178 011      B2 0B     LDA (&0B)
897B  .  201 046      C9 2E     CMP#&2E
897D     208 015      D0 0F     BNE 15 --> &898E
897F  7  032 055 189  20 37 BD  JSR &BD37 Output Character/Token on screen
8982     202          CA        DEX
8983     208 002      D0 02     BNE 2 --> &8987
8985     162 001      A2 01     LDX#&01
8987     200          C8        INY
8988     177 011      B1 0B     LDA (&0B),Y
898A  N  196 078      C4 4E     CPY &4E
898C     208 241      D0 F1     BNE -15 --> &897F
898E     032 191 189  20 BF BD  JSR &BDBF Output X number of Spaces
8991     136          88        DEY
```

8992		200	C8	INY
8993		209 011	D1 0B	CMP (&0B),Y
8995		240 251	F0 FB	BEQ -5 --> &8992
8997		177 011	B1 0B	LDA (&0B),Y
8999	:	201 058	C9 3A	CMP#&3A
899B		240 010	F0 0A	BEQ 10 --> &89A7
899D		201 013	C9 0D	CMP#&0D
899F		240 010	F0 0A	BEQ 10 --> &89AB
89A1	7	032 055 189	20 37 BD	JSR &BD37 Output Character/Token on screen
89A4		200	C8	INY
89A5		128 240	80 F0	BRA -16 --> &8997
89A7		196 010	C4 0A	CPY &0A
89A9		144 246	90 F6	BCC -10 --> &89A1
89AB		032 146 186	20 92 BA	JSR &BA92 Output a new line
89AE		164 010	A4 0A	LDY &0A
89B0		136	88	DEY
89B1		200	C8	INY
89B2		177 011	B1 0B	LDA (&0B),Y
89B4	:	201 058	C9 3A	CMP#&3A
89B6		240 004	F0 04	BEQ 4 --> &89BC
89B8		201 013	C9 0D	CMP#&0D
89BA		208 245	D0 F5	BNE -11 --> &89B1
89BC		032 168 155	20 A8 9B	JSR &9BA8 Check end of Statement
89BF		178 011	B2 0B	LDA (&0B)
89C1	:	201 058	C9 3A	CMP#&3A
89C3		240 012	F0 0C	BEQ 12 --> &89D1
89C5		165 012	A5 0C	LDA &0C
89C7		201 007	C9 07	CMP#&07
89C9		208 003	D0 03	BNE 3 --> &89CE
89CB	L	076 134 143	4C 86 8F	JMP &8F86 Read & execute command line input
89CE		032 222 155	20 DE 9B	JSR &9BDE Skip Program Line Number
89D1	L	076 032 137	4C 20 89	JMP &8920 Assemble next assembly statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

89EB Assemble Assembly instruction/statement

Submitted by Steve Fewell

Description:

[89EB] Get the first non-space character from BASIC Text Pointer A.
Set Y and location &3D (the Mnemonic Code LSB location) to 0.
If the next character is ':', then jump to &8A5E to store current instruction (if any) and exit.
If the next character is '<cr>' [carriage return], then jump to &8A5E to store current instruction (if any) and exit.
Note: No need to check for 'ELSE', as 'ELSE' is invalid in assembly mode.

If next character is '\' then jump to &8A5E (Store current instruction (if any) and skip comment).

If the next characer is '.' then we need to create a label, as follows:

* Call routine &98AE to evaluate the variable name (and create it if the variable doesn't already exist and isn't a direct memory access.
* If the zero flag is set on return from &98AE, then a variable was not specified, so generate 'Syntax error'.
* If the carry flag is set on return from &98AE, then the variable was a String-variable, so generate 'Syntax error' as an address value cannot be stored in a string variable.
* Push variable address and type details (&2A-&2B) to the stack.
* Set the IWA value to P% (&0440-&0441).
* Store variable type (#&40 - Integer) in location &27.
* Call routine &B32B to set the Numberic variable (details on Stack) to the IWA Integer value.
* Call routine &9275 to update the BASIC Text Pointer A offset to equal the Text Pointer B offset value.
* Store the BASIC Text Pointer A offset value in location &4E.
* return to &89EB to skip spaces, and check for end of statement.

Decrement the BASIC Text Pointer A offset value (as the character we got last was not recognised as having a special function).

[&8A04] Encode and validate the 3-character Mnemonic value
Set X to 3 (as we will check the next 3 characters to see if they match an assembly Mnemonic Code).
[&8A06] Get next character from PTR A.

If the character is negative, then we have found a BASIC token, check the token for a Mnemonic code, as follows [&8A38]:

* If token is &80 ('AND'), then set X (Mnemonic position in Mnemonic lookup table) to #&29 and jump to &8A5E to process the assembly Mnemonic.
* If token is &82 ('EOR'), then set X (Mnemonic position in Mnemonic lookup table) to #&2A and jump to &8A5E to process the assembly Mnemonic.
* If token is &84 ('OR'), then check next character. If next character is 'A' or 'a' (zeroing Bit 5 of the ASCII value removes the upper-lower case distinction) then Set X (Mnemonic position in Mnemonic lookup table) to &2B, & jump to &8A5E to process the assembly Mnemonic. Otherwise (next character is not 'A' or 'a') issue a 'Syntax error', as the Assembly Mnemonic 'ORA' is not present, and we do not have a valid Assembly Mnemonic code.
* Otherwise (as the token isn't 'AND', 'OR' or 'EOR'), the token is not a valid Assembly Mnemonic, so generate a 'Syntax error'.

If the character is a space (' '), then jump to &8A22 to check the mnemonic code found so far (as space terminates the Mnemonic code).

Now, before we test the Mnemonic, we need to compact each character of the Mnemonic (usually 3 characters in total) into a 2-byte code. This has 2 main advantages: (1) it saves space, and (2) it strips off the case bit from the ASCII character, so that Mnemonics are recognised irrespective of whether they appear in upper or lower case (or a mixture of both).
Set Y to 5 (as we will pack each character into 5-bits).
Multiply the current ASCII character by 8 (by shifting all bits left 3 places, and loosing the top 3 bits).
This leaves us with the bottom 5 bits of the ASCII code. The character has now lost it's case distinction (bit 5), so now 'A' and 'a' result in the same 5-bit value. Additionally, the #&40 (64) offset is removed, so that 'A' is now 1, 'B' is 2, etc...
Now, move the top 5-bits of the character (the bits that are left after the earlier shift), into the bottom bit of the 2-byte location &3D-&3E. &3D-&3E will (after all 3 Mnemonic characters have been processed in this way) contain the 2-byte Mnemonic Code, which will uniquely identify the Assembly Mnemonic.
Decrement X (number of characters to process), and, if X is not 0, jump back to &8A06 to process the next character.

[&8A22] Check the Mnemonic code:
Now that a maximum of 3 characters have been encoded, and &3D-&3E contain 15 bits from the 3-character Assembly Mnemonic. the coded Mnemonic (&3D-&3E) needs to be checked against the Mnemonic code lookup tables located between address &884D and &88D6.
The Encoded Mnemonic LSB value (&3D) is looked up in the Mnemonic LSB lookup table (&884D-&8891) (looking at the last value in the table first, then moving down towards the first value).
When a matching LSB Mnemonic value is found, the Mnemonic MSB value is then checked against the corresponding entry in the Mnemonic MSB table (&8893-&88D6). If both codes matche then we have a valid Mnemonic Code, so jump to &8A5E to process the Mnemonic statement.

Otherwise, continue searching with the next Mnemonic LSB code in the Mnemonic LSB table (&884D-&8891).
If the end of the Assembly Mnemonic LSB table (&884D-&8891) is reached, and the encoded Mnemonic 2-byte code was not matched against the entries in the table, then an invalid Mnemonic was specified, so issue a 'Syntax error'.

[&8A53] A valid Assembly Mnemonic has been found! Now, start to process it
Now, X points to the position of the Assembly Mnemonic in the lookup tables.
Store the default opcode (from the Default Opcode lookup table, located at &88D7-&891B at position X) in location &29.
Set Y = 1 (default length of Assembly Instruction is 1-byte (i.e. a single-byte Mnemonic code with no parameters)).

The Mnemonic codes in the Mnemonic lookup tables are listed in an order that enables them to be processed more easily.
I.e. the first &1F (31) Mnemonics are single-byte Instructions.

1) Check if we have a single-byte Mnemonic
If X (the Mnemonic position) is less than #&20 then we have a single-byte Mnemonic (i.e. One of the following:
BRK, CLC, CLD, CLI, CLV, DEX, DEY, INX, INY, NOP, PHA, PHP, PLA, PLP, RTI, RTS, SEC, SED, SEI, TAX, TAY, TSX, TXA, TXS, TYA, DEA, INA, PHY, PHX, PLY, or PLX).
Single byte instructions have no parameters, and no indexing modes, so the Default Opcode (&29) is the Opcode that we need to use, so jump to &8A5E to store the assembly instruction.

2) Check if we have a branch-statement Mnemonic

If X (Mnemonic position) is < #&29 then the Mnemonic is a Branch statement (i.e. one of: BCC, BCS, BEQ, BMI, BNE, BPL, BVC, BVS or BRA), so process as follows:

* Get the Integer result of the expression at BASIC Text pointer A (or 'Type mismatch' error if a String value was found).
* The Integer result will be a 16-bit address value (in &2A-&2B).
* Subtract the current physical location (P% + 1) from the address specified (by the result value).
* (Now, Y is the LSB difference, and A is the MSB difference in the address values).
* If Y is 0 then subtract 1 from A and decrement Y (difference is &100, treated as &FFFF (-1)).
* If A is 0 then: * Check Y; if Y is negative (top bit set) then issue 'Out of range' error.
  * Otherwise, goto &8ADF with Y = number of Bytes to branch, as Y is within the valid range of 1 to 127.
* If A is &FF and Y is negative, then we are branching backwards (to a previous instruction), and the branch is within the required ramge of -1 to -128, so goto &8ADF with Y = number of bytes to branch.
* Otherwise, if OPT flag bit 2 is not set (errors are not reported), then skip the out of range error, and set Y (the number of bytes to branch) to 0 (as a placemaker, until we have the actual Branch value), and goto &8ADF.
* [Note: OPT flag bit 2 is not set for OPT codes 0, 1, 4 & 5 (don't report errors); and it is set for 2, 3, 6 & 7 (report errors))]
* Otherwise, if OPT flag bit 2 is set then issue an 'Out of range' error.
* [8ADF]: Now we have a valid branch, so Store Y (bytes to branch) in &2A, set Y to 2 (length of instruction), and jump to &8A5E to store the assembly instruction.

3) Process Mnemonics AND, EOR, ORA, ADC, CMP, LDA, SBC (and also used by STA, see later)
If X (Mnemonic position) is < #&30 then the Mnemonic is one of the following: AND, EOR, ORA, ADC, CMP, LDA or SBC, so process as follows:

* If the next non-space character is '#' then we have an immediate value indexing mode. Process as follows:
  * Add 8 to the defalt opcode (in location &29)
  * Get the Integer result of the expression at BASIC Text Pointer A (Type mismatch error if a String value was found).

* Issue a 'Byte' error if the Integer result is more than 255 (#&FF).
* Set Y to 2 (length of instruction is 2 bytes) and goto &8A5E to store the instruction.
* This handles the following Mnemonic instructions:
* AND#nn (opcode &29).
* EOR#nn (opcode &49).
* ORA#nn (opcode &09).
* ADC#nn (opcode &69).
* CMP#nn (opcode &C9).
* LDA#nn (opcode &A9).
* SBC#nn (opcode &E9).
* [Note: STA does not have an immediate mode, so for STA Mnemonic, this routine is entered at &8B07].

* [8B07] If the next non-space character is not '(' then we have an absolute or zero page addressing mode. So:
* [8B44] Get the Integer result of the expression at BASIC Text Pointer A (Type mismatch error if a String value was found).
* If the next non-space character is ',' [comma] then process index-offset mode as follows:
> Add 16 to default opcode value (&29).
> If next character is 'Y' then add 8 to the Opcode, set Y (length of instruction) to 3, & goto &8A5E to store the instruction.
> -- This handles the following Mnemonic instructions:
> AND abs,Y (opcode &39).
> EOR abs,Y (opcode &59).
> ORA abs,Y (opcode &19).
> ADC abs,Y (opcode &79).
> CMP abs,Y (opcode &D9).
> LDA abs,Y (opcode &B9).
> SBC abs,Y (opcode &F9).
> STA abs,Y (opcode &99).
> Otherwise, add 4 to the Opcode.
> If the Integer value is a 16-bit address (more than 255) then add 8 to Opcode, set Y to 3 (instruction is 3 bytes long) and goto &8A5E to store the instruction.
> -- This handles the following Mnemonic instructions:
> AND abs (opcode &2D).
> EOR abs (opcode &4D).
> ORA abs (opcode &0D).
> ADC abs (opcode &6D).
> CMP abs (opcode &CD).
> LDA abs (opcode &AD).
> SBC abs (opcode &ED).
> STA abs (opcode &8D).
> AND abs,X (opcode &3D).
> EOR abs,X (opcode &5D).
> ORA abs,X (opcode &1D).
> ADC abs,X (opcode &7D).
> CMP abs,X (opcode &DD).
> LDA abs,X (opcode &BD).

> SBC abs,X (opcode &FD).
> STA abs,X (opcode &9D).
> Otherwise, we have an 8-bit address, so set Y to 2 (length of instruction is 2 bytes),
and goto &8A5E to store the instruction.
> -- This handles the following Mnemonic instructions:
> AND zp (opcode &25).
> EOR zp (opcode &45).
> ORA zp (opcode &05).
> ADC zp (opcode &65).
> CMP zp (opcode &C5).
> LDA zp (opcode &A5).
> SBC zp (opcode &E5).
> STA zp (opcode &85).
> AND zp,X (opcode &35).
> EOR zp,X (opcode &55).
> ORA zp,X (opcode &15).
> ADC zp,X (opcode &75).
> CMP zp,X (opcode &D5).
> LDA zp,X (opcode &B5).
> SBC zp,X (opcode &F5).
> STA zp,X (opcode &95).

* [8B0B] If the next non-space character is '(' then we have an indirect addressing mode. So:
* Get the Integer result of the expression at BASIC Text Pointer A (Type mismatch error if a String value was found).
* If the next character is ")" then we do not have an indirectly indexed instruction, so process as follows:
> Add 16 to default opcode value (&29).
> If next character is not ',' [comma] then Add 1 to the opcode and issue 'Byte' error
if the integer value is > 255; otherwise, set Y to 2 (length of instruction) and goto &8A5E to store the instruction.
> -- This handles the following Mnemonic instructions:
> AND(zp) (opcode &32).
> EOR(zp) (opcode &52).
> ORA(zp) (opcode &12).
> ADC(zp) (opcode &72).
> CMP(zp) (opcode &D2).
> LDA(zp) (opcode &B2).
> SBC(zp) (opcode &F2).
> STA(zp) (opcode &92).
> If next character is ',' [comma] then issue 'Index' error if the next non-space character
is not 'Y' or 'y', or 'Byte' error if the Integer result is > 255; otherwise,
set Y to 2 (length of instruction) and goto &8A5E to store the instruction.
> -- This handles the following Mnemonic instructions:
> AND(zp),Y (opcode &31).
> EOR(zp),Y (opcode &51).
> ORA(zp),Y (opcode &11).
> ADC(zp),Y (opcode &71).

> CMP(zp),Y (opcode &D1).
> LDA(zp),Y (opcode &B1).
> SBC(zp),Y (opcode &F1).
> STA(zp),Y (opcode &91).
> .
> .
> .

* Otherwise (next character is not ')') issue 'Index' error if a ',' [Comma] is not the next non-space character.
* Issue 'Index' error if the next non-space character after the ',' is not 'X' or 'x'.
* Issue 'Index' error if the next non-space character after the 'X', or 'x', is not ')'.
* Issue 'Byte' error if the Integer value is more than 255.
* [Note: no opcode changes for this indexing mode, as the opcode used is the default Opcode for the Mnemonic].
* Set Y to 2 (length of instruction is 2 bytes) and goto &8A5E to store the instruction.
* This handles the following Mnemonic instructions:
* AND(zp,X) (opcode &21).
* EOR(zp,X) (opcode &41).
* ORA(zp,X) (opcode &01).
* ADC(zp,X) (opcode &61).
* CMP(zp,X) (opcode &C1).
* LDA(zp,X) (opcode &A1).
* SBC(zp,X) (opcode &E1).
* STA(zp,X) (opcode &81).

4) Process Mnemonic STA

If X (Mnemonic position) is < #&41 then the Mnemonic is STA. STA is handled in the same way as AND, EOR, ORA, ADC, CMP, LDA and SBC, except that it doesn't have an immediate '#' addressing mode.
so process by jumping to &8B07 (above), which skips the Immediate addressing mode and continues to process the other addressing modes.

5) Process Mnemonics ASL, LSR, ROL, ROR, DEC and INC (and also used by BIT, see later)

[&8B67] If X (Mnemonic position) is < #&36 then the Mnemonic is one of the following: ASL, LSR, ROL, ROR, DEC or INC, so process as follows:

* If the next non-space character is 'A' or 'a' then the Mnemonic could be an Accumulator operator, so:
    * Check the next character after the 'A' or 'a'. If the next character is not valid variable name character ('A'-'Z','a'-'z','0'-'9','_' or '£') then the A refers to the Accumulator, so:
        * If X (Mnemonic position) < #&34 (ASL, LSR, ROL or ROR) then add 4 to the opcode (&29).
        * Otherwise, if X = #&34 (DEC) then set the opcode to #&3A.
        * Otherwise, if X = #&35 (INC) then set the opcode to #&1A.
        * Set Y to 1 (length of instruction) and goto &8A5E to store the instruction.
        -- This handles the following Mnemonic instructions:
            ASL A (opcode &0A).
            LSR A (opcode &4A).
            ROL A (opcode &2A).

ROR A (opcode &6A).
DEC A (opcode &3A).
INC A (opcode &1A).
[Note: BIT does not have an accumulator mode, so for BIT Mnemonic, this routine is entered at &8B74].

* Otherwise:
  * [&8B74] Get the Integer result of the expression at BASIC Text pointer A.
  * Get the next non-space character at BASIC Text pointer A.
  * If the next character is "," [comma] then :
    * Add 16 to the Opcode (&29).
    * If the next non-space character is not "X" or "x" then 'Index' error
  * [8B61] If 16-bit Integer then add 8 to Opcode (&29), Set Y = 3 and Store instruction.
  * Otherwise, Set Y to 2 and Store instruction.
  * This handles the following Mnemonic instructions:
    ASL abs (opcode &0E).
    LSR abs (opcode &4E).
    ROL abs (opcode &2E).
    ROR abs (opcode &6E).
    DEC abs (opcode &CE).
    INC abs (opcode &EE).
    BIT abs (opcode &2C).
    ASL abs,X (opcode &1E).
    LSR abs,X (opcode &5E).
    ROL abs,X (opcode &3E).
    ROR abs,X (opcode &7E).
    DEC abs,X (opcode &DE).
    INC abs,X (opcode &FE).
    BIT abs,X (opcode &3C).
    ASL zp (opcode &06).
    LSR zp (opcode &46).
    ROL zp (opcode &26).
    ROR zp (opcode &66).
    DEC zp (opcode &C6).
    INC zp (opcode &E6).
    BIT zp (opcode &24).
    ASL zp,X (opcode &16).
    LSR zp,X (opcode &56).
    ROL zp,X (opcode &36).
    ROR zp,X (opcode &76).
    DEC zp,X (opcode &D6).
    INC zp,X (opcode &F6).
    BIT zp,X (opcode &34).

6) Process Mnemonics CLR and STZ

If X (Mnemonic position) is < #&38 then the Mnemonic is one of the following: CLR or STZ.

Both of these Mnemonics represent the same Opcode (default Opcode value=&9C). Process as follows:

* Get the Integer result of the expression at BASIC Text pointer A (or 'Type mismatch' error if a String value was found).
* If Integer is an 8-bit address then set X to #&0F, Opcode (&29) to #&64 and Y to 2.
* If Integer is a 16-bit address then set X to #&01 and Y to 3.
* [8BB7] Store Y to the Stack.
* If the next non-space characetr is not "," then pop Y and Store Intruction.
* If the next non-space character is not 'X' or 'x', then issue a 'Index' error.
* Otherwise, add (1 + X) to the Opcode value (&29), Pop Y from the stack (length of instruction), and Store instruction.
* This handles the following Mnemonic instructions:
  STZ (or CLR) zp (opcode &64).
  STZ (or CLR) zp,X (opcode &74).
  STZ (or CLR) abs (opcode &9C).
  STZ (or CLR) abs,X (opcode &9E).

7) Process Mnemonics CPX, CPY, TSB or TRB

If X (Mnemonic position) is < #&3C then the Mnemonic is one of the following: CPX, CPY, TSB or TRB,
so process as follows:

* If the Mnemonic is CPX or CPY then check for Immediaate mode (TSB and TRB do not have an Immediate addressing mode):
  If the next non-space character is '#' then [8BE7] Get Integer value of expression, if the Integer is greater than 255 then issue 'Byte' error, otherwise set Y to 2, and Store Intruction.
  This handles the following Mnemonic instructions:
    CPX# nn (Opcode &E0)
    CPY# nn (Opcode &C0)
  Otherwise (if char not '#'), decrement BASIC Text Pointer A offset.
* [&8BD9] Get Integer result of expression from BASIC Text Pointer A (Convert Float value to Integer or issue 'Type mismatch' error if a String value was found).
* [&8B5E] Add 4 to the Opcode value (&29).
* If Integer is an 8-bit value (< 256) then (zero page addressing mode) Set Y to 2 (length is 2 bytes) & Store Instruction.
* Otherwise (Integer is an 16-bit value) (> 255), Add 8 to Opcode (&29), Set Y to 3 (length is 3 bytes) and Store Instruction.
* This handles the following Mnemonic instructions:
  CPX zp (opcode &E4).
  CPY zp (opcode &C4).
  TSB zp (opcode &04).
  TRB zp (opcode &14).
  CPX abs (opcode &EC).
  CPY abs (opcode &CC).
  TSB abs (opcode &0C).
  TRB abs (opcode &1C).

8) Process Mnemonics BIT, JSR and JMP

If X (Mnemonic position) is < #&3F then the Mnemonic is one of the following: BIT, JSR or JMP,
so process as follows:

* If the Mnemonic is BIT, then:

* [&8BDE] If the next non-space character is not '#' then goto &8B74 (in step 5 above) to process BIT in the same way as the Absolute, Absolute,X, Zero Page and Zero Page,X addressing modes for Mnemonics ASL, LSR, ROL, ROR, DEC and INC.
* Otherwise, we have BIT with Immediate mode addressing (which couldn't be handled by &8B74, as ASL, LSR, ROL, ROR, DEC and INC do not have an immediate mode. Process as follows:
* Set the Opcode (&29) to #&89 (Instruction: BIT# nn).
* [&8BF2] Get Integer result of the expression at BASIC Text Pointer A.
* If the Integer result is > 255 then issue 'Byte' error. Otherwise set Y to 2 and Store Instruction.
* This handles the following Mnemonic instruction:
BIT#nn (opcode &89).

* If the Mnemonic is JSR, then:
* [&8BFB] Get Integer value of the expression at BASIC Text Pointer A.
* Set Y to 3 (length of instruction is 3 bytes) and goto &8A5E to store the instruction.
* This handles the following Mnemonic instruction:
JSR abs (opcode &20).

* If the Mnemonic is JMP, then:
* If the next non-space character is not '(' then Decrement BASIC Text Pointer A offset and get Integer result of expression at Text Pointer A, Set Y to 3 and Store Instruction.
This handles the following Mnemonic instruction:
JMP abs (opcode &4C - the default opcode for JMP).
* Otherwise:
* Add 32 to the opcode (&29).
* Get the Integer result of expression at BASIC Text pointer A ('Type mismatch' error if String found).
* If the next non-space character is ')' then Set Y to 3 and Store Instruction.
This handles the following Mnemonic instruction:
JMP (abs) (opcode &6C).
* Otherwise:
* Issue 'Index' error if the next non-space character is not ',' [comma].
* Add 16 to the Opcode (&29).
* Issue 'Index' error if the next non-space character is not 'X' or 'x'.
* Issue 'Index' error if the next non-space character is not ')'.
* Set Y to 3 (length of instruction is 3 bytes) and goto &8A5E to store the instruction.
* This handles the following Mnemonic instruction:
JMP (abs,X) (opcode &7C).

9) Process Mnemonics LDX, LDY, STX or STY

If X (Mnemonic position) is < #&44 then the Mnemonic is one of the following: LDX, LDY, STX or STY, so process as follows:

* Load Encoded Mnemonic LSB byte in A.
* EOR the LSB Mnemonic code with #&01 (to reverse bit 0) and AND it with #&1F (to clear the top 3 bits).
(Now A contains the last letter of the Mnemonic code (minus 64).
* Store A to the Stack.

* If the Mnemonic is LDX or LDY then:
    * (LDX and LDY have Immediate and Absolute,X (or Absolute,Y) addressing modes that STX and STY do not have).
    * [&8C38] If the next non-space character is '#' then Retrieve A from the stack (as no longer needed), Get Integer result from expression at BASIC text pointer A, Set Y to 2, issue 'Byte' error if Integer is > 255; otherwise Store Instruction.
        This handles the following Mnemonic instructions:
        LDX#nn (opcode &A2).
        LDY#nn (opcode &A0).
    * Otherwise:
    * [&8C40] Decrement BASIC Text Pointer A offset & get the Integer result of the expression at Text Pointer A.
    * Retrieve A from the stack and store it in location &37.
    * If the next non-space character is ',' [comma] then:
        * Get the next non-space character and AND the value with #&1F (to remove the &64 ASCII letter offset, and also removes case sensitivity).
        * Compare the result with &37 location. If equal then issue 'Index' error, as the Index character after the comma is the same as the register operator in the Mnemonic - which is invalid.
        * Add 16 to the Opcode (&29) and continue with &8B5E, below:
    * [&8B5E] Add 4 to the opcode (&29).
    * If the Integer is > 255 (16-bit address) then add 8 to the Opcode (&29), Set Y to 3 and Store Instruction.
    * Otherwise (Integer is < 256 (8-bit zero-page address)), so set Y to 2 and Store Instruction.
    * This handles the following Mnemonic instructions:
    * LDX abs (opcode &AE).
    * LDY abs (opcode &AC).
    * LDX abs,Y (opcode &BE).
    * LDY abs,X (opcode &BC).
    * LDX zp (opcode &A6).
    * LDY zp (opcode &A4).
    * LDX zp,Y (opcode &B6).
    * LDY zp,X (opcode &B4).

* If the Mnemonic is STX or STY then:
    * (STX and STY do not have the Immediate and Absolute,X (or Absolute,Y) addressing modes that LDX and LDY have).
    * [&8C59] Get the Integer result of the expression at Text Pointer A.
    * Retrieve A from the stack and store it in location &37.
    * If the next non-space character is ',' [comma] then:
        * Get the next non-space character and AND the value with #&1F (to remove the &64 ASCII letter offset (and also removes case sensitivity)).
        * Compare the result with &37 location. If equal then issue 'Index' error, as the Index character after the comma is the same as the register operator in the Mnemonic - which is invalid.
        * Add 16 to the Opcode (&29).
        * If the Integer is > 255 then issue 'Byte' error as only zero page addressing is allowed (STX zp,Y & STY zp,X).
        * Otherwise, continue with &8B61, below:
    * [&8B61] If the Integer is > 255 (16-bit address) then add 8 to the Opcode (&29), Set Y to 3 and Store Instruction.
    * Otherwise (Integer is < 256 (8-bit zero-page address)), so set Y to 2 and Store Instruction.
    * This handles the following Mnemonic instructions:
    * STX abs (opcode &8E).

* STY abs (opcode &8C).
* STX zp (opcode &86).
* STY zp (opcode &84).
* STX zp,Y (opcode &96).
* STY zp,X (opcode &94).

10) Process Pseudo Mnemonic OPT

If X (Mnemonic position) is = #&44 then the Mnemonic is OPT so process as follows:
* Get the Integer result of the expression at Text Pointer A.
* Store LSB of the Integer result (&2A) in &28 [OPT flag location].
* Set Y to 0 (No bytes to store) and goto &8A5E to store the instruction (nothing!) & exit.

11) Process Pseudo Mnemonic EQU

If X (Mnemonic position) is = #&45 then the Mnemonic is EQU so process as follows:
* Increment BASIC Text Pointer A offset and get the next character after the EQU Mnemonic.
* AND the character code with #&DF to clear the case bit (bit 5), Now we have the Upper Case representation of the letter.
* If the next character is '**B**' then Get Integer result of the expression at BASIC Stack pointer A, Store IWA value in locations &29 - &2C. Set Y to 1 (as we will store only the first byte (&2A) of the IWA value) and goto &8A5E to store.
* If the next character is '**W**' then Get Integer result of the expression at BASIC Stack pointer A, Store IWA value in locations &29 - &2C. Set Y to 2 (as we will store 2 bytes of the IWA value) and goto &8A5E to store.
* If the next character is '**D**' then Get Integer result of the expression at BASIC Stack pointer A, Store IWA value in locations &29 - &2C. Set Y to 4 (as we will store 4 bytes of the IWA value) and goto &8A5E to store.
* If the next character is not 'S' then issue 'Syntax' error as the Instruction Line does not contain a valid Mnemonic.
* If the next character is '**S**' then Store &28 (OPT flag) to the stack (for safe keeping), Get the result of the expression at BASIC Stack pointer A, ('Type mismatch' error if not String value). Retrieve OPT Flag from the Stack. Call &9275 to set BASIC Text Pointer A Offset to BASIC Text Pointer B Offset. Set Y to #&FF (to indicate that the storage bytes are in the SWA) and goto &8A5E to store.

[&8A5E] Store Assembly Instruction

Now, &29 contains the appropriate Opcode and &2A-&2B contain any parameters, and Y contains the length (number of bytes) of the Assembly instruction.
Store Y (Length of instruction) in location &39.
Store P% LSB (&0440) in location &37.
Store P% MSB (&0441) in location &38.
X = P% MSB and A = P% LSB.
If OPT (&28) >= 4 (relocate), then X = O% MSB (&043D) and A = O% LSB (&043C).
Store A in &3A and X in &3B.
Now: &37-&38 contain the execution location (P%) and &3A-&3B contain the physical location (P%, or O% (if OPT > 3)).

If the number of bytes (Y) is 0 then exit (RTS), as there are no instruction bytes to store.
If the number of bytes is a negative value, then the storage length is determined by the length of the SWA,

as we are storing a String value (EQUS), so, copy the SWA value (&0600-&0600+&36) to the physical location (&3A-&3B), and increment P% (&0400-&0441) by 1 for each byte stored.
Otherwise, copy locations &29 to (&28 + Number of Bytes (Y)) to the physical location (&3A-&3B), and increment P% by 1 for each byte stored.
Note: Y is used as an offset pointer when copying, so no need to increment the physical location pointer, &3A-&3B.

If carry flag is set (meaning that OPT (&28) was >= 4 --> as no statements between &8A67 and &8A98 affect the carry flag), then increment O% for each byte stored.
Exit (RTS), as the assembly instruction has now been stored.

Table of Assembly instructions and Opcodes (Included here for reference)

[where: 'rel' represents a relative address (i.e. +06), 'abs' an absolute (or 16-bit) address i.e. 8005), 'zp' a zero-page (or 8-bit) address (i.e. 56) and 'nn' a literal].
[a '*' represents an Instruction that is not in the 6502 processor, and is new for the 6502C12 (BBC Master) processor].

Opcode	Instruction	Opcode	Instruction
00	BRK	80	BRA rel *
01	ORA (zp,X)	81	STA (zp,X)
02	---	82	---
03	---	83	---
04	TSB zp *	84	STY zp
05	ORA zp	85	STA zp
06	ASL zp	86	STX zp
07	---	87	---
08	PHP	88	DEY
09	ORA #nn	89	BIT #nn *
0A	ASL A	8A	TXA
0B	---	8B	---
0C	TSB abs *	8C	STY abs
0D	ORA abs	8D	STA abs
0E	ASL abs	8E	STX abs
0F	---	8F	---
10	BPL rel	90	BCC rel
11	ORA (zp),Y	91	STA (zp),Y
12	ORA (zp) *	92	STA (zp) *
13	---	93	---
14	TRB zp *	94	STY zp
15	ORA zp,X	95	STA zp,X
16	ASL zp,X	96	STX zp,Y

17	---	97	---
18	CLC	98	TYA
19	ORA abs,Y	99	STA abs,Y
1A	INC A *	9A	TXS
1B	---	9B	---
1C	TRB abs *	9C	STZ abs *
1D	ORA abs,X	9D	STA abs,X
1E	ASL abs,X	9E	STZ abs,X *
1F	---	9F	---
20	JSR abs	A0	LDY #nn
21	AND (zp,X)	A1	LDA (zp,X)
22	---	A2	LDX #nn
23	---	A3	---
24	BIT zp	A4	LDY zp
25	AND zp	A5	LDA zp
26	ROL zp	A6	LDX zp
27	---	A7	---
28	PLP	A8	TAY
29	AND #nn	A9	LDA #nn
2A	ROL A	AA	TAX
2B	---	AB	---
2C	BIT abs	AC	LDY abs
2D	AND abs	AD	LDA abs
2E	ROL abs	AE	LDX abs
2F	---	AF	---
30	BMI rel	B0	BCS rel
31	AND (zp),Y	B1	LDA (zp),Y
32	AND (zp) *	B2	LDA (zp) *
33	---	B3	---
34	BIT zp,X *	B4	LDY zp
35	AND zp,X	B5	LDA zp,X
36	ROL zp,X	B6	LDX sp,Y
37	---	B7	---
38	SEC	B8	CLV
39	AND abs,Y	B9	LDA abs,Y
3A	DEC A *	BA	TSX
3B	---	BB	---

3C	BIT abs,X *	BC	LDY abs,X
3D	ORA abs,X	BD	LDA abs,X
3E	ASL abs,X	BE	LDX abs,Y
3F	---	BF	---
40	RTI	C0	CPY #nn
41	EOR (zp,X)	C1	CMP (zp,X)
42	---	C2	---
43	---	C3	---
44	---	C4	CPY zp
45	EOR zp	C5	CMP zp
46	LSR zp	C6	DEC zp
47	---	C7	---
48	PHA	C8	INY
49	EOR #nn	C9	CMP #nn
4A	LSR A	CA	DEX
4B	---	CB	---
4C	JMP abs	CC	CPY abs
4D	EOR abs	CD	CMP abs
4E	LSR abs	CE	DEC abs
4F	---	CF	---
50	BVC rel	D0	BNE rel
51	EOR (zp),Y	D1	CMP (zp),Y
52	EOR (zp) *	D2	CMP (zp) *
53	---	D3	---
54	---	D4	---
55	EOR zp,X	D5	CMP zp,X
56	LSR zp,X	D6	DEC zp,X
57	---	D7	---
58	CLI	D8	CLD
59	EOR abs,Y	D9	CMP abs,Y
5A	PHY *	DA	PHX *
5B	---	DB	---
5C	---	DC	---
5D	EOR abs,X	DD	CMP abs,X
5E	LSR abs,X	DE	DEC abs,X
5F	---	DF	---
60	RTS	E0	CPX #nn

61	ADC (zp,X)	E1	SBC (zp,X)
62	---	E2	---
63	---	E3	---
64	STZ zp *	E4	CPX zp
65	ADC zp	E5	SBC zp
66	ROR zp	E6	INC zp
67	---	E7	---
68	PLA	E8	INX
69	ADC #nn	E9	SBC #nn
6A	ROR A	EA	NOP
6B	---	EB	---
6C	JMP (abs)	EC	CPX abs
6D	ADC abs	ED	SBC abs
6E	ROR abs	EE	INC abs
6F	---	EF	---
70	BCS rel	F0	BEQ rel
71	ADC (zp),Y	F1	SBC (zp),Y
72	ADC (zp) *	F2	SBC (zp) *
73	---	F3	---
74	STZ zp,X *	F4	---
75	ADC zp,X	F5	SBC zp,X
76	ROR zp,X	F6	INC zp,X
77	---	F7	---
78	SEI	F8	SED
79	ADC abs,Y	F9	SBC abs,Y
7A	PLY *	FA	PLX *
7B	---	FB	---
7C	JMP (abs,X) *	FC	---
7D	ADC abs,X	FD	SBC abs,X
7E	ROR abs,X	FE	INC abs,X
7F	---	FF	---

[&884D-&891B] Assembly Mnemonic and default Opcode tables

There are 3 tables are stored at locations &884D-&8891 (Mnemonic MSB), &8892-&88D6 (Mnemonic LSB) and &88D7-&891B (default Opcode). These tables help to decode the Assembly Mnemonics and to obtain the correct opcodes.
The values contained in these tables is as follows:

Num	Address	Encoded Mnemonic MSB byte	Address	Encoded Mnemonic LSB byte	Address	Default Opcode	Mnemonic
01	884D	4B	8892	0A	88D7	00	BRK
02	884E	83	8893	0D	88D8	18	CLC
03	884F	84	8894	0D	88D9	D8	CLD
04	8850	89	8895	0D	88DA	58	CLI
05	8851	96	8896	0D	88DB	B8	CLV
06	8852	B8	8897	10	88DC	CA	DEX
07	8853	B9	8898	10	88DD	88	DEY
08	8854	D8	8899	25	88DE	E8	INX
09	8855	D9	889A	25	88DF	C8	INY
0A	8856	F0	889B	39	88E0	EA	NOP
0B	8857	01	889C	41	88E1	48	PHA
0C	8858	10	889D	41	88E2	08	PHP
0D	8859	81	889E	41	88E3	68	PLA
0E	885A	90	889F	41	88E4	28	PLP
0F	885B	89	88A0	4A	88E5	40	RTI
10	885C	93	88A1	4A	88E6	60	RTS
11	885D	A3	88A2	4C	88E7	38	SEC
12	885E	A4	88A3	4C	88E8	F8	SED
13	885F	A9	88A4	4C	88E9	78	SEI
14	8860	38	88A5	50	88EA	AA	TAX
15	8861	39	88A6	50	88EB	A8	TAY
16	8862	78	88A7	52	88EC	BA	TSX
17	8863	01	88A8	53	88ED	8A	TXA
18	8864	13	88A9	53	88EE	9A	TXS
19	8865	21	88AA	53	88EF	98	TYA
1A	8866	A1	88AB	10	88F0	3A	DEA [New 6502C12 Mnemonic]
1B	8867	C1	88AC	25	88F1	1A	INA [New 6502C12 Mnemonic]
1C	8868	19	88AD	41	88F2	5A	PHY [New 6502C12 Mnemonic]
1D	8869	18	88AE	41	88F3	DA	PHX [New 6502C12 Mnemonic]
1E	886A	99	88AF	41	88F4	7A	PLY [New 6502C12 Mnemonic]
1F	886B	98	88B0	41	88F5	FA	PLX [New 6502C12 Mnemonic] [End of single-byte Mnemonics]
20	886C	63	88B1	08	88F6	90	BCC [Start of Branch Mnemonics]
21	886D	73	88B2	08	88F7	B0	BCS
22	886E	B1	88B3	08	88F8	F0	BEQ
23	886F	A9	88B4	09	88F9	30	BMI

24	8870	C5	88B5	09	88FA	D0	BNE
25	8871	0C	88B6	0A	88FB	10	BPL
26	8872	C3	88B7	0A	88FC	50	BVC
27	8873	D3	88B8	0A	88FD	70	BVS
28	8874	41	88B9	0A	88FE	80	BRA [New 6502C12 Mnemonic] [End of Branch Mnemonics]
29	8875	C4	88BA	05	88FF	21	AND
2A	8876	F2	88BB	15	8900	41	EOR
2B	8877	41	88BC	3E	8901	01	ORA
2C	8878	83	88BD	04	8902	61	ADC
2D	8879	B0	88BE	0D	8903	C1	CMP
2E	887A	81	88BF	30	8904	A1	LDA
2F	887B	43	88C0	4C	8905	E1	SBC
30	887C	6C	88C1	06	8906	06	ASL
31	887D	72	88C2	32	8907	46	LSR
32	887E	EC	88C3	49	8908	26	ROL
33	887F	F2	88C4	49	8909	66	ROR
34	8880	A3	88C5	10	890A	C6	DEC
35	8881	C3	88C6	25	890B	E6	INC
36	8882	92	88C7	0D	890C	9C	CLR
37	8883	9A	88C8	4E	890D	9C	STZ
38	8884	18	88C9	0E	890E	E0	CPX
39	8885	19	88CA	0E	890F	C0	CPY
3A	8886	62	88CB	52	8910	00	TSB
3B	8887	42	88CC	52	8911	10	TRB
3C	8888	34	88CD	09	8912	24	BIT
3D	8889	B0	88CE	29	8913	4C	JMP
3E	888A	72	88CF	2A	8914	20	JSR
3F	888B	98	88D0	30	8915	A2	LDX
40	888C	99	88D1	30	8016	A0	LDY
41	888D	81	88D2	4E	8917	81	STA
42	888E	98	88D3	4E	8918	86	STX
43	888F	99	88D4	4E	8919	8A	STY
44	8890	14	88D5	3E	891A	3A	OPT
45	8891	35	88D6	16	891B	85	EQU

Disassembly for the Assemble Assembly instruction/statement routine

89D4		032 174 152	20 AE 98	JSR &98AE Evaluate variable name & create new variable
89D7	\	240 092	F0 5C	BEQ 92 --> &8A35 [JMP &9B69 Syntax error]
89D9	Z	176 090	B0 5A	BCS 90 --> &8A35 [JMP &9B69 Syntax error]
89DB	C	032 067 188	20 43 BC	JSR &BC43 Push &2A, &2B & &2C to the Stack
89DE		032 132 173	20 84 AD	JSR &AD84 Set IWA to P%
89E1	'	133 039	85 27	STA &27
89E3	+	032 043 179	20 2B B3	JSR &B32B Set numeric variable
89E6	u	032 117 146	20 75 92	JSR &9275 Set PTR A Offset to PTR B Offset
89E9	N	132 078	84 4E	STY &4E
89EB		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
89EE		160 000	A0 00	LDY#&00
89F0	d=	100 061	64 3D	STZ &3D
89F2	:	201 058	C9 3A	CMP#&3A
89F4	h	240 104	F0 68	BEQ 104 --> &8A5E Store current instruction (if any) and exit
89F6		201 013	C9 0D	CMP#&0D
89F8	d	240 100	F0 64	BEQ 100 --> &8A5E Store current instruction (if any) and exit
89FA	\	201 092	C9 5C	CMP#&5C
89FC	`	240 096	F0 60	BEQ 96 --> &8A5E Store current instruction (if any) and exit
89FE	.	201 046	C9 2E	CMP#&2E
8A00		240 210	F0 D2	BEQ -46 --> &89D4 Create label
8A02		198 010	C6 0A	DEC &0A
8A04		162 003	A2 03	LDX#&03
8A06		164 010	A4 0A	LDY &0A
8A08		230 010	E6 0A	INC &0A
8A0A		177 011	B1 0B	LDA (&0B),Y
8A0C	0*	048 042	30 2A	BMI 42 --> &8A38 Check for BASIC tokens that are also Mnemonics
8A0E		201 032	C9 20	CMP#&20
8A10		240 016	F0 10	BEQ 16 --> &8A22
8A12		160 005	A0 05	LDY#&05
8A14		010	0A	ASL A
8A15		010	0A	ASL A
8A16		010	0A	ASL A
8A17		010	0A	ASL A
8A18	&=	038 061	26 3D	ROL &3D

8A1A	&>	038 062	26 3E	ROL &3E
8A1C		136	88	DEY
8A1D		208 248	D0 F8	BNE -8 --> &8A17
8A1F		202	CA	DEX
8A20		208 228	D0 E4	BNE -28 --> &8A06
8A22	E	162 069	A2 45	LDX#&45
8A24	=	165 061	A5 3D	LDA &3D
8A26	L	221 076 136	DD 4C 88	CMP &884C,X
8A29		208 007	D0 07	BNE 7 --> &8A32
8A2B		188 145 136	BC 91 88	LDY &8891,X
8A2E	>	196 062	C4 3E	CPY &3E
8A30	!	240 033	F0 21	BEQ 33 --> &8A53
8A32		202	CA	DEX
8A33		208 241	D0 F1	BNE -15 --> &8A26
8A35	Li	076 105 155	4C 69 9B	JMP &9B69 Syntax error
8A38	)	162 041	A2 29	LDX#&29
8A3A		201 128	C9 80	CMP#&80
8A3C		240 021	F0 15	BEQ 21 --> &8A53
8A3E		232	E8	INX
8A3F		201 130	C9 82	CMP#&82
8A41		240 016	F0 10	BEQ 16 --> &8A53
8A43		232	E8	INX
8A44		201 132	C9 84	CMP#&84
8A46		208 237	D0 ED	BNE -19 --> &8A35 [JMP &9B69 Syntax error]
8A48		230 010	E6 0A	INC &0A
8A4A		200	C8	INY
8A4B		177 011	B1 0B	LDA (&0B),Y
8A4D	)	041 223	29 DF	AND#&DF
8A4F	A	201 065	C9 41	CMP#&41
8A51		208 226	D0 E2	BNE -30 --> &8A35 [JMP &9B69 Syntax error]
8A53		189 214 136	BD D6 88	LDA &88D6,X
8A56	)	133 041	85 29	STA &29
8A58		160 001	A0 01	LDY#&01
8A5A		224 032	E0 20	CPX#&20
8A5C	H	176 072	B0 48	BCS 72 --> &8AA6
8A5E	@	173 064 004	AD 40 04	LDA &0440

8A61	7	133 055	85 37	STA &37
8A63	9	132 057	84 39	STY &39
8A65	(	166 040	A6 28	LDX &28
8A67		224 004	E0 04	CPX#&04
8A69	A	174 065 004	AE 41 04	LDX &0441
8A6C	8	134 056	86 38	STX &38
8A6E		144 006	90 06	BCC 6 --> &8A76
8A70	<	173 060 004	AD 3C 04	LDA &043C
8A73	=	174 061 004	AE 3D 04	LDX &043D
8A76	:	133 058	85 3A	STA &3A
8A78	;	134 059	86 3B	STX &3B
8A7A		152	98	TYA
8A7B	(	240 040	F0 28	BEQ 40 --> &8AA5
8A7D		016 004	10 04	BPL 4 --> &8A83
8A7F	6	164 054	A4 36	LDY &36
8A81	"	240 034	F0 22	BEQ 34 --> &8AA5
8A83		136	88	DEY
8A84	)	185 041 000	B9 29 00	LDA &0029,Y
8A87	$9	036 057	24 39	BIT &39
8A89		016 003	10 03	BPL 3 --> &8A8E
8A8B		185 000 006	B9 00 06	LDA &0600,Y
8A8E	:	145 058	91 3A	STA (&3A),Y
8A90	@	238 064 004	EE 40 04	INC &0440
8A93		208 003	D0 03	BNE 3 --> &8A98
8A95	A	238 065 004	EE 41 04	INC &0441
8A98		144 008	90 08	BCC 8 --> &8AA2
8A9A	<	238 060 004	EE 3C 04	INC &043C
8A9D		208 003	D0 03	BNE 3 --> &8AA2
8A9F	=	238 061 004	EE 3D 04	INC &043D
8AA2		152	98	TYA
8AA3		208 222	D0 DE	BNE -34 --> &8A83
8AA5	`	096	60	RTS
8AA6	)	224 041	E0 29	CPX#&29
8AA8	<	176 060	B0 3C	BCS 60 --> &8AE6
8AAA	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8AAD		024	18	CLC
8AAE	*	165 042	A5 2A	LDA &2A

```
8AB0  @  237 064 004  ED 40 04  SBC &0440
8AB3     168          A8        TAY
8AB4  +  165 043      A5 2B     LDA &2B
8AB6  A  237 065 004  ED 41 04  SBC &0441
8AB9     192 001      C0 01     CPY#&01
8ABB     136          88        DEY
8ABC     233 000      E9 00     SBC#&00
8ABE     240 027      F0 1B     BEQ 27 --> &8ADB
8AC0     026          1A        INC A
8AC1     208 003      D0 03     BNE 3 --> &8AC6
8AC3     152          98        TYA
8AC4  0  048 025      30 19     BMI 25 --> &8ADF
8AC6  (  165 040      A5 28     LDA &28
8AC8  )  041 002      29 02     AND#&02
8ACA     240 018      F0 12     BEQ 18 --> &8ADE
8ACC                           ... Out of range error...
8ADB     152          98        TYA
8ADC  0  048 232      30 E8     BMI -24 --> &8AC6
8ADE     168          A8        TAY
8ADF  *  132 042      84 2A     STY &2A
8AE1     160 002      A0 02     LDY#&02
8AE3  L^ 076 094 138  4C 5E 8A  JMP &8A5E Store current instruction (if any) and exit
8AE6  0  224 048      E0 30     CPX#&30
8AE8     176 022      B0 16     BCS 22 --> &8B00
8AEA     032 223 140  20 DF 8C  JSR &8CDF Get next non-space char (PTR A) and compare with '#'
8AED     208 024      D0 18     BNE 24 --> &8B07
8AEF     032 204 140  20 CC 8C  JSR &8CCC Add 8 to Opcode (&29)
8AF2  o  032 111 146  20 6F 92  JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8AF5  +  165 043      A5 2B     LDA &2B
8AF7     240 232      F0 E8     BEQ -24 --> &8AE1
8AF9                           ... Byte error...
8B00  A  224 065      E0 41     CPX#&41
8B02  c  208 099      D0 63     BNE 99 --> &8B67
8B04     032 224 142  20 E0 8E  JSR &8EE0 Get next non-space char (PTR A)
8B07  (  201 040      C9 28     CMP#&28
8B09  9  208 057      D0 39     BNE 57 --> &8B44
```

8B0B	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8B0E		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8B11	)	201 041	C9 29	CMP#&29
8B13		208 023	D0 17	BNE 23 --> &8B2C
8B15		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8B18		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
8B1B		240 004	F0 04	BEQ 4 --> &8B21
8B1D	)	230 041	E6 29	INC &29
8B1F		128 212	80 D4	BRA -44 --> &8AF5 If IWA = 8-bit value, Y = 2 & Store assembly instruction; otherwise, Byte error
8B21		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8B24	)	041 223	29 DF	AND#&DF
8B26	Y	201 089	C9 59	CMP#&59
8B28		240 203	F0 CB	BEQ -53 --> &8AF5 If IWA = 8-bit value, Y = 2 & Store assembly instruction; otherwise, Byte error
8B2A		128 016	80 10	BRA 16 --> &8B3C Index error
8B2C	,	201 044	C9 2C	CMP#&2C
8B2E		208 012	D0 0C	BNE 12 --> &8B3C Index error
8B30		032 215 140	20 D7 8C	JSR &8CD7 Compare next non-space [PTR A] character with 'X' or 'x'
8B33		208 007	D0 07	BNE 7 --> &8B3C Index error
8B35		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8B38	)	201 041	C9 29	CMP#&29
8B3A		240 185	F0 B9	BEQ -71 --> &8AF5 If IWA = 8-bit value, Y = 2 & Store assembly instruction; otherwise, Byte error
8B3C				... Index error...
8B44	m	032 109 146	20 6D 92	JSR &926D Decrement TEXT POINTER A offset & evaluate Expression [PTR A] & convert result to integer
8B47		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
8B4A		208 018	D0 12	BNE 18 --> &8B5E
8B4C		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8B4F		032 215 140	20 D7 8C	JSR &8CD7 Compare next non-space [PTR A] character with 'X' or 'x'
8B52		240 010	F0 0A	BEQ 10 --> &8B5E
8B54	Y	201 089	C9 59	CMP#&59
8B56		208 228	D0 E4	BNE -28 --> &8B3C Index error
8B58		032 204 140	20 CC 8C	JSR &8CCC Add 8 to Opcode (&29)
8B5B	L	076 254 139	4C FE 8B	JMP &8BFE Set number of bytes (Y) = 3 & Store assembly instruction
8B5E		032 207 140	20 CF 8C	JSR &8CCF Add 4 to Opcode (&29)
8B61	+	165 043	A5 2B	LDA &2B
8B63		208 243	D0 F3	BNE -13 --> &8B58

8B65		128 144	80 90	BRA -112 --> &8AF7 If IWA = 8-bit value, Y = 2 & Store assembly instruction; otherwise, Byte error
8B67	6	224 054	E0 36	CPX#&36
8B69	6	176 054	B0 36	BCS 54 --> &8BA1
8B6B		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8B6E	)	041 223	29 DF	AND#&DF
8B70	A	201 065	C9 41	CMP#&41
8B72		240 018	F0 12	BEQ 18 --> &8B86
8B74	m	032 109 146	20 6D 92	JSR &926D Decrement TEXT POINTER A offset & evaluate Expression [PTR A] & convert result to integer
8B77		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
8B7A		208 229	D0 E5	BNE -27 --> &8B61
8B7C		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8B7F		032 215 140	20 D7 8C	JSR &8CD7 Compare next non-space [PTR A] character with 'X' or 'x'
8B82		240 221	F0 DD	BEQ -35 --> &8B61
8B84		128 182	80 B6	BRA -74 --> &8B3C Index error
8B86		200	C8	INY
8B87		177 011	B1 0B	LDA (&0B),Y
8B89		032 132 141	20 84 8D	JSR &8D84 Check for Variable name character or digit (in A)
8B8C		176 230	B0 E6	BCS -26 --> &8B74
8B8E		160 022	A0 16	LDY#&16
8B90	4	224 052	E0 34	CPX#&34
8B92		144 006	90 06	BCC 6 --> &8B9A
8B94		208 002	D0 02	BNE 2 --> &8B98
8B96	6	160 054	A0 36	LDY#&36
8B98	)	132 041	84 29	STY &29
8B9A		032 207 140	20 CF 8C	JSR &8CCF Add 4 to Opcode (&29)
8B9D		160 001	A0 01	LDY#&01
8B9F	_	128 095	80 5F	BRA 95 --> &8C00
8BA1	8	224 056	E0 38	CPX#&38
8BA3	%	176 037	B0 25	BCS 37 --> &8BCA
8BA5	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8BA8		160 003	A0 03	LDY#&03
8BAA		162 001	A2 01	LDX#&01
8BAC	+	165 043	A5 2B	LDA &2B
8BAE		208 007	D0 07	BNE 7 --> &8BB7
8BB0		162 015	A2 0F	LDX#&0F
8BB2	d	169 100	A9 64	LDA#&64

8BB4	)	133 041	85 29	STA &29
8BB6		136	88	DEY
8BB7	Z	090	5A	PHY
8BB8		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
8BBB		208 010	D0 0A	BNE 10 --> &8BC7
8BBD		032 215 140	20 D7 8C	JSR &8CD7 Compare next non-space [PTR A] character with 'X' or 'x'
8BC0		208 194	D0 C2	BNE -62 --> &8B84
8BC2		138	8A	TXA
8BC3	e)	101 041	65 29	ADC &29
8BC5	)	133 041	85 29	STA &29
8BC7	z	122	7A	PLY
8BC8	6	128 054	80 36	BRA 54 --> &8C00
8BCA	<	224 060	E0 3C	CPX#&3C
8BCC		176 028	B0 1C	BCS 28 --> &8BEA
8BCE	:	224 058	E0 3A	CPX#&3A
8BD0		176 007	B0 07	BCS 7 --> &8BD9
8BD2		032 223 140	20 DF 8C	JSR &8CDF Get next non-space char (PTR A) and compare with '#'
8BD5		240 016	F0 10	BEQ 16 --> &8BE7
8BD7		198 010	C6 0A	DEC &0A
8BD9	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8BDC		128 128	80 80	BRA -128 --> &8B5E
8BDE		032 223 140	20 DF 8C	JSR &8CDF Get next non-space char (PTR A) and compare with '#'
8BE1		208 145	D0 91	BNE -111 --> &8B74
8BE3		160 137	A0 89	LDY#&89
8BE5	)	132 041	84 29	STY &29
8BE7	L	076 242 138	4C F2 8A	JMP &8AF2 Get Integer value; if 8-bit value, Y = 2 & Store assembly instruction; otherwise, Byte error
8BEA		240 242	F0 F2	BEQ -14 --> &8BDE
8BEC	>	224 062	E0 3E	CPX#&3E
8BEE		240 011	F0 0B	BEQ 11 --> &8BFB
8BF0	7	176 055	B0 37	BCS 55 --> &8C29
8BF2		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8BF5	(	201 040	C9 28	CMP#&28
8BF7		240 010	F0 0A	BEQ 10 --> &8C03
8BF9		198 010	C6 0A	DEC &0A
8BFB	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8BFE		160 003	A0 03	LDY#&03

8C00	L^	076 094 138	4C 5E 8A	JMP &8A5E Store current instruction (if any) and exit
8C03		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8C06		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8C09	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8C0C		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8C0F	)	201 041	C9 29	CMP#&29
8C11		240 235	F0 EB	BEQ -21 --> &8BFE Set number of bytes (Y) = 3 & Store assembly instruction
8C13	,	201 044	C9 2C	CMP#&2C
8C15		208 015	D0 0F	BNE 15 --> &8C26
8C17		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8C1A		032 215 140	20 D7 8C	JSR &8CD7 Compare next non-space [PTR A] character with 'X' or 'x'
8C1D		208 007	D0 07	BNE 7 --> &8C26
8C1F		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8C22	)	201 041	C9 29	CMP#&29
8C24		240 216	F0 D8	BEQ -40 --> &8BFE Set number of bytes (Y) = 3 & Store assembly instruction
8C26	L<	076 060 139	4C 3C 8B	JMP &8B3C Index error
8C29	D	224 068	E0 44	CPX#&44
8C2B	M	176 077	B0 4D	BCS 77 --> &8C7A
8C2D	=	165 061	A5 3D	LDA &3D
8C2F	I	073 001	49 01	EOR#&01
8C31	)	041 031	29 1F	AND#&1F
8C33	H	072	48	PHA
8C34	A	224 065	E0 41	CPX#&41
8C36	!	176 033	B0 21	BCS 33 --> &8C59
8C38		032 223 140	20 DF 8C	JSR &8CDF Get next non-space char (PTR A) and compare with '#'
8C3B		208 003	D0 03	BNE 3 --> &8C40
8C3D	h	104	68	PLA
8C3E		128 167	80 A7	BRA -89 --> &8BE7
8C40	m	032 109 146	20 6D 92	JSR &926D Decrement TEXT POINTER A offset & evaluate Expression [PTR A] & convert result to integer
8C43	h	104	68	PLA
8C44	7	133 055	85 37	STA &37
8C46		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
8C49		208 145	D0 91	BNE -111 --> &8BDC
8C4B		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8C4E	)	041 031	29 1F	AND#&1F

8C50	7	197 055	C5 37	CMP &37
8C52		208 210	D0 D2	BNE -46 --> &8C26
8C54		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8C57		128 131	80 83	BRA -125 --> &8BDC
8C59	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8C5C	h	104	68	PLA
8C5D	7	133 055	85 37	STA &37
8C5F		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
8C62		208 019	D0 13	BNE 19 --> &8C77
8C64		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
8C67	)	041 031	29 1F	AND#&1F
8C69	7	197 055	C5 37	CMP &37
8C6B		208 185	D0 B9	BNE -71 --> &8C26
8C6D		032 201 140	20 C9 8C	JSR &8CC9 Add 16 to Opcode (&29)
8C70	+	165 043	A5 2B	LDA &2B
8C72		240 003	F0 03	BEQ 3 --> &8C77
8C74	L	076 249 138	4C F9 8A	JMP &8AF9 Byte error
8C77	La	076 097 139	4C 61 8B	JMP &8B61
8C7A		208 011	D0 0B	BNE 11 --> &8C87
8C7C	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8C7F	*	165 042	A5 2A	LDA &2A
8C81	(	133 040	85 28	STA &28
8C83		160 000	A0 00	LDY#&00
8C85	*	128 042	80 2A	BRA 42 --> &8CB1
8C87		162 001	A2 01	LDX#&01
8C89		164 010	A4 0A	LDY &0A
8C8B		230 010	E6 0A	INC &0A
8C8D		177 011	B1 0B	LDA (&0B),Y
8C8F	)	041 223	29 DF	AND#&DF
8C91	B	201 066	C9 42	CMP#&42
8C93		240 018	F0 12	BEQ 18 --> &8CA7
8C95		232	E8	INX
8C96	W	201 087	C9 57	CMP#&57
8C98		240 013	F0 0D	BEQ 13 --> &8CA7
8C9A		162 004	A2 04	LDX#&04
8C9C	D	201 068	C9 44	CMP#&44

8C9E		240 007	F0 07	BEQ 7 --> &8CA7
8CA0	S	201 083	C9 53	CMP#&53
8CA2		240 019	F0 13	BEQ 19 --> &8CB7
8CA4	Li	076 105 155	4C 69 9B	JMP &9B69 Syntax error
8CA7		218	DA	PHX
8CA8	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
8CAB	)	162 041	A2 29	LDX#&29
8CAD		032 198 189	20 C6 BD	JSR &BDC6 Save Integer (IWA) to zero page location
8CB0	z	122	7A	PLY
8CB1	L^	076 094 138	4C 5E 8A	JMP &8A5E Store current instruction (if any) and exit
8CB4	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
8CB7	(	165 040	A5 28	LDA &28
8CB9	H	072	48	PHA
8CBA	/	032 047 157	20 2F 9D	JSR &9D2F Ptr B = Ptr A & Get result of expression
8CBD		208 245	D0 F5	BNE -11 --> &8CB4
8CBF	h	104	68	PLA
8CC0	(	133 040	85 28	STA &28
8CC2	u	032 117 146	20 75 92	JSR &9275 Set PTR A Offset to PTR B Offset
8CC5		160 255	A0 FF	LDY#&FF
8CC7		128 232	80 E8	BRA -24 --> &8CB1
8CC9		032 204 140	20 CC 8C	JSR &8CCC Add 8 to Opcode (&29)
8CCC		032 207 140	20 CF 8C	JSR &8CCF Add 4 to Opcode (&29)
8CCF	)	165 041	A5 29	LDA &29
8CD1		024	18	CLC
8CD2	i	105 004	69 04	ADC#&04
8CD4	)	133 041	85 29	STA &29
8CD6	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

BD37 Output ASCII characeter or BASIC Token to screen

Submitted by Steve Fewell

Routine:PrintToken
Name: Output ASCII characeter or BASIC Token to screen
Starting Address: &BD37
Entry criteria: A contains the ASCII value to output to the screen.

Description:

Store A in location &37.
If A is less then #&80 then the ASCII character will be output via the &BD94 routine.

&BD94 outputs the character with correct handling of the following:

* The COUNT variable
* New line when COUNT exceeds the WIDTH value
* Redirection of output (to memory) when *EDIT-mode is active
* New line when character to output is &0D ('<cr>')

Otherwise, a BASIC token is been specified by the ASCII character, so output the BASIC Keyword (in ASCII text) as follows:

*1) Set pointer &38-&39 to point to address &8456 -> which is the address of the start of the Token table
*2) Save Y (the current program/command line position) to the Stack
*3) Set Y to 0 (to point to the start of the next Keyword in the &38-&39 token table list.
*4) Load the next character (of the token) from the Token Table (pointed to by (&38,&39) + Y)
*5) Skip all ASCII characters (less then #&80) [incrementing Y for each character skipped]
Now, the character is the token number that represents the BASIC Keyword specified by the skipped characters
*6) Compare the character against the token we want to output (stored in location &37)
If the token (character) is equal to &37 then output the Keyword text as follows [&BD5E]:
* Reset Y to 0 (the start of the ASCII Text Keyword)
* Load the next character of the Keyword from (&38-&39) + Y.
* If the character is negative (the token number) then goto Step 9 to exit (as token is now printed).
* Otherwise, call routine &BD94 to output the ASCII character to the screen.
* Increment Y (to point to the next character of the BASIC Keyword)
* If Y is not 0 (less than 255 characters have been printed) then jump back to load the next character of the Keyword
* Jump to step 9 to exit.
*7) Set the token table pointer (&38,&39) to point to the position after the Token Number (i.e. Add Y + 1 to the pointer).
*8) Jump back to Step 3 to check the next BASIC Keyword in the Token Table list
*9) Retrieve Y from the Stack and exit

Disassembly for the Output ASCII characeter or BASIC Token to screen routine

BD37	7	133 055	85 37	STA &37
BD39		201 128	C9 80	CMP#&80
BD3B	W	144 087	90 57	BCC 87 --> &BD94 Output character to the screen
BD3D	V	169 086	A9 56	LDA#&56
BD3F	8	133 056	85 38	STA &38
BD41		169 132	A9 84	LDA#&84
BD43	9	133 057	85 39	STA &39
BD45	Z	090	5A	PHY
BD46		160 000	A0 00	LDY#&00
BD48		200	C8	INY
BD49	8	177 056	B1 38	LDA (&38),Y
BD4B		016 251	10 FB	BPL -5 --> &BD48
BD4D	7	197 055	C5 37	CMP &37
BD4F		240 013	F0 0D	BEQ 13 --> &BD5E
BD51		200	C8	INY
BD52		152	98	TYA
BD53	8	056	38	SEC
BD54	e8	101 056	65 38	ADC &38
BD56	8	133 056	85 38	STA &38
BD58		144 236	90 EC	BCC -20 --> &BD46
BD5A	9	230 057	E6 39	INC &39
BD5C		128 232	80 E8	BRA -24 --> &BD46
BD5E		160 000	A0 00	LDY#&00
BD60	8	177 056	B1 38	LDA (&38),Y
BD62	0	048 006	30 06	BMI 6 --> &BD6A
BD64		032 148 189	20 94 BD	JSR &BD94 Output character to the screen
BD67		200	C8	INY
BD68		208 246	D0 F6	BNE -10 --> &BD60
BD6A	z	122	7A	PLY
BD6B	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9BCF Check & skip Program Line end

Submitted by Steve Fewell

Description:

Firstly, this routine calls routine &9BA6 to check for the end of statement character (':', '<cr>', 'ELSE') if end of Statement was not found then issue 'Syntax error'.

If the Line terminator character was ':', then exit (as nothing more to do - as we are not at the line end yet).

Now, either '<cr>' or 'ELSE' have been found, so we have to skip to the next statement/program line.
If the BASIC Text pointer A MSB is 7 (i.e. PTR A points to an address between &0700 and &07FF) then jump to &8F86 to return to the command line input prompt (as we are not currently executing a program).

Otherwise, the next line number needs to be processed and skipped before we continue running the program.
Get the next byte after the line terminator.
If the byte is negative (i.e. &FF) then jump to &8F86 to prompt for the next conmmand line input - as we have reached the end of the program that we were executing.

If the TRACE flag (&20) is not 0 then TRACE is currently on, so set the IWA to the 2-byte line number (located after the line terminator character) & call routine &9C4B to display the line number on the screen between square braces (i.e.[xxx]).

[&9BF2] Add 3 to the BASIC Text pointer A address (&0B-&0C) to skip the Line Number and Line length, set the BASIC Text Pointer A Offset (&0A) to 1 and exit.

Disassembly for the Check & skip Program Line end routine

9BCF		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
9BD2		178 011	B2 0B	LDA (&0B)
9BD4	:	201 058	C9 3A	CMP#&3A
9BD6		240 246	F0 F6	BEQ -10 --> &9BCE [RTS]
9BD8		165 012	A5 0C	LDA &0C
9BDA		201 007	C9 07	CMP#&07
9BDC	$	240 036	F0 24	BEQ 36 --> &9C02
9BDE		160 001	A0 01	LDY#&01
9BE0		177 011	B1 0B	LDA (&0B),Y

9BE2	0	048 030	30 1E	BMI 30 --> &9C02
9BE4		166 032	A6 20	LDX &20
9BE6		240 010	F0 0A	BEQ 10 --> &9BF2
9BE8	+	133 043	85 2B	STA &2B
9BEA		200	C8	INY
9BEB		177 011	B1 0B	LDA (&0B),Y
9BED	*	133 042	85 2A	STA &2A
9BEF	K	032 075 156	20 4B 9C	JSR &9C4B Display current line number (IWA) on screen [if TRACE is on]
9BF2		169 003	A9 03	LDA#&03
9BF4		024	18	CLC
9BF5	e	101 011	65 0B	ADC &0B
9BF7		133 011	85 0B	STA &0B
9BF9		144 002	90 02	BCC 2 --> &9BFD
9BFB		230 012	E6 0C	INC &0C
9BFD		160 001	A0 01	LDY#&01
9BFF		132 010	84 0A	STY &0A
9C01	`	096	60	RTS
9C02	L	076 134 143	4C 86 8F	JMP &8F86 Read & execute command line input

---

BASIC IV ROM Routines

Page Last Altered: undefined

BE93 EXT =

Submitted by Steve Fewell

Description:

Set A to 3 (this is the OSARGS action number for 'Set EXT for an open file').

Call &BE99 (in the 'PTR =' routine) to:

* Store A to the 6502 stack
* Check for '#' character ('Missing #' error if not found)
* Get Integer result of expression (file channel number) and store LSB byte to the 6502 stack
* Check for '=' character ('Syntax error' if not found)
* Get the result of expression and convert to Integer (IWA) if Floating-Point value, or issue 'Type mismatch' error if String value
* Check for the end of statement ('Syntax error' if ':', '<cr>' or 'ELSE' not found)
* Retrieve Y from the 6502 stack (the file channel number)
* Set X to #&2A (the start of the 4-byte zero page parameter block that holds the new file length (EXT) value)
* Retrieve A from the 6502 stack (this retrieves #&03, the OSARGS action for 'set EXT value')
* Call the Operating System OSARGS routine (&FFDA) to set the EXT value for the open file (specified by channel Y)
* Jump to &9005 to start processing the next command line or program statement

Disassembly for the EXT = routine

BE93	169 003	A9 03	LDA#&03
BE95	128 002	80 02	BRA 2 --> &BE99 Get '#' char, file channel number, '=' char & result of expression and call OSARGS (A=3)

BASIC IV ROM Routines

Page Last Altered: undefined

BE97 PTR =

Submitted by Steve Fewell

Description:

Store #&01 on the 6502 stack.
Call routine &BA3E to copy BASIC Text pointer A address to BASIC text pointer B, Check that the next character is '#' ('Missing #' error if not), set Y to the number after the '#' (the file channel number).
Store Y to the 6502 stack.

Call routine &9B52 to check that the next character is '=' ('Syntax error' if not), retrieve the result of the expression after the '=' character, and check for the end of statement ('Syntax error' if not).

Call routine &96BC to check the expression result type (location &27), issue 'Type mismatch' error if the result is a String value (SWA), or convert the result to Integer if the result type is a Floating-Point value.

Retrieve Y from the 6502 stack.
Now the IWA contains the Integer Pointer (PTR) value to assign to the open file at the channel number stored in Y.
Set X to point to the first byte of the IWA (#&2A, i.e. the start of the 4-byte zero page control block containing the new PTR value).
Retrieve A from the stack (this value specifies the OSARGS action that is required, in this case #&01 (Set file PTR value)).

Call the Operating System OSARGS routine (&FFDA) which will perform the requested operation on the open file.
Now the Pointer of the Open file (with the channel number specified in Y) has been set to the Integer value in the IWA.
Jump to &9005 to start processing the next command line or program statement.

Disassembly for the PTR = routine

BE97		169 001	A9 01	LDA#&01
BE99	H	072	48	PHA
BE9A	>	032 062 186	20 3E BA	JSR &BA3E Copy PTR A to PTR B, check for '#', get file channel number & store it in Y
BE9D	Z	090	5A	PHY
BE9E	R	032 082 155	20 52 9B	JSR &9B52 Check for '=' & get result of expression; check end of statement
BEA1		032 188 150	20 BC 96	JSR &96BC Check result type (location &27) - if Float then convert to Integer
BEA4	z	122	7A	PLY
BEA5	*	162 042	A2 2A	LDX#&2A
BEA7	h	104	68	PLA
BEA8		032 218 255	20 DA FF	JSR &FFDA OSARGS
BEAB	L	076 005 144	4C 05 90	JMP &9005 Process the next program statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

9B52 Check for '=', get expression result & check end of statement

Submitted by Steve Fewell

Description:

If called from 9B46, then copy BASIC Text Pointer A to BASIC Text Pointer B.

Increment BASIC Text pointer B offset, load the next non-space character pointed to by BASIC Text pointer B. If the next character is not '=' then issue a Mistake error, as the current statement is not correct BASIC.
Otherwise, all is well, so jump to &9B8E, which does the following:
Evaluate the expression after the '='.
Then set A = X (this is the next non-space charater after the expression).
Next, load Y wth the BASIC Text pointer B offset (so that routine &9BB0 can update the BASIC Text pointer A position correctly).
Finally, jump to &9BB0 to check for the end of statement (issuing a Syntax error if the statement is not terminated correctly), and update BASIC Text Pointer A to point to the location after the end of the current statement, and exit.

Disassembly for the Check for '=', get expression result & check end of statement routine

9B46		165 011	A5 0B	LDA &0B
9B48		133 025	85 19	STA &19
9B4A		165 012	A5 0C	LDA &0C
9B4C		133 026	85 1A	STA &1A
9B4E		165 010	A5 0A	LDA &0A
9B50		133 027	85 1B	STA &1B
9B52		164 027	A4 1B	LDY &1B
9B54		230 027	E6 1B	INC &1B
9B56		177 025	B1 19	LDA (&19),Y
9B58		201 032	C9 20	CMP#&20
9B5A		240 246	F0 F6	BEQ -10 --> &9B52
9B5C	=	201 061	C9 3D	CMP#&3D
9B5E	.	240 046	F0 2E	BEQ 46 --> &9B8E Evaluate expression & check end of statement

9B60 ... Mistake error

Evaluate expression and check for end of statement

9B8E	; 032 059 157	20 3B 9D	JSR &9D3B Evaluate expression
9B91	138	8A	TXA
9B92	164 027	A4 1B	LDY &1B
9B94	128 026	80 1A	BRA 26 --> &9BB0 Check for end of Statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

960F HIMEM=

Submitted by Steve Fewell

Description:

Call routine &96B9 to Set BASIC Text Pointer B to BASIC Text Pointer A value, check that the next non-space character is an "=", and issue 'Mistake' error if it isn't. Get Result of the expression at BASIC Text Pointer B. Check that the statement terminates correctly (with a '<cr>', ':' or 'ELSE'), (issue 'Syntax error if not terminated correctly), and convert the numeric result value to Integer (or issue 'Type mismatch' error if the value is a String).

Set HIMEM (&06-&07) to the Integer result's 2 LSB bytes (&2A-&2B).
Set the Stack Pointer to the HIMEM value.
Note: this will destroy any information currently on the stack.

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the HIMEM= routine

960F		032 185 150	20 B9 96	JSR &96B9
9612	*	165 042	A5 2A	LDA &2A
9614		133 006	85 06	STA &06
9616		133 004	85 04	STA &04
9618	+	165 043	A5 2B	LDA &2B
961A		133 007	85 07	STA &07
961C		133 005	85 05	STA &05
961E		128 027	80 1B	BRA 27 --> &963B [JMP &9005] Process the next statement

Set Ptr B = Ptr A; Check for '='; Get Integer result of expression & check end of statement

96B9	F	032 070 155	20 46 9B	JSR &9B46 Ptr B = Ptr A; Check for '=' & get result of expression; check end of statement
96BC	'	165 039	A5 27	LDA &27
96BE				... Check if Float value & convert to Integer

BASIC IV ROM Routines

Page Last Altered: undefined

8FA4 '*'-Command

Submitted by Steve Fewell

Description:

Update the BASIC Text Pointer A address (&0B-&0C) to include the PTR A offset value (in Y) (i.e. add Y to PTR A).
BASIC Text pointer A now points to the start of the '*' command.
Set X (LSB - &0B) and Y (MSB - &0C) to the address of BASIC Text pointer A.
Call OSCLI (&FFF7) to execute the Operating System ('*') command.
Continue to &8FAE to skip the rest of the line and execute the next command line or program statement (or to return to the command line prompt if the program has finished or we were not running a program.

Disassembly for the '*'-Command routine

8FA4	032 188 155	20 BC 9B	JSR &9BBC Update BASIC Text pointer A (Add offset value & then reset offset to 1)
8FA7	166 011	A6 0B	LDX &0B
8FA9	164 012	A4 0C	LDY &0C
8FAB	032 247 255	20 F7 FF	JSR &FFF7 OSCLI
8FAE			...Execute next command line / program statement...

BASIC IV ROM Routines

Page Last Altered: undefined

904A LET

Submitted by Steve Fewell

Description:

Call routine &98AE to evaluate the variable name - check whether it exists or not
and create the variable if it does not exist. &98AE also returns with &2A and &2B pointing
to the address of the variable's value (which will be blank (zero) if the variable was newly created),
and &2C set to the variable's type.

If the zero flag is set, then there was an error and the text after the LET command
was not a valid variable name, so issue a Syntax Error.

If the Carry flag is clear then the variable has a numerical value type (E.g. Integer or Floating-Point),
so do the following:
* Store &2A, &2B and &2C on the stack (to save variable info for the Set numerical variable
routine (&B32B) to retrieve)
* Call routine &9B52 to check for '=' (if no '=' then Mistake error), get result of expression and check
that the Statement Line is terminated correctly (if it isn't then Syntax Error).
* Call routine &B32B to set the numerical variable to the value of the expression, or issue
Type mismatch error if the expression returned a String value.
* Jump to routine &9005 to continue running the BASIC program/command line.

If the Carry flag is set then the variable has a String value type, so do the following:

* Push the IWA (&2A-&2D) to the BASIC stack (to save variable info for the Set String variable
routine (&90AB) to retrieve)
* Call routine &9B52 to check for '=' (if no '=' then Mistake error), get result of expression and check
that the Statement Line is terminated correctly (if it isn't then Syntax Error).
* Issue a Type mismatch error if the expression returned a numerical (non string) value.
* Call routine &90AB to set the String variable to the value of the expression.
* Jump to routine &9005 to continue running the BASIC program/command line.

Disassembly for the LET routine

904A		032 174 152	20 AE 98	JSR &98AE Evaluate Variable Name & Allocate variable space (if necessary)
904D	4	240 052	F0 34	BEQ 52 --> &9083 [JMP &9B69 Syntax error]

904F	!	144 033	90 21	BCC 33 --> &9072 Variable is numeric
9051	&	032 038 188	20 26 BC	JSR &BC26 Push IWA to Basic Stack
9054	R	032 082 155	20 52 9B	JSR &9B52 Check for '=' & get result of expression; check end of statement
9057	'	165 039	A5 27	LDA &27
9059	7	208 055	D0 37	BNE 55 --> &9092 Type mismatch error
905B		032 171 144	20 AB 90	JSR &90AB Set String variable (with check length)
905E		128 165	80 A5	BRA -91 --> &9005 Continue running program (next statement)
....				
9072	*	165 042	A5 2A	LDA &2A
9074	H	072	48	PHA
9075	+	165 043	A5 2B	LDA &2B
9077	H	072	48	PHA
9078	,	165 044	A5 2C	LDA &2C
907A	H	072	48	PHA
907B	R	032 082 155	20 52 9B	JSR &9B52 Check for '=' & get result of expression; check end of statement
907E	+	032 043 179	20 2B B3	JSR &B32B Set numeric variable
9081		128 130	80 82	BRA -126 --> &9005 Continue running program (next statement)
9083	Li	076 105 155	4C 69 9B	JMP &9B69 Syntax error

BASIC IV ROM Routines

Page Last Altered: undefined

Extract Hex Number

Submitted by Steve Fewell

Routine: ExtrHex
Name: Extract Hex Number
Starting Address: &ADB7
Entry criteria: Text Pointer B points to the "&" character of the Hex Number. Y contains the Text Pointer B Offset.
Exit: IWA contains the extracted Integer number.

Description:
Clears the IWA to zero (using the FALSE routine - this also [importantly] sets X to zero) . Increment Y to point to the first character of the Hex number (after the "&").

Check the current Digit (&ADBB):
Read the current character pointed to by Text Pointer B. If it is less than "0" then the Hex digit is invalid, so branch to the error routine (&ADE4). If the character is numeric ("0" to "9"), then jump to the process digit routine (&ADCF). If the character is more than "9" then Subtract 55 (so "A" becomes 10 and "F" becomes 15). If the new value is less than 10 or more than 15 then jump to the error routine (&ADE4) otherwise process the Hex digit.

Process the Hex digit (&ADCF):
Multiply the number by 16 (This moves all bits to the most significant half of the byte), this has two functions, firstly it removes the &30 (48) part of the ASCII value of the character, so that if the character is a digit between "0" and "9", it becomes it's numerical equivalent. And, secondary, it prepares the Hex digit so that the 4-bits specifying the digit's value can be pulled off the end.

Next a loop is repeated 4 times, in order to insert each bit of the 4-bit digit into the IWA. The loop moves the bits in the digit right 1 place (ASL), so that the carry contains the previous Most Significant Bit (and hence, the next bit of the digit). The Carry is rolled into the Least significant byte of the IWA, moving all of the existing bits left a position. Note: X equals -1 (&FF) after the loop has finished and the Hex digit has been processed.

Next, increment Y to the next character. If the end of the Text Line has not been reached (The character pointer equals 0 when this situation arises) then branch back to the Check Current Digit routine.

Error Routine (&ADE4):
Check X, if X = 0 then the Process Hex Digit routine has never been reached, i.e. The first character wasn't a valid Hex digit, so a "Bad Hex" error is generated. Otherwise if X is &FF (negative), then at least one digit has been extracted, so the fact that the next character is invalid signifies that the end of the Hex number has been reached, if this is the case then store Y back in &1B (the Text Pointer B Offset). Exit with A = #&40 (as an Integer number is being processed).

Note: No error checking is done to make sure that the Hex Number is not too large to fit in the IWA, instead the Most Significant bits of the IWA are moved out and lost if an overflow condition occurs (without the event being signified by an error message).

Disassembly for the Extract Hex Number routine

ADB7		032 232 171	20 E8 AB	JSR &ABE8 FALSE
ADBA		200	C8	INY
ADBB		177 025	B1 19	LDA (&19),Y
ADBD	0	201 048	C9 30	CMP#&30
ADBF	#	144 035	90 23	BCC 35 --> &ADE4
ADC1	:	201 058	C9 3A	CMP#&3A
ADC3		144 010	90 0A	BCC 10 --> &ADCF
ADC5	7	233 055	E9 37	SBC#&37
ADC7		201 010	C9 0A	CMP#&0A
ADC9		144 025	90 19	BCC 25 --> &ADE4
ADCB		201 016	C9 10	CMP#&10
ADCD		176 021	B0 15	BCS 21 --> &ADE4
ADCF		010	0A	ASL A
ADD0		010	0A	ASL A

ADD1		010	0A	ASL A
ADD2		010	0A	ASL A
ADD3		162 003	A2 03	LDX#&03
ADD5		010	0A	ASL A
ADD6	&*	038 042	26 2A	ROL &2A
ADD8	&+	038 043	26 2B	ROL &2B
ADDA	&,	038 044	26 2C	ROL &2C
ADDC	&-	038 045	26 2D	ROL &2D
ADDE		202	CA	DEX
ADDF		016 244	10 F4	BPL -12 --> &ADD5
ADE1		200	C8	INY
ADE2		208 215	D0 D7	BNE -41 --> &ADBB
ADE4		138	8A	TXA
ADE5		016 187	10 BB	BPL -69 --> &ADA2 Bad Hex error
ADE7		132 027	84 1B	STY &1B
ADE9	@	169 064	A9 40	LDA#&40
ADEB	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

FALSE (Clear IWA)

Submitted by Steve Fewell

Routine: FALSE
Name: Clear IWA [also: IWA = FALSE]
Starting Address: &ABE8
Entry criteria: None
Exit: IWA contains zero.

Description:
Clears the IWA to zero, and sets A to #&40 to show that an Integer number is being processed.

Disassembly for the Clear IWA routine

ABE8	162 000	A2 00	LDX#&00
ABEA	128 241	80 F1	BRA -15 --> &ABDD Store X (zero) in every byte of the IWA

Store X in every byte of the IWA:

ABDD	*	134 042	86 2A	STX &2A
ABDF	+	134 043	86 2B	STX &2B
ABE1	,	134 044	86 2C	STX &2C
ABE3	-	134 045	86 2D	STX &2D

ABE5	@	169 064	A9 40	LDA#&40
ABE7	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B1A0 Load Variable

Submitted by Steve Fewell

Routine:loadvar
Name: Load Variable
Starting Address: &B1C2
Entry criteria: &2A and &2B point to the address of the variable's value.
&2C contains the Type of the value to load.
Exit: The value has been loaded into the appropriate storage location (IWA, FWA or SWA).
The address in (&2A and &2B) could be destroyed.

Description:

If &2C is negative (probable &80 or &81) then we are loading a string value, so jump
to routine &B1F7.
If &2C is zero then we are loading a 1-byte Integer value, so jump to routine &B1C2.
If &2C is &05 then we are loading a Floating-Point value, so jump to routine &B1C7.
Otherwise, &2C must be &04, so we are loading an Integer value, so continue with routine &B1AA.

Disassembly for the Load Variable routine

B1A0	,	164 044	A4 2C	LDY &2C
B1A2	0S	048 083	30 53	BMI 83 --> &B1F7 Load SWA with String Value
B1A4		240 028	F0 1C	BEQ 28 --> &B1C2 Load IWA with 1-byte Integer value
B1A6		192 005	C0 05	CPY#&05
B1A8		240 029	F0 1D	BEQ 29 --> &B1C7 Load FWA with Floating-Point Value (ain)
B1AA				...Load IWA with Integer Value

BASIC IV ROM Routines

Page Last Altered: undefined

B1F7 Load SWA with String value

Submitted by Steve Fewell

Routine:loadstring
Name: Load String value
Starting Address: &B1F7
Entry criteria: &2A and &2B point to the address of the variable's value; or
&2B contains 0 and &2A contains the character for a 1-character string.
Y is &80 for a 1 character string or a String input that is terminated with &0D.
Y is anything else (probably &81) for a String variable).
Exit: The SWA contains the value of the variable.

Description:

If Y is &80 then the value is either a string input (which is terminated by a carriage return (<CR> character) or a single character (determined by the address MSB being zero).
Otherwise (Y is &81) the string value is a String variable.
Each of these cases is explained in their own section below:

String Variable
The address pointed to by (&2A, &2B) is the string parameter block.
The first 2 bytes of this parameter block point to the address of the String value [first byte = Low byte, second byte = High byte of address].
The third byte contains the maximum size allocated to the string value (not needed during the Load operation).
The forth byte contains the current length of the string.
To load the string variable, the following is done:
* Store the forth byte of the parameter block (length of string) in &36 (length of SWA).
* Set (&37 and &38) to the address of the string value (low byte first).
* Load each character of the string value (last character first), and store the character at the appropriate SWA potition.
* Exit with A = &00 (indicating a string result) when the entire string has been copied to the SWA.

String Input (terminated by a carriage return character)
The address pointed to by (&2A, &2B) is the first character of the value.
Store each character of the value in the next available SWA position (Starting at &0600).
When a &0D (carriage return) character is found then store the &0D character in the SWA, stop getting further characters, set &36 to the current SWA position and exit.

After 255 characters have been processed, without a &0D character being found,
exit with A = &00 and &36 (SWA length) also set to zero.

Single Character value
&2A contains the character to place in the SWA and &2B contains zero.
Load A with the character (&2A) and jump to &AE6C (part of GET$) to put
the ASCII character (in A) into the SWA, set the SWA length (&36 to 1) and exit
with A = &00 (as the result is a string value).

Disassembly for the Load SWA with String value routine

B1F7		192 128	C0 80	CPY#&80
B1F9		240 030	F0 1E	BEQ 30 --> &B219 Load SWA with String input (terminated with <CR>) or a 1 char value
B1FB		160 003	A0 03	LDY#&03
B1FD	*	177 042	B1 2A	LDA (&2A),Y
B1FF	6	133 054	85 36	STA &36
B201		240 021	F0 15	BEQ 21 --> &B218
B203		160 001	A0 01	LDY#&01
B205	*	177 042	B1 2A	LDA (&2A),Y
B207	8	133 056	85 38	STA &38
B209	*	178 042	B2 2A	LDA (&2A)
B20B	7	133 055	85 37	STA &37
B20D	6	164 054	A4 36	LDY &36
B20F		136	88	DEY
B210	7	177 055	B1 37	LDA (&37),Y
B212		153 000 006	99 00 06	STA &0600,Y
B215		152	98	TYA
B216		208 247	D0 F7	BNE -9 --> &B20F
B218	`	096	60	RTS
B219	+	165 043	A5 2B	LDA &2B
B21B		240 021	F0 15	BEQ 21 --> &B232 Set SWA to 1-character value
B21D		160 000	A0 00	LDY#&00
B21F	*	177 042	B1 2A	LDA (&2A),Y
B221		153 000 006	99 00 06	STA &0600,Y
B224	I	073 013	49 0D	EOR#&0D
B226		240 004	F0 04	BEQ 4 --> &B22C
B228		200	C8	INY
B229		208 244	D0 F4	BNE -12 --> &B21F
B22B		152	98	TYA
B22C	6	132 054	84 36	STY &36
B22E	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B22F CHR$

Submitted by Steve Fewell

Description:

Call routine &96B4 to evaluate the expression after the CHR$ keyword and set the IWA to the Integer result value of the expression (or issue 'Type mismatch' error if the result was not numeric).

Set A to the IWA's LSB byte (&2A), i.e. 1-byte value specified by the numeric result of the expression.
This value should be the ASCII code of the character required to be returned by the CHR$ function.
Call routine &AE6C to set the SWA to the character specified by the ASCII code (located in A).

Disassembly for the CHR$ routine

B22F		032 180 150	20 B4 96	JSR &96B4 IWA = Integer value at PTR B
B232	*	165 042	A5 2A	LDA &2A
B234	Ll	076 108 174	4C 6C AE	JMP &AE6C Set SWA to 1-byte value (A)

BASIC IV ROM Routines

Page Last Altered: undefined

AE69 GET$

Submitted by Steve Fewell

Description:

Call OSRDCH operating system routine to wait for a key to be pressed on the keyboard
(if the keyboard buffer is empty) and return in A the next character read from the
Keyboard buffer.
Store the ASCII code for the Key pressed in the SWA (first position, &0600) and
set the SWA length (&36 to 1).
Exit with A=0 (as the result is a String).

Disassembly for the GET$ routine

AE69	032 224 255	20 E0 FF	JSR &FFE0 OSRDCH
AE6C	141 000 006	8D 00 06	STA &0600
AE6F	169 001	A9 01	LDA#&01
AE71	128 028	80 1C	BRA 28 --> &AE8F Set SWA length (&36) to A & exit with A = 0 (result is a String value)

Disassembly for the Set SWA length to A and exit with A = 0 routine

AE8F	6 133 054	85 36	STA &36
AE91	169 000	A9 00	LDA#&00
AE93	` 096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B1C2 Load IWA with 1-byte Integer value

Submitted by Steve Fewell

Routine:loadbyte
Name: Load IWA with 1-byte Integer value
Starting Address: &B1C2
Entry criteria: &2A and &2B point to the address of the variable's value.
Y is 0.
Exit: The IWA contains the value of the variable.
The address in (&2A and &2B) is destroyed.

Description:

Load the 1-byte Integer from the address pointed to by (&2A, &2B) and exit
via the IWA = 8-bit Integer routine which sets the Least significant byte of the IWA
to the 1-byte value [A] (&2A), &2B to the value of Y (which is zero), &2C to
zero and &2D (IWA most significant byte) to zero.

Disassembly for the Load IWA with 1-byte Integer value routine

B1C2	*	177 042	B1 2A	LDA (&2A),Y
B1C4	L	076 026 174	4C 1A AE	JMP &AE1A Load IWA with 2-byte value (A and Y)

BASIC IV ROM Routines

Page Last Altered: undefined

B1C7 Load FWA with Float Variable

Submitted by Steve Fewell

Routine:ain
Name: Copy Float Variable to the FWA
Starting Address: &B1C7
Entry criteria: &2A and &2B point to a 5-byte Floating-Point variable to unpack and load.
Y = 5 (size of variable).
Exit: The FWA contains the value of the variable.

Description:

Zero the FWA Rounding and FWA Overflow bytes.
Store the 5th byte of the variable (pointed to by (&2A, &2B)) in FWA Mantissa byte 4.
Store the 4th byte of the variable (pointed to by (&2A, &2B)) in FWA Mantissa byte 3.
Store the 3rd byte of the variable (pointed to by (&2A, &2B)) in FWA Mantissa byte 2.
Store the 2nd byte of the variable (pointed to by (&2A, &2B)) in FWA Sign byte.
Store the 1st byte of the variable (pointed to by (&2A, &2B)) in FWA Exponent byte.

If exponent = 0 and sign = 0 and FWA Man 2 = 0 and FWA Man 3 = 0 and FWA Man 4 = 0 then the number is zero so jump to end of routine (B1F2) to store zero in FWA Mantissa 1 and exit with A = #&FF (indicating a float result).

If exponent is non-zero, copy sign byte to FWA Mantissa byte 1 (setting the top bit, as the sign is now unpacked and held separately from the top byte of the mantissa).
Exit with A=#&FF.

Disassembly for the Load FWA with Float Variable routine

B1C7	d5	100 053	64 35	STZ &35

B1C9	d/	100 047	64 2F	STZ &2F
B1CB		136	88	DEY
B1CC	*	177 042	B1 2A	LDA (&2A),Y
B1CE	4	133 052	85 34	STA &34
B1D0		136	88	DEY
B1D1	*	177 042	B1 2A	LDA (&2A),Y
B1D3	3	133 051	85 33	STA &33
B1D5		136	88	DEY
B1D6	*	177 042	B1 2A	LDA (&2A),Y
B1D8	2	133 050	85 32	STA &32
B1DA		136	88	DEY
B1DB	*	177 042	B1 2A	LDA (&2A),Y
B1DD	.	133 046	85 2E	STA &2E
B1DF		168	A8	TAY
B1E0	*	178 042	B2 2A	LDA (&2A)
B1E2	0	133 048	85 30	STA &30
B1E4		208 009	D0 09	BNE 9 --> &B1EF
B1E6		152	98	TYA
B1E7	2	005 050	05 32	ORA &32
B1E9	3	005 051	05 33	ORA &33
B1EB	4	005 052	05 34	ORA &34
B1ED		240 003	F0 03	BEQ 3 --> &B1F2
B1EF		152	98	TYA
B1F0		009 128	09 80	ORA#&80
B1F2	1	133 049	85 31	STA &31
B1F4		169 255	A9 FF	LDA#&FF
B1F6	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

Load IWA with Integer from Address

Submitted by Steve Fewell

Routine: iin
Name: Load Integer from Address
Starting Address: &B1AA
Entry criteria: &2A and &2B (lo, hi) contain the address of the Least Significant (first) byte of a 32-bit Integer variable.
Exit: IWA contains the 32-bit Integer number pointed to by &2A, &2B.

Description:
Loads the IWA with the 32-bit Integer pointed to by the zero page locations &2A (low byte) and &2B (high byte). The bytes are copied most significant byte first. The 2nd byte (which should go into IWA location &2B) is temporally stored in X, as &2B is also being used as the address pointer, and we don't want to corrupt it before we've read the complete Integer variable!

The routine sets A to #&40 to indicate that an Integer number is being processed before exiting.

Disassembly for the Load IWA with Integer from Address routine

B1AA		160 003	A0 03	LDY#&03
B1AC	*	177 042	B1 2A	LDA (&2A),Y
B1AE	-	133 045	85 2D	STA &2D
B1B0		136	88	DEY
B1B1	*	177 042	B1 2A	LDA (&2A),Y

```
B1B3   ,   133 044     85 2C     STA &2C
B1B5       136         88        DEY
B1B6   *   177 042     B1 2A     LDA (&2A),Y
B1B8       170         AA        TAX
B1B9   *   178 042     B2 2A     LDA (&2A)
B1BB   *   133 042     85 2A     STA &2A
B1BD   +   134 043     86 2B     STX &2B
B1BF   @   169 064     A9 40     LDA#&40
B1C1   `   096         60        RTS
```

BASIC IV ROM Routines

Page Last Altered: undefined

A027 '^' Operator (raise FWA to the Power)

Submitted by Steve Fewell

Description:

Convert value to a Float if it is an Integer. If it is a String, then Type mismatch error.
Push FWA to the Stack.
Obtain second value, convert to Float if it is an Integer. If it is a String, then Type mismatch error.
If the FWA exponent of the Power value is less than #&87 then do the following:

* Call &82E0 to Split the FWA into an Integer value (&49) and a Fractional value (FWA).
* If the Fractional Part of the FWA is 0.00 then Pop the Float back from Stack to the FWA; call &A5BE to raise the FWA to the power of the Integer in &49; and return to level 6 of the expression handler with FWA = the result.
* Otherwise, store the fractional power value in the Temporary Floating-Point variable at &0476.
* Unpack the Float value from the Stack (without popping it, as we will pop it later) into the FWA.
* Raise the FWA to the Integer Part of the Power value (&49).

Otherwise, If the FWA exponent of the Power value is >= #&87 then do the following:

* Store the Power value (in the FWA) in the Temporary Floating-Point variable at &0476.
* Set the FWA = 1.0.

Now we have the power value in &0476 and the FWA set to any result calculated so far.
Next, store the result calculated so far (in the FWA) to Temporary Floating-Point variable at &0471.
Pop the Float from the Stack (the first value - that is the value to raise to the power) to the FWA.
Call LN routine to get the natural logorithm of the Float value.
Multiply the natural logorithm result with the Float Power value (&0476).
Call EXP routine to obtain the exponental of this result.
This gives us the first value raised to the power of the Float Power value.

Next, we need to multiply this result by any result we have already calculated (in &0471), as the Integer part of the power may have already been calculated separately.
Lastly, exit with A = #&FF (as we have a Floating-Point result).

Disassembly for the '^' Operator routine

A027	168	A8	TAY

A028		032 221 150	20 DD 96	JSR &96DD Check Float value (Convert if Integer)
A02B		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
A02E		032 218 150	20 DA 96	JSR &96DA Get & Check Float Value (Convert if Integer)
A031	0	165 048	A5 30	LDA &30
A033		201 135	C9 87	CMP#&87
A035	B	176 066	B0 42	BCS 66 --> &A079
A037		032 224 130	20 E0 82	JSR &82E0 Split FWA into Float/Integer parts
A03A		208 013	D0 0D	BNE 13 --> &A049
A03C		032 232 187	20 E8 BB	JSR &BBE8 Pop Float from Stack
A03F	A	032 065 165	20 41 A5	JSR &A541 Unpack Float variable to FWA
A042	I	165 073	A5 49	LDA &49
A044		032 190 165	20 BE A5	JSR &A5BE Raise FWA to the power of the value in A
A047	,	128 044	80 2C	BRA 44 --> &A075
A049		032 013 165	20 0D A5	JSR &A50D Store FWA to &0476
A04C		165 004	A5 04	LDA &04
A04E	J	133 074	85 4A	STA &4A
A050		165 005	A5 05	LDA &05
A052	K	133 075	85 4B	STA &4B
A054	A	032 065 165	20 41 A5	JSR &A541 Unpack Float variable to FWA
A057	I	165 073	A5 49	LDA &49
A059		032 190 165	20 BE A5	JSR &A5BE Raise FWA to power value in A
A05C	q	169 113	A9 71	LDA#&71
A05E		032 019 165	20 13 A5	JSR &A513 Store FWA to location &0471
A061		032 232 187	20 E8 BB	JSR &BBE8 Pop Float from Stack
A064	A	032 065 165	20 41 A5	JSR &A541 Unpack float variable to FWA
A067	I	032 073 167	20 49 A7	JSR &A749 LN
A06A		032 159 169	20 9F A9	JSR &A99F Multiply FWA by &0476
A06D		032 226 169	20 E2 A9	JSR &A9E2 EXP
A070	q	169 113	A9 71	LDA#&71
A072		032 161 169	20 A1 A9	JSR &A9A1 Multiply FWA by &0400 + A
A075		169 255	A9 FF	LDA#&FF
A077		128 156	80 9C	BRA -100 --> &A015 Expression handler level 6
A079		032 013 165	20 0D A5	JSR &A50D Store FWA to &0476
A07C		032 216 165	20 D8 A5	JSR &A5D8 Set FWA to 1.0
A07F		128 219	80 DB	BRA -37 --> &A05C

BASIC IV ROM Routines

Page Last Altered: undefined

82E0 Split Floating-Point Value into Integer/Fractional part

Submitted by Steve Fewell

Routine:SplitFWA
Name: Split Floating-Point Value
Starting Address: &82E0
Entry criteria: The FWA contains the value to split.
Exit: The FWA contains the fractional value.
&49 contains the 1-byte Integer value.
Z (zero flag) is set if the fractrional value is zero.

Description:

If the FWA exponent is positive then return FWA sign (sets Z flag to 1 if zero) and set &49 to 0, as the Integer part of the number is zero. The number in the FWA is a fractional value (there is no integer part).

Otherwise, the FWA value contains an Integer part.
Clear the FWB (in &8275 routine), and keep incrementing the exponent and moving FWA Mantissa bits right (ROR, divide by 2) moving any overflowed bits into the FWB mantissa. Until the exponent = #&:A0. This removes the fractional part of the FWA Mantissa value into the FWB Mantissa. If the FWA sign byte is set (negative value) then the Integer value in the FWA Mantissa will be complemented (to negate it).

Store FWA Mantissa byte 4 in &49, this is the LSB of the Integer value in the FWA. The rest of the Integer value in the FWA will be lost. &49 now contains the 1-byte Integer value of the Integer part of the FWA value.

Copy the FWB Mantissa to the FWA Mantissa, so that the FWA Mantissa contains the Fractional part of the original value.
Set the FWA exponent to #&80, as the whole of the FWA Mantissa now contains a fractional value.

If the FWA Mantissa 1 is not negative, then Normalise FWA (&81F7) and exit.
(I'm not too sure what the meaning of the next 3 lines is! Anyone know?)
Reverse the FWA sign byte (EOR &2E with #&80).
If the FWA sign is negative then decrement &49 and complement the FWA value.
If the FWA sign is positive then increment &49 and complement the FWA value.

Exit with the required values in the FWA and &49.

Disassembly for the Split FWA into Integer/Fractional parts routine

82E0	0	165 048	A5 30	LDA &30
82E2	0	048 005	30 05	BMI 5 --> &82E9
82E4	dI	100 073	64 49	STZ &49
82E6	L	076 242 163	4C F2 A3	JMP &A3F2 Floating-Point Sign
82E9	u	032 117 130	20 75 82	JSR &8275 Move fractional part of the FWA value to FWB
82EC	4	165 052	A5 34	LDA &34
82EE	I	133 073	85 49	STA &49
82F0	S	032 083 131	20 53 83	JSR &8353 Copy FWB Mantissa + Rounding bytes to FWA
82F3		169 128	A9 80	LDA#&80
82F5	0	133 048	85 30	STA &30
82F7	1	166 049	A6 31	LDX &31
82F9		016 015	10 0F	BPL 15 --> &830A Normalise FWA and exit
82FB	E.	069 046	45 2E	EOR &2E
82FD	.	133 046	85 2E	STA &2E
82FF		016 004	10 04	BPL 4 --> &8305
8301	I	230 073	E6 49	INC &49
8303		128 002	80 02	BRA 2 --> &8307
8305	I	198 073	C6 49	DEC &49
8307		032 200 130	20 C8 82	JSR &82C8 Reverse order complement of FWA
830A	L	076 247 129	4C F7 81	JMP &81F7 Normalise FWA#1

8275: Move fractional part of FWA to FWB

8275	0	165 048	A5 30	LDA &30
8277		016 246	10 F6	BPL -10 --> &826F
8279	p	032 112 165	20 70 A5	JSR &A570 Clear FWB
827C	1	164 049	A4 31	LDY &31
827E				... Jump to Reverse Order Complement

BASIC IV ROM Routines

Page Last Altered: undefined

Clear FWB

Submitted by Steve Fewell

Routine: bclear
Name: Clear FWB
Starting Address: &A570
Entry criteria: None
Exit: Every byte in the FWB contains the value 0 (this indicates the number 0).

Description:
Sets every byte of the FWB (locations &3B to &41) to zero. This uniquely identifies the number zero.

This routine can be jumped in at address A576, in this case all bytes of the FWB are cleared, except for the Sign, Exponent and Mantissa byte 1 (the most significant byte of the mantissa).

Disassembly for the Clear FWB routine

A570	d;	100 059	64 3B	STZ &3B
A572	d<	100 060	64 3C	STZ &3C
A574	d=	100 061	64 3D	STZ &3D
A576	d>	100 062	64 3E	STZ &3E
A578	d?	100 063	64 3F	STZ &3F
A57A	d@	100 064	64 40	STZ &40
A57C	dA	100 065	64 41	STZ &41

A57E ` 096 60 RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A746 LN

Submitted by Steve Fewell

Routine:LN
Name: LN (Natural Logarithm)
Starting Address: &A746
Entry criteria: The FWA contains the value to work with
Exit: The FWA contains the result.

Description:

Get the Numeric value (if integer then convert to Float, if String then Type Mismatch error).
If the number is zero or negative then generate a Log range error.

Set the FWB value to -1.0.
Scale the value in the FWA to a number between 1 and 2. [This is done by setting the FWA exponent to #&81 if the Mantissa byte 1 value is less than #&B5, or to #&80 if the FWA value is 0.0 or the Mantissa 1 value is more than #&B4].
The previous exponent is stored on the Stack (X). This will be used to calculate the difference between the original and scaled exponents later on.
Note: If the FWA value is 0.0 or the Mantissa 1 value is more than #&B4 then the value or the original exponent (on the stack) is incremented, to represent the exponent variance (the actual difference) rather than the exact original value.

Subtract 1 from the FWA value to further reduce the the FWA value to a number between 0 and 1 (a fractional value). [This is done by adding FWB (value=-1) to the FWA].
Store this fractional value in Temporary Floating-Point variable address &047B.

Evaluate the LN continued-fraction expansion series (A861) with X=#&51, A=#&6F and Y=#&02.
Thus the parameters to the evaluate series routine are as follows:
Default value (if FWA value too small to calculate) = &BF6F = -0.5
First Floating-Point constant to use = &BF51.
Number of cycles to evaluate = #&02.

The series is evaluated as follows:
If FWA too small then return -0.5 (&BF6F).
FWA = 1/FWA (store in Temporary Floating-Point variable address &046C)
FWA = 0.546254168 + FWA [Floating-Point constant &BF51]
FWA = -0.0513882861 / FWA [Floating-Point constant &BF56]

FWA = 0.583293331 + FWA [Floating-Point constant &BF5B]
FWA = &046C [1/orig FWA] + FWA
FWA = -0.0374986753 / FWA [Floating-Point constant &BF60]
FWA = 0.750000063 + FWA [Floating-Point constant &BF65]
FWA = &046C [1/orig FWA] + FWA
FWA = 0.33333334 / FWA [Floating-Point constant &BF6A]
FWA = -0.5 + FWA [Floating-Point constant &BF6F]

Multiply the series result by 5 * &047B (the scaled FWA value before series calculation was done)
This is achieved by multiplying the result by &047B twice and then adding &047B.
Store the FWA value in Temporary Floating-Point location &046C. This is the LN result
for the scaled value.

Subtract #&81 from the original exponent before the value was scaled. This will give
the difference between the exponent of the original value and the exponent of the scaled
value that was used for the series calculation.
Place this value into the FWA.
Multiply the FWA (exponent difference) by Floating-Point constant &BF4C (0.693147181).
Add &046C (the result calculated with the scaled value) to the FWA.
The value in the FWA is now the LN value of the original value.

Example 1:
Calculate LN(3.14)
3.14 => Exponent = &82, Mantissa 1 = &48, Mantissa 2 = &F5, Mantissa 3 = &C2, Mantissa 4 = &8F
Change Exponent to &81 to scale the value to 1.57 (exponent difference = 1)
Subtract 1 from the FWA to give 0.57, store this value in &047B.
Calculate reciple -> FWA = 1/FWA = 1.75438596. Store this value in location &046C.
FWA = 0.546254168 + FWA = 2.30064013291
FWA = -0.0513882861 / FWA = -0.0223365164
FWA = 0.583293331 + FWA = 0.56095681458
FWA = &046C + FWA = 2.3153427745823
FWA = -0.0374986753 / FWA = -0.0161957338
FWA = 0.750000063 + FWA = 0.73380432917
FWA = &046C + FWA = 2.488190289171
FWA = 0.33333334 / FWA = 0.133966176723
FWA = -0.5 + FWA = -0.366033823

FWA = FWA * 0.57 (&047B) = -0.20863927926778
FWA = FWA * 0.57 (again) = -0.11892438918
FWA = FWA + 0.57 = 0.451075610756108
This is the LN value of the scaled number (1.57). Store in &046C.
FWA = 1 (exponent difference) * 0.693147181 (&BF4C) = 0.693147181.
FWA = 0.451075610756108 + FWA = 1.14422279. This is the LN value of 3.14.

Example 2:
Calculate LN(15)
15 => Exponent = &84, Mantissa 1 = &70, Mantissa 2 = &00, Mantissa 3 = &00, Mantissa 4 = &00
Change Exponent to &81 to scale the value to 1.875 (exponent difference = 3)
Subtract 1 from the FWA to give 0.875, store this value in &047B.
Calculate reciple -> FWA = 1/FWA = 1.142857142857. Store this value in location &046C.
FWA = 0.546254168 + FWA = 1.689111310857
FWA = -0.0513882861 / FWA = -0.0304232680047
FWA = 0.583293331 + FWA = 0.552870062995
FWA = &046C + FWA = 1.69572720585
FWA = -0.0374986753 / FWA = -0.02211362486288

FWA = 0.750000063 + FWA = 0.727886438137
FWA = &046C + FWA = 1.8707435809941
FWA = 0.33333334 / FWA = 0.17818227114956
FWA = -0.5 + FWA = -0.3218177288504

FWA = FWA * 0.875 (&047B) = -0.28159051274413
FWA = FWA * 0.875 (again) = -0.246391698651115
FWA = FWA + 0.875 = 0.6286083013488848
This is the LN value of the scaled number (1.875). Store in &046C.
FWA = 3 (exponent difference) * 0.693147181 (&BF4C) = 2.079441543.
FWA = 0.6286083013488848 + FWA = 2.708049844. This is the LN value of 15.

Disassembly for the LN routine

A746		032 218 150	20 DA 96	JSR &96DA Get and check Float value
A749		032 242 163	20 F2 A3	JSR &A3F2 Float Sign
A74C		240 002	F0 02	BEQ 2 --> &A750
A74E		016 022	10 16	BPL 22 --> &A766
A750				...Log range and -ve root Error messages...
A766	v	032 118 165	20 76 A5	JSR &A576 Clear FWB Mantissa bytes 2 to 5
A769		160 128	A0 80	LDY#&80
A76B	;	132 059	84 3B	STY &3B
A76D	=	132 061	84 3D	STY &3D
A76F		200	C8	INY
A770	<	132 060	84 3C	STY &3C
A772	0	166 048	A6 30	LDX &30
A774		240 006	F0 06	BEQ 6 --> &A77C
A776	1	165 049	A5 31	LDA &31
A778		201 181	C9 B5	CMP#&B5
A77A		144 002	90 02	BCC 2 --> &A77E
A77C		232	E8	INX
A77D		136	88	DEY
A77E		218	DA	PHX
A77F	0	132 048	84 30	STY &30
A781		032 146 166	20 92 A6	JSR &A692 FWA = FWA + FWB
A784	{	169 123	A9 7B	LDA#&7B
A786		032 019 165	20 13 A5	JSR &A513 Store FWA to &047B
A789	Q	162 081	A2 51	LDX#&51
A78B	o	169 111	A9 6F	LDA#&6F
A78D		160 002	A0 02	LDY#&02
A78F	a	032 097 168	20 61 A8	JSR &A861 Evaluate continued-fraction expansion series
A792	{	169 123	A9 7B	LDA#&7B
A794		032 161 169	20 A1 A9	JSR &A9A1 Multiply FWA by &047B
A797		032 166 166	20 A6 A6	JSR &A6A6 Multiply the FWA by argp [i.e. &047B again]
A79A		032 141 166	20 8D A6	JSR &A68D Add argp to the FWA [argp = &047B]

A79D		032 017 165	20 11 A5	JSR &A511 Store FWA to &046C
A7A0	h	104	68	PLA
A7A1	8	056	38	SEC
A7A2		233 129	E9 81	SBC#&81
A7A4		032 213 129	20 D5 81	JSR &81D5 Set FWA to 1-byte value (i.e. FWA = the exponent difference)
A7A7	L	169 076	A9 4C	LDA#&4C
A7A9		032 212 169	20 D4 A9	JSR &A9D4 Multply FWA by &BF4C
A7AC		032 146 165	20 92 A5	JSR &A592 Set argp to &046C
A7AF		032 141 166	20 8D A6	JSR &A68D FWA = FWA + argp [&046C]
A7B2		169 255	A9 FF	LDA#&FF
A7B4	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A90E COS

Submitted by Steve Fewell

Description:

On entry to the COS routine, the carry flag will be set - this indicates that the Cosine result is required (a clear carry flag indicates that the Sine result is required!).

Store the 6502 flags to the 6502 stack, so that the carry flag can be checked later.
Call routine &A93A to obtain the Floating-Point argument and calculate the Sine or Cosine value (the value returned will be either COS or SIN - depending on the value of the Floating-Point argument.

Routine &A93A does the following:

* Call routine &96DA to obtain the value at the Text pointer B location (convert it to Float if it is Integer or issue 'Type mismatch' error if it is a String value).
* If the FWA exponent (location &30) is more than or equal to #&98 then issue 'Accuracy lost' error as the FWA value is too large to use to calculate the SIN/COS value.
* Store the FWA value to temporary Floating point value location &046C-&0470.
* Set the argument pointer (&4A-&4B) to &BF2E (which points to the Float value 1.57079633 in the Floating-Point constant table).
* Call &A4E0 to unpack the Floating-Point constant at &BF2E (1.57079633, which is PI/2) to the FWB
* Store the FWA sign (location &2E) in the FWB sign byte (location &3B) so that the PI/2 value has the same sign as the argument (the FWA value).
* Decrement the FWB exponent (location &3C) to divide the FWB value by 2 - so the FWB now contains the value 0.785398165.
* Call routine &A692 to add the FWB value (PI/4) to the FWA value.
* Set A to #&33.
* Call routine &A9D4 to set the argument pointer (&4A-&4B) to &BF00 + A = &BF33 (which points to the Float value 0.636619772 (which is 2/PI) in the Floating-Point constant table) and then multiply the FWA by the Float value pointed to by the argument pointer (&4A-&4B).
* Call routine &96C3 to convert the Float value (FWA) to an Integer and place the result in the IWA.
* Store A (the LSB of the IWA value - from location &2A) in location &49.
* If the Integer value (&2A-&2C) is zero then jump to &A98D (as the FWA value does not need to be reduced to a value within the required range -(PI/4) to (PI/4) [i.e. -0.785398 to 0.785398]), so [&A98D] load the FWA with the original Sine/Cosine argument (from location &046C-&0470).
* Otherwise, the FWA value is out of the Sine/Cosine range, and needs to be reduced, so:
    * Call &8189 to convert the Integer value (IWA) to a Float and store the result in the temporary Floating-point variable located at &0471-&0475.

    * Then set the argp pointer (&4A-&4B) to pointer to &BF24 (which is the Floating-point constant value -1.57080078) and multiply the FWA by this constant.
    * Set the argp pointer (&4A-&4B) to point to the temporary Floating-point variable location &046C (which is the original FWA Sine/Cosine argument value) and add the Floating-Point value at &046C-&0470 to the FWA value.
    * Store the result in temporary Floating-Point value location &046C-&0470.
    * Set the FWA to the Floating-Point value stored at location &0471-&0475 (the IWA value).
    * Set the argp pointer (&4A-&4B) to point to &BF29 (the address of the value 0.00000445445511 in the Floating-Point constant table).
    * Multiply the FWA by the Floating-Point constant 0.00000445445511 (store the result in the FWA).
    * Set the argp pointer (&4A-&4B) to &046C and add the Floating-Point value at &046C-&0470 to the FWA.
    * Jump to &A990 to perform the Sine/Cosine calculation as the value is now within the required range.
* [A990] Store the FWA value (the actual Sine/Cosine argument, which has been reduced where necessary) to location &0476-&047A.
* Multiply the FWA by the Floating-point value at &0476-&047A (i.e. square the FWA value).
* Set X to #&74 (i.e. the LSB of the first fraction to apply)
* Set A to #&92 (i.e. the LSB of the default value (if the FWA value is too small to evaluate)
* Set Y to #&02 (the number of continued-fraction expansion cycles to evaluate)
* Call routine &A861 to evaluate the continued-fraction expansions, as follows:
    * If the FWA exponent is less than #&40 then the FWA value is too small to evaluate, so exit with the FWA set to the Floating-Point constant at &BF92 (i.e. 1.0).
    * Calculate the reciple of the FWA value (FWA = 1/FWA) and store the result in the FWA and location &046C-&0470
    * Add the Floating-Point constant at &BF74 (-0.0119090311) to the FWA
    * **Cycle 1:**
    * Divide the Floating-Point constant at &BF79 (0.000107499459) by the FWA, storing the result in the FWA.
    * Add the Floating-Point constant at &BF7E (-0.0171640246) to the FWA.
    * Add the Reciple value (from location &046C-&0470) to the FWA.
    * **Cycle 2:**
    * Divide the Floating-Point constant at &BF83 (0.0013095369) by the FWA, storing the result in the FWA.
    * Add the Floating-Point constant at &BF88 (0.0499999922) to the FWA.
    * Add the Reciple value (from location &046C-&0470) to the FWA.

    Last Cycle calculation:
    * Divide the Floating-Point constant at &BF8D (-0.166666666) by the FWA, storing the result in the FWA.
    * Add the Floating-Point constant at &BF92 (1.0) to the FWA.
* Multiply the FWA by the Temporary Floating-Point variable stored at location &0476-&047A (the argument value).
* Exit with A = #&FF (as the result is a Floating-Point value in the FWA)

Now, the FWA contains either the Sine result or the Cosine result (which will have to be converted to the value requested by the user).

Retrieve the carry flag value from the 6502 stack.
As the carry flag is set (Cosine result required), increment the value stored at location &49 (the integer part of the FWA argument value).

[&A923] If bit 0 of location &49 is clear (i.e. the original argument was an even value (in the case of Sine) - or an odd value (in the case of Cosine)) then Set A to #&FF (to indicate that the result is a Floating-point value) and [&A917] check the value in location &49 - if bit 1 is not set then complement the FWA value (FWA=-FWA) and exit; otherwise, just exit.

Otherwise, bit 0 of location &49 is set (indicating that the original argument was odd (in the case of Sine) or even (in the case of Cosine); so, adjust the calculated result as follows:

* Square the FWA value (FWA = FWA * FWA),
* Subtract the FWA value from 1 (FWA=1.0-FWA) and
* Set the FWA value to the square root of the FWA (FWA=SQR(FWA)).

[&A917] check the value in location &49 - if bit 1 is not set then complement the FWA value (FWA=-FWA) and exit; otherwise, just exit.

Example 1: COS(1.5)

Set &046C = 1.5 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 2.285398
FWA = FWA * 0.636619772 = 1.4549294
?&49 = 1 (Integer part of FWA value). FWA = IWA = 1
Store the FWA in &0471
FWA = FWA * -1.57080078 = -1.57080078
FWA = FWA + &046C = -0.07080078
Store FWA in &046C
Store &0471 (1) in FWA
FWA = FWA * 0.0000044544511 = 0.0000044544511
FWA = FWA + &046C = -0.0707963255
Store the FWA in &0476
FWA = FWA * FWA = 0.0050121197
[&A861]: FWA = 1 / FWA = 199.51638
Store FWA in &046C
FWA = FWA + -0.0119090311 = 199.504474778
FWA = 0.000107499 / FWA = 0.00000005388
FWA = -0.017164024 + FWA = -0.017163485
FWA = &046C + FWA = 199.499216515
FWA = 0.0013095369 / FWA = 0.000006564
FWA = 0.04999999 + FWA = 0.050006554
FWA = &046C + FWA = 199.566387
FWA = -0.16666666 / FWA = -0.00083514
FWA = 1 + FWA = 0.999164856
FWA = FWA * &0476 = -0.070737200377

Increment &49 -> now &49 = 2
Bit 0 of &49 is clear and Bit 1 of &49 is set, indicating that the FWA value needs to be complemented - so exit with FWA = - FWA = 0.070737200377

Example 2: COS(-0.75)

Set &046C = -0.75 (the argument)
Set FWB to -0.785398165 -(PI/4)
FWA = FWA + FWB = -1.535398165
FWA = FWA * 0.636619772 = -0.9774646
?&49 = 0 (Integer part of FWA value). FWA = IWA = 0
[&A98D] As &49 is 0, store value at &046C in FWA = -0.75
Store the FWA in &0476
FWA = FWA * FWA = 0.5625
[&A861]: FWA = 1 / FWA = 1.777777
Store FWA in &046C

FWA = FWA + -0.0119090311 = 1.7896868
FWA = 0.000107499 / FWA = 0.00006
FWA = -0.017164024 + FWA = -0.0171039
FWA = &046C + FWA = 1.7606737
FWA = 0.0013095369 / FWA = 0.0007437
FWA = 0.04999999 + FWA = 0.0507436
FWA = &046C + FWA = 1.8285213
FWA = -0.16666666 / FWA = -0.0911482
FWA = 1 + FWA = 0.9088517
FWA = FWA * &0476 = -0.6816387

Increment &49 -> now &49 = 1
Bit 0 of &49 is set, so adjust the calculated value as follows:
    FWA = FWA * FWA = 0.4646313
    FWA = 1 - FWA = 0.5353686
    FWA = SQR(FWA) = 0.7316889
Bit 1 of &49 is not set, so exit with FWA unchanged.

Example 3: COS(0.0)

Set &046C = 0.0 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 0.785398165
FWA = FWA * 0.636619772 = 0.50000000
?&49 = 0 (Integer part of FWA value). FWA = IWA = 0
[&A98D] As &49 is 0, store value at &046C in FWA = 0.0
Store the FWA in &0476
FWA = FWA * FWA = 0.0
[&A861]: FWA = 1.0 (as argument is zero)
FWA = FWA * &0476 = 0.0

Increment &49 -> now &49 = 1
Bit 0 of &49 is set, so adjust the calculated value as follows:
    FWA = FWA * FWA = 0.0
    FWA = 1 - FWA = 1.0
    FWA = SQR(FWA) = 1.0
Bit 1 of &49 is not set, so exit with FWA unchanged.

Example 4: COS(0.25)

Set &046C = 0.25 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 1.0353982
FWA = FWA * 0.636619772 = 0.6591548
?&49 = 0 (Integer part of FWA value). FWA = IWA = 0
[&A98D] As &49 is 0, store value at &046C in FWA = 0.25
Store the FWA in &0476
FWA = FWA * FWA = 0.0625
[&A861]: FWA = 1 / FWA = 16
Store FWA in &046C
FWA = FWA + -0.0119090311 = 15.988091

FWA = 0.000107499 / FWA = 0.0000067
FWA = -0.0171640246 + FWA = -0.0171572
FWA = &046C + FWA = 15.982843
FWA = 0.0013095369 / FWA = 0.0000819
FWA = 0.04999999 + FWA = 0.0500818
FWA = &046C + FWA = 16.050082
FWA = -0.16666666 / FWA = -0.0103841
FWA = 1 + FWA = 0.9896158
FWA = FWA * &0476 = 0.2474039

Increment &49 -> now &49 = 1
Bit 0 of &49 is set, so adjust the calculated value as follows:
    FWA = FWA * FWA = 0.0612086
    FWA = 1 - FWA = 0.9387913
    FWA = SQR(FWA) = 0.9689124
Bit 1 of &49 is not set, so exit with FWA unchanged.

Example 5: COS(2.41)

Set &046C = 2.41 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 3.195398
FWA = FWA * 0.636619772 = 2.0342533
?&49 = 2 (Integer part of FWA value). FWA = IWA = 2
Store the FWA in &0471
FWA = FWA * -1.57080078 = -3.1416014
FWA = FWA + &046C = -0.7316014
Store FWA in &046C
Store &0471 (2) in FWA
FWA = FWA * 0.0000044544511 = 0.0000089
FWA = FWA + &046C = -0.7315924
Store the FWA in &0476
FWA = FWA * FWA = 0.5352275
[&A861]: FWA = 1 / FWA = 1.8683641
Store FWA in &046C
FWA = FWA + -0.0119090311 = 1.8564551
FWA = 0.000107499 / FWA = 0.0000057
FWA = -0.017164024 + FWA = -0.0171583
FWA = &046C + FWA = 1.8512058
FWA = 0.0013095369 / FWA = 0.0007073
FWA = 0.04999999 + FWA = 0.0507072
FWA = &046C + FWA = 1.9190714
FWA = -0.16666666 / FWA = -0.0868475
FWA = 1 + FWA = 0.9131524
FWA = FWA * &0476 = -0.6680554

Increment &49 -> now &49 = 3
Bit 0 of &49 is set, so adjust the calculated value as follows:
    FWA = FWA * FWA = 0.446298
    FWA = 1 - FWA = 0.5537019
    FWA = SQR(FWA) = 0.7441115
Bit 1 of &49 is set, so FWA = - FWA = -0.7441115

Example 6: COS(5.63)

Set &046C = 5.63 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 6.415398
FWA = FWA * 0.636619772 = 4.0841688
?&49 = 4 (Integer part of FWA value). FWA = IWA = 4
Store the FWA in &0471
FWA = FWA * -1.57080078 = -6.2832028
FWA = FWA + &046C = -0.6532028
Store FWA in &046C
Store &0471 (4) in FWA
FWA = FWA * 0.0000044544511 = 0.0000178
FWA = FWA + &046C = -0.6531849
[&A990] Store the FWA in &0476
FWA = FWA * FWA = 0.4266506
[&A861]: FWA = 1 / FWA = 2.3438381
Store FWA in &046C
FWA = FWA + -0.0119090311 = 2.3319291
FWA = 0.000107499 / FWA = 0.000046
FWA = -0.017164024 + FWA = -0.0171179
FWA = &046C + FWA = 2.3267202
FWA = 0.0013095369 / FWA = 0.0056282
FWA = 0.04999999 + FWA = 0.0556281
FWA = &046C + FWA = 2.3994662
FWA = -0.16666666 / FWA = -0.0694598
FWA = 1 + FWA = 0.9305401
FWA = FWA * &0476 = -0.6078147

Increment &49 -> now &49 = 5
Bit 0 of &49 is set, so adjust the calculated value as follows:
    FWA = FWA * FWA = 0.3694387
    FWA = 1 - FWA = 0.6305612
    FWA = SQR(FWA) = 0.7940788
Bit 1 of &49 is not set, so exit with FWA unchanged.

Example 7: COS(90)

Set &046C = 90 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 90.785398
FWA = FWA * 0.636619772 = 57.795773
?&49 = 57 (Integer part of FWA value). FWA = IWA = 57
Store the FWA in &0471
FWA = FWA * -1.57080078 = -89.53564
FWA = FWA + &046C = 0.4643601
Store FWA in &046C
Store &0471 (57) in FWA
FWA = FWA * 0.0000044544511 = 0.0002539
FWA = FWA + &046C = 0.464614
Store the FWA in &0476

FWA = FWA * FWA = 0.2158661
[&A861]: FWA = 1 / FWA = 4.6324998
Store FWA in &046C
FWA = FWA + -0.0119090311 = 4.6205908
FWA = 0.000107499 / FWA = 0.0000023
FWA = -0.017164024 + FWA = -0.0171616
FWA = &046C + FWA = 4.6153381
FWA = 0.0013095369 / FWA = 0.0002837
FWA = 0.04999999 + FWA = 0.0502836
FWA = &046C + FWA = 4.6827834
FWA = -0.16666666 / FWA = -0.0355913
FWA = 1 + FWA = 0.9644086
FWA = FWA * &0476 = 0.4480777

Increment &49 -> now &49 = 58 (0011 1010)
Bit 0 of &49 is clear and Bit 1 of &49 is set, indicating that the FWA value needs to be complemented - so exit with FWA = - FWA = -0.4480777

This diagram shows the relationship between the trig functions.

Disassembly for the COS routine

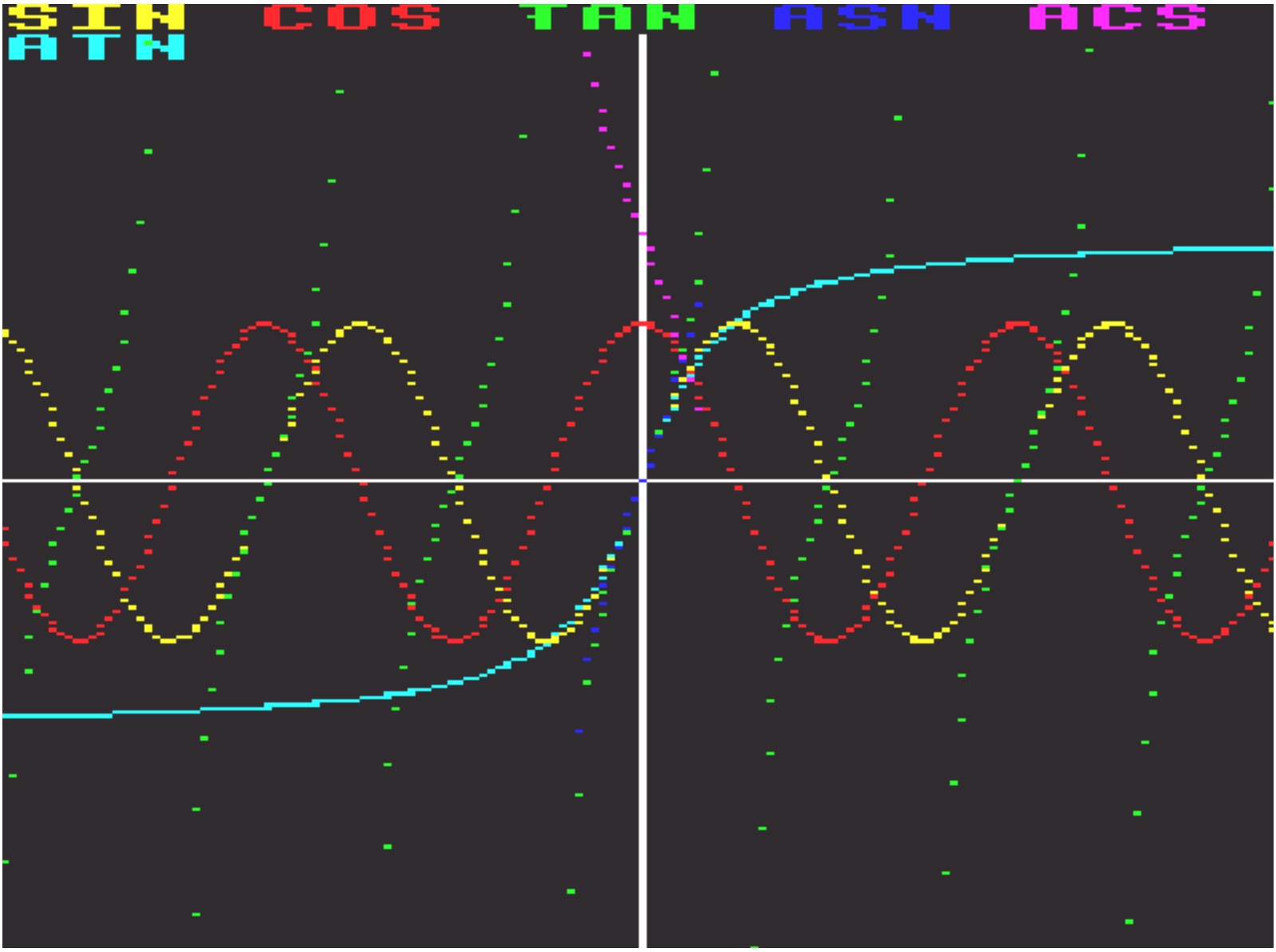
A90E		008	08	PHP
A90F	:	032 058 169	20 3A A9	JSR &A93A Get value and calculate Sine or Cosine wave
A912	(	040	28	PLP
A913		144 002	90 02	BCC 2 --> &A917
A915	I	230 073	E6 49	INC &49
A917	I	165 073	A5 49	LDA &49
A919		137 002	89 02	BIT#&02
A91B		240 006	F0 06	BEQ 6 --> &A923
A91D	#	032 035 169	20 23 A9	JSR &A923
A920	L	076 202 172	4C CA AC	JMP &ACCA Compliment FWA value [FWA=-FWA]
A923	J	074	4A	LSR A
A924		176 003	B0 03	BCS 3 --> &A929
A926		169 255	A9 FF	LDA#&FF
A928	`	096	60	RTS
A929		032 017 165	20 11 A5	JSR &A511 Store FWA to &046C & set argp=&046C
A92C		032 166 166	20 A6 A6	JSR &A6A6 Multiply the FWA by argp [i.e. &046C]
A92F		169 146	A9 92	LDA#&92
A931		032 139 165	20 8B A5	JSR &A58B Set argp to &BF00 + A
A934		032 138 166	20 8A A6	JSR &A68A Floating-Point Subtraction [FWA=argp-FWA]
A937	L	076 184 167	4C B8 A7	JMP &A7B8 Floating-point Square Root [FWA = SQRT(FWA)]
A93A		032 218 150	20 DA 96	JSR &96DA Get and Check Float (convert if Int)
A93D	0	165 048	A5 30	LDA &30
A93F		201 152	C9 98	CMP#&98

A941	j	176 106	B0 6A	BCS 106 --> &A9AD 'Accuracy lost' error
A943		032 017 165	20 11 A5	JSR &A511 Store FWA to &046C & set argp=&046C
A946		032 137 165	20 89 A5	JSR &A589 Set argp to &BF2E
A949		032 224 164	20 E0 A4	JSR &A4E0 Unpack (&4A, &4B) variable to FWB
A94C	.	165 046	A5 2E	LDA &2E
A94E	;	133 059	85 3B	STA &3B
A950	<	198 060	C6 3C	DEC &3C
A952		032 146 166	20 92 A6	JSR &A692 FWA = FWA + FWB (PI/4)
A955	3	169 051	A9 33	LDA#&33
A957		032 212 169	20 D4 A9	JSR &A9D4 Multiply FWA by &BF33 (0.636619772 - 2/PI)
A95A		032 195 150	20 C3 96	JSR &96C3 Convert Float to Integer
A95D	I	133 073	85 49	STA &49
A95F	+	005 043	05 2B	ORA &2B
A961	,	005 044	05 2C	ORA &2C
A963	(	240 040	F0 28	BEQ 40 --> &A98D
A965		032 137 129	20 89 81	JSR &8189 Convert Integer value (IWA) to Floating-Point (FWA)
A968	q	169 113	A9 71	LDA#&71
A96A		032 019 165	20 13 A5	JSR &A513 Store FWA to &047B
A96D	$	169 036	A9 24	LDA#&24
A96F		032 212 169	20 D4 A9	JSR &A9D4 Multiply FWA by &BF24 (-1.57080078)
A972		032 146 165	20 92 A5	JSR &A592 Set argp to &046C
A975		032 141 166	20 8D A6	JSR &A68D Floating-Point Addition [FWA=argp+FWA]
A978		032 025 165	20 19 A5	JSR &A519 Store FWA to argp address (&4A-&4B)
A97B	q	169 113	A9 71	LDA#&71
A97D	;	032 059 165	20 3B A5	JSR &A53B Load FWA from variable at &0400 + A (i.e. &0471)
A980	)	169 041	A9 29	LDA#&29
A982		032 212 169	20 D4 A9	JSR &A9D4 Multiply FWA by &BF29 (0.00000445445511)
A985		032 146 165	20 92 A5	JSR &A592 Set argp to &046C
A988		032 141 166	20 8D A6	JSR &A68D Floating-Point Addition [FWA=argp+FWA]
A98B		128 003	80 03	BRA 3 --> &A990
A98D	9	032 057 165	20 39 A5	JSR &A539 Load FWA from &046C
A990		032 013 165	20 0D A5	JSR &A50D Store FWA to &0476 and set argp to &0476
A993		032 166 166	20 A6 A6	JSR &A6A6 Multiply the FWA by argp [i.e. &0476]
A996	t	162 116	A2 74	LDX#&74
A998		169 146	A9 92	LDA#&92
A99A		160 002	A0 02	LDY#&02
A99C	a	032 097 168	20 61 A8	JSR &A861 Evaluate continued-fraction expansion series
A99F	v	169 118	A9 76	LDA#&76

A9A1 Multiply FWA by &0400 + A

A9A1		160 004	A0 04	LDY#&04
A9A3	K	132 075	84 4B	STY &4B

A9A5	J	133 074	85 4A	STA &4A
A9A7		032 166 166	20 A6 A6	JSR &A6A6 Multiply the FWA by argp [i.e. &0400 + A]
A9AA		169 255	A9 FF	LDA#&FF
A9AC	`	096	60	RTS


SIN
COS
TAN
ASN
ACS
ATN

BASIC IV ROM Routines

Page Last Altered: undefined

A7B5 SQR

Submitted by Steve Fewell

Description:

Call routine &96DA to get the value from text pointer B generating
Type Mismatch error is the value is a String, and converting any Integer value to a Floating-Point value.
If the sign of the Floating-Point value is 0 (meaning FWA = 0.0) then exit with FWA unchanged (as result = 0.0) and exit with A = &FF (as current value is Floating-Point).
If the sign of the Floating-Point value is negative (meaning FWA is negative) then generate a '-ve root' error as SQR cannot evaluate negative values.

Divide the FWA exponent by 2 & add #&41 to obtain a 'guess value' (usually FWA value divided by 2?). If overflow occurred (LSB bit was lost from Exponent) the divide the FWA mantissa by 2 (to make up for the lost bit, and retain the required value).

Initialise the Result Mantissa value (&3D to &41 [FWB Mantissa]) to #&40.
Initialise the Test Mantissa value (&42 to &46) to #&40.
Start by subtracting #&40 from the FWA Mantissa (Byte 1).

Loop for each bit in the result value (&3D-&41):
During this loop X will specify which byte the current bit is in (i.e. #&FB specifies the first byte (&3D) and #&FF specifies the last/5th byte (&41)).
Also, Y specifies the current bit number within the current byte.
For the first byte, only the lower 5 bits are tested (Y starts at #&10 meaning start with bit 4, then test bit 3, and then bit 2, and then bit 1 and lastly bit 0)
[as each loop affects the value of the previous bit, the value of the first 6 bits of the first byte of the result could be changed].
for all other bytes (&3E to &41), all bits are tested, MSB first.
Only 1 bit of the result is processed at a time.

For each bit (each loop iteration), the following processing is done:

*1) Copy the current Result value to the Test value (as only the current byte can change, then only the current byte is copied). Set the current bit position (the bit we are currently testing) in the Test Value (i.e. set the relevent bit to test). So: &42-&46 is an exact copy of the current result (&3D-&41) with the addition that the current bit is set (i.e. the "test bit").ÿ(br>

*2) Compare the FWA Mantissa with the Test value (byte by byte, stopping at the first difference). If the FWA Mantissa >&eq; Test value then subtract the test value from the

FWA mantissa, and set the previous bit in the result (this works even if the previous bit is in the previous byte!). Update any change to the Test value (overwriting the previous 'Result value + "test bit"' contents of the Temp value. If the previous byte was changed, then also update the current byte value to the test result (to set back the "test bit").

If the FWA Mantissa < Test value then do nothing.

*3) Multiply the FWA Mantissa by 2.

*4) Loop again to test the next bit, or if we have reached the end of the last byte (&41), then we have finished.

When we have completed the loop, copy the FWB Mantissa (the result value) to the FWA Mantissa & [&A854] normalise the value (if necessary). Round the FWA Mantissa value to 4 bytes (loosing the rounding byte) and exit with A=&FF (indicating a Floating-Point result).

I am not exactly sure how this routine works, so if anyone else can provide a better description or an example of the algorithum using decimal (base 10) values, then please do so.

Examples:

Example 1:
FWA = 9 (Exponent = &84, Mantissa = &90).
"Guess value" = 4.5 (Exponent = &83, Mantissa = &90).
FWB Mantissa = &40
&42-&46 = &40
Subtract &40 from FWA Mantissa (FWA => Exponent = &83, Mantissa = &50).

Current Byte = &3D (X = &FB), Current Bit = 4 (Y = &10)
Test value = &50 (&40 + &10 (bit 4)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA => Exponent = &83, Mantissa = &00.
Set previous Result bit; Result = &60 (&40 + &20 (previous bit position))
FWA = FWA * 2; FWA => Exponent = &83, Mantissa = &00.

Current Byte = &3D (X = &FB), Current Bit = 3 (Y = &08)
Test value = &68 (&60 + &08 (bit 3)).
FWA Mantissa is < Test Value, so do nothing.
FWA = FWA * 2; FWA => Exponent = &83, Mantissa = &00.

... From now onwards the FWA Mantissa value is always less than the test value, so we can stop here as the result will not alter now. [Note: BASIC does not stop, it keeps going until all the bits have been tested).
Copy FWB Mantissa to FWA Mantissa; FWA => Exponent = &83, Mantissa = &60.
Normalise FWA and exit with FWA => Exponent = &82, Mantissa = &C0 (i.e. 3.0) which is the SQR of 9.

Example 2:
FWA = 28.512 (Exponent = &85, Mantissa = &E418937500).
"Guess value" = 14.256 (Exponent = &84, Mantissa = &720C49BA80).
FWB Mantissa = &40
&42-&46 = &40
Subtract &40 from FWA Mantissa (FWA => Exponent = &84, Mantissa = &320C49BA80).

Current Byte = &3D (X = &FB), Current Bit = 4 (Y = &10)

Test value = &50 (&40 + &10 (bit 4)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &6418937500

Current Byte = &3D (X = &FB), Current Bit = 3 (Y = &08)
Test value = &48 (&40 + &08 (bit 3)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &1C18937500.
Set previous Result bit; Result = &50 (&40 + &10 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &383126EA00

Current Byte = &3D (X = &FB), Current Bit = 2 (Y = &04)
Test value = &54 (&50 + &04 (bit 2)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &70624DD400

Current Byte = &3D (X = &FB), Current Bit = 1 (Y = &02)
Test value = &52 (&50 + &02 (bit 1)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &1E624DD400.
Set previous Result bit; Result = &54 (&50 + &04 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &3CC49BA800

Current Byte = &3D (X = &FB), Current Bit = 0 (Y = &01)
Test value = &55 (&54 + &01 (bit 0)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &7989375000

Current Byte = &3E (X = &FC), Current Bit = 7 (Y = &80)
Test value = &5480 (&5400 + &80 (bit 7)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &2509375000.
Set previous Result bit; Result = &5500 (&5400 + &100 (previous bit position in previous byte))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &4A126EA000

Current Byte = &3E (X = &FC), Current Bit = 6 (Y = &40)
Test value = &5540 (&5500 + &40 (bit 6)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &9424DD4000

Current Byte = &3E (X = &FC), Current Bit = 5 (Y = &20)
Test value = &5520 (&5500 + &20 (bit 5)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &3F04DD4000.
Set previous Result bit; Result = &5540 (&5500 + &40 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &7E09BA8000

Current Byte = &3E (X = &FC), Current Bit = 4 (Y = &10)
Test value = &5550 (&5540 + &10 (bit 4)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &28B9BA8000.
Set previous Result bit; Result = &5560 (&5540 + &20 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &5173750000

Current Byte = &3E (X = &FC), Current Bit = 3 (Y = &08)
Test value = &5568 (&5560 + &08 (bit 3)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &A2E6EA0000

Current Byte = &3E (X = &FC), Current Bit = 2 (Y = &04)
Test value = &5564 (&5560 + &04 (bit 2)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &4D82EA0000.
Set previous Result bit; Result = &5568 (&5560 + &08 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &9B05D40000

Current Byte = &3E (X = &FC), Current Bit = 1 (Y = &02)
Test value = &556A (&5568 + &02 (bit 1)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &459BD40000.
Set previous Result bit; Result = &556C (&5568 + &04 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &8B37A80000

Current Byte = &3E (X = &FC), Current Bit = 0 (Y = &01)
Test value = &556D (&556C + &01 (bit 0)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &35CAA80000.
Set previous Result bit; Result = &556E (&556C + &02 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &6B95500000

Current Byte = &3F (X = &FD), Current Bit = 7 (Y = &80)
Test value = &556E80 (&556E00 + &80 (bit 7)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &1626D00000.
Set previous Result bit; Result = &556F00 (&556E00 + &100 (previous bit position (in previous byte)))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &2C4DA00000

Current Byte = &3F (X = &FD), Current Bit = 6 (Y = &40)
Test value = &556F40 (&556F00 + &40 (bit 6)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &589B400000

Current Byte = &3F (X = &FD), Current Bit = 5 (Y = &20)
Test value = &556F20 (&556F00 + &20 (bit 5)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa

FWA Mantissa = &032C200000.
Set previous Result bit; Result = &556F40 (&556F00 + &40 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &0658400000

Current Byte = &3F (X = &FD), Current Bit = 4 (Y = &10)
Test value = &556F50 (&556F40 + &10 (bit 4)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &0CB0800000

Current Byte = &3F (X = &FD), Current Bit = 3 (Y = &08)
Test value = &556F48 (&556F40 + &08 (bit 3)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &1961000000

Current Byte = &3F (X = &FD), Current Bit = 2 (Y = &04)
Test value = &556F44 (&556F40 + &04 (bit 2)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &32C2000000

Current Byte = &3F (X = &FD), Current Bit = 1 (Y = &02)
Test value = &556F42 (&556F40 + &02 (bit 1)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &6584000000

Current Byte = &3F (X = &FD), Current Bit = 0 (Y = &01)
Test value = &556F41 (&556F40 + &01 (bit 0)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &1014BF0000.
Set previous Result bit; Result = &556F42 (&556F40 + &02 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &20297E0000

Current Byte = &40 (X = &FE), Current Bit = 7 (Y = &80)
Test value = &556F4280 (&556F4200 + &80 (bit 7)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &4052FC0000

Current Byte = &40 (X = &FE), Current Bit = 6 (Y = &40)
Test value = &556F4240 (&556F4200 + &40 (bit 6)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &80A5F80000

Current Byte = &40 (X = &FE), Current Bit = 5 (Y = &20)
Test value = &556F4220 (&556F4200 + &20 (bit 5)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &2B36B5E000.
Set previous Result bit; Result = &556F4240 (&556F4200 + &40 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &566D6BC000

Current Byte = &40 (X = &FE), Current Bit = 4 (Y = &10)
Test value = &556F4250 (&556F4240 + &10 (bit 4)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &00FE297000.
Set previous Result bit; Result = &556F4260 (&556F4240 + &20 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &01FC52E000

Current Byte = &40 (X = &FE), Current Bit = 3 (Y = &08)
Test value = &556F4268 (&556F4260 + &08 (bit 3)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &03F8A5C000

Current Byte = &40 (X = &FE), Current Bit = 2 (Y = &04)
Test value = &556F4264 (&556F4260 + &04 (bit 2)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &07F14B8000

Current Byte = &40 (X = &FE), Current Bit = 1 (Y = &02)
Test value = &556F4262 (&556F4260 + &02 (bit 1)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &0FE2970000

Current Byte = &40 (X = &FE), Current Bit = 0 (Y = &01)
Test value = &556F4261 (&556F4260 + &01 (bit 0)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &1FC52E0000

Current Byte = &41 (X = &FF), Current Bit = 7 (Y = &80)
Test value = &556F426080 (&556F426000 + &80 (bit 7)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &3F8A5C0000

Current Byte = &41 (X = &FF), Current Bit = 6 (Y = &40)
Test value = &556F426040 (&556F426000 + &40 (bit 6)).
FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &7F14B80000

Current Byte = &41 (X = &FF), Current Bit = 5 (Y = &20)
Test value = &556F426020 (&556F426000 + &20 (bit 5)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &29A5759FE0.
Set previous Result bit; Result = &556F426040 (&556F426000 + &40 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &534AEB3FC0

Current Byte = &41 (X = &FF), Current Bit = 4 (Y = &10)
Test value = &556F426050 (&556F426040 + &10 (bit 4)).

FWA Mantissa is < Test Value, so nothing to update to Result
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &A695D67F80

Current Byte = &41 (X = &FF), Current Bit = 3 (Y = &08)
Test value = &556F426048 (&556F426040 + &08 (bit 3)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &5126941F38.
Set previous Result bit; Result = &556F426050 (&556F426040 + &10 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &A24D283E70

Current Byte = &41 (X = &FF), Current Bit = 2 (Y = &04)
Test value = &556F426054 (&556F426050 + &04 (bit 2)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &4CDDE5DE1C.
Set previous Result bit; Result = &556F426058 (&556F426050 + &08 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &99BBCBBC38

Current Byte = &41 (X = &FF), Current Bit = 1 (Y = &02)
Test value = &556F42605A (&556F426058 + &02 (bit 1)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &444C895BDE.
Set previous Result bit; Result = &556F42605C (&556F426058 + &04 (previous bit position))
FWA Mantissa = FWA Mantissa * 2; FWA Mantissa = &889912B7BC

Current Byte = &41 (X = &FF), Current Bit = 0 (Y = &01)
Test value = &556F42605D (&556F42605C + &01 (bit 0)).
FWA Mantissa is >&eq; Test Value, so subtract test value from FWA Mantissa
FWA Mantissa = &3329D0575F.
Set previous Result bit; Result = &556F42605E (&556F42605C + &02 (previous bit position))

FWA => (Exponent = &84, Mantissa = &556F42605E).
Normalise & Round (Round Mantissa to 4-bytes) FWA. FWA => (Exponent = &83, Mantissa = &AADE84C100).
FWA Value (in decimal) = 5.33966291 which is the SQR of 28.512.

Disassembly for the SQR routine

A7B5		032 218 150	20 DA 96	JSR &96DA Get and Check Float value
A7B8		032 242 163	20 F2 A3	JSR &A3F2 Floating-Point Sign
A7BB		240 245	F0 F5	BEQ -11 --> &A7B2 [LDA#&FF : RTS]
A7BD	0	048 156	30 9C	BMI -100 --> &A75B -ve root error
A7BF	0	165 048	A5 30	LDA &30
A7C1	J	074	4A	LSR A
A7C2		008	08	PHP
A7C3	iA	105 065	69 41	ADC#&41
A7C5	0	133 048	85 30	STA &30
A7C7	(	040	28	PLP

A7C8		144 010	90 0A	BCC 10 --> &A7D4
A7CA	F1	070 049	46 31	LSR &31
A7CC	f2	102 050	66 32	ROR &32
A7CE	f3	102 051	66 33	ROR &33
A7D0	f4	102 052	66 34	ROR &34
A7D2	f5	102 053	66 35	ROR &35
A7D4	p	032 112 165	20 70 A5	JSR &A570 Clear FWB
A7D7	dC	100 067	64 43	STZ &43
A7D9	dD	100 068	64 44	STZ &44
A7DB	dE	100 069	64 45	STZ &45
A7DD	dF	100 070	64 46	STZ &46
A7DF	@	169 064	A9 40	LDA#&40
A7E1	=	133 061	85 3D	STA &3D
A7E3	B	133 066	85 42	STA &42
A7E5		162 251	A2 FB	LDX#&FB
A7E7		160 016	A0 10	LDY#&10
A7E9	8	056	38	SEC
A7EA	1	165 049	A5 31	LDA &31
A7EC	@	233 064	E9 40	SBC#&40
A7EE	1	133 049	85 31	STA &31
A7F0		152	98	TYA
A7F1	UB	085 066	55 42	EOR &42,X
A7F3	G	149 071	95 47	STA &47,X
A7F5	1	165 049	A5 31	LDA &31
A7F7	B	197 066	C5 42	CMP &42
A7F9		208 013	D0 0D	BNE 13 --> &A808
A7FB		218	DA	PHX
A7FC		162 252	A2 FC	LDX#&FC
A7FE	6	181 054	B5 36	LDA &36,X
A800	G	213 071	D5 47	CMP &47,X
A802		208 003	D0 03	BNE 3 --> &A807
A804		232	E8	INX
A805		208 247	D0 F7	BNE -9 --> &A7FE
A807		250	FA	PLX
A808	)	144 041	90 29	BCC 41 --> &A833
A80A	5	165 053	A5 35	LDA &35
A80C	F	229 070	E5 46	SBC &46
A80E	5	133 053	85 35	STA &35
A810	4	165 052	A5 34	LDA &34
A812	E	229 069	E5 45	SBC &45
A814	4	133 052	85 34	STA &34
A816	3	165 051	A5 33	LDA &33
A818	D	229 068	E5 44	SBC &44
A81A	3	133 051	85 33	STA &33

A81C	2	165 050	A5 32	LDA &32
A81E	C	229 067	E5 43	SBC &43
A820	2	133 050	85 32	STA &32
A822	1	165 049	A5 31	LDA &31
A824	B	229 066	E5 42	SBC &42
A826	1	133 049	85 31	STA &31
A828		152	98	TYA
A829		010	0A	ASL A
A82A		144 011	90 0B	BCC 11 --> &A837
A82C		026	1A	INC A
A82D	UA	085 065	55 41	EOR &41,X
A82F	A	149 065	95 41	STA &41,X
A831	F	149 070	95 46	STA &46,X
A833	B	181 066	B5 42	LDA &42,X
A835		128 004	80 04	BRA 4 --> &A83B
A837	UB	085 066	55 42	EOR &42,X
A839	B	149 066	95 42	STA &42,X
A83B	G	149 071	95 47	STA &47,X
A83D	5	006 053	06 35	ASL &35
A83F	&4	038 052	26 34	ROL &34
A841	&3	038 051	26 33	ROL &33
A843	&2	038 050	26 32	ROL &32
A845	&1	038 049	26 31	ROL &31
A847		152	98	TYA
A848	J	074	4A	LSR A
A849		168	A8	TAY
A84A		144 164	90 A4	BCC -92 --> &A7F0
A84C		160 128	A0 80	LDY#&80
A84E		232	E8	INX
A84F		208 159	D0 9F	BNE -97 --> &A7F0
A851	S	032 083 131	20 53 83	JSR &8353 Copy FWB Mantissa to FWA Mantissa
A854	1	165 049	A5 31	LDA &31
A856	0	048 003	30 03	BMI 3 --> &A85B
A858		032 251 129	20 FB 81	JSR &81FB Normalise FWA
A85B		032 149 166	20 95 A6	JSR &A695 Round FWA Mantissa to 4 bytes
A85E		169 255	A9 FF	LDA#&FF
A860	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A9CF LOG

Submitted by Steve Fewell

Description:

Call the &A746 routine to get the Float value and obtain the result calculation for LN.
Jump to &A9A3 with A = &42 and Y = &BF to multiply the FWA by the Floating-Point constant at location &BF42 in the Floating-Point constant table.
The constant value at &BF42 is 0.434294482. This value is 1/Log(10). Multiplying the LN result by this value gives the LOG result.

Disassembly for the LOG routine

A9CF	F	032 070 167	20 46 A7	JSR &A746 LN
A9D2	B	169 066	A9 42	LDA#&42
A9D4		160 191	A0 BF	LDY#&BF
A9D6		128 203	80 CB	BRA -53 --> &A9A3 Multiply FWA by &BF42 (0.434294482)

BASIC IV ROM Routines

Page Last Altered: undefined

81D5 Store 1-byte value in FWA

Submitted by Steve Fewell

Routine:BytetoFWA
Name: Store 1-byte value in FWA
Starting Address: &81D5
Entry criteria: A contains the 1-byte value to store in the FWA.
Exit: The FWA contains the 1-byte value.

Description:

Clear the FWA.
If the value to set the FWA to is positive then Set FWA to the value, Normalise and exit.
If the value to set the FWA to is negative, then Set the FWA Sign byte to Negative
and reverse the bits in the number and add 1 to make the value positive.
Now it is ok to set the FWA to the value, Normalise the FWA and exit.
We set Y to &88 when the Normalise routine is called, to tell the routine that
our FWA value is 8-bits in size, so set the Initial exponent to &80 + 8 (=&88),
so that the '.' is located after the first 8 bits.

Disassembly for the Store 1-byte value in FWA routine

81D5		032 180 166	20 B4 A6	JSR &A6B4 Clear FWA
81D8		168	A8	TAY
81D9		016 005	10 05	BPL 5 --> &81E0 Normalise FWA#2
81DB	.	133 046	85 2E	STA &2E
81DD	I	073 255	49 FF	EOR#&FF
81DF		026	1A	INC A

81E0	160 136	A0 88	LDY#&88
81E2			...Normalise FWA#2

BASIC IV ROM Routines

Page Last Altered: undefined

A9DF EXP

Submitted by Steve Fewell

Routine:EXP
Name: EXP (Exponential)
Starting Address: &A9DF
Entry criteria: The FWA contains the value to work with
Exit: The FWA contains the result.

Description:

Get the Numeric value (if integer then convert to Float, if String then Type Mismatch error).
If the FWA value has an exponent greater than #&87 (or equal to #&87 with a mantissa value >= &B3)
and the FWA sign is positive, then generate a EXP range error. Otherwise, if the FWA contains
a negative value then run the Clear FWA routine and exit with FWA = 0.0.
Thus 89.49999 is the largest value before the EXP range error is issued. No error is given
for negative values, as negative numbers with a large exponents return an EXP result of 0.

Call &82E0 to Split the FWA value into its Integer and Fractional parts. This is done as the
separate parts need to be calculated separately. This routine puts the 1-byte Integer value into &49
and the fractional value in the FWA.

Evaluate the EXP continued-fraction expansion series (A861) with X=#&CE, A=#&F6 and Y=#&03.
Thus the parameters to the evaluate series routine are as follows:
Default value (if FWA value too small to calculate) = &BFF6 = 1
First Floating-Point constant to use = &BFCE.
Number of cycles to evaluate = #&03.

The series is evaluated as follows:
If FWA too small then return 1 (&BFF6).
FWA = 1/FWA (store in Temporary Floating-Point variable address &046C)
FWA = 0.071206464 + FWA [Floating-Point constant &BFCE]
FWA = 0.00710252642 / FWA [Floating-Point constant &BFD3]
FWA = 0.000254009799 + FWA [Floating-Point constant &BFD8]
FWA = &046C [1/orig FWA] + FWA
FWA = 0.0166665235 / FWA [Floating-Point constant &BFDD]
FWA = 0.000000662400541 + FWA [Floating-Point constant &BFE2]
FWA = &046C [1/orig FWA] + FWA
FWA = 0.0833333324 / FWA [Floating-Point constant &BFE7]

FWA = -0.499999997 + FWA [Floating-Point constant &BFEC]
FWA = &046C [1/orig FWA] + FWA
FWA = 1 / FWA [Floating-Point constant &BFF1]
FWA = 1 + FWA [Floating-Point constant &BFF6]

This gives the EXP value for the fractional part of the value.
Store the series result to Temporary Floating-Point variable location &0476.
Unpack the Floating point constant at location &BF47 to the FWA.
This value is 2.71828183 (or EXP(1)).
Set A to &49 (the Integer part of the original FWA value).
Call &A5BE to raise the FWA to the power of the Integer value in A.
Multiply the FWA (the EXP result for the Integer part of the value) by
&0476 (the EXP result for the fractional value).
Now, we have the complete EXP result, so exit with the FWA set to this result.

Example 1:
Calculate EXP(1.245)
1.245 => Exponent = &81, Mantissa 1 = &1F, Mantissa 2 = &5C, Mantissa 3 = &28, Mantissa 4 = &F6
?&49 = 1.
FWA = 0.245
Calculate reciple -> FWA = 1/FWA = 4.08163265. Store this value in location &046C.
FWA = 0.071206464 + FWA = 4.15283911706
FWA = 0.00710252642 / FWA = 0.0017102821033
FWA = 0.000254009799 + FWA = 0.0019642919023
FWA = &046C + FWA = 4.083596941902
FWA = 0.0166665235 / FWA = 0.0040813341123
FWA = 0.000000662400541 + FWA = 0.0040819965128
FWA = &046C + FWA = 4.08571464651287
FWA = 0.0833333324 / FWA = 0.02039626836669
FWA = -0.499999997 + FWA = -0.4796037286333
FWA = &046C + FWA = 3.60202892136669
FWA = 1 / FWA = 0.27762131338484
FWA = 1 + FWA = 1.27762131338484. Store in &0476.

FWA = 2.71828183 Raised to the power of 1 = 2.71828183
FWA = 1.27762131338484 * FWA = 3.47293480179. This is the EXP value of 1.245.

Example 2:
Calculate EXP(8.17)
8.17 => Exponent = &84, Mantissa 1 = &02, Mantissa 2 = &B8, Mantissa 3 = &51, Mantissa 4 = &EC
?&49 = 8.
FWA = 0.17
Calculate reciple -> FWA = 1/FWA = 5.882352941176. Store this value in location &046C.
FWA = 0.071206464 + FWA = 5.953559405176
FWA = 0.00710252642 / FWA = 0.00119298825066
FWA = 0.000254009799 + FWA = 0.00144699804966
FWA = &046C + FWA = 5.88379993922566
FWA = 0.0166665235 / FWA = 0.00283261220166
FWA = 0.000000662400541 + FWA = 0.0028332746022
FWA = &046C + FWA = 5.8851862157782
FWA = 0.0833333324 / FWA = 0.014159846323398
FWA = -0.499999997 + FWA = -0.4858401506766
FWA = &046C + FWA = 5.396512790499398
FWA = 1 / FWA = 0.185304851266267
FWA = 1 + FWA = 1.185304851266267. Store in &0476.

FWA = 2.71828183 Raised to the power of 8 = 2980.9580005606
FWA = 1.185304851266267 * FWA = 3533.3439794855. This is the EXP value of 8.17.

Disassembly for the EXP routine

A9DF		032 218 150	20 DA 96	JSR &96DA Get and check Float value
A9E2	0	165 048	A5 30	LDA &30
A9E4		201 135	C9 87	CMP#&87
A9E6		144 015	90 0F	BCC 15 --> &A9F7
A9E8		208 006	D0 06	BNE 6 --> &A9F0
A9EA	1	164 049	A4 31	LDY &31
A9EC		192 179	C0 B3	CPY#&B3
A9EE		144 007	90 07	BCC 7 --> &A9F7
A9F0	.	165 046	A5 2E	LDA &2E
A9F2		016 200	10 C8	BPL -56 --> &A9BC EXP range error
A9F4	L	076 180 166	4C B4 A6	JMP &A6B4 Clear FWA
A9F7		032 224 130	20 E0 82	JSR &82E0 Split FWA (into Integer/Fractional parts)
A9FA		162 206	A2 CE	LDX#&CE
A9FC		169 246	A9 F6	LDA#&F6
A9FE		160 003	A0 03	LDY#&03
AA00	a	032 097 168	20 61 A8	JSR &A861 Evaluate continued-fraction expansion series
AA03		032 013 165	20 0D A5	JSR &A50D Store FWA to &0476
AA06	G	169 071	A9 47	LDA#&47
AA08		032 150 168	20 96 A8	JSR &A896 Load FWA with FP constant at &BF00 + A
AA0B	I	165 073	A5 49	LDA &49
AA0D		032 190 165	20 BE A5	JSR &A5BE Raise FWA to power of Integer value in A
AA10		128 141	80 8D	BRA -115 --> &A99F Multiply FWA by &0476

BASIC IV ROM Routines

Page Last Altered: undefined

99FE Get address of the specified Array element

Submitted by Steve Fewell

Routine:ArrayElemAddr
Name: Get address of the specified Array element
Starting Address: &99FE
Entry criteria: &37 and &38 point to the start of the variable name.
Y = length of the variable name (Y points to the '(' character of the variable name.
&2C contains the return type of the variable.
Exit: (&2A, &2B) = the address of the required data element (&2A = LSB).

Description:

Increment the X and Y pointers to point to after the '(' character of the variable name.
Call routine &8085 to get the address of the variable block (stored in (&2A, &2B).
If the variable was not found (Zero flag set), then the variable hasn't been defined
so, generate 'Array' error.
Update the Text Pointer B offset value (&1B) with X.

Push the variable return type, the variable pointer LSB and variable pointer MSB to the stack.
Load the first byte from the variable pointer address (&2A, &2B). This value is
the pointer to position after the last dimension.
If the pointer to position after the last dimension is 4 (indicating 1 dimension) then goto &9A85 to
handle a 1 dimensional-array.

[&9A16] Handle Multiple Array dimensions
Clear the IWA and set byte &2D of the IWA to 1. [Bytes &2A and &2B is the requested element
value. Byte &2D is the pointer to the dimension subscript (max value)].
[&9A1D] Push the IWA value to the stack. This is the current subscript value & pointer to max subscript.
Call routine &96AF to evaluate the expression after the '(' character, and convert the
expression result to an Integer (if it was a Float). A Type Mismatch error is generated
if the result is a String. Now the IWA contains the requested element.
If the next character is not a comma then generate an Array error, as the declared Array
is a multi-dimensional array, but the array usage does not specify a second dimension, so
the array the text pointer is refering to does not exist!
Increment the Text pointer B offset (to point to after the comma character).

Set (&37,&38) to Variable block pointer & retrieve current subscript from stack to &39-&3C
Pop the Integer value stored to the stack to locations &39 to &3C.

Set Y = the current dimension offset in the variable block (the 4th byte of the Integer variable).
Store the Variable block pointer MSB in &38 (keeping the value on the stack).
Store the Variable block pointer LSB in &37 (keeping the value on the stack).

Check subscript value in the IWA & get next dimension's max subscript value
[&9A3A] Call routine &9AD3 to check the dimension subscript.
Update &2D with the value returned by the check subscript routine. This value now points to the Dimensioned subscript value for the next dimension in the variable parameter block [each 2-byte dimensioned subscript value is stored, one after the other, after the pointer to the position after the last dimension (1-byte) value in the variable parameter block].
Load the second dimension's subscript (maximum element number) and store this 2-byte value in locations &3F-&40.

Add the current subscript value (in the IWA) to the previous subscript value (&39-&3A) and store the result in &2A-&2B, as the current subscript value now takes into account any previous dimensions that have been specified.
The previous subscript value is the previous IWA value (before the current element subscript) was obtained. This value is the total of all previous subscript values.
Call routine &9508 to multiply the current subscript value (&2A, &2B) (or "the Upper limit", as it is the total element value from all earlier elements) by the Lower Limit (&3F-&40). The lower limit is the maximum number of elements in the next dimension.
We need to multiply the current subscript by the lower limit (of the next element) in order to take into account the relative position of the element we require within the Array as a whole.
E.g. "DIM r(100,100):r(37, 50)=1" will return the element value for r as: 37 * 100 (= 3700) + 50 = 3750.
The 37 * 100 part is the value that is calculated by the section of the routine described above.

Load the pointer to the position after the last dimension of the array (the first byte of the variable parameter block). Subtract the current dimension maximim subscript pointer (&2D) from the pointer to the position after the last dimension.
If this value is >&eq;3 then there is more than 1 more dimension to process, so goto &9A1D to handle the other dimensions.

One more element left to process - obtain details for last element
Push the IWA to the Stack, the IWA contains the current element location)&2A-&2B) and a pointer to the next element details in the Variable Parameter block (&2D).
Call routine &96A7 to evaluate the next expression pointed to by Text pointer B.Convert the [&ADAC] If the next character after the expression is not an ")" then generate a Missing) error.
Convert the expression result to an Integer (if it was a Float). A Type Mismatch error is generated if the result is a String. Now the IWA contains the requested element value of the last element.

Pop the Array variable parameter block pointer address from the Stack (and place it in &37-&38).
Pop the Integer value from the stack to locations &39-&3C (this is the current element location).
Set Y = the Dimension offset in the variable paremeter block (?&3C).
Call routine &9AD3 to check the dimension subscript.
Add the current subscript value (in the IWA) to the previous subscript value (&39-&3A) and store the result in &2A-&2B, as the current subscript value now takes into account any previous dimensions that have been specified. Now IWA bytes &2A-&2B contain the required element position within the array.
Jump to &9A96 to calculate the address of the element's value.

[&9A85] Handle Single/Last dimension of array
Call routine &ADAC to evaluate the next expression pointed to by Text pointer B.Convert the
If the next character after the expression is not an ")" then generate a Missing) error.
Convert the expression result to an Integer (if it was a Float). A Type Mismatch error is generated

if the result is a String. Now the IWA contains the requested element value.

Pop the Array variable parameter block pointer address from the Stack (and place it in &37-&38).
Set Y = 1 (the Dimension offset in the variable paremeter block).
Call routine &9AD3 to check the dimension subscript.
Continue to &9A96 to calculate the address of the element's value.

&9A96 Calculate the address of the element's value.
Pop the variable return type from the Stack.
If the variable return type = 5 (Floating-Point value) then Store the original element
position (&2A-&2B) in A and X and Multiply the element position by 4 and then
add the original element position to the result. This will have multiplied the value by 5 in total.
Each Floating-Point value is 5-bytes in length, so now bytes &2A-&2B contain the
address of the required value. Jump to &9ABC to finish off and tweek this value to the correct address.

If the variable return type is either Integer or String (both values are 4-bytes long), then
Multiply the element position by 4, so that bytes &2A-&2B now point to the required value.

[&9ABC] However, as the values are offset from the variable paremeter block, we need to add the
length of the array parameter block to the address we generated. This is because, depending on the
number of dimensions, the parameter block could be different lengths for each array - not a fixed length.
Add Y (Byte &2D of the IWA value), which is the offset of the last dimension in the parameter block,
to &2A, and increment byte &2B if an overflow occurred.
Now, to obtain the direct address of the specified value we need to add the value we calculated
in &2A-&2B to the base address of our variable paremeter block (&37-&38). This will
give us the exact memory start address for the first byte of our value.
Exit with &2A-&2B = The value address and &2C = value type.

[&9AD3] Check the dimension subscript
Load byte 2 of the Integer subscript value in the IWA & 'AND' the value with #&C0,
then 'ORA' the result with &2C and &2D. This checks the subscript value, if the value
is > 8392 or < 0 (&2C or &2D contain a value or one of the top 2 bits of &2B are set).
then generate a Subscript error, as the subscript value is too large or negative.
Compare the first byte (&2A) of the subscript value with the dimension subscript LSB value
in the variable parameter block.]. Increment Y.
Compare the second byte (&2B) of the subscript value with
the dimension subscript MSB value (maximum size of subscript, MSB).
If either of the compares exceed the maximum subscript value for that dimension then
generate Subscript error. Increment Y to point to the next byte of the variable block.
The subscript is ok, so return to the calling routine.

[&9508] Multiply specified upper dimension by the lower dinension
(&2A, &2B) return the specified upper dimension adjusted to take into account the lower elements.
Multiplies &2A, &2B by &3F-&40.
Firstly: zero X and Y.
[&950C] Divide copy of 2nd dimension's max subscript (&3F-&40) by 2 (shift right).
Next, if the lost bit (from the right-hand end) was 0 then multiply &2A,&2B by 2 and
if &3F and &40 are not zero then loop again [&950C].
Otherwise: clear carry flag, add &2A to Y & add &2B to X. If the addition
    overflowed, then generate a Bad Dim error, as the element value is >&FFFF)
    (this should only occur during the DIM statement, and not during access to
    the array - as the DIM would have already generated the error).
    Multiply &2A-&2B by 2.
    If &3F-&40 is not zero then loop again [&950C].
Now, the second value (&3F-&40) is zero, and X & Y contain the result of
the first value (&2A-&2B) multiplied by the second value (&3F-&40).

Set &2A = Y and &2B = X and return with the updated element location.

Disassembly for the Get address of specified Array element routine

99FE		232	E8	INX
99FF		200	C8	INY
9A00		032 133 128	20 85 80	JSR &8085 Get pointer to variable block (array)
9A03		240 241	F0 F1	BEQ -15 --> &99F6 Array error
9A05		134 027	86 1B	STX &1B
9A07	,	165 044	A5 2C	LDA &2C
9A09	H	072	48	PHA
9A0A	*	165 042	A5 2A	LDA &2A
9A0C	H	072	48	PHA
9A0D	+	165 043	A5 2B	LDA &2B
9A0F	H	072	48	PHA
9A10	*	178 042	B2 2A	LDA (&2A)
9A12		201 004	C9 04	CMP#&04
9A14	o	144 111	90 6F	BCC 111 --> &9A85
9A16		032 232 171	20 E8 AB	JSR &ABE8 FALSE - Clear IWA
9A19		169 001	A9 01	LDA#&01
9A1B	-	133 045	85 2D	STA &2D
9A1D	&	032 038 188	20 26 BC	JSR &BC26 Push IWA to Stack
9A20		032 175 150	20 AF 96	JSR &96AF Get Integer result of expression
9A23		230 027	E6 1B	INC &1B
9A25	,	224 044	E0 2C	CPX#&2C
9A27		208 205	D0 CD	BNE -51 --> &99F6 Array error
9A29	9	162 057	A2 39	LDX#&39
9A2B		032 008 189	20 08 BD	JSR &BD08 Pop Integer from Stack to Zero page address
9A2E	<	164 060	A4 3C	LDY &3C
9A30	h	104	68	PLA
9A31	8	133 056	85 38	STA &38
9A33	h	104	68	PLA
9A34	7	133 055	85 37	STA &37
9A36	H	072	48	PHA
9A37	8	165 056	A5 38	LDA &38
9A39	H	072	48	PHA
9A3A		032 211 154	20 D3 9A	JSR &9AD3 Check Dimension Subscript
9A3D	-	132 045	84 2D	STY &2D
9A3F	7	177 055	B1 37	LDA (&37),Y
9A41	?	133 063	85 3F	STA &3F
9A43		200	C8	INY
9A44	7	177 055	B1 37	LDA (&37),Y
9A46	@	133 064	85 40	STA &40

9A48	*	165 042	A5 2A	LDA &2A
9A4A	e9	101 057	65 39	ADC &39
9A4C	*	133 042	85 2A	STA &2A
9A4E	+	165 043	A5 2B	LDA &2B
9A50	e:	101 058	65 3A	ADC &3A
9A52	+	133 043	85 2B	STA &2B
9A54		032 008 149	20 08 95	JSR &9508 Multiply upper dimension by the lower dimenson
9A57	8	056	38	SEC
9A58	7	178 055	B2 37	LDA (&37)
9A5A	-	229 045	E5 2D	SBC &2D
9A5C		201 003	C9 03	CMP#&03
9A5E		176 189	B0 BD	BCS -67 --> &9A1D
9A60	&	032 038 188	20 26 BC	JSR &BC26 Push IWA to Stack
9A63		032 167 150	20 A7 96	JSR &96A7 Extract Integer result of expression & check for ')'
9A66	h	104	68	PLA
9A67	8	133 056	85 38	STA &38
9A69	h	104	68	PLA
9A6A	7	133 055	85 37	STA &37
9A6C	9	162 057	A2 39	LDX#&39
9A6E		032 008 189	20 08 BD	JSR &BD08 Pop Integer from Stack to Zero page address
9A71	<	164 060	A4 3C	LDY &3C
9A73		032 211 154	20 D3 9A	JSR &9AD3 Check Dimension Subscript
9A76		024	18	CLC
9A77	9	165 057	A5 39	LDA &39
9A79	e*	101 042	65 2A	ADC &2A
9A7B	*	133 042	85 2A	STA &2A
9A7D	:	165 058	A5 3A	LDA &3A
9A7F	e+	101 043	65 2B	ADC &2B
9A81	+	133 043	85 2B	STA &2B
9A83		144 017	90 11	BCC 17 --> &9A96
9A85		032 172 173	20 AC AD	JSR &ADAC Evaluate expression & check for ')'
9A88		032 191 150	20 BF 96	JSR &96BF Check value & convert to Integer (if float)
9A8B	h	104	68	PLA
9A8C	8	133 056	85 38	STA &38
9A8E	h	104	68	PLA
9A8F	7	133 055	85 37	STA &37
9A91		160 001	A0 01	LDY#&01
9A93		032 211 154	20 D3 9A	JSR &9AD3 Check Dimension Subscript
9A96	h	104	68	PLA
9A97	,	133 044	85 2C	STA &2C
9A99		201 005	C9 05	CMP#&05
9A9B		208 023	D0 17	BNE 23 --> &9AB4
9A9D	+	166 043	A6 2B	LDX &2B

9A9F	*	165 042	A5 2A	LDA &2A
9AA1	*	006 042	06 2A	ASL &2A
9AA3	&+	038 043	26 2B	ROL &2B
9AA5	*	006 042	06 2A	ASL &2A
9AA7	&+	038 043	26 2B	ROL &2B
9AA9	e*	101 042	65 2A	ADC &2A
9AAB	*	133 042	85 2A	STA &2A
9AAD		138	8A	TXA
9AAE	e+	101 043	65 2B	ADC &2B
9AB0	+	133 043	85 2B	STA &2B
9AB2		128 008	80 08	BRA 8 --> &9ABC
9AB4	*	006 042	06 2A	ASL &2A
9AB6	&+	038 043	26 2B	ROL &2B
9AB8	*	006 042	06 2A	ASL &2A
9ABA	&+	038 043	26 2B	ROL &2B
9ABC		152	98	TYA
9ABD	e*	101 042	65 2A	ADC &2A
9ABF	*	133 042	85 2A	STA &2A
9AC1		144 003	90 03	BCC 3 --> &9AC6
9AC3	+	230 043	E6 2B	INC &2B
9AC5		024	18	CLC
9AC6	7	165 055	A5 37	LDA &37
9AC8	e*	101 042	65 2A	ADC &2A
9ACA	*	133 042	85 2A	STA &2A
9ACC	8	165 056	A5 38	LDA &38
9ACE	e+	101 043	65 2B	ADC &2B
9AD0	+	133 043	85 2B	STA &2B
9AD2	`	096	60	RTS

Disassembly for the Check Dimension Subscript routine

9AD3	+	165 043	A5 2B	LDA &2B
9AD5	)	041 192	29 C0	AND#&C0
9AD7	,	005 044	05 2C	ORA &2C
9AD9	-	005 045	05 2D	ORA &2D
9ADB		208 013	D0 0D	BNE 13 --> &9AEA Subscript error
9ADD	*	165 042	A5 2A	LDA &2A
9ADF	7	209 055	D1 37	CMP (&37),Y
9AE1		200	C8	INY
9AE2	+	165 043	A5 2B	LDA &2B
9AE4	7	241 055	F1 37	SBC (&37),Y
9AE6		176 002	B0 02	BCS 2 --> &9AEA Subscript error
9AE8		200	C8	INY
9AE9	`	096	60	RTS

Multiply specified upper dimension by the lower dimenson subscript (2-byte multiplication)

9503	?	162 063	A2 3F	LDX#&3F
9505		032 008 189	20 08 BD	JSR &BD08 Pop Integer from Stack to Zero page address
9508		162 000	A2 00	LDX#&00
950A		160 000	A0 00	LDY#&00
950C	F@	070 064	46 40	LSR &40
950E	f?	102 063	66 3F	ROR &3F
9510		144 011	90 0B	BCC 11 --> &951D
9512		024	18	CLC
9513		152	98	TYA
9514	e*	101 042	65 2A	ADC &2A
9516		168	A8	TAY
9517		138	8A	TXA
9518	e+	101 043	65 2B	ADC &2B
951A		170	AA	TAX
951B		176 015	B0 0F	BCS 15 --> &952C Bad Dim error
951D	*	006 042	06 2A	ASL &2A
951F	&+	038 043	26 2B	ROL &2B
9521	?	165 063	A5 3F	LDA &3F
9523	@	005 064	05 40	ORA &40
9525		208 229	D0 E5	BNE -27 --> &950C
9527	*	132 042	84 2A	STY &2A
9529	+	134 043	86 2B	STX &2B
952B	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B4F1 NEXT

Submitted by Steve Fewell

Description:

Call routine &98F5 to set BASIC text pointer B to the BASIC text pointer A value, evaluate the variable name at pointed to by BASIC text pointer B and obtain the address of the variable's value.

If A is not zero then the variable was valid and has been declared, so jump to &B4FF.

Otherwise, A contains zero (the zero flag is set) on return from &98F5 then the variable name is either not declared or invalid; so, check the FOR level (&26). Set X to the current FOR level (from location &26).
If the FOR level is zero then issue 'No FOR' error (as this has priority over 'Syntax error'). If the carry flag is not set then issue 'Syntax error' (as the variable name, after the NEXT keyword, was not declared - so it does not match any active FOR-variable [This is strange, as I would have expected 'No such variable' to be issued instead!]).

If the carry flag is set then no valid variable name was found after the NEXT keyword, so jump to &B539 to process the NEXT statement, as we have found a matching FOR statement (we already know that at least one FOR-loop is active; otherwise, 'No FOR' error would have been issued above). X contains a pointer to the current FOR level (from location &26), and, more importantly, X points to the details of the last FOR-statement on the FOR stack - therefore, &B539 will process the NEXT for the most recent FOR loop reached. Note: If a variable name is not present after the NEXT keyword, BASIC assumes that a NEXT-statement without a variable is present, and matches the latest FOR loop. E.g. in the case of the statement: *140 NEXT %* - the latest FOR loop will be processed, and 'Syntax error' will be issued after the FOR loop is complete - after the NEXT statement has been processed!

[&B4FF] A variable name was found after the NEXT-keyword - and the variable has been declared. So, check the carry flag. If the carry flag is set then the variable has a String value - so issue 'Syntax error' as a FOR-variable cannot be a String.

Set X to the FOR level (from location &26).

If X is zero, then there are no active FOR loops, so issue 'No FOR' error, as the NEXT statement does not have a matching FOR-statement.

[&B505] Compare the variable's address (stored in locations &2A-&2B) and type (location &2C) with the 3-byte FOR-variable details on the FOR stack (pointed to by X) for the latest FOR-loop (starting at location: &0519 + X, and ending at location: &051B + X).
If the variable matches the variable for the FOR loop on the stack (pointed to by X) then we have found a matching FOR loop for the variable, so jump to &B539 to process the NEXT statement.

[&B51A] Otherwise, Subtract #&0F (15) from X, so that X now points to the previous FOR-loop on the FOR stack.
Store X back to location &26 - to update the FOR stack. This will in effect cancel the most recent FOR loop - as the NEXT statements are not allowed to be in different orders, so the inner-most loop will be cencelled. I.e.:

```
10 FOR I = 1 TO 10
20 FOR K = 1 TO 10
30 PRINT I,K
40 NEXT I
50 NEXT K
```

At line 40, the 'K'-FOR loop will be cancelled, so 'No FOR' error will be issued at line 50. This behaviour makes sense, as otherwise, BASIC would be forced to jump back into the 'FOR I...'-loop when the 'NEXT K' statement is executed, which would cause problems.

If X is not zero then jump back to &B505, to compare the FOR-variable of the FOR loop (on the FOR stack) pointed to by X. Otherwise, the FOR-variable couldn't be matched with any of the FOR loops on the FOR-stack, so issue 'Can't match FOR' error.

[&B539] We have found a corresponding FOR-loop
X points to the details of the matching FOR-loop on the FOR-stack.
Store the start address of the variable's value (from the FOR-stack locations &0519 + X and &051A + X) in locations &2A (LSB) and &2B (MSB of address).

Check the variable's type (at location &051B + X on the FOR-stack).

If the variable is a Floating-point variable (the variable type is 5 (as a Float is a 5-byte value)), then:

* [&B5C0] Call routine &B1C7 to load the FWA with the FOR-variable's value (pointed to by &2A-&2B)
* Add #&1C to the X value to obtain a pointer to the Floating-Point STEP value for the FOR loop on the FOR-stack. Store this LSB pointer in location &4A and set the Most significant byte of the STEP value location to #&05 (stored in &4B).
* Add the Floating-Point STEP value (pointed to by &4A-&4B) to the FWA (FOR-Variable value).
* Set &37-&38 to &2A-&2B (i.e. the address of the FOR-Variable's value.
* Call routine &B369 to store the FWA value to the 5-byte location starting at the address pointed to by &37-&38.
* Load X with the value at location &26 (the current FOR level)
* Add #&21 to X and store the result in location &4A. This is the LSB of the FOR-loop's TO value

Note: &4B still contains #&05 (the MSB value). This will update the FOR-Variable value.
* Call routine &9C8F to unpack the Floating-Point value pointed to by &4A-&4B to the FWB and compare the FWB value with the FWA value.
* If the values are equal then we have not exceeded the TO-value yet (but should do when the STEP value is next added to the FOR-Variable's value), so jump to &B59E to repeat the FOR-loop again.
* Check the second byte of the STEP value (at location &051D + X), which contains the STEP value's sign.
* If the STEP value is negative, and the carry flag is set (FOR-Variable (FWA) > TO value (FWB)), then jump to &B5AE to exit the FOR-loop; otherwise, jump to &B59E to continue the FOR-loop.
* If the STEP value is positive, and the carry flag is clear (FOR-Variable (FWA) < TO value (FWB)), then jump to &B5AE to exit the FOR-loop; otherwise, jump to &B59E to continue the FOR-loop.

Otherwise, the variable is an Integer variable (the variable type is 4 (as an Integer is a 4-byte value)), so:
* [&B54A] Add the STEP value to the current value of the FOR-variable (pointed to by &2A-&2B).
The 4-byte STEP value starts at location &051C + X on the FOR-Stack [LSB value] and finishes at &051F + X.
The result of the addition is stored as the FOR-variable's value, and also in locations &37-&39 and Y.
* [B571] Subtract the 4-byte TO-value (stored on the FOR-stack starting at location &0521 + X and ending at location &0524 + X), from the FOR-variable's current value (stored in locations &37-&39 and Y).
To do the subtraction, firstly, the TO-value's LSB is subtracted and the result is stored in &37.
next, the TO-value's next significant byte is subtracted and the result Tests and Sets (TSB) bits in &37.
next, the TO-value's next significant byte is subtracted and the result Tests and Sets (TSB) bits in &37.
next, lastly, the TO-value's Most significant byte (MSB) is subtracted and the result is OR-ed (ORA) with &37.
Note: the Test and Set bits (TSB) operation is effectively the same as ORA - except that the result is stored to memory.
* If location &37 contains zero, then the TO-value is the same as the FOR-variable's current value, so jump to &B59E to repeat the FOR-loop again (as we have not exceeded the TO-value yet!).
* Check the sign (top bit of the MSB value) of the STEP value and the TO value and the FOR-Variable value (Y).
This is done as follows:
Result = (FOR-Variable's MSB) EOR (MSB of the STEP value) EOR (MSB of the TO value)

The result will be determined from the sign of the STEP, TO and (new) FOR-Variable values as follows:

FOR-Variable-sign	STEP-sign	TO-sign	Result
Pos.....	Pos.....	Pos....	Pos
Pos.....	Pos.....	Neg....	Neg
Pos.....	Neg.....	Pos....	Neg
Pos.....	Neg.....	Neg....	Pos
Neg.....	Pos.....	Pos....	Neg
Neg.....	Pos.....	Neg....	Pos
Neg.....	Neg.....	Pos....	Pos
Neg.....	Neg.....	Neg....	Neg

This result determines how we should handle the subtraction result. If the result is positive, then if the result of (FOR-Variable - TO-value) is less than 0 then we have exceeded the TO value;

otherwise, if the result of (FOR-Variable - TO-value) is more than 0 then we have exceeded the TO value.

* If the EORed-sign result is positive (Most significant bit not set), then check the carry flag. If the carry flag (after the subtraction) is not set then repeat the FOR-loop again, as we haven't exceeded the TO-value yet, so jump to &B59E to repeat the FOR loop again. Otherwise, if the Carry-flag is set then we have exceeded the TO-value, so jump to &B5AE to exit the FOR-loop.
* If the EORed-sign result is negative (Most significant bit is set), then check the carry flag. If the carry flag (after the subtraction) is set then we haven't exceeded the TO-value yet, so jump to &B59E to repeat the FOR loop again. Otherwise, if the Carry-flag is clear then we have exceeded the TO-value, so jump to &B5AE to exit the FOR-loop.

[&B59E] Repeat the FOR-loop again
We have not exceeded the TO-value yet, so jump back to the first statement within the FOR-loop and begin executing that statement. To do this, first set BASIC Text pointer A (&0B-&0C) to the Program address following the FOR-statement (from the FOR-stack locations &0526 + X and &0527 + X, call routine &9BC6 to set the BASIC Text Pointer A offset (&0A) to 1 and check for [Escape-Key]-pressed condition (if Escape is pressed, then issue 'Escape' error.), and jump to &900B to execute the first statement of the FOR-loop again.

[&B5AE] Exit the FOR-loop
Subtract #&0F (15) from the FOR level (in X) and store the new value back to &26, so that &26 now points to the FOR-loop before the one we have just exited from (if one exists).
Set BASIC Text pointer A offset to the BASIC Text pointer B offset (to update the BASIC text pointer A so that it points to the Program Code following the NEXT-statement.
If the next character (pointed to by BASIC text pointer A) is a comma ',', then jump back to the start of the NEXT routine - &&B4F1, (as the NEXT statement includes multiple FOR-variables) to process the NEXT for the next FOR-Loop.
Otherwise (comma not found), jump to &9000 to issue for 'Syntax error' if end of statement wasn't found and execute the next program statement.

Disassembly for the NEXT routine

B4F1		032 245 152	20 F5 98	JSR &98F5 PtrB=PtrA & evaluate variable name & obtain address of value
B4F4		208 009	D0 09	BNE 9 --> &B4FF
B4F6	&	166 038	A6 26	LDX &26
B4F8	8	240 056	F0 38	BEQ 56 --> &B532 'No FOR' error
B4FA	=	176 061	B0 3D	BCS 61 --> &B539 We have found a matching FOR-statement, so process the NEXT-statement
B4FC	Li	076 105 155	4C 69 9B	JMP &9B69 Syntax error
B4FF		176 251	B0 FB	BCS -5 --> &B4FC issue 'Syntax error'
B501	&	166 038	A6 26	LDX &26
B503	-	240 045	F0 2D	BEQ 45 --> &B532 'No FOR' error

```
B505   *  165 042      A5 2A     LDA &2A
B507      221 025 005  DD 19 05  CMP &0519,X
B50A      208 014      D0 0E     BNE 14 --> &B51A
B50C   +  165 043      A5 2B     LDA &2B
B50E      221 026 005  DD 1A 05  CMP &051A,X
B511      208 007      D0 07     BNE 7 --> &B51A
B513   ,  165 044      A5 2C     LDA &2C
B515      221 027 005  DD 1B 05  CMP &051B,X
B518      240 031      F0 1F     BEQ 31 --> &B539 We have found a matching FOR-statement, so process the NEXT-statement
B51A      138          8A        TXA
B51B   8  056          38        SEC
B51C      233 015      E9 0F     SBC#&0F
B51E      170          AA        TAX
B51F   &  134 038      86 26     STX &26
B521      208 226      D0 E2     BNE -30 --> &B505 Compare the next FOR loop's variable
B523                             'Can't match FOR' error
B532                             'No FOR' error
B539      189 025 005  BD 19 05  LDA &0519,X
B53C   *  133 042      85 2A     STA &2A
B53E      189 026 005  BD 1A 05  LDA &051A,X
B541   +  133 043      85 2B     STA &2B
B543      188 027 005  BC 1B 05  LDY &051B,X
B546      192 005      C0 05     CPY#&05
B548   v  240 118      F0 76     BEQ 118 --> &B5C0 Increment FOR-variable & Check for end of FOR-loop (FOR-variable is Float)
B54A   *  178 042      B2 2A     LDA (&2A)
B54C   }  125 028 005  7D 1C 05  ADC &051C,X
B54F   *  146 042      92 2A     STA (&2A)
B551   7  133 055      85 37     STA &37
B553      160 001      A0 01     LDY#&01
B555   *  177 042      B1 2A     LDA (&2A),Y
B557   }  125 029 005  7D 1D 05  ADC &051D,X
B55A   *  145 042      91 2A     STA (&2A),Y
B55C   8  133 056      85 38     STA &38
B55E      200          C8        INY
B55F   *  177 042      B1 2A     LDA (&2A),Y
```

B561	}	125 030 005	7D 1E 05	ADC &051E,X
B564	*	145 042	91 2A	STA (&2A),Y
B566	9	133 057	85 39	STA &39
B568		200	C8	INY
B569	*	177 042	B1 2A	LDA (&2A),Y
B56B	}	125 031 005	7D 1F 05	ADC &051F,X
B56E	*	145 042	91 2A	STA (&2A),Y
B570		168	A8	TAY
B571	7	165 055	A5 37	LDA &37
B573	8	056	38	SEC
B574	!	253 033 005	FD 21 05	SBC &0521,X
B577	7	133 055	85 37	STA &37
B579	8	165 056	A5 38	LDA &38
B57B	"	253 034 005	FD 22 05	SBC &0522,X
B57E	7	004 055	04 37	TSB &37
B580	9	165 057	A5 39	LDA &39
B582	#	253 035 005	FD 23 05	SBC &0523,X
B585	7	004 055	04 37	TSB &37
B587		152	98	TYA
B588	$	253 036 005	FD 24 05	SBC &0524,X
B58B	7	005 055	05 37	ORA &37
B58D		240 015	F0 0F	BEQ 15 --> &B59E Repeat the FOR-loop again
B58F		152	98	TYA
B590	]	093 031 005	5D 1F 05	EOR &051F,X
B593	]$	093 036 005	5D 24 05	EOR &0524,X
B596		016 004	10 04	BPL 4 --> &B59C If carry set then Exit FOR loop otherwise repeat FOR-loop
B598		176 004	B0 04	BCS 4 --> &B59E Repeat the FOR-loop again
B59A		128 018	80 12	BRA 18 --> &B5AE Exit the FOR-loop
B59C		176 016	B0 10	BCS 16 --> &B5AE Exit the FOR-loop
B59E	&	188 038 005	BC 26 05	LDY &0526,X
B5A1	'	189 039 005	BD 27 05	LDA &0527,X
B5A4		132 011	84 0B	STY &0B
B5A6		133 012	85 0C	STA &0C
B5A8		032 198 155	20 C6 9B	JSR &9BC6 Set PTR A Offset to 1 & Check for Escape error condition
B5AB	L	076 011 144	4C 0B 90	JMP &900B Process next BASIC program statement

B5AE		138	8A	TXA
B5AF	8	056	38	SEC
B5B0		233 015	E9 0F	SBC#&0F
B5B2	&	133 038	85 26	STA &26
B5B4		164 027	A4 1B	LDY &1B
B5B6		132 010	84 0A	STY &0A
B5B8		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
B5BB	8	208 056	D0 38	BNE 56 --> &B5F5 Start processing the next Program statement
B5BD	L	076 241 180	4C F1 B4	JMP &B4F1 Check whether there are any additional NEXT statements to process
B5C0		032 199 177	20 C7 B1	JSR &B1C7 Load FWA with Floating-Point Value (ain)
B5C3		138	8A	TXA
B5C4		024	18	CLC
B5C5	i	105 028	69 1C	ADC#&1C
B5C7	J	133 074	85 4A	STA &4A
B5C9		169 005	A9 05	LDA#&05
B5CB	K	133 075	85 4B	STA &4B
B5CD		032 141 166	20 8D A6	JSR &A68D Floating-Point Addition [FWA=argp+FWA]
B5D0	*	165 042	A5 2A	LDA &2A
B5D2	7	133 055	85 37	STA &37
B5D4	+	165 043	A5 2B	LDA &2B
B5D6	8	133 056	85 38	STA &38
B5D8	i	032 105 179	20 69 B3	JSR &B369 Store the FWA value to the variable's memory address
B5DB	&	166 038	A6 26	LDX &26
B5DD		138	8A	TXA
B5DE		024	18	CLC
B5DF	i!	105 033	69 21	ADC#&21
B5E1	J	133 074	85 4A	STA &4A
B5E3		032 143 156	20 8F 9C	JSR &9C8F Unpack argp to the FWB and compare it's value with the FWA value
B5E6		240 182	F0 B6	BEQ -74 --> &B59E Repeat the FOR-loop again
B5E8		189 029 005	BD 1D 05	LDA &051D,X
B5EB	0	048 004	30 04	BMI 4 --> &B5F1 If carry set then Exit FOR loop otherwise repeat FOR-loop
B5ED		176 175	B0 AF	BCS -81 --> &B59E Repeat the FOR-loop again
B5EF		128 189	80 BD	BRA -67 --> &B5AE Exit the FOR-loop
B5F1		144 171	90 AB	BCC -85 --> &B59E Repeat the FOR-loop again
B5F3		128 185	80 B9	BRA -71 --> &B5AE Exit the FOR-loop

B5F5 L 076 000 144 4C 00 90 JMP &9000 Check end of statement & execute the next statement on the Command/Program line

---

BASIC IV ROM Routines

Page Last Altered: undefined

B75B ON

Submitted by Steve Fewell

Description:

Get the next non-space character pointed to by BASIC Text Pointer A.

If the character is #&85 ('ERROR-token') then jump to &B741 to set the BASIC ERROR pointer, as follows:

* [&B741] Get the next non-space character after the 'ERROR' keyword
* If the character is #&87 ('OFF-Token') then jump to &B739 to check for the end of statement (&9BA6), and issue 'Syntax error' if the end of statement was not found (i.e. the next character wasn't ':', '<cr>' (carriage return) or 'ELSE-token'). Call routine &B2A6 to reset BASIC's ON ERROR pointer, so that it points to BASIC's default error handling routine (which starts at location &B2AF) and jump to &9005 to continue executing the Program.
* Otherwise, Set Y to the BASIC Text Pointer A offset (location &0A) and decrement Y (so that it points to the first non-space character after the ERROR-keyword (the character that we just checked).
* Call &9BBC to Add the BASIC Text Pointer A offset (&0A) to the Base BASIC Text Pointer A address (&0B-&0C), and to reset the Offset (&0A) to 1.
* Zero BASIC text pointer A Offset (&0A).
* Set BASIC's ON ERROR handler (stored at locations &16-&17) to point to the address in BASIC text pointer A (&0B-&0C) - so that it points to the program code following the ON ERROR keywords.
* Jump to &8FAE to skip the rest of the current program line (as this is the error code that is only executed when an error occurs) and continue executing the program starting at the next program line.

Otherwise, the next character should be a numeric variable name or numeric expression, so decrement the BASIC text pointer A offset value (&0A) to point to the start of this expression.
Call routine &926F to evaluate the expression at BASIC Text Pointer A and issue 'Type mismatch' error is a String expression is found, a Floating-Point result will be converted to Integer.

Now, the IWA contains the Integer result of the expression (or the contents of the Integer variable specified) and X contains the next character after the expression. This routine also sets BASIC Text Pointer B to the BASIC Text Pointer A value.

If the next character (X) is #&E5 (GOTO-token) or #&E4 (GOSUB-token) then increment Y (to skip over the keyword).
If the next character (X) is not #&F2 (PROC-token), #&E5 (GOTO-token) or #&E4 (GOSUB-token) then issue an 'ON syntax' error as a valid ON statement (ON ERROR..., ON...GOTO, ON...GOSUB, ON...PROC) was not found.

[&B774] Push X to the 6502 Stack (this is the Keyword which indicates the type of ON-statement that we are processing).
Check whether IWA bytes &2B-&2D are zero, if they are not the the IWA contains a value greater than 255 - which is not within the permitted ON range (1-127), so jump to &B7D5 to check each character of the current program line (starting from the current character), looking for an 'ELSE-token' (#&8B). If an 'ELSE-token' is found then [&B817] update the BASIC text pointer A offset (&0A) to point to the location of the ELSE-token, and call routine &9C29 to execute the ELSE-statement. If a carriage return character ('<cr>') is found without finding an 'ELSE-token' then issue 'ON range' error. Please see the note at the bottom for a flaw with this ELSE-statement search.

[&B77D] Decrement the IWA LSB (&2A) value.
If the IWA LSB (&2A) value is now zero then the original IWA value was 1, so jump to &B7B6 to execute the PROC/GOTO/GOSUB statement that is pointed to by the BASIC Text Pointer A (i.e. the first statement after the GOTO/GOSUB/PROC keyword).

[B781] If the IWA (&2A) value was less than 0 or greater than 127 (i.e. negative flag set) after the decrement, then the value is not within the permitted ON range (1-127), so jump to &B7D5 to check each character of the current program line (starting from the current character), looking for an 'ELSE-token' (#&8B). If an 'ELSE-token' is found then [&B817] update the BASIC text pointer A offset (&0A) to point to the location of the ELSE-token, and call routine &9C29 to execute the ELSE-statement. If a carriage return character ('<cr>') is found without finding an 'ELSE-token' then issue 'ON range' error. Please see the note at the bottom for a flaw with this ELSE-statement search.

[&B783] If the current character (pointed to by PTR A (&0B-&0C) plus offset Y) is either a carriage return ('<cr>'), a colon (':') or an 'ELSE-token' clause, then the end of the ON statement has been found, (or the ELSE-part of the ON-statement has been found), this means that the GOSUB,GOTO or PROC statement corresponding to the original value specified in the IWA was not located within the statement, so jump to &B7D5 to check each character of the current program line (starting from the current character), looking for an 'ELSE-token' (#&8B). If an 'ELSE-token' is found then [&B817] update the BASIC text pointer A offset (&0A) to point to the location of the ELSE-token, and call routine &9C29 to execute the ELSE-statement. If a carriage return character ('<cr>') is found without finding an 'ELSE-token' then issue 'ON range' error. Please see the note at the bottom for a flaw with this ELSE-statement search.

[&B791] Skip the current GOTO, GOSUB or PROC statement, and move forward to the next GOTO line, GOSUB line or PROC call in the ON-statement (that is the next statement after a comma ',' has been located). This is done as follows:
Increment Y offset (to point to the next character). The accumulator (A), however, still contains the current character.
If the current character (in A) was a double quote (''), then Exclusive-OR the character with the contents of location &2B (this flag indicates whether the current character is inside a String literal or not).
If the value of the String Literal flag (location &2B) is not zero, then we are currently inside a String Literal, so ignore the current character and jump back to &B783 to process the next character.
Note: the initial value of &2B is zero (as the IWA now only contains an 8-bit value).

If the current character is a close bracket ')', then decrement location &2C. Location &2C is a flag that determines whether we have reached the end of a function call such as MID$(a$,3,4).
If the current character is an open bracket '(' then increment location &2C (the bracket counter).

If the current character is not a comma then jump back to &B783 to process the next character, as we haven't found the next GOTO, GOSUB or PROC call yet.

Now, we have found a comma. However, if location &2C (the bracket counter) contains a positive value, then the comma forms part of a function call, and is, therefore, not the ON list separator, so jump back to &B783 to process the next character.

Now, we have found the next list element in the On-Statement, so decrement the IWA value (&2A) and check the value of the IWA (location &2A). If the IWA has reached zero, then we have found the ON GOSUB, GOTO or PROC statement for the original value of the IWA, so jump to &B783 to execute the statement.
Otherwise, we haven't found the GOTO, GOSUB or PROC call that corresponds to the original IWA value, so jump back to &B783 to process the next character.

[&B7B6] Execute the PROC/GOTO/GOSUB statement:
Retrieve the ON statement type (either PROC/GOTO/GOSUB) from the 6502 stack.
If the statement type is 'PROC' then jump to &B803 to execute the PROCedure call.
Store Y back to the BASIC text pointer A offset (to update the offset value). This didn't need to be done for PROC, as PROC sets the BASIC Text pointer B offset instead.

If the statement type is 'GOSUB' then:

* [&B7CA] Call routine &B82A to Read the Line Number at BASIC Text pointer A and find the Program address for the start of that Program line.
* Set Y to the BASIC Text pointer A offset (&0A), i.e. the character after the Line Number.
* Call &B81D to skip the rest of the ON-Statement until a ':' or carriage return ('<cr>') character is found. This is so that on return from the GOSUB routine, the program will correctly continue to execute at the next program statement.
* Jump to &B6DC to execute the GOSUB statement by checking for end of statement & GOSUBing the Line Number specified

Otherwise, the statement type must be 'GOTO', so:

* [&B7C1] Call routine &B82A to Read the Line Number at BASIC Text pointer A and find the Program address for the start of that Program line.
* Call routine &9BC6 to set the BASIC Text Pointer A offset (&0A) to 1 and to check for the Escape-Key pressed condition - if the Escape key has been pressed, then issue 'Escape' error.
* Jump to &B723 to Display the Line Number on the screen (if TRACE is currently active), Set the BASIC Text Pointer A Offset to 4 (to skip the 3-byte Line Number and 1-byte offset to the next line which are located at the start of the Program Line), Set the BASIC Text Pointer A to the address specified in locations &3D-&3E and jump to &900B to start executing the statement at the start of the specified Program Line.

* Note: The rest of the ON-Statement does not need to be skipped (using &B81D), as it is in the GOSUB call, as we will not be returning from the GOTO call.

[&B803] Execute the PROC statement:
Store Y in location &1B (to update BASIC text pointer B to the current character on the line (pointed to by Y) - this should be the PROC-token).
Get the next non-space character pointed to by the BASIC text pointer B.
If the character is not PROC (note: that each comma-separated statement in an ON...PROC statement must begin with the PROC-keyword (i.e. ON...PROCxxx,PROCyyy,PROCzzz)) then issue 'ON syntax' error.
Call the routine &B019 to execute the PROC statement.
Set Y to the BASIC text pointer B offset (the value after the PROCedure name)
Call routine &B81D to skip the rest of the program statement (until a ':' or carriage return ('<cr>') character is found).
Jump to &9002 to execute the next program statement.

Flaws with the ON routine
The ON routine has one flaw, found in routine &B7D5. This routine keeps checking the rest of the Program line until either an ELSE-token or a carriage-return character is found.
If an ELSE-token is found then the ELSE-statement will be executed, if a carriage return ('<cr>') character is found 'ON range' error will be issued, as an out of range value was found and the ON statement doesn't have an ELSE-Statement part.
However, the problem is that the 'ON' keyword is a statement, which ends when a carriage return ('<cr>') or ':' character is found - however BASIC will keep searching for the 'ELSE-statement' past the end of the ON-Statement, if the ON-statement terminates with a colon.

E.g.:

```
5 ON ERROR A = A / B : GOTO 40
10 INPUT A
20 INPUT B
30 ON A GOSUB 100,110,120,130 : IF B > 0 THEN LET A = A / B ELSE LET A = 0
40 PRINT A
50 END
100 LET A = A * 20:RETURN
110 LET A = A * 30:RETURN
120 LET A = A * 40:RETURN
130 LET A = A * 50:RETURN
```

In the above program, if 5 is entered for A (which is out of range for the ON statement, as only 4 options are present), then the ELSE-Statement part of the 'IF-Statement' will be executed - even though the ON-Statement terminated (with a ':' character) before the IF statement started. This is basically due to BASIC not checking for a colon ':' character as well as a carriage return. Therefore, in the above example, if A = 5 then A will be set to 0 - instead of being divided by B (as per the code following the ON ERROR statement, as the ON statement should have issued 'ON Range' error).

Disassembly for the ON routine

B739		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
B73C		032 166 178	20 A6 B2	JSR &B2A6 Reset 'ON ERROR' pointer to BASIC's default error handling routine
B73F		128 217	80 D9	BRA -39 --> &B71A Jump to &9005 to continue running the program
B741		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space character pointed to by Ptr A
B744		201 135	C9 87	CMP#&87
B746		240 241	F0 F1	BEQ -15 --> &B739 Execute ON ERROR OFF statement
B748		164 010	A4 0A	LDY &0A
B74A		136	88	DEY
B74B		032 188 155	20 BC 9B	JSR &9BBC Update BASIC Text pointer A (Add offset value & then reset offset to 1)
B74E	d	100 010	64 0A	STZ &0A
B750		165 011	A5 0B	LDA &0B
B752		133 022	85 16	STA &16
B754		165 012	A5 0C	LDA &0C
B756		133 023	85 17	STA &17
B758	L	076 174 143	4C AE 8F	JMP &8FAE Skip the rest of the line and process the next program line
B75B		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space character pointed to by Ptr A
B75E		201 133	C9 85	CMP#&85
B760		240 223	F0 DF	BEQ -33 --> &B741 Execute 'ON ERROR' statement
B762		198 010	C6 0A	DEC &0A
B764	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A & convert the result to integer
B767		224 242	E0 F2	CPX#&F2
B769		240 009	F0 09	BEQ 9 --> &B774
B76B		200	C8	INY
B76C		224 229	E0 E5	CPX#&E5
B76E		240 004	F0 04	BEQ 4 --> &B774
B770		224 228	E0 E4	CPX#&E4
B772	v	208 118	D0 76	BNE 118 --> &B7EA 'ON syntax' error
B774		218	DA	PHX
B775	+	165 043	A5 2B	LDA &2B
B777	,	005 044	05 2C	ORA &2C
B779	-	005 045	05 2D	ORA &2D
B77B	X	208 088	D0 58	BNE 88 --> &B7D5 Check for ELSE-token (if found then execute ELSE statement) otherwise issue 'ON range' error

```
B77D  *   198 042   C6 2A     DEC &2A
B77F  5   240 053   F0 35     BEQ 53 --> &B7B6 Execute the PROC/GOTO/GOSUB statement
B781  0R  048 082   30 52     BMI 82 --> &B7D5 Check for ELSE-token (if found then execute ELSE statement) otherwise issue 'ON range' error
B783      177 011   B1 0B     LDA (&0B),Y
B785      201 013   C9 0D     CMP#&0D
B787  L   240 076   F0 4C     BEQ 76 --> &B7D5 Check for ELSE-token (if found then execute ELSE statement) otherwise issue 'ON range' error
B789  :   201 058   C9 3A     CMP#&3A
B78B  H   240 072   F0 48     BEQ 72 --> &B7D5 Check for ELSE-token (if found then execute ELSE statement) otherwise issue 'ON range' error
B78D      201 139   C9 8B     CMP#&8B
B78F  D   240 068   F0 44     BEQ 68 --> &B7D5 Check for ELSE-token (if found then execute ELSE statement) otherwise issue 'ON range' error
B791      200       C8        INY
B792  "   201 034   C9 22     CMP#&22
B794      208 004   D0 04     BNE 4 --> &B79A
B796  E+  069 043   45 2B     EOR &2B
B798  +   133 043   85 2B     STA &2B
B79A  +   166 043   A6 2B     LDX &2B
B79C      208 229   D0 E5     BNE -27 --> &B783
B79E  )   201 041   C9 29     CMP#&29
B7A0      208 002   D0 02     BNE 2 --> &B7A4
B7A2  ,   198 044   C6 2C     DEC &2C
B7A4  (   201 040   C9 28     CMP#&28
B7A6      208 002   D0 02     BNE 2 --> &B7AA
B7A8  ,   230 044   E6 2C     INC &2C
B7AA  ,   201 044   C9 2C     CMP#&2C
B7AC      208 213   D0 D5     BNE -43 --> &B783
B7AE  ,   166 044   A6 2C     LDX &2C
B7B0      208 209   D0 D1     BNE -47 --> &B783
B7B2  *   198 042   C6 2A     DEC &2A
B7B4      208 205   D0 CD     BNE -51 --> &B783
B7B6  h   104       68        PLA
B7B7      201 242   C9 F2     CMP#&F2
B7B9  H   240 072   F0 48     BEQ 72 --> &B803 Execute the PROC statement
B7BB      132 010   84 0A     STY &0A
B7BD      201 228   C9 E4     CMP#&E4
B7BF      240 009   F0 09     BEQ 9 --> &B7CA
```

```
B7C1  *   032 042 184  20 2A B8  JSR &B82A Get Line Number & find Program Address of the Line Number
B7C4      032 198 155  20 C6 9B  JSR &9BC6 Set PTR A Offset to 1 & Check for Escape error condition
B7C7  L#  076 035 183  4C 23 B7  JMP &B723 Jump to Program Line spec ified
B7CA  *   032 042 184  20 2A B8  JSR &B82A Get Line Number & find Program Address of the Line Number
B7CD      164 010      A4 0A     LDY &0A
B7CF      032 029 184  20 1D B8  JSR &B81D Skip the rest of the program statement (until a ':' or carriage return ('<cr>') character is found)
B7D2  L   076 220 182  4C DC B6  JMP &B6DC Check for end of statement & GOSUB the Line Number specified in the IWA
B7D5  h   104          68        PLA
B7D6      177 011      B1 0B     LDA (&0B),Y
B7D8      200          C8        INY
B7D9      201 139      C9 8B     CMP#&8B
B7DB  :   240 058      F0 3A     BEQ 58 --> &B817 Update PTR A Offset & execute the ELSE-statement
B7DD      201 013      C9 0D     CMP#&0D
B7DF      208 245      D0 F5     BNE -11 --> &B7D6
B7E1                             'ON range' error
B7EA                             'ON syntax' error
B7F4                             'No such line' error
B803      132 027      84 1B     STY &1B
B805      032 213 142  20 D5 8E  JSR &8ED5 Get next non-space character (PTR B)
B808      201 242      C9 F2     CMP#&F2
B80A      208 222      D0 DE     BNE -34 --> &B7EA 'ON syntax' error
B80C      032 025 176  20 19 B0  JSR &B019 PROC
B80F      164 027      A4 1B     LDY &1B
B811      032 029 184  20 1D B8  JSR &B81D Skip the rest of the program statement (until a ':' or carriage return ('<cr>') character is found)
B814  L   076 002 144  4C 02 90  JMP &9002 'Syntax error' if not end of statement; otherwise, execute next statement/program line
B817      132 010      84 0A     STY &0A
B819  L)  076 041 156  4C 29 9C  JMP &9C29 Find and execute the ELSE statement (if one is found)
B81C      200          C8        INY
B81D      177 011      B1 0B     LDA (&0B),Y
B81F      201 013      C9 0D     CMP#&0D
B821      240 004      F0 04     BEQ 4 --> &B827
B823  :   201 058      C9 3A     CMP#&3A
B825      208 245      D0 F5     BNE -11 --> &B81C
B827      132 010      84 0A     STY &0A
```

B829 ` 096 60 RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B707 RETURN

Submitted by Steve Fewell

Description:

Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.

If the GOSUB Level value (at location &25) is zero, then no outstanding GOSUB calls have been executed, so issue 'No GOSUB' error.

Decrement the &25 GOSUB Level value (so that its value is the return address offset) to reduce the number of active GOSUBs.
Retrieve the return address value from the GOSUB return address Stack (The LSB of the return address is retrieved from location &05CC + ?&25. The MSB of the return address is retrieved from location &05E6 + ?&25).
Store the return address value in the BASIC Text Pointer A (&0B-&0C).

Jump to &9005 to continue executing from the program position after the original GOSUB statement.

Disassembly for the RETURN routine

B707		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
B70A	%	166 037	A6 25	LDX &25
B70C		240 242	F0 F2	BEQ -14 --> &B700 No GOSUB error
B70E	%	198 037	C6 25	DEC &25
B710		188 203 005	BC CB 05	LDY &05CB,X

B713		189 229 005	BD E5 05	LDA &05E5,X
B716		132 011	84 0B	STY &0B
B718		133 012	85 0C	STA &0C
B71A	L	076 005 144	4C 05 90	JMP &9005 Continue running program (execute next statement)

BASIC IV ROM Routines

Page Last Altered: undefined

B82A Get Line Number & return the Line Number's Program Address

Submitted by Steve Fewell

Routine:GetLineNumberAddress
Name: Get Program Line Number and Program Address
Starting Address: &B82A
Entry criteria: The BASIC Text Pointer A points to a tokenised Line Number
Exit: Locations &3D-&3E contain a pointer to the specified Line Number's start address location within the Program Code

Description:

Call routine &9B1E to Skip any leading spaces and detokenise the Line Number pointed to by BASIC Text Pointer A.

If a Line Number was not found (carry flag is not set) then, the Line Number is probably a variable or calculation, so:

* Call routine &926F to evaluate the expression at BASIC Text Pointer A & convert the result to Integer
* The IWA should now contain the Line Number value. Only 16-Bits of the IWA value will be used (that is locations &2A-&2B). As the Line Number cannot be greater than 32767, reset bit 7 (#&80) of the Line Number high-byte (location &2B) to reduce the number to a valid Line Number value (if it was > 32767 (#&7FFF)).

[&B836] Now the IWA contains the specified Line Number.
Call routine &80CD to search for the first program line that is greater than or equal to the Line Number specified in the IWA.
If &80CD returned with the Carry flag clear, then the specified line was not found in the program, so issue 'No such line' error, as we are unable to perform an operation on a Line that doesn't exist in the Program.

Otherwise exit with &3D-&3E containing a pointer to the specified Line Number's start address location within

the Program Code.

Disassembly for the Get Line Number & return the Line Number's Program Address routine

B82A		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
B82D		176 007	B0 07	BCS 7 --> &B836 Set &3D-&3E to the IWA Line's Start address & issue error if No Such Line
B82F	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
B832		169 128	A9 80	LDA#&80
B834	+	020 043	14 2B	TRB &2B
B836		032 205 128	20 CD 80	JSR &80CD Search for Program Line >= the Line Number in the IWA
B839		144 185	90 B9	BCC -71 --> &B7F4 No such line error
B83B	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B71D GOTO

Submitted by Steve Fewell

Description:

Call routine &B82A to get the Line Number after the GOTO keyword, Issue 'No such line' error if the Line doesn't exist and set &3D-&3E to the start address of the specified Program Line.
Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.

If the TRACE flag (location &20) is not zero, then call routine &9C4B to check that TRACE is on, and that the Line Number (locationed in the IWA) is in the specified TRACE range, if it is then the Line number will be displayed on the screen.

Set BASIC Text Pointer A to the start of the specified program line, by:

* [&B72A] Set #&0A (the BASIC Text Pointer A Offset) to 4 (a pointer to the first character of the Program Line (skipping the preceding Line number (3 bytes) and Next Line offset (1 byte) values).
* Set &0B (the BASIC Text Pointer A LSB) to the Line Number Start Address LSB value at location &3D
* Set &0C (the BASIC Text Pointer A MSB) to the Line Number Start Address MSB value at location &3E

Jump to &900B to start executing the code at the GOTO Line (the new BASIC Text Pointer A location).

Disassembly for the GOTO routine

B71D	*	032 042 184	20 2A B8	JSR &B82A Get Line Number & find Program Address of the Line Number
B720		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement

B723		165 032	A5 20	LDA &20
B725		240 003	F0 03	BEQ 3 --> &B72A
B727	K	032 075 156	20 4B 9C	JSR &9C4B Display current line number (IWA) on screen [if TRACE is on]
B72A		160 004	A0 04	LDY#&04
B72C		132 010	84 0A	STY &0A
B72E	=	164 061	A4 3D	LDY &3D
B730	>	165 062	A5 3E	LDA &3E
B732		132 011	84 0B	STY &0B
B734		133 012	85 0C	STA &0C
B736	L	076 011 144	4C 0B 90	JMP &900B Process next BASIC program statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

B6D9 GOSUB

Submitted by Steve Fewell

Description:

Call routine &B82A to get the Line Number after the GOTO keyword, Issue 'No such line' error if the Line doesn't exist and set &3D-&3E to the start address of the specified Program Line.
Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.

If the GOSUB Level (at location &25) is more than or equal to &1A (26) then issue 'Too many GOSUBs' error, as the maximum nested GOSUB level is 26.

Store the current Text Pointer A location on the GOSUB return address stack, so that the correct position in the program can be returned to when the RETURN statement corresponding to this GOSUB is reached.
The LSB Byte of the current program position is placed on the GOSUB return address stack at position &05CC + Y (where Y is the current GOSUB level, i.e. the value of &25 before it is incremented (in the range 0-24)).
The MSB Byte of the current program position is placed on the GOSUB return address stack at position &05E6 + Y (where Y is the current GOSUB level, i.e. the value of &25 before it is incremented (in the range 0-24)).

Increment the GOSUB level [&25] (to increase the current number of active GOSUBs and the GOSUB Stack level value).

Jump to &B723 to position BASIC Text Pointer A at the start of the specified Line
and begin execution of the GOSUB routine, as follows:

* If the TRACE flag (location &20) is not zero, then call routine &9C4B to display the Line Number (located in the IWA) on the screen [if TRACE is currently active].
* Set BASIC Text Pointer A to the start of the specified program line (&3D-&3E, offset by 4 bytes).
* Jump to &900B to start executing the code at the GOTO Line (the new BASIC Text Pointer A location).

Disassembly for the GOSUB routine

B6D9	*	032 042 184	20 2A B8	JSR &B82A Get Line Number & find Program Address of the Line Number
B6DC		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
B6DF	%	164 037	A4 25	LDY &25
B6E1		192 026	C0 1A	CPY#&1A
B6E3		176 014	B0 0E	BCS 14 --> &B6F3 Too many GOSUBs error
B6E5		165 011	A5 0B	LDA &0B
B6E7		153 204 005	99 CC 05	STA &05CC,Y
B6EA		165 012	A5 0C	LDA &0C
B6EC		153 230 005	99 E6 05	STA &05E6,Y
B6EF	%	230 037	E6 25	INC &25
B6F1	0	128 048	80 30	BRA 48 --> &B723 Jump to program line specified

---

BASIC IV ROM Routines

Page Last Altered: undefined

9C08 IF

Submitted by Steve Fewell

Description:

Call routine &9D2F to set BASIC text pointer B to the BASIC text pointer A value and get the result of the expression after the IF-Keyword.

If the result of the expression after ther 'IF-Keyword' is a String value then, a boolean or numerical IF-condition has not been specified, so issue 'Type mismatch' error (i.e. IF "Hello" THEN was specified instead of IF A$ = "Hello" THEN).

If the result of the expression evaluation was a Floating-Point value (i.e. The negative flag is set), then call routine &96C3 to convert the Floating-Point value (in the FWA) to an Integer value and place the result in the IWA.

[&9C12] Update the BASIC Text Pointer A Offset (at location &0A) to the BASIC Text Pointer B offset (this will place the BASIC Text Pointer A so that is points to the location after the 'IF-Condition' expression.

If the IWA value is zero, then the IF-condition is False so, jump to &9C37 to process the ELSE part of the IF-statement.
Otherwise continue to &9C20 to process the True part of the IF-Statement.

[&9C20] IF-Condition is True so, Process the Main part of the IF-Statement
If the next character after the IF-Condition expression (in X) is not #&8C 'THEN-Keyword', then jump to &900B to execute the next program statement found after the IF-Condition.
Note: This is why you must include a THEN if you want to specify a GOTO without using the GOTO Keyword (as in IF 1=1 THEN 1000); as without the THEN, BASIC would attempt to execute the Line Number and fail, as a Keyword/variable name was not found!
Note: The Statement will automatically terminate if the ELSE keyword is found, so there is no need to skip the 'ELSE-statement' here.

Otherwise, (The 'THEN-Keyword' was found), Increment the BASIC Text Pointer A Offset (location &0A) to skip over the 'THEN-Token'.
Call routine &9B1E to check for a tokenised Line Number at the BASIC Text Pointer A location, and if a tokenised Line Number is found then set the IWA to the untokenised Line Number value.

If a Line Number was found (carry flag is set) then jump to that Program Line, as follows:

* Call routine &B836 to set &3D-&3E to the Start Address within the Program Code where the Line Number (specified in the IWA) is located and issue 'No such line' error if the specified line doesn't exist in the Program.
* Call routine &9BC6 to Set the BASIC Text Pointer A offset to 1 (the start of the Program Line) and check for an Escape condition (If the Escape-Key has been pressed, then issue an 'Escape' error)
* Jump to &B723 to Display the Line Number on the screen (if TRACE is currently active), Set the BASIC Text Pointer A Offset to 4 (to skip the 3-byte Line Number and 1-byte offset to the next line which are located at the start of the Program Line), Set the BASIC Text Pointer A to the address specified in locations &3D-&3E and jump to &900B to start executing the statement at the start of the specified Program Line.

Otherwise, (no tokenised Line Number found), jump to &900B to execute the next statement after the 'THEN-Keyword'.

[&9C37] If-Condition is False so, Process the ELSE part of the IF-Statement
Set Y to the BASIC Text Pointer A offset value (from location &0A).
[&9C39] Keep checking the character at the current position of the BASIC Text Pointer A (&0B-&0C plus offset value (Y)). Set A to the character found at the current location.

If the current character (A) is '<cr>' (carriage return) then there is no ELSE-statement part to this IF-Statement, so jump to &8FB8 to:

* Add the BASIC Text Pointer A Offset value (location &0A) to the BASIC Text Pointer A value (&0B-&0C)
* Reset the BASIC Text Pointer A Offset (location &0A) to 1
* Skip the rest of the current Program Line, Find and execute the next Program Line

Increment Y, to move to the next character on the current Program Line.
If the character in A (the character read by &9C39) is not the 'ELSE-token' then, we haven't found the ELSE-Statement part (if one exists) of the IF-Statement yet, so jump back to &9C39 to check the next character of the current Program Line.
Otherwise, we have located the ELSE-Statement part of the IF-Statement, so Store Y back in the BASIC Text Pointer A Offset (location &0A), so that BASIC Text Pointer A now points to the character after the 'ELSE-Token' and execute the ELSE-part by jumping to &9C29, as follows:
[&9C29] Call routine &9B1E to check for a tokenised Line Number at the BASIC Text Pointer A location, and if a tokenised Line Number is found then set the IWA to the untokenised Line Number value.

If a Line Number was found (carry flag is set) then jump to that Program Line, as follows:

* Call routine &B836 to set &3D-&3E to the Start Address within the Program Code where the Line Number

(specified in the IWA) is located and issue 'No such line' error if the specified line doesn't exist in the Program.

* Call routine &9BC6 to Set the BASIC Text Pointer A offset to 1 (the start of the Program Line) and check for an Escape condition (If the Escape-Key has been pressed, then issue an 'Escape' error)
* Jump to &B723 to Display the Line Number on the screen (if TRACE is currently active), Set the BASIC Text Pointer A Offset to 4 (to skip the 3-byte Line Number and 1-byte offset to the next line which are located at the start of the Program Line), Set the BASIC Text Pointer A to the address specified in locations &3D-&3E and jump to &900B to start executing the statement at the start of the specified Program Line.

Otherwise, (no tokenised Line Number found), jump to &900B to execute the next statement after the 'ELSE-Keyword'.

Note: As intended, BASIC does not stop checking for an 'ELSE-Token' if another 'IF-Statement' is found on the current Line, as the ELSE will match both the outer IF-Statement and the inner IF-Statement.
I.e. IF A$ = "W" THEN w = w + 1:IF w > 25 THEN PRINT "A" ELSE PRINT "B"
In the above statement, "B" will be printed if A$ is not "W", or if A$ is "W" and w > 25! This is correct behaviour!

This also means that the ELSE part of an ON-GOTO-Statement will be handled in the same way.
I.e. IF A$ = "W" THEN ON A GOTO 10, 20, 30 ELSE 40
In the above statement, the Program will execute Line 40 if A$ is not "W", or if A$ is "W" and A is not 1, 2 or 3
This is correct as when BASIC is looking for an Else-Condition, it will handle the first 'ELSE-Token' it finds on the Program Line.

Disassembly for the IF routine

9C05	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
9C08	/	032 047 157	20 2F 9D	JSR &9D2F Ptr B = Ptr A & Get result of expression [PTR B]
9C0B		240 248	F0 F8	BEQ -8 --> &9C05 Type mismatch error
9C0D		016 003	10 03	BPL 3 --> &9C12
9C0F		032 195 150	20 C3 96	JSR &96C3 Convert Float (FWA) to Integer (IWA)
9C12		164 027	A4 1B	LDY &1B
9C14		132 010	84 0A	STY &0A
9C16	*	165 042	A5 2A	LDA &2A
9C18	+	005 043	05 2B	ORA &2B
9C1A	,	005 044	05 2C	ORA &2C
9C1C	-	005 045	05 2D	ORA &2D
9C1E		240 023	F0 17	BEQ 23 --> &9C37 Process the ELSE-part of the IF-Statement
9C20		224 140	E0 8C	CPX#&8C
9C22		240 003	F0 03	BEQ 3 --> &9C27

9C24	L	076 011 144	4C 0B 90	JMP &900B Process next BASIC program statement
9C27		230 010	E6 0A	INC &0A
9C29		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
9C2C		144 246	90 F6	BCC -10 --> &9C24
9C2E	6	032 054 184	20 36 B8	JSR &B836 Set &3D-&3E to the IWA Line's Start address & issue error if No Such Line
9C31		032 198 155	20 C6 9B	JSR &9BC6 Set PTR A Offset to 1 & Check for Escape error condition
9C34	L#	076 035 183	4C 23 B7	JMP &B723 Jump to Program Line specified
9C37		164 010	A4 0A	LDY &0A
9C39		177 011	B1 0B	LDA (&0B),Y
9C3B		201 013	C9 0D	CMP#&0D
9C3D		240 009	F0 09	BEQ 9 --> &9C48
9C3F		200	C8	INY
9C40		201 139	C9 8B	CMP#&8B
9C42		208 245	D0 F5	BNE -11 --> &9C39
9C44		132 010	84 0A	STY &0A
9C46		240 225	F0 E1	BEQ -31 --> &9C29
9C48	L	076 184 143	4C B8 8F	JMP &8FB8 Update PTR A (Add offset value & reset offset to 1) & Process the next Program Line

---

BASIC IV ROM Routines

Page Last Altered: undefined

BA58 REPEAT

Submitted by Steve Fewell

Description:

Set X to the current value of the REPEAT level (location &24).
If the current REPEAT level is greater than or equal to #&14 (20) then issue 'Too many REPEATs' error, as a maximum of 20 REPEATs can be nested.

Call routine &9BBC to add the BASIC text pointer A offset (&0A) to the BASIC text pointer A base (&0B-&0C), and reset the BASIC text pointer A offset to 1.

Store the current Program location (&0B-&0C) on the REPEAT start address stack (&0500-&0527), where the MSB of the current program location (&0C) is stored at &0514 + X (where X is the current REPEAT level) and LSB of the current program location (&0B) is stored at &0500 + X (where X is the current REPEAT level).

Increment &24 (the REPEAT level) to increase the number of nested REPEAT statements.
Jump to &900B to continue executing the program from the next statement after the REPEAT keyword.

Disassembly for the REPEAT routine

BA58	$	166 036	A6 24	LDX &24
BA5A		224 020	E0 14	CPX#&14
BA5C		176 167	B0 A7	BCS -89 --> &BA05 Too many REPEATs error
BA5E		032 188 155	20 BC 9B	JSR &9BBC Update BASIC Text pointer A (Add offset value & then reset offset to 1)
BA61		165 011	A5 0B	LDA &0B

```
BA63      157 000 005     9D 00 05    STA &0500,X
BA66      165 012         A5 0C       LDA &0C
BA68      157 020 005     9D 14 05    STA &0514,X
BA6B   $  230 036         E6 24       INC &24
BA6D   L  076 011 144     4C 0B 90    JMP &900B Process next BASIC program statement
```

BASIC IV ROM Routines

Page Last Altered: undefined

BA17 UNTIL

Submitted by Steve Fewell

Description:

Call routine &9D2F to set BASIC text Pointer B to the BASIC text pointer A value and get the result of the expression pointed to by BASIC text pointer B.

Call routine &9B91 to check for the end of the Statement - if the end of statement (either a ':', '<cr>' or 'ELSE' character) is not found after the expression then issue 'Syntax error', as the UNTIL statement does not finish correctly.

Call routine &96BC to check the result type of the expression (location &27), and convert the expression result value to an Integer if it is a Floating-Point value, or issue 'Type mismatch' error if it is a String value.
The IWA now contains the value of the result of the expression.

Load X with the current REPEAT level (from location &24).
If X is zero, then no outstanding REPEAT statements are active, so issue 'No REPEAT' error.

Check the IWA result value - if the IWA contains zero, then the result of the UNTIL expression is false, so [&BA33] load Y with the LSB of the REPEAT start address from the REPEAT start address stack (&04FF + X (current REPEAT level)) and load A with the MSB of the REPEAT start address from the REPEAT start address stack (&0513 + X (current REPEAT level)), and jump to &B732 to set the BASIC text pointer A to the REPEAT start address value (in Y and A) and jump to &900B to continue executing the program from the REPEAT start address.

Otherwise, the UNTIL expression is true, and we have finished the REPEAT...UNTIL loop, so decrement the current REPEAT level - so that it is now set to the level before the REPEAT loop began, and jump to &9005 to continue executing the program, starting with the next program statement after the UNTIL statement.

Disassembly for the UNTIL routine

BA17	/	032 047 157	20 2F 9D	JSR &9D2F Ptr B = Ptr A & Get result of expression [PTR B]
BA1A		032 145 155	20 91 9B	JSR &9B91 X = A; Check for end of Statement (PTR B)
BA1D		032 188 150	20 BC 96	JSR &96BC Check result type (location &27) - if Float then convert to Integer
BA20	$	166 036	A6 24	LDX &24
BA22		240 215	F0 D7	BEQ -41 --> &B9FB No REPEAT error
BA24	*	165 042	A5 2A	LDA &2A
BA26	+	005 043	05 2B	ORA &2B
BA28	,	005 044	05 2C	ORA &2C
BA2A	-	005 045	05 2D	ORA &2D
BA2C		240 005	F0 05	BEQ 5 --> &BA33
BA2E	$	198 036	C6 24	DEC &24
BA30	L	076 005 144	4C 05 90	JMP &9005 Continue running program (execute next statement)
BA33		188 255 004	BC FF 04	LDY &04FF,X
BA36		189 019 005	BD 13 05	LDA &0513,X
BA39	L2	076 050 183	4C 32 B7	JMP &B732 Jump to the address in Y (MSB) and A (LSB) and continue executing the program from this address

---

BASIC IV ROM Routines

Page Last Altered: undefined

9646 TRACE

Submitted by Steve Fewell

Description:

Call routine &9B1E to check if there is a tokenised Line Number following the TRACE Keyword. If a Line Number is found then detokenise the Line Number and set the IWA to the Line Number value.

If the carry flag is not set, then a Line Number was not found and A contains the character after the TRACE Keyword, so:

* If A is the 'ON-Token' [#&EE] then jump to the TRACE ON routine at &9667
* If A is the 'OFF-Token' [#&87] then jump to the TRACE OFF routine at &9670
* Otherwise, the TRACE Line Number must be a variable/expression, so call routine &926F to evaluate the variable/expression, convert the result to Integer and store the result in the IWA

Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.
Set the TRACE Line Number (locations &21-&22) to the 16-bit Line Number from the IWA (locations &2A-&2B).
Set the TRACE flag (location &20) to #&FF to indicate that TRACEing is currently activated.

Jump to &9005 to continue executing the next statement of the Program.

[&9667] TRACE ON routine
Increment the BASIC Text Pointer A offset (&0A) to skip over the ON-Token.
Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.
Set A to #&FF.

Jump to &965F to:

* Set the TRACE Line Number MSB (location &22) to A (#&FF) to indicate that there is no maximum TRACE Line Number.
* Set the TRACE flag (location &20) to #&FF to indicate that TRACEing is currently activated.
* Jump to &9005 to continue executing the next statement of the Program.

[&9670] TRACE OFF routine
Increment the BASIC Text Pointer A offset (&0A) to skip over the OFF-Token.
Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.
Set A to #&00.

Jump to &9663 to:
* Set the TRACE flag (location &20) to A (#&00) to indicate that TRACEing is NOT currently activated.
* Jump to &9005 to continue executing the next statement of the Program.

Disassembly for the TRACE routine

9646		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
9649		176 011	B0 0B	BCS 11 --> &9656
964B		201 238	C9 EE	CMP#&EE
964D		240 024	F0 18	BEQ 24 --> &9667 TRACE ON
964F		201 135	C9 87	CMP#&87
9651		240 029	F0 1D	BEQ 29 --> &9670 TRACE OFF
9653	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
9656		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
9659	*	165 042	A5 2A	LDA &2A
965B	!	133 033	85 21	STA &21
965D	+	165 043	A5 2B	LDA &2B
965F	"	133 034	85 22	STA &22
9661		169 255	A9 FF	LDA#&FF
9663		133 032	85 20	STA &20
9665		128 212	80 D4	BRA -44 --> &963B [JMP &9005] Process the next statement
9667		230 010	E6 0A	INC &0A
9669		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
966C		169 255	A9 FF	LDA#&FF
966E		208 239	D0 EF	BNE -17 --> &965F

9670	230 010	E6 0A	INC &0A
9672	032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
9675	169 000	A9 00	LDA#&00
9677	128 234	80 EA	BRA -22 --> &9663

BASIC IV ROM Routines

Page Last Altered: undefined

B94D RESTORE

Submitted by Steve Fewell

Description:

Set locations &3D-&3E to PAGE (The Program Start address) [Where the LSB (&3D) is #&00].
This defaults the RESTORE Line Number to be PAGE - This &3D-&3E value will be overwritten by the Line Number specified after the RESTORE Keyword (if a Line Number is specified).

Call &8EE0 to get the next non-space character pointed to by BASIC Text Pointer A.
Decrement the BASIC Text Pointer A offset [&0A] so that Text Pointer A points to the non-space character.

If the next non-space character is not the end of Statement (I.e. ':', '<cr>', 'ELSE-token') then we need to read the Line Number to RESTORE from, by:

* [&B964] Calling routine &B82A to get the Line Number after the RESTORE keyword, Issue 'No such line' error if the Line doesn't exist and set &3D-&3E to the start address of the specified Program Line.

[&B967] Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.

Set the DATA Pointer (&1C-&1D) to the DATA start address specified (the address stored in &3D-&3E, i.e. either PAGE, or the start address of the Program Line specified).

Jump to &9005 to continue running the program by executing the next program statement.

Disassembly for the RESTORE routine

B94D	d=	100 061	64 3D	STZ &3D
B94F		164 024	A4 18	LDY &18
B951	>	132 062	84 3E	STY &3E
B953		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space character pointed to by Ptr A
B956		198 010	C6 0A	DEC &0A
B958	:	201 058	C9 3A	CMP#&3A
B95A		240 011	F0 0B	BEQ 11 --> &B967
B95C		201 013	C9 0D	CMP#&0D
B95E		240 007	F0 07	BEQ 7 --> &B967
B960		201 139	C9 8B	CMP#&8B
B962		240 003	F0 03	BEQ 3 --> &B967
B964	*	032 042 184	20 2A B8	JSR &B82A Get Line Number & find Program Address of the Line Number
B967		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
B96A	=	165 061	A5 3D	LDA &3D
B96C		133 028	85 1C	STA &1C
B96E	>	165 062	A5 3E	LDA &3E
B970		133 029	85 1D	STA &1D
B972	L	076 005 144	4C 05 90	JMP &9005 Continue running program (execute next statement)

BASIC IV ROM Routines

Page Last Altered: undefined

B97D READ

Submitted by Steve Fewell

Description:

[&B975] *Check for a comma, if found - jump to READ (&B97D) otherwise end of READ statement (jump to &9000)*
Call routine &8CE5 to check that the next non-space character is a comma ',', if it is then jump to &B97D
to READ the new data value into the variable specified after the comma.
If the next non-space character was not a comma, then we have reached the end of the READ statement so jump to &9000 to
execute the next program statement.

[&B97D] READ statement

Call &98AE to evaluate the variable name after the READ keyword [or after the comma, if we have already
processed the first variable of the READ statement].
If the zero flag is set on return from &98AE then a valid variable name was not found; so, jump to &B975 to check
whether the next non-space character at PTR A is a comma, if it is then try reading a variable name after the comma;
otherwise, jump to &9000, as the end of the READ statement has been found. Note: &9000 will handle any Syntax
Error messages if the command at PTR A is not reconised.
If the carry flag is 1 then the variable is a String variable; so, jump to &B98F to process the String variable
Otherwise, the variable is numeric, so continue with &B984, to READ the numeric value.

[&B984] READ a value into a Numeric variable
Call &B9AC to set the BASIC Text Pointer B to point to the location before the start of the next DATA element.
Call &BC43 to push the variable address & type information (&2A, &2B and &2C) to the 6502 Stack.
Call &B328 to evaluate the numeric expression at BASIC Text Pointer B, retrieve the variable information from
the 6502 stack and set the variable's value to the result of the numeric expression, or issue 'Type mismatch' error
if a non-numeric (i.e. String) value was found.
Jump to &B99D to update the (&1C, &1D) DATA pointer and to continue processing the READ statement.

[&B98F] READ a value into a String variable
Call &B9AC to set the BASIC Text Pointer B to point to the location before the start of the next DATA element.
Call &BC26 to push the IWA value (which contains the Variable address and type information) to the BASIC stack.
Call &ACF8 to Extract the Next Field starting from the location pointed to by BASIC text pointer B, storing the

result in the SWA.
Store A (the result type) in location &27.
Call &90AB to retrieve the String variable details from the BASIC stack and set the String variable to the SWA value.
Continue to &B99D to update the (&1C, &1D) DATA pointer and to continue processing the READ statement.
Note: Because the Extract Next field routine (&ACF8) is used, a value of any type (Integer, Float or String) can be READ into a String variable (i.e. This is fine: 10 READ A$ 20 DATA 2.34).

[&B99D] Update the DATA Pointer and READ the next value (if any more are specified)
Add the BASIC Text Pointer B value (&19, &1A) to the DATA pointer (&1C, &1D).
Jump back to &B975 to check for a comma after the variable name in the READ Statement (BASIC text pointer A), if a comma is found then process the READ statement for the next variable, otherwise end of READ statement found, so jump to &9000 to start executing the next Statement in the Program.

[&B9AC] Move BASIC Text Pointer B to next DATA element location
Call &9275 to update the BASIC Text Pointer A offset to equal the BASIC Text Pointer B offset.
Set BASIC text pointer B to point to the value at locations &1C-&1D (this is either the last DATA element READ or, if nothing has been ready yet, it will be the start of the BASIC program. Zero the BASIC text pointer B offset
Note: If (&1C-&1D) pointed to the start of the program, then the character pointed to will be the carriage return ('<cr>') character which indicates the start of the program.

Get the next non-space character at the location pointed to by BASIC Text Pointer B.
If the character is a comma then exit (RTS).
If the character is the 'DATA-token' (&DC) then exit (RTS).
If the character is a carriage return ('<cr>') then jump to &B9CF to move to the next line.

[&B9C6] Get the next non-space character at the location pointed to by BASIC Text Pointer B.
If the character is a comma then exit (RTS).
If the character is not '<cr>' then jump back to &B9C6 to check the next character.

[&B9CF] Check the next character pointed to by BASIC Text pointer B (plus offset - &1B), if it is negative (i.e. #&FF) then the end of the program has been reached without finding the required data, so issue 'Out of DATA' error; Otherwise (not negative), start processing the next program line by:

* Incrementing Y (offset) by 2 (to skip the Line Number)
* Setting X to the character at Text pointer B (plus offset) - this is the pointer to the next program line.
* Incrementing Y (offset) by 1 (to skip the pointer to the next Program Line)
* Skipping any spaces at the beginning of the Program Line
* If the first non-space character on the program line is 'DATA-token' then jump to &BA13 to increment Y (offset), store Y back in &1B (PTR B offset) and exit.
* Otherwise, Add X to BASIC Text Pointer B (&19, &1A) and junmp back to &B9CF to move to the next line

---

Thus, in other words, &B9AC simply looks for a Comma or DATA-Token at the current DATA pointer location (&1C, &1D) and, if neither of these are found, then BASIC will skip the rest of the line until a comma is found, or if a carriage return character is found (or a comma was not found on the current program line) then this routine will look for a Program Line that begins with a DATA-token (as the first non-space character). 'Out of DATA' error is issued if no DATA is found.

This format allows comments to be included inside the DATA statement! This is illustrated by the following program:

```
10 READ A, B, C
20 PRINT A, B, C
```

30 DATA 2 (This is A) , 3 'This is B, 4 : This is C
This program correctly outputs 2 3 4
The best characters to use to separate the DATA element from the remark is '(' (open bracket), """ (quote) or ':' (colon).
However, DATA statements will not be reconised if they do not begin at the start of the line (i.e. 30 : DATA 2, 3, 4)

Disassembly for the READ routine

B975		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
B978		240 003	F0 03	BEQ 3 --> &B97D READ
B97A	L	076 000 144	4C 00 90	JMP &9000 Check end of statement & execute the next statement on the Command/Program line
B97D		032 174 152	20 AE 98	JSR &98AE Evaluate variable name & create it if it's a new variable
B980		240 243	F0 F3	BEQ -13 --> &B975 Check for a comma (if not found then end of READ statement)
B982		176 011	B0 0B	BCS 11 --> &B98F Process String Variable
B984		032 172 185	20 AC B9	JSR &B9AC Set the BASIC Text Pointer B to point to the location before the start of the next DATA element
B987	C	032 067 188	20 43 BC	JSR &BC43 Push &2A, &2B & &2C (the variable info) to the Stack
B98A	(	032 040 179	20 28 B3	JSR &B328 Evaluate expression and set Numeric variable
B98D		128 014	80 0E	BRA 14 --> &B99D
B98F		032 172 185	20 AC B9	JSR &B9AC Set the BASIC Text Pointer B to point to the location before the start of the next DATA element
B992	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value (variable info) to the BASIC Stack [pushi]
B995		032 248 172	20 F8 AC	JSR &ACF8 Extract next field (PTR B)
B998	'	133 039	85 27	STA &27
B99A		032 171 144	20 AB 90	JSR &90AB Set String variable
B99D		024	18	CLC
B99E		165 027	A5 1B	LDA &1B
B9A0	e	101 025	65 19	ADC &19
B9A2		133 028	85 1C	STA &1C
B9A4		165 026	A5 1A	LDA &1A
B9A6	i	105 000	69 00	ADC#&00
B9A8		133 029	85 1D	STA &1D
B9AA		128 201	80 C9	BRA -55 --> &B975 Check for a comma (if not found then end of READ statement)
B9AC	u	032 117 146	20 75 92	JSR &9275 Set PTR A Offset to PTR B Offset
B9AF		165 028	A5 1C	LDA &1C
B9B1		133 025	85 19	STA &19
B9B3		165 029	A5 1D	LDA &1D
B9B5		133 026	85 1A	STA &1A
B9B7	d	100 027	64 1B	STZ &1B
B9B9		032 235 142	20 EB 8E	JSR &8EEB Get next non-space char (PTR B) & compare with ','
B9BC	X	240 088	F0 58	BEQ 88 --> &BA16 RTS

B9BE		201 220	C9 DC	CMP#&DC
B9C0	T	240 084	F0 54	BEQ 84 --> &BA16 RTS
B9C2		201 013	C9 0D	CMP#&0D
B9C4		240 009	F0 09	BEQ 9 --> &B9CF
B9C6		032 235 142	20 EB 8E	JSR &8EEB Get next non-space char (PTR B) & compare with ','
B9C9	K	240 075	F0 4B	BEQ 75 --> &BA16 RTS
B9CB		201 013	C9 0D	CMP#&0D
B9CD		208 247	D0 F7	BNE -9 --> &B9C6
B9CF		164 027	A4 1B	LDY &1B
B9D1		177 025	B1 19	LDA (&19),Y
B9D3	0	048 028	30 1C	BMI 28 --> &B9F1 Out of DATA error
B9D5		200	C8	INY
B9D6		200	C8	INY
B9D7		177 025	B1 19	LDA (&19),Y
B9D9		170	AA	TAX
B9DA		200	C8	INY
B9DB		177 025	B1 19	LDA (&19),Y
B9DD		201 032	C9 20	CMP#&20
B9DF		240 249	F0 F9	BEQ -7 --> &B9DA
B9E1		201 220	C9 DC	CMP#&DC
B9E3	.	240 046	F0 2E	BEQ 46 --> &BA13
B9E5		138	8A	TXA
B9E6		024	18	CLC
B9E7	e	101 025	65 19	ADC &19
B9E9		133 025	85 19	STA &19
B9EB		144 226	90 E2	BCC -30 --> &B9CF
B9ED		230 026	E6 1A	INC &1A
B9EF		128 222	80 DE	BRA -34 --> &B9CF

Disassembly for BA13 - Increment Y & Update Text Pointer Offset (&1B=Y)

BA13		200	C8	INY
BA14		132 027	84 1B	STY &1B
BA16	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A59B TAN

Submitted by Steve Fewell

Description:

This routine calls &A93A to obtain the Floating-point argument and use continued-fraction expressions to calculate either the Sine or Cosine value (depending on the value of the argument) of the argument.
This FWA result is stored in temporary Floating-point location &047B-&047F.

Call &A917 (SIN) to calculate the Sine value from the FWA result.
Store the Sine result in the temporary Floating-point location &0476-&047A.

Set the FWA back to the &A93A result (from the Temporary Floating-point variable at &047B-&047F).
Call &A915 (COS) to calculate the Cosine value from the &A93A result.
Note: Storing the &A93A value avoids the need to calculate it again!

Set argp to point to &0476 (the Sine result).
Call &A5EE to divide argp by the FWA [FWA = argp / FWA], where argp points to the Sine result and the FWA contains the Cosine result.
Dividing the Sine result by the Cosine result gives the Tangent value.

Exit with A = #&FF (as the result is Floating-point and is located in the FWA).

Example 1: TAN(90)
Divide the SIN(90) result by the COS(90) result
FWA = 0.8939945 / -0.4480777 --> -1.9951774 (= the TAN result)

Example 2: TAN(2.41)
Divide the SIN(2.41) result by the COS(2.41) result
FWA = 0.6680554 / -0.7441115 --> -0.8977893 (= the TAN result)

Example 3: TAN(1.5)
Divide the SIN(1.5) result by the COS(1.5) result
FWA = 0.9974949 / 0.0707372 --> 14.1014191 (= the TAN result)

Example 4: TAN(5.63)
Divide the SIN(5.63) result by the COS(5.63) result

FWA = -0.6078147 / 0.7940788 --> -0.7654337 (= the TAN result)

Example 5: TAN(0)
Divide the SIN(0) result by the COS(0) result
FWA = 0 / 1 --> 0 (= the TAN result)

Example 6: TAN(-0.75)
Divide the SIN(-0.75) result by the COS(-0.75) result
FWA = -0.6816387 / 0.7316889 --> -0.9315963 (= the TAN result)

Example 7: TAN(0.25)
Divide the SIN(0.25) result by the COS(0.25) result
FWA = 0.2474039 / 0.9689124 --> 0.2553418 (= the TAN result)

This diagram shows the relationship between the trig functions.

Disassembly for the TAN routine

A59B	:	032 058 169	20 3A A9	JSR &A93A Get value and calculate Sine/Cosine value
A59E	{	169 123	A9 7B	LDA#&7B
A5A0		032 019 165	20 13 A5	JSR &A513 Store FWA to &047B
A5A3		032 023 169	20 17 A9	JSR &A917 SIN (convert the result of &A93A to the Sine result)
A5A6	v	169 118	A9 76	LDA#&76
A5A8		032 019 165	20 13 A5	JSR &A513 Store FWA to &047B
A5AB	{	169 123	A9 7B	LDA#&7B
A5AD	;	032 059 165	20 3B A5	JSR &A53B Load FWA from variable at &0400 + A (i.e. &047B)
A5B0		032 021 169	20 15 A9	JSR &A915 COS (convert the result of &A93A to the Cosine result)
A5B3	v	169 118	A9 76	LDA#&76
A5B5		032 148 165	20 94 A5	JSR &A594 Set argp to &0400 + A (i.e. &0476)
A5B8		032 238 165	20 EE A5	JSR &A5EE Floating-Point Division [FWA=argp/FWA]
A5BB		169 255	A9 FF	LDA#&FF
A5BD	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A89C ACS

Submitted by Steve Fewell

Description:

Call routine &A8A1 to get the Float value (at the BASIC text pointer location, converting Integer to Float - if the value was Integer) and calculate the Arc-Sine (ASN) result of the Float value.
Call routine &A8E1 to: set the FWA = (1.57079633 (constant at &BF2E) - FWA) and exit with A=#&FF (as the result is a Floating-point value in the FWA).

The ACS calculation is simply:

ACS(x) = 1.57079633 - ASN(x)

Example 1: ACS(0)

FWA = ASN(FWA) = 0.0
FWA = 1.57079633 - FWA = 1.57079633

Example 2: ACS(0.25)

FWA = ASN(FWA) = 0.2526802
FWA = 1.57079633 - FWA = 1.3181161

Example 3: ASN(-0.75)

FWA = ASN(FWA) = -0.848062
FWA = 1.57079633 - FWA = 2.4188583

This diagram shows the relationship between the trig functions.

Disassembly for the ACS routine

A89C		032 161 168	20 A1 A8	JSR &A8A1 Get Float value and calculate ASN result
A89F	@	128 064	80 40	BRA 64 --> &A8E1 FWA = 1.57079633 (constant at &BF2E) - FWA & exit with A=#&FF

BASIC IV ROM Routines

Page Last Altered: undefined

A8A1 ASN

Submitted by Steve Fewell

Description:

Call routine &96DA to get the numeric value (at the current position of the BASIC text pointer) and convert it to a Floating-Point value (if it was an Integer).

If the FWA sign is positive then jump to &A8AF to calculate the ASN result and exit.

If the FWA sign is negative then:
* Clear the FWA Sign byte (to make the FWA value positive)
* Call routine &A8AF to calculate the ASN result for the FWA value
* Jump to &A8D2 to store A (#&FF) in the FWA sign byte (to make the FWA value negative) and exit

[A8AF] Calculate ASN result:
Call routine &A50D to store the FWA value in temporary Floating-point location &0476-&047A.

Call routine &A929 to:
* FWA = FWA * FWA (calculate the square of FWA)
* FWA = 1.0 [&BF92] - FWA
* FWA = SQR(FWA) (calculate the square root of FWA)

If the FWA Mantissa byte 1 is zero then jump to &A896 to exit with the FWA set to the floating-point constant at location &BF2E [which is 1.57079633].

Call &A5B3 to set the FWA = (as the result is a Floating-Point value).
Jump to &A8C6 to set the FWA to the Arc-Tangent of the FWA value [FWA = ATN(FWA)].

The ASN calculation for a non-zero positive value is therefore:
ASN(x) = ATN(x / SQR(1 - (x * x)))

Example 1: ASN(0)
[A8AF] Store FWA in &0476
Store FWA in &046C
FWA = FWA * &046C = 0.0
FWA = 1.0 - FWA = 1.0

FWA = SQR(FWA) = 1.0
FWA = &0476 (0) / FWA = 0.0
FWA = ATN(FWA) = 0.0

Example 2: ASN(0.25)

[A8AF] Store FWA in &0476
Store FWA in &046C
FWA = FWA * &046C = 0.0625
FWA = 1.0 - FWA = 0.9375
FWA = SQR(FWA) = 0.9682458
FWA = &0476 (0.25) / FWA = 0.2581988
FWA = ATN(FWA) = 0.2526802

Example 3: ASN(-0.75)

[A8A4] Sign byte of FWA is negative, so clear the sign of the FWA = 0.75
[A8AF] Store FWA in &0476
Store FWA in &046C
FWA = FWA * &046C = 0.5625
FWA = 1.0 - FWA = 0.4375
FWA = SQR(FWA) = 0.6614378
FWA = &0476 (0.75) / FWA = 1.1338934
FWA = ATN(FWA) = 0.848062
[A8AD] Store A (#&FF) in the FWA sign byte. FWA = -0.848062

This diagram shows the relationship between the trig functions.

Disassembly for the ASN routine

A8A1		032 218 150	20 DA 96	JSR &96DA Get and Check Float (convert if Int)
A8A4	.	165 046	A5 2E	LDA &2E
A8A6		016 007	10 07	BPL 7 --> &A8AF
A8A8	d.	100 046	64 2E	STZ &2E
A8AA		032 175 168	20 AF A8	JSR &A8AF
A8AD	#	128 035	80 23	BRA 35 --> &A8D2 Store A in location &2E (i.e. make result negative) and exit
A8AF		032 013 165	20 0D A5	JSR &A50D Store FWA to &0476 and set argp to &0476
A8B2	)	032 041 169	20 29 A9	JSR &A929
A8B5	1	165 049	A5 31	LDA &31
A8B7		240 005	F0 05	BEQ 5 --> &A8BE
A8B9		032 179 165	20 B3 A5	JSR &A5B3 FWA=&0476/FWA & exit with A=#&FF
A8BC		128 008	80 08	BRA 8 --> &A8C6 Calculate ATN result [FWA=ATN(FWA)]
A8BE	.	169 046	A9 2E	LDA#&2E
A8C0	L	076 150 168	4C 96 A8	JMP &A896 Load FWA with FP constant at &BF00 + A

BASIC IV ROM Routines

Page Last Altered: undefined

A8C3 ATN

Submitted by Steve Fewell

Description:

Call routine &96DA to obtain the Floating-point value (at the BASIC text pointer location) and convert iy to a Float value (if it was an Integer) and store it in the FWA.

If the FWA Number is zero then exit with FWA = 0.0 and A = #&FF (to indicate a Floating-Point result).
If the FWA value is negative then clear the FWA sign byte (to make the value positive), call the ATN calculation routine, set the result's (FWA) sign byte to #&FF (to make the value negative) [as after the ATN calculation, A = #&FF] and exit.
Otherwise (FWA value is positive), jump to the ATN calculation routine and exit.

[&A8D5] The ATN Calculation routine:
If the FWA exponent is less than #&81 (i.e. the FWA value is less than 1) then jump to &A8EA to calculate ATN result for an argument in the range 0.0 - 1.0) and exit.

Otherwise, the FWA value is too large to calculate ATN with, so reduce the FWA value as follows:

* Call &A5E9 to calculate the reciple of the FWA value (FWA = 1 / FWA). The result will be in the range 0.0 - 1.0
* As the FWA value is now in the required range, call &A8EA to calculate the ATN result from the FWA value.
* Set the argp pointer to point to the Floating-Point constant value at location &BF2E, which is the value 1.57079683
* Subtract the FWA (ATN result) value from 1.57079683 [Pi / 2] (FWA = argp - FWA).
* Now, the FWA contains the correct ATN result, so exit with A = #&FF (to indicate that the result is Floating-Point).

[&A8EA] The ATN restricted range (0.0 - 1.0) Calculation routine:
IF the FWA exponent is less than #&73 then the FWA value is too small to calculate (i.e. it is practically 0.0), so exit with the FWA value unchanged, and with A = #&FF (to indicate a Floating-Point result).

Store the FWA value to temporary Floating-point location &0476 - &047A.
Set the FWB to contain the value -0.5 (using &A576 (Clear FWB) routine, and setting FWB exponent = #&80, FWB Mantissa byte 1 to #&80 and FWB Sign byte to #&80).
Add the FWB value (-0.5) to the FWA [FWA = FWB + FWA].
Set X to #&97 -> the LSB of the first fraction to apply [i.e. this points to the constant at location &BF97]
Set A to #&C9 -> the LSB of the default value (if the FWA value is too small to evaluate) [i.e. this points to the constant at location &BFC9]
Set Y to #&04 (the number of cycles to evaluate).

Call routine &A861 which will evaluate the continued-fraction expansion series, as follows:

* If the FWA exponent is less than #&40 thern the FWA is too small to be calculated with, so exit with the FWA = the default value at &BFC9, which is 0.927295218.
* Calculate the reciple of the FWA value [FWA = 1 / FWA], store the reciple in &046C. Set argp to &BF97.
* Add argp (&BF97 --> -0.0800532048 (in the Floating-point constant table)) to the FWA value.

* **Cycle 1:**
* Argp = &BF9C (0.155279656). FWA = argp / FWA.
* Argp = &BFA1 (0.440292008). FWA = argp + FWA.
* FWA = &046C (the reciple value) + FWA.

* **Cycle 2:**
* Argp = &BFA6 (0.215397413). FWA = argp / FWA.
* Argp = &BFAB (0.240938382). FWA = argp + FWA.
* FWA = &046C (the reciple value) + FWA.

* **Cycle 3:**
* Argp = &BFB0 (0.0767796872). FWA = argp / FWA.
* Argp = &BFB5 (1.07231162). FWA = argp + FWA.
* FWA = &046C (the reciple value) + FWA.

* **Cycle 4:**
* Argp = &BFBA (0.956538983). FWA = argp / FWA.
* Argp = &BFBF (-0.513841612). FWA = argp + FWA.
* FWA = &046C (the reciple value) + FWA.

* Argp = &BFC4 (-0.254590442). FWA = argp / FWA.
* Argp = &BFC9 (0.927295218). FWA = argp + FWA.

Jump to &A99F to set the argp to the temporary Floating-Point value at location &0476,
multiply the FWA by argp [FWA = argp * FWA] and exit with A = #&FF (to indicate a Floating-point result).

Example 1: ATN(0.0)

FWA is zero, so exit with FWA = 0.0!

Example 2: ATN(-0.75)

Make FWA sign positive (FWA = 0.75)
[&A8EA] Store FWA (0.75) to &0476
FWA = FWA + -0.5 = 0.25
[&A861] FWA = 1 / FWA = 4
Store FWA to &046C
FWA = -0.0800532 + FWA = 3.9199468
FWA = 0.1552796 / FWA = 0.0396126
FWA = 0.4402920 + FWA = 0.4799046
FWA = &046C + FWA = 4.4799046
FWA = 0.2153974 / FWA = 0.0480808
FWA = 0.2409384 + FWA = 0.2890192
FWA = &046C + FWA = 4.2890192
FWA = 0.0767797 / FWA = 0.0179014
FWA = 1.0723116 + FWA = 1.0902131
FWA = &046C + FWA = 5.0902131
FWA = 0.9565390 / FWA = 0.1879172
FWA = -0.5138416 + FWA = -0.3259243

FWA = &046C + FWA = 3.6740757
FWA = -0.2545904 / FWA = -0.0692937
FWA = 0.9272952 + FWA = 0.8580014
FWA = &0476 (0.75) * FWA = 0.6435011
Make FWA sign negative (FWA = -0.6435011)

Example 3: ATN(2.41)

FWA Exponent >= #&81, so: [A8DB] FWA = 1 / FWA = 0.4149377
[&A8EA] Store FWA (2.41) to &0476
FWA = FWA + -0.5 = -0.0850622
[&A861] FWA = 1 / FWA = -11.756098
Store FWA to &046C
FWA = -0.0800532 + FWA = -11.836151
FWA = 0.1552796 / FWA = -0.013119
FWA = 0.4402920 + FWA = 0.4271729
FWA = &046C + FWA = -11.328925
FWA = 0.2153974 / FWA = -0.019013
FWA = 0.2409384 + FWA = 0.2219253
FWA = &046C + FWA = -11.534173
FWA = 0.0767797 / FWA = -0.0066567
FWA = 1.0723116 + FWA = 1.0656549
FWA = &046C + FWA = -10.690443
FWA = 0.9565390 / FWA = -0.089476
FWA = -0.5138416 + FWA = -0.6033176
FWA = &046C + FWA = -12.359416
FWA = -0.2545904 / FWA = 0.0205989
FWA = 0.9272952 + FWA = 0.9478941
FWA = &0476 (0.4149377) * FWA = 0.393317
As the original FWA value has an exponent >= #'&81, FWA = 1.5707963 - FWA = 1.1774793

Example 4: ATN(0.25)

[&A8EA] Store FWA (0.25) to &0476
FWA = FWA + -0.5 = -0.25
[&A861] FWA = 1 / FWA = -4
Store FWA to &046C
FWA = -0.0800532 + FWA = -4.0800532
FWA = 0.1552796 / FWA = -0.0380582
FWA = 0.4402920 + FWA = 0.4022337
FWA = &046C + FWA = -3.5977662
FWA = 0.2153974 / FWA = -0.0598697
FWA = 0.2409384 + FWA = 0.1810686
FWA = &046C + FWA = -3.8189314
FWA = 0.0767797 / FWA = -0.020105
FWA = 1.0723116 + FWA = 1.0522066
FWA = &046C + FWA = -2.9477934
FWA = 0.9565390 / FWA = -0.3244932
FWA = -0.5138416 + FWA = -0.8383348
FWA = &046C + FWA = -4.8383348
FWA = -0.2545904 / FWA = 0.0526194
FWA = 0.9272952 + FWA = 0.9799146
FWA = &0476 (0.25) * FWA = 0.2449786

Example 5: ATN(5.63)

FWA Exponent >= #&81, so: [A8DB] FWA = 1 / FWA = 0.1776198
[&A8EA] Store FWA (0.1776198) to &0476
FWA = FWA + -0.5 = -0.3223802
[&A861] FWA = 1 / FWA = -3.1019275
Store FWA to &046C
FWA = -0.0800532 + FWA = -3.1819807
FWA = 0.1552796 / FWA = -0.0487996
FWA = 0.4402920 + FWA = 0.3914923
FWA = &046C + FWA = -2.7104352
FWA = 0.2153974 / FWA = -0.0794696
FWA = 0.2409384 + FWA = 0.1614686
FWA = &046C + FWA = -2.9404589
FWA = 0.0767797 / FWA = -0.0261114
FWA = 1.0723116 + FWA = 1.0462001
FWA = &046C + FWA = -2.0557274
FWA = 0.9565390 / FWA = -0.4653043
FWA = -0.5138416 + FWA = -0.9791459
FWA = &046C + FWA = -4.0810735
FWA = -0.2545904 / FWA = 0.0623831
FWA = 0.9272952 + FWA = 0.9896783
FWA = &0476 (0.1776198) * FWA = 0.1757864
As the original FWA value has an exponent >= #'&81, FWA = 1.5707963 - FWA = 1.3950098

Example 6: ATN(1.5)

FWA Exponent >= #&81, so: [A8DB] FWA = 1 / FWA = 0.6666666
[&A8EA] Store FWA (0.6666666) to &0476
FWA = FWA + -0.5 = -0.1666666
[&A861] FWA = 1 / FWA = 6
Store FWA to &046C
FWA = -0.0800532 + FWA = 5.9199468
FWA = 0.1552796 / FWA = 0.0262298
FWA = 0.4402920 + FWA = 0.4665219
FWA = &046C + FWA = 6.4665219
FWA = 0.2153974 / FWA = 0.0333096
FWA = 0.2409384 + FWA = 0.274248
FWA = &046C + FWA = 6.274248
FWA = 0.0767797 / FWA = 0.0122372
FWA = 1.0723116 + FWA = 1.089017
FWA = &046C + FWA = 7.089017
FWA = 0.9565390 / FWA = 0.1349325
FWA = -0.5138416 + FWA = -0.378909
FWA = &046C + FWA = 5.6210909
FWA = -0.2545904 / FWA = -0.0452919
FWA = 0.9272952 + FWA = 0.8820032
FWA = &0476 (0.6666666) * FWA = 0.588002
As the original FWA value has an exponent >= #'&81, FWA = 1.5707963 - FWA = 0.9827942

Example 7: ATN(90)

FWA Exponent >= #&81, so: [A8DB] FWA = 1 / FWA = 0.0111111
[&A8EA] Store FWA (0.0111111) to &0476
FWA = FWA + -0.5 = -0.4888888
[&A861] FWA = 1 / FWA = -2.0454545
Store FWA to &046C
FWA = -0.0800532 + FWA = -2.1255077
FWA = 0.1552796 / FWA = -0.0730552
FWA = 0.4402920 + FWA = 0.3672367
FWA = &046C + FWA = -1.6782178
FWA = 0.2153974 / FWA = -0.1283489
FWA = 0.2409384 + FWA = 0.1125895
FWA = &046C + FWA = -1.932865
FWA = 0.0767797 / FWA = -0.0397232
FWA = 1.0723116 + FWA = 1.0325883
FWA = &046C + FWA = -1.0128662
FWA = 0.9565390 / FWA = -0.9443883
FWA = -0.5138416 + FWA = -1.4582299
FWA = &046C + FWA = -3.5036844
FWA = -0.2545904 / FWA = 0.0726636
FWA = 0.9272952 + FWA = 0.9999588
FWA = &0476 (0.0111111) * FWA = 0.0111106
As the original FWA value has an exponent >= #'&81, FWA = 1.5707963 - FWA = 1.5596857

This diagram shows the relationship between the trig functions.

Disassembly for the ATN routine

A8C3		032 218 150	20 DA 96	JSR &96DA Get and Check Float (convert if Int)
A8C6		032 242 163	20 F2 A3	JSR &A3F2 Obtain Sign of the FWA Floating-Point value
A8C9	[	240 091	F0 5B	BEQ 91 --> &A926 Set A to #&FF and exit
A8CB		016 008	10 08	BPL 8 --> &A8D5
A8CD	d.	100 046	64 2E	STZ &2E
A8CF		032 213 168	20 D5 A8	JSR &A8D5
A8D2	.	133 046	85 2E	STA &2E
A8D4	`	096	60	RTS
A8D5	0	165 048	A5 30	LDA &30
A8D7		201 129	C9 81	CMP#&81
A8D9		144 015	90 0F	BCC 15 --> &A8EA
A8DB		032 233 165	20 E9 A5	JSR &A5E9 Floating-Point Reciple (FWA=1/FWA)
A8DE		032 234 168	20 EA A8	JSR &A8EA
A8E1		032 137 165	20 89 A5	JSR &A589 Set argp to &BF2E
A8E4		032 138 166	20 8A A6	JSR &A68A Floating-Point Subtraction [FWA=argp-FWA]
A8E7		169 255	A9 FF	LDA#&FF
A8E9	`	096	60	RTS

A8EA	0	165 048	A5 30	LDA &30
A8EC	s	201 115	C9 73	CMP#&73
A8EE	6	144 054	90 36	BCC 54 --> &A926 Set A to #&FF and exit
A8F0		032 013 165	20 0D A5	JSR &A50D Store FWA to &0476 and set argp to &0476
A8F3	v	032 118 165	20 76 A5	JSR &A576 Clear FWB Mantissa bytes 2 to 5
A8F6		169 128	A9 80	LDA#&80
A8F8	<	133 060	85 3C	STA &3C
A8FA	=	133 061	85 3D	STA &3D
A8FC	;	133 059	85 3B	STA &3B
A8FE		032 146 166	20 92 A6	JSR &A692 FWA = FWA + FWB
A901		162 151	A2 97	LDX#&97
A903		169 201	A9 C9	LDA#&C9
A905		160 004	A0 04	LDY#&04
A907	a	032 097 168	20 61 A8	JSR &A861 Evaluate continued-fraction expansion series
A90A	L	076 159 169	4C 9F A9	JMP &A99F Multiply FWA by variable at &0476

BASIC IV ROM Routines

Page Last Altered: undefined

A90D SIN

Submitted by Steve Fewell

Description:

Clear the carry flag - to indicate that the Sine result is required.
Continue to the COS routine (&A90E) to continue the calculation as follows.

Store the 6502 flags to the 6502 stack, so that the carry flag can be checked later.
Call routine &A93A to obtain the Floating-Point argument and calculate the Sine or Cosine value (the value returned will be either COS or SIN - depending on the value of the Floating-Point argument.

Routine &A93A does the following:

* Call routine &96DA to obtain the value at the Text pointer B location (convert it to Float if it is Integer or issue 'Type mismatch' error if it is a String value).
* If the FWA exponent (location &30) is more than or equal to #&98 then issue 'Accuracy lost' error as the FWA value is too large to use to calculate the SIN/COS value.
* Store the FWA value to temporary Floating point value location &046C-&0470.
* Set the argument pointer (&4A-&4B) to &BF2E (which points to the Float value 1.57079633 in the Floating-Point constant table).
* Call &A4E0 to unpack the Floating-Point constant at &BF2E (1.57079633, which is PI/2) to the FWB
* Store the FWA sign (location &2E) in the FWB sign byte (location &3B) so that the PI/2 value has the same sign as the argument (the FWA value).
* Decrement the FWB exponent (location &3C) to divide the FWB value by 2 - so the FWB now contains the value 0.785398165.
* Call routine &A692 to add the FWB value (PI/4) to the FWA value.
* Set A to #&33.
* Call routine &A9D4 to set the argument pointer (&4A-&4B) to &BF00 + A = &BF33 (which points to the Float value 0.636619772 (which is 2/PI) in the Floating-Point constant table) and then multiply the FWA by the Float value pointed to by the argument pointer (&4A-&4B).
* Call routine &96C3 to convert the Float value (FWA) to an Integer and place the result in the IWA.
* Store A (the LSB of the IWA value - from location &2A) in location &49.
* If the Integer value (&2A-&2C) is zero then jump to &A98D (as the FWA value does not need to be reduced to a value within the required range -(PI/4) to (PI/4) [i.e. -0.785398 to 0.785398]), so [&A98D] load the FWA with the original Sine/Cosine argument (from location &046C-&0470).
* Otherwise, the FWA value is out of the Sine/Cosine range, and needs to be reduced, so:
    * Call &8189 to convert the Integer value (IWA) to a Float and store the result in the temporary Floating-point variable located at &0471-&0475.

    * Then set the argp pointer (&4A-&4B) to pointer to &BF24 (which is the Floating-point constant value -1.57080078) and multiply the FWA by this constant.
    * Set the argp pointer (&4A-&4B) to point to the temporary Floating-point variable location &046C (which is the original FWA Sine/Cosine argument value) and add the Floating-Point value at &046C-&0470 to the FWA value.
    * Store the result in temporary Floating-Point value location &046C-&0470.
    * Set the FWA to the Floating-Point value stored at location &0471-&0475 (the IWA value).
    * Set the argp pointer (&4A-&4B) to point to &BF29 (the address of the value 0.00000445445511 in the Floating-Point constant table).
    * Multiply the FWA by the Floating-Point constant 0.00000445445511 (store the result in the FWA).
    * Set the argp pointer (&4A-&4B) to &046C and add the Floating-Point value at &046C-&0470 to the FWA.
    * Jump to &A990 to perform the Sine/Cosine calculation as the value is now within the required range.
* [A990] Store the FWA value (the actual Sine/Cosine argument, which has been reduced where necessary) to location &0476-&047A.
* Multiply the FWA by the Floating-point value at &0476-&047A (i.e. square the FWA value).
* Set X to #&74 (i.e. the LSB of the first fraction to apply)
* Set A to #&92 (i.e. the LSB of the default value (if the FWA value is too small to evaluate)
* Set Y to #&02 (the number of continued-fraction expansion cycles to evaluate)
* Call routine &A861 to evaluate the continued-fraction expansions, as follows:
    * If the FWA exponent is less than #&40 then the FWA value is too small to evaluate, so exit with the FWA set to the Floating-Point constant at &BF92 (i.e. 1.0).
    * Calculate the reciple of the FWA value (FWA = 1/FWA) and store the result in the FWA and location &046C-&0470
    * Add the Floating-Point constant at &BF74 (-0.0119090311) to the FWA
    * **Cycle 1:**
    * Divide the Floating-Point constant at &BF79 (0.000107499459) by the FWA, storing the result in the FWA.
    * Add the Floating-Point constant at &BF7E (-0.0171640246) to the FWA.
    * Add the Reciple value (from location &046C-&0470) to the FWA.
    * **Cycle 2:**
    * Divide the Floating-Point constant at &BF83 (0.0013095369) by the FWA, storing the result in the FWA.
    * Add the Floating-Point constant at &BF88 (0.0499999922) to the FWA.
    * Add the Reciple value (from location &046C-&0470) to the FWA.

    Last Cycle calculation:
    * Divide the Floating-Point constant at &BF8D (-0.166666666) by the FWA, storing the result in the FWA.
    * Add the Floating-Point constant at &BF92 (1.0) to the FWA.
* Multiply the FWA by the Temporary Floating-Point variable stored at location &0476-&047A (the argument value).
* Exit with A = #&FF (as the result is a Floating-Point value in the FWA)

Now, the FWA contains either the Sine result or the Cosine result (which will have to be converted to the value requested by the user).

Retrieve the carry flag value from the 6502 stack.
The carry flag is clear, indicating that a Sine result is required (therefore, the value at location &49 is not incremented.

[&A923] If bit 0 of location &49 is clear (i.e. the original argument was an even value (in the case of Sine) - or an odd value (in the case of Cosine)) then Set A to #&FF (to indicate that the result is a Floating-point value) and [&A917] check the value in location &49 - if bit 1 is not set then complement the FWA value (FWA=-FWA) and exit; otherwise, just exit.

Otherwise, bit 0 of location &49 is set (indicating that the original argument was odd (in the case of Sine) or even (in the case of Cosine); so, adjust the calculated result as follows:

* Square the FWA value (FWA = FWA * FWA),
* Subtract the FWA value from 1 (FWA=1.0-FWA) and
* Set the FWA value to the square root of the FWA (FWA=SQR(FWA)).

[&A917] check the value in location &49 - if bit 1 is not set then complement the FWA value (FWA=-FWA) and exit; otherwise, just exit.

Example 1: SIN(1.5)

Set &046C = 1.5 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 2.285398
FWA = FWA * 0.636619772 = 1.4549294
?&49 = 1 (Integer part of FWA value). FWA = IWA = 1
Store the FWA in &0471
FWA = FWA * -1.57080078 = -1.57080078
FWA = FWA + &046C = -0.07080078
Store FWA in &046C
Store &0471 (1) in FWA
FWA = FWA * 0.0000044544511 = 0.0000044544511
FWA = FWA + &046C = -0.0707963255
Store the FWA in &0476
FWA = FWA * FWA = 0.0050121197
[&A861]: FWA = 1 / FWA = 199.51638
Store FWA in &046C
FWA = FWA + -0.0119090311 = 199.504474778
FWA = 0.000107499 / FWA = 0.00000005388
FWA = -0.017164024 + FWA = -0.017163485
FWA = &046C + FWA = 199.499216515
FWA = 0.0013095369 / FWA = 0.000006564
FWA = 0.04999999 + FWA = 0.050006554
FWA = &046C + FWA = 199.566387
FWA = -0.16666666 / FWA = -0.00083514
FWA = 1 + FWA = 0.999164856
FWA = FWA * &0476 = -0.070737200377

Location &49 contains the value 1
Bit 0 of &49 is set, so adjust the calculated value as follows:
    FWA = FWA * FWA = 0.005003751517
    FWA = 1 - FWA = 0.9949962484828
    FWA = SQR(FWA) = 0.99749498669558
Bit 1 of &49 is not set, so exit with FWA unchanged.

Example 2: SIN(-0.75)

Set &046C = -0.75 (the argument)
Set FWB to -0.785398165 -(PI/4)
FWA = FWA + FWB = -1.535398165
FWA = FWA * 0.636619772 = -0.9774646
?&49 = 0 (Integer part of FWA value). FWA = IWA = 0
[&A98D] As &49 is 0, store value at &046C in FWA = -0.75
Store the FWA in &0476

FWA = FWA * FWA = 0.5625
[&A861]: FWA = 1 / FWA = 1.777777
Store FWA in &046C
FWA = FWA + -0.0119090311 = 1.7896868
FWA = 0.000107499 / FWA = 0.00006
FWA = -0.017164024 + FWA = -0.0171039
FWA = &046C + FWA = 1.7606737
FWA = 0.0013095369 / FWA = 0.0007437
FWA = 0.04999999 + FWA = 0.0507436
FWA = &046C + FWA = 1.8285213
FWA = -0.16666666 / FWA = -0.0911482
FWA = 1 + FWA = 0.9088517
FWA = FWA * &0476 = -0.6816387

Location &49 contains the value 0
Bits 0 and 1 of &49 are not set, so exit with FWA unchanged.

Example 3: SIN(0.0)

Set &046C = 0.0 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 0.785398165
FWA = FWA * 0.636619772 = 0.50000000
?&49 = 0 (Integer part of FWA value). FWA = IWA = 0
[&A98D] As &49 is 0, store value at &046C in FWA = 0.0
Store the FWA in &0476
FWA = FWA * FWA = 0.0
[&A861]: FWA = 1.0 (as argument is zero)
FWA = FWA * &0476 = 0.0

Location &49 contains the value 0
Bits 0 and 1 of &49 are not set, so exit with FWA unchanged.

Example 4: SIN(0.25)

Set &046C = 0.25 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 1.0353982
FWA = FWA * 0.636619772 = 0.6591548
?&49 = 0 (Integer part of FWA value). FWA = IWA = 0
[&A98D] As &49 is 0, store value at &046C in FWA = 0.25
Store the FWA in &0476
FWA = FWA * FWA = 0.0625
[&A861]: FWA = 1 / FWA = 16
Store FWA in &046C
FWA = FWA + -0.0119090311 = 15.988091
FWA = 0.000107499 / FWA = 0.0000067
FWA = -0.0171640246 + FWA = -0.0171572
FWA = &046C + FWA = 15.982843
FWA = 0.0013095369 / FWA = 0.0000819
FWA = 0.04999999 + FWA = 0.0500818

FWA = &046C + FWA = 16.050082
FWA = -0.16666666 / FWA = -0.0103841
FWA = 1 + FWA = 0.9896158
FWA = FWA * &0476 = 0.2474039

Location &49 contains the value 0
Bits 0 and 1 of &49 are not set, so exit with FWA unchanged.

Example 5: SIN(2.41)

Set &046C = 2.41 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 3.195398
FWA = FWA * 0.636619772 = 2.0342533
?&49 = 2 (Integer part of FWA value). FWA = IWA = 2
Store the FWA in &0471
FWA = FWA * -1.57080078 = -3.1416014
FWA = FWA + &046C = -0.7316014
Store FWA in &046C
Store &0471 (2) in FWA
FWA = FWA * 0.0000044544511 = 0.0000089
FWA = FWA + &046C = -0.7315924
Store the FWA in &0476
FWA = FWA * FWA = 0.5352275
[&A861]: FWA = 1 / FWA = 1.8683641
Store FWA in &046C
FWA = FWA + -0.0119090311 = 1.8564551
FWA = 0.000107499 / FWA = 0.0000057
FWA = -0.017164024 + FWA = -0.0171583
FWA = &046C + FWA = 1.8512058
FWA = 0.0013095369 / FWA = 0.0007073
FWA = 0.04999999 + FWA = 0.0507072
FWA = &046C + FWA = 1.9190714
FWA = -0.16666666 / FWA = -0.0868475
FWA = 1 + FWA = 0.9131524
FWA = FWA * &0476 = -0.6680554

Location &49 contains the value 2
Bit 0 of &49 is not set, so no need to adjust the calculated value
Bit 1 of &49 is set, so FWA = - FWA = 0.6680554

Example 6: SIN(5.63)

Set &046C = 5.63 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 6.415398
FWA = FWA * 0.636619772 = 4.0841688
?&49 = 4 (Integer part of FWA value). FWA = IWA = 4
Store the FWA in &0471
FWA = FWA * -1.57080078 = -6.2832028
FWA = FWA + &046C = -0.6532028

Store FWA in &046C
Store &0471 (4) in FWA
FWA = FWA * 0.0000044544511 = 0.0000178
FWA = FWA + &046C = -0.6531849
[&A990] Store the FWA in &0476
FWA = FWA * FWA = 0.4266506
[&A861]: FWA = 1 / FWA = 2.3438381
Store FWA in &046C
FWA = FWA + -0.0119090311 = 2.3319291
FWA = 0.000107499 / FWA = 0.000046
FWA = -0.017164024 + FWA = -0.0171179
FWA = &046C + FWA = 2.3267202
FWA = 0.0013095369 / FWA = 0.0056282
FWA = 0.04999999 + FWA = 0.0556281
FWA = &046C + FWA = 2.3994662
FWA = -0.16666666 / FWA = -0.0694598
FWA = 1 + FWA = 0.9305401
FWA = FWA * &0476 = -0.6078147

Location &49 contains the value 4
Bits 0 and 1 of &49 are not set, so exit with FWA unchanged.

Example 7: SIN(90)

Set &046C = 90 (the argument)
Set FWB to 0.785398165 (PI/4)
FWA = FWA + FWB = 90.785398
FWA = FWA * 0.636619772 = 57.795773
?&49 = 57 (Integer part of FWA value). FWA = IWA = 57
Store the FWA in &0471
FWA = FWA * -1.57080078 = -89.53564
FWA = FWA + &046C = 0.4643601
Store FWA in &046C
Store &0471 (57) in FWA
FWA = FWA * 0.0000044544511 = 0.0002539
FWA = FWA + &046C = 0.464614
Store the FWA in &0476
FWA = FWA * FWA = 0.2158661
[&A861]: FWA = 1 / FWA = 4.6324998
Store FWA in &046C
FWA = FWA + -0.0119090311 = 4.6205908
FWA = 0.000107499 / FWA = 0.0000023
FWA = -0.017164024 + FWA = -0.0171616
FWA = &046C + FWA = 4.6153381
FWA = 0.0013095369 / FWA = 0.0002837
FWA = 0.04999999 + FWA = 0.0502836
FWA = &046C + FWA = 4.6827834
FWA = -0.16666666 / FWA = -0.0355913
FWA = 1 + FWA = 0.9644086
FWA = FWA * &0476 = 0.4480777

Location &49 contains the value 57 (0011 1001)
Bit 0 of &49 is set, so adjust the calculated value as follows:

FWA = FWA * FWA = 0.2007736
FWA = 1 - FWA = 0.7992263
FWA = SQR(FWA) = 0.8939945
Bit 1 of &49 is not set, so exit with FWA unchanged.

This diagram shows the relationship between the trig functions.

Disassembly for the SIN routine

A90D	024	18	CLC
A90E			...COS routine...

BASIC IV ROM Routines

Page Last Altered: undefined

8000 BASIC ROM header

Submitted by Steve Fewell

Description:

The first 6 bytes contain code to jump to the BASIC ROM startup routine.
This code does the following:

If Accumulator is 1 then jump to start of the language address (&802B);
otherwise, exit (RTS) ('A=1' indicates a normal language entry).
The NOP command in byte &8005 is just to fill space!

Any service address 'jump to startup' code is usually in bytes &8003 to &8005, but
as BASIC doesn't have a service address, this space is used to extend the jump to
the startup routine.

Byte &8006 = ROM Type. &60 indicates a language ROM (i.e. it has a language entry address
& a second processor relocation address).
Byte &8007 = Pointer to the end of title address [minus &8000]. Therefore, as
the value in this byte is &0E, the end of the title is &800E.
Byte &8008 = Internal version number of the ROM = &4 indicating BASIC 4.

Bytes &8009 to &800E contain the title of the ROM ("BASIC"). Ending with a binary 0.
This binary zero (&800E) is the address pointed to by the byte in &8007.
Bytes &800F to &8011 contain the copyright symbol -> "(C)".
Bytes &8012 to &801E contain the copyright string. Ending with a binary 0.
This string contains "1984 Acorn" + chr$(10) [New Line] + chr$(13) [carriage return].
Bytes &801F to &8022 contain the Tube relocation address (32-bit number).
The Tube relocation address for the BASIC ROM is &00008000.

BASIC has no service entry routine.

On entry to the BASIC language, the title string ("BASIC") is automatically printed
(on ROM selection, as it is a language ROM).

Also on entry, the error message pointer (&FD, &FE) is setup to point to
the copyright message (as the BASIC command REPORT proves!) starting at byte &800E as

the error message (BRK) string begins with the Error Number (0 in this case!).

Disassembly for the BASIC ROM header routine

8000		201 001	C9 01	CMP#&01
8002	'	240 039	F0 27	BEQ 39 --> &802B BASIC ROM startup
8004	`	096	60	RTS
8005		234	EA	NOP
8006	`	096	60	EQUB &60
8007		014	0E	EQUB &E Points to &800E
8008		004	04	EQUB &4
8009	BASIC	066 065 083 073 067	42 41 53 49 43	EQUS "BASIC"
800E		000	00	EQUB &0
800F	(C)	040 067 041	28 43 29	EQUS "(C)"
8012	1984 Acorn	049 057 056 052 032 065 099 111 114 110	31 39 38 34 20 41 63 6F 72 6E	EQUS "1984 Acorn"
801C		010	0A	EQUB &0A
801D		013	0D	EQUB &0D
801E		000	00	EQUB &00
801F		000 128 000 000	00 80 00 00	EQUD &00008000

BASIC IV ROM Routines

Page Last Altered: undefined

802B Startup Initialisations

Submitted by Steve Fewell

Description:

If the Random Number seed value (&0D-&11) is zero then default the seed value to &0000575241, where:
byte &0D contains #&41, byte &0E contains #&52, byte &0F contains #&57, byte &10 contains 0
and byte &11 contains 0.
Otherwise, leave the Random Number seed at its current value.

Call OSBYTE &84 (Read top of user RAM address) and set HIMEM (&06-&07) to the address returned.
Call OSBYTE &83 (Read operating system high water mark (OSHWM)) and set PAGE (&18) to the MSB of the address returned.

Zero byte &1F (LISTO Flag).
Initialise the @% variable:
-> Zero &0402 (@% field 2 - number format = 'general')
-> Zero &0403 (@% field 3 - STR$ number format = 'do not use @% for STR$')
-> Set @% field 0 (Field width) to #&0A (10 characters wide)
-> Set @% field 1 (Number of Decimals to display) to #&09 (Show 9 digits after the decimal point)

Set the BRK vector (&0202-&0203) to address &B278 this is the routine that BASIC will run when an error is generated.
Clear the 6502 Interrupt flag (so that operating system interrupts are now allowed).
jump to &8F2D for further initialisation.

Disassembly for the Startup Initialisations routine

802B	%	037 017	25 11	AND &11
802D		005 013	05 0D	ORA &0D
802F		005 014	05 0E	ORA &0E
8031		005 015	05 0F	ORA &0F
8033		005 016	05 10	ORA &10
8035		208 012	D0 0C	BNE 12 --> &8043
8037	A	169 065	A9 41	LDA#&41
8039		133 013	85 0D	STA &0D
803B	I	073 019	49 13	EOR#&13

803D		133 014	85 0E	STA &0E
803F	I	073 005	49 05	EOR#&05
8041		133 015	85 0F	STA &0F
8043		169 132	A9 84	LDA#&84
8045		032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
8048		134 006	86 06	STX &06
804A		132 007	84 07	STY &07
804C	:	058	3A	DEC A
804D		032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
8050		132 024	84 18	STY &18
8052	d	100 031	64 1F	STZ &1F
8054		156 002 004	9C 02 04	STZ &0402
8057		156 003 004	9C 03 04	STZ &0403
805A		162 255	A2 FF	LDX#&FF
805C	#	134 035	86 23	STX &23
805E		162 010	A2 0A	LDX#&0A
8060		142 000 004	8E 00 04	STX &0400
8063		202	CA	DEX
8064		142 001 004	8E 01 04	STX &0401
8067	x	169 120	A9 78	LDA#&78
8069		141 002 002	8D 02 02	STA &0202
806C		169 178	A9 B2	LDA#&B2
806E		141 003 002	8D 03 02	STA &0203
8071	X	088	58	CLI
8072	L-	076 045 143	4C 2D 8F	JMP &8F2D BASIC ROM Startup initialisation (part 2)

BASIC IV ROM Routines

Page Last Altered: undefined

8F2D Startup initialisations (part 2)

Submitted by Steve Fewell

Description:

Set the IWA to value #&F2.
Call &BEE2 to read the contents of memory location &000000F2. This value is the LSB value of the Operating System's Text pointer. Store this value in X.
Call &BEE2 to read the contents of memory location &000000F3. This value is the MSB value of the Operating System's Text pointer. Store the pointer's address in the IWA (&2A = X (contents of &F2); &2B = contents of &F3).
The *EDIT command uses the Operating system's text pointer (&F2-&F3) to point to the 2-byte code '@' followed by '<cr>' and pointer (&00-&01) to store the address of the edited BASIC program (in ASCII text).

Set X to #&14 (the number of attempts to try finding the byte value that we are expecting).
[&8F3F] Decrement X. If X has reached zero then [&8F80] create a NEW empty program and jump to the command line prompt as startup initialisation is now complete.

Call &BEE2 to read the byte at the memory location pointed to by the address in the IWA, and then increment the IWA address.
If the byte is '<cr>' then no program text found at the IWA location, so [&8F80] create a NEW empty program and jump to the command line prompt as startup initialisation is now complete.

If the byte is not #&40 ('@'), then jump back to &8F3F to check the next byte at the IWA address.

Otherwise, we have found byte #&40 ('@'), which indicates that BASIC has been called from *EDIT (or possibly another application).
[&8F4D] Call &BEE2 to read the byte at the memory location pointed to by the address in the IWA, and then increment the IWA address. This is the next byte after the '@' character.

If the byte is not '<cr>' then no program text found at the IWA location, so [&8F80] create a NEW empty program and jump to the command line prompt as startup initialisation is now complete.

Now we have found the byte sequence #&40 ('@') followed by #&0D ('<cr>').
Call &BEFE (NEW) to create an empty program that consists of the following bytes:
* #&0D ('<cr>') - to indicate the start of the program
* #&FF - to indicate the end of the program

[&8F57] Set BASIC Text pointer A (&0B-&0C) value to &0700.
Set Y (pointer offset) to 0.
[&8F5F] Load the next character pointed to by pointer (&00-&01). *EDIT places the start address of the BASIC

program (in ASCII text) in locations (&00-&01).

If the character is &00 (or 'null', ASCII 0) then this indicates the end of the BASIC program, so jump to &8F83 to initialise variable workspace (etc...) and prompt for the command line entry as the startup initialisations and tokenisation of the *EDIT program are now complete.

Otherwise store the character in the command line buffer (&0700, pointed to by (&0B, &0C) at offset Y and increment Y to next next character in the command line buffer).
If Y has now rolled over from &FF to 0, then the BASIC program ASCII text line is too long, and cannot have been created from the *EDIT application, so [&8F80] create a new empty program and jump to &8F83 to initialise variable workspace (etc...) and prompt for the command line entry as the startup initialisations are now complete.
Note: This wipes out the entire program (all lines entered so far), as they were not the *EDIT ASCII text program lines (that we had thought they were).

Increment the (&00-&01) pointer (to the next character on the program line).
If the character that we have just stored at the end of the command line buffer is not '<cr>', then we haven't reached the end of the line yet, so jump back to &8F5F to copy the next character to the command line buffer.

Otherwise, the end of the libne has been reached, so we now need to tokenise the line and insert it into the BASIC program as follows:

* Check whether the *EDIT program text pointer MSB (&01) is more than or equal to the HIMEM MSB address (&07). If it is then do not insert the current line into the program, and do not insert any further lines from the *EDIT text pointer. Instead, jump to &8F83 to initialise variable workspace (etc...) and prompt for the command line entry as the startup initialisations (and tokenisation of the *EDIT program) tasks are now complete.
* Call routine &BAEB to tokenise the ASCII program line text contained within the command line buffer (&0700) and insert the tokenised program line into the program (at the line number specified at the start of the line text).
* Jump back to &8F57 to copy and tokenise the next *EDIT program line.

Disassembly for the Startup initialisations (part 2) routine

8F2D		169 242	A9 F2	LDA#&F2
8F2F		032 024 174	20 18 AE	JSR &AE18 Set IWA to the 8-bit value in A
8F32		032 226 190	20 E2 BE	JSR &BEE2 Read byte from I/O processor (at the address in IWA) & increment IWA
8F35		170	AA	TAX
8F36		032 226 190	20 E2 BE	JSR &BEE2 Read byte from I/O processor (at the address in IWA) & increment IWA
8F39	+	133 043	85 2B	STA &2B
8F3B	*	134 042	86 2A	STX &2A
8F3D		162 020	A2 14	LDX#&14
8F3F		202	CA	DEX
8F40	>	240 062	F0 3E	BEQ 62 --> &8F80 NEW (create null program)
8F42		032 226 190	20 E2 BE	JSR &BEE2 Read byte from I/O processor (at the address in IWA) & increment IWA
8F45		201 013	C9 0D	CMP#&0D
8F47	7	240 055	F0 37	BEQ 55 --> &8F80 NEW (create null program)
8F49	@	201 064	C9 40	CMP#&40
8F4B		208 242	D0 F2	BNE -14 --> &8F3F
8F4D		032 226 190	20 E2 BE	JSR &BEE2 Read byte from I/O processor (at the address in IWA) & increment IWA
8F50		201 013	C9 0D	CMP#&0D
8F52	,	208 044	D0 2C	BNE 44 --> &8F80 NEW (create null program)

8F54		032 254 190	20 FE BE	JSR &BEFE NEW - reset the current program to an empty program
8F57		160 000	A0 00	LDY#&00
8F59	d	100 011	64 0B	STZ &0B
8F5B		169 007	A9 07	LDA#&07
8F5D		133 012	85 0C	STA &0C
8F5F		178 000	B2 00	LDA (&00)
8F61		240 032	F0 20	BEQ 32 --> &8F83 Initialise & prompt for (and execute) command line input
8F63		145 011	91 0B	STA (&0B),Y
8F65		200	C8	INY
8F66		240 024	F0 18	BEQ 24 --> &8F80 NEW (create null program)
8F68		230 000	E6 00	INC &00
8F6A		208 002	D0 02	BNE 2 --> &8F6E
8F6C		230 001	E6 01	INC &01
8F6E		201 013	C9 0D	CMP#&0D
8F70		208 237	D0 ED	BNE -19 --> &8F5F
8F72		165 001	A5 01	LDA &01
8F74		197 007	C5 07	CMP &07
8F76		176 011	B0 0B	BCS 11 --> &8F83 Prompt for command line and execute the entered command(s)
8F78		032 235 186	20 EB BA	JSR &BAEB Tokenise Line of program text and Insert Line into Program
8F7B		128 218	80 DA	BRA -38 --> &8F57

BASIC IV ROM Routines

Page Last Altered: undefined

BEE2 Read byte from I/O processor (at the address in IWA) & increment IWA

Submitted by Steve Fewell

Routine:ReadIObyte
Name: Read byte from I/I processor memory
Starting Address: &BEE2
Entry criteria: The IWA contains the address of the required I/O processor memory location.
Exit: The IWA has been incremented by 1.
A contains the byte at the specified location.

Description:

Store X on the stack while the OSWORD call (below) is set up and performed.
Set A to #&05, X to #&2A and Y to #&00 as the parameter block location is &002A.
Call OSWORD (A = 5) to read byte from I/O processor memory.
The parameter block for the OSWORD 5 call contains the following:
* &2A - I/O processor memory address (LSB)
* &2B - I/O processor memory address byte 2
* &2C - I/O processor memory address byte 3
* &2D - I/O processor memory address (LSB)
* &2E - Returned value: the byte at the I/O processor memory address location

Retrieve X from the stack.

Disassembly for the Read byte from I/O processor at address (IWA) & increment IWA routine

BEE2		169 005	A9 05	LDA#&05
BEE4		218	DA	PHX
BEE5	*	162 042	A2 2A	LDX#&2A
BEE7		160 000	A0 00	LDY#&00
BEE9		032 241 255	20 F1 FF	JSR &FFF1 OSWORD
BEEC		250	FA	PLX
BEED	.	165 046	A5 2E	LDA &2E
BEEF				..Increment IWA value...

BASIC IV ROM Routines

Page Last Altered: undefined

8F7D NEW

Submitted by Steve Fewell

Description:

Call routine &9BA6 to check that the BASIC statement is correct and ends correctly

Call routine &BEFE to carry out the NEW command as follows:

* Set TOP (&12-&13) to PAGE (&18, LSB of PAGE is always zero)
* Set TRACE flag (&20) to 0 (meaning that TRACE is OFF)
* Store #&0D as the first byte of the program, i.e. the '<cr>' character
* Store #&FF as the second byte of the program (previously this byte contained the line number MSB value)
  These 2 characters specify a null program ('<cr>' specifies program start and #&FF specifies program end)
* Set TOP to point to the next byte after these two characters

After the NEW command has been carried out, continue to routine &8F83 to prompt for the next command line and then execute any entered commands.

Disassembly for the NEW routine

8F7D	032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
8F80	032 254 190	20 FE BE	JSR &BEFE reset the program to an empty program
8F83			... Prompt for command line and execute the entered command(s) ...

&BEFE - reset the program to an empty program

BEFE		169 013	A9 0D	LDA#&0D
BF00		164 024	A4 18	LDY &18
BF02		132 019	84 13	STY &13
BF04	d	100 018	64 12	STZ &12
BF06	d	100 032	64 20	STZ &20
BF08		146 018	92 12	STA (&12)
BF0A		169 255	A9 FF	LDA#&FF
BF0C		160 001	A0 01	LDY#&01

BF0E	145 018	91 12	STA (&12),Y
BF10	200	C8	INY
BF11	132 018	84 12	STY &12
BF13	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

831E Calculate next Random Number Seed value

Submitted by Steve Fewell

Routine:CalcRNDSeed
Name: Calculate the Next Random Number Seed value
Starting Address: &831E
Entry criteria: The Random Number Seed (&0D-&11) contain a 5-byte Random Number Seed value.
Exit: The Random Number Seed (&0D-&11) contain the next Random Number Seed value (based on a specific calculated sequence).

Description:

This routine calculates the next seed value (in sequence) from the current seed value (in locations &0D-&11).
The calculation is repeated 4 times (once for each byte to be calculated), during each cycle a new byte is calculated and the other 4 bytes are moved along a position within the Random Number seed value. This enables the calculation to produce varied (non duplicate) results and to always return a valid 'random' result (i.e. never evaluating to a value of 0, which would be useless as a seed value). The calculation is designed not to give frequent duplicate values (which would put the routine into a repitive loop).

This routines sets Y to 4 at the start and repeats the Random Number seed calculation loop (decrementing Y for each loop), until Y is 0.

Each Random Number seed calculation loop obtains a 1-byte result (and moves all other bytes along a position) as follows:

* Set temp-var1 to the value of location &0F divided by 2 [LSR]
* Set temp-var2 to the top 4 bits of location &0E plus the bottom 4 bits of temp-var1
* Copy bottom 2 bits of temp-var2 (bit 0 and bit 1) to the top 2 bits (bit 6 and bit 7) of temp-var3
* Set bit 5 of temp-var3 to the Carry flag status
* Set carry flag to bit 2 of temp-var2
* Copy bit 7, bit 6, bit 5, bit 4 and bit 3 of temp-var2 to bit 4, bit 3, bit 2, bit 1 and bit 0 of temp-var3
* ROR location &11 (move the bits right, set bit 7 to carry and set carry to the bit lost during the move)
  (where carry is bit 2 of temp-var1 in the previous cycle (or 0 if this is the first cycle))
* Set temp-var4 to temp-var3 EOR location &11
* Set location &0D to temp-var4
* Set location &11 to the byte value at location &10
* Set location &10 to the byte value at location &0F
* Set location &0F to the byte value at location &0E
* Set location &0E to the byte value at location &0D

After the 4 cycles, the location &11 is populated with the value of location &0D value from the previous Random

Number Seed value.
This 5-byte value is used as the mantissa for a Floating-Point value in order to generate a random floating point value between 0 and 1.

Example 1:

This example shows the calculation of the first &831E call (where the Random Number Seed is currently set to the initial values: location &0D=>&41; location &0E=>&52; location &0F=>&57;
location &10=>&00 and location &11=>&00.

On entry Carry flag = 0 (?).
On entry: &0D=&41; &0E=&52; &0F=&57; &10=&00; &11=&11.

Y = 4 (number of cycles to process).
[Cycle 1:]
ROR &11 (00000000) => &11 = &00, Carry = 0.
A = &10 => A = &00.
X = A => X = &00.
ROR A => A = &00, carry = 0.
&11 = A => &11 = &00.
A = &0F => A = &57.
&10 = A => &10 = &57.
LSR A (01010111) => A = 00101011 (&2B), carry = 1.
EOR &0E (01010010) => A = 01111001 (&79).
AND #&0F => A = &09.
EOR &0E (01010010) => A = 01011011 (&5B).
ROR A => A = 10101101, carry = 1.
ROR A => A = 11010110, carry = 1.
ROR A => A = 11101011, carry = 0.
ROR A => A = 01110101 (&75), carry = 1.
EOR &11 (00000000) => A = 01110101 (&75).
&11 = X => &11 = &00.
&0F = &0E => &0F = &52.
&0E = &0D => &0E = &41.
&0D = A => &0D = &75.
Now: &0D=&75; &0E=&41; &0F=&52; &10=&57; &11=&00.

Y = Y - 1 => Y = 3.
[Cycle 2:]
ROR &11 (00000000) => &11 = 10000000 (&80) [as Carry was 1], Carry = 0.
A = &10 => A = &57.
X = A => X = &57.
ROR A (01010111) => A = 00101011 (&2B), carry = 1.
&11 = A => &11 = &2B.
A = &0F => A = &52.
&10 = A => &10 = &52.
LSR A (01010010) => A = 00101001 (&29), carry = 0.
EOR &0E (01000001) => A = 01101000 (&68).
AND #&0F => A = &08.
EOR &0E (01000001) => A = 01001001 (&49).
ROR A => A = 00100100, carry = 1.
ROR A => A = 10010010, carry = 0.
ROR A => A = 01001001, carry = 0.

ROR A => A = 00100100 (&24), carry = 1.
EOR &11 (00101011) => A = 00001111 (&0F).
&11 = X => &11 = &57.
&0F = &0E => &0F = &41.
&0E = &0D => &0E = &75.
&0D = A => &0D = &0F.
Now: &0D=&0F; &0E=&75; &0F=&41; &10=&52; &11=&57.

Y = Y - 1 => Y = 2.
[Cycle 3:]
ROR &11 (01010111) => &11 = 10101011 (&AB), Carry = 1.
A = &10 => A = &52.
X = A => X = &52.
ROR A (01010010) => A = 10101001 (&A9), carry = 0.
&11 = A => &11 = &A9.
A = &0F => A = &41.
&10 = A => &10 = &41.
LSR A (01000001) => A = 00100000 (&20), carry = 1.
EOR &0E (01110101) => A = 01010101 (&55).
AND #&0F => A = &05.
EOR &0E (01110101) => A = 01110000 (&70).
ROR A => A = 10111000, carry = 0.
ROR A => A = 01011100, carry = 0.
ROR A => A = 00101110, carry = 0.
ROR A => A = 00010111 (&17), carry = 0.
EOR &11 (10101001) => A = 10111110 (&BE).
&11 = X => &11 = &52.
&0F = &0E => &0F = &75.
&0E = &0D => &0E = &0F.
&0D = A => &0D = &BE.
Now: &0D=&BE; &0E=&0F; &0F=&75; &10=&41; &11=&52.

Y = Y - 1 => Y = 1.
[Cycle 4:]
ROR &11 (01010010) => &11 = 00101001 (&29), Carry = 0.
A = &10 => A = &41.
X = A => X = &41.
ROR A (01000001) => A = 00100000 (&20), carry = 1.
&11 = A => &11 = &20.
A = &0F => A = &75.
&10 = A => &10 = &75.
LSR A (01110101) => A = 00111010 (&3A), carry = 1.
EOR &0E (00001111) => A = 00110101 (&35).
AND #&0F => A = &05.
EOR &0E (00001111) => A = 00001010 (&0A).
ROR A => A = 10000101, carry = 0.
ROR A => A = 01000010, carry = 1.
ROR A => A = 10100001, carry = 0.
ROR A => A = 01010000 (&75), carry = 1.
EOR &11 (00100000) => A = 01110000 (&70).
&11 = X => &11 = &41.
&0F = &0E => &0F = &0F.
&0E = &0D => &0E = &BE.
&0D = A => &0D = &70.
Now: &0D=&70; &0E=&BE; &0F=&0F; &10=&75; &11=&41.

Y = Y - 1 => Y = 0.
The new seed value is: &0D=&70; &0E=&BE; &0F=&0F; &10=&75; &11=&41.

Table showing the initial sequence of Random Number Seed values

(based on the initial startup value and assuming the Seed value is not explicitly reset at any time [using RND(-x)]).

Sequence#	&0D	&0E	&0F	&10	&11
Initial values	&41	&52	&57	&00	&00
1 (831E called once)	&70	&BE	&0F	&75	&41
2 (831E called twice)	&2E	&DB	&60	&41	&70
3	&47	&8F	&02	&2D	&2E
4	&44	&34	&75	&3E	&47
5	&E5	&D6	&7E	&CC	&44
6	&C7	&33	&51	&8B	&E5
7	&8A	&E4	&94	&D6	&C7
8	&15	&D8	&02	&A5	&8A
9	&FA	&3B	&00	&7F	&15
10	&3E	&B6	&3F	&BC	&FA
11	&48	&31	&7C	&A5	&3E
12	&BE	&91	&AA	&91	&48
13	&C3	&A6	&CE	&E1	&BE
14	&C9	&6A	&8B	&9A	&C3
15	&DA	&22	&E9	&7B	&C9
16	&90	&33	&D9	&2F	&DA
17	&85	&91	&D5	&84	&90
18	&75	&99	&72	&1B	&85
19	&F8	&16	&2E	&A4	&75
20	&4B	&88	&78	&33	&F8
After setting seed value to -1 [RND(-1)]	&FF	&FF	&FF	&FF	&40
#1	&FF	&07	&00	&80	&FF
#2	&F8	&FF	&7F	&C0	&FF
After setting seed value to -&FFFFFFFF	&E0	&7D	&BD	&DE	&17
#1	&78	&BD	&80	&38	&E0
#2	&C5	&DF	&97	&17	&78

Disassembly for the Calculate next Random Number Seed value routine

831E		160 004	A0 04	LDY#&04
8320	f	102 017	66 11	ROR &11
8322		165 016	A5 10	LDA &10
8324		170	AA	TAX

8325	j	106	6A	ROR A
8326		133 017	85 11	STA &11
8328		165 015	A5 0F	LDA &0F
832A		133 016	85 10	STA &10
832C	J	074	4A	LSR A
832D	E	069 014	45 0E	EOR &0E
832F	)	041 015	29 0F	AND#&0F
8331	E	069 014	45 0E	EOR &0E
8333	j	106	6A	ROR A
8334	j	106	6A	ROR A
8335	j	106	6A	ROR A
8336	j	106	6A	ROR A
8337	E	069 017	45 11	EOR &11
8339		134 017	86 11	STX &11
833B		166 014	A6 0E	LDX &0E
833D		134 015	86 0F	STX &0F
833F		166 013	A6 0D	LDX &0D
8341		134 014	86 0E	STX &0E
8343		133 013	85 0D	STA &0D
8345		136	88	DEY
8346		208 216	D0 D8	BNE -40 --> &8320
8348	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

8769 BASIC Keyword Execution Address List

Submitted by Steve Fewell

Description:

The BASIC ROM contains a list of execution addresses between locations &8769 and &884C. For each address, the list contains 2-bytes of information specifying which address should be jumped to in order to execute that command (LSB first, MSB second).

The table contains an address for each Keyword Token in the range &8E to &FF.
Plus three extra addresses the meanings of which are currently unknown.
BASIC jumps to the appropriate Execution address, depending of the value of the Token.

Note:
The following Tokens do not need an execution address, as BASIC handles these tokens separately (either in relation to other Keywords or as part of expressions):

- &80 - AND
- &81 - DIV
- &82 - EOR
- &83 - MOD
- &84 - OR
- &85 - ERROR
- &86 - LINE
- &87 - OFF
- &88 - STEP
- &89 - SPC
- &8A - TAB(
- &8B - ELSE
- &8C - THEN
- &8D - Missing [+ ASCII 32 (Space)]

Location of address	Execution adress	Keyword/Desc
&8769	&AADF	OPENIN (token 8E)
&876B	&AAC9	=PTR (token 8F)
&876D	&AE08	=PAGE (token 90)
&876F	&AE44	=TIME (token 91)
&8771	&AE29	=LOMEM (token 92)
&8773	&AE2F	=HIMEM (token 93)
&8775	&ACB7	ABS (token 94)
&8777	&A89C	ACS (token 95)
&8779	&ADEC	ADVAL (token 96)
&877B	&ABB3	ASC (token 97)
&877D	&A8A1	ASN (token 98)
&877F	&A8C3	ATN (token 99)
&8781	&AAD7	BGET (token 9A)
&8783	&A90E	COS (token 9B)
&8785	&AE25	COUNT (token 9C)
&8787	&A9D8	DEG (token 9D)
&8789	&AE35	ERL (token 9E)
&878B	&AE3B	ERR (token 9F)
&878D	&AB05	EVAL (token A0)
&878F	&A9DF	EXP (token A1)
&8791	&AAC5	EXT (token A2)
&8793	&ABE8	FALSE (token A3)
&8795	&B017	FN (token A4)
&8797	&AE3F	GET (token A5)
&8799	&ABC2	INKEY (token A6)
&879B	&AC36	INSTR((token A7)
&879D	&AB8A	INT (token A8)
&879F	&AE11	LEN (token A9)
&87A1	&A746	LN (token AA)
&87A3	&A9CF	LOG (token AB)
&87A5	&AA93	NOT (token AC)
&87A7	&AAE7	OPENUP (token AD)
&87A9	&AAE3	OPENOUT (token AE)
&87AB	&AAFF	PI (token AF)

&87AD	&AC0E	POINT((token B0)
&87AF	&AAA3	POS (token B1)
&87B1	&A9C8	RAD (token B2)
&87B3	&AA73	RND (token B3)
&87B5	&ABF5	SGN (token B4)
&87B7	&A90D	SIN (token B5)
&87B9	&A7B5	SQR (token B6)
&87BB	&A59B	TAN (token B7)
&87BD	&ADF9	TO (token B8)
&87BF	&ABDB	TRUE (token B9)
&87C1	&AAA9	USR (token BA)
&87C3	&AB49	VAL (token BB)
&87C5	&AABC	VPOS (token BC)
&87C7	&B22F	CHR$ (token BD)
&87C9	&AE69	GET$ (token BE)
&87CB	&AEB3	INKEY$ (token BF)
&87CD	&AE73	LEFT$((token C0)
&87CF	&AEC5	MID$((token C1)
&87D1	&AE74	RIGHT$((token C2)
&87D3	&AF1C	STR$((token C3)
&87D5	&AF47	STRING$((token C4)
&87D7	&ABCF	EOF (token C5)

End of function execution addresses. Below the list continues with the execition addresses for the BASIC commands.

&87D9	&9489	AUTO (token C6)
&87DB	&9317	DELETE (token C7)
&87DD	&8F20	LOAD (token C8)
&87DF	&B3DD	LIST (token C9)
&87E1	&8F7D	NEW (token CA)
&87E3	&8F00	OLD (token CB)
&87E5	&9384	RENUMBER (token CC)
&87E7	&BE55	SAVE (token CD)
&87E9	&B393	EDIT (token CE)
&87EB	&BE97	PTR= (token CF)

&87ED	&9634	PAGE= (token D0)
&87EF	&9679	TIME= (token D1)
&87F1	&9620	LOMEM= (token D2)
&87F3	&960F	HIMEM= (token D3)
&87F5	&B2C8	SOUND (token D4)
&87F7	&BEBD	BPUT (token D5)
&87F9	&92BE	CALL (token D6)
&87FB	&8EFB	CHAIN (token D7)
&87FD	&963E	CLEAR (token D8)
&87FF	&BEAE	CLOSE (token D9)
&8801	&97E0	CLG (token DA)
&8803	&97E7	CLS (token DB)
&8805	&8FAE	DATA (token DC)
&8807	&8FAE	DEF (token DD)
&8809	&9534	DIM (token DE)
&880B	&97A6	DRAW (token DF)
&880D	&8F25	END (token E0)
&880F	&9B9A	ENDPROC (token E1)
&8811	&B2EC	ENVELOPE (token E2)
&8813	&B618	FOR (token E3)
&8815	&B6D9	GOSUB (token E4)
&8817	&B71D	GOTO (token E5)
&8819	&9741	GCOL (token E6)
&881B	&9C08	IF (token E7)
&881D	&B8B6	INPUT (token E8)
&881F	&904A	LET (token E9)
&8821	&9703	LOCAL (token EA)
&8823	&975F	MODE (token EB)
&8825	&97A2	MOVE (token EC)
&8827	&B4F1	NEXT (token ED)
&8829	&B75B	ON (token EE)
&882B	&980D	VDU (token EF)
&882D	&97B1	PLOT (token F0)
&882F	&918D	PRINT (token F1)
&8831	&96E4	PROC (token F2)

&8833	&B97D	READ (token F3)
&8835	&8FAE	REM (token F4)
&8837	&BA58	REPEAT (token F5)
&8839	&97F4	REPORT (token F6)
&883B	&B94D	RESTORE (token F7)
&883D	&B707	RETURN (token F8)
&883F	&8F12	RUN (token F9)
&8841	&9086	STOP (token FA)
&8843	&9755	COLOUR (token FB)
&8845	&9646	TRACE (token FC)
&8847	&BA17	UNTIL (token FD)
&8849	&B317	WIDTH (token FE)
&884B	&BE87	OSCLI (token FF)

BASIC IV ROM Routines

Page Last Altered: undefined

AADF OPENIN

Submitted by Steve Fewell

Description:

Set A to #&40. This is the OSFIND (&FFCE) option to open a file for input access only.
Call routine &AAE9 (part of OPENUP) to obtain the filename and place it in the SWA (or issue 'Type mismatch' error if a String value was not found), Append a '<cr>' character to the end of the SWA (so that the filename is correctly terminated for the OSFIND routine's requirements), Set X and Y to point to the SWA (&0600), call OSFIND (&FFCE) to open the file, and return the opened file's channel number in the IWA.

Disassembly for the OPENIN routine

AADF	@	169 064	A9 40	LDA#&40
AAE1		128 006	80 06	BRA 6 --> &AAE9 Get filename, call OSFIND to open the filename specified in the SWA & set IWA to channel#

BASIC IV ROM Routines

Page Last Altered: undefined

AAE7 OPENUP

Submitted by Steve Fewell

Description:

Set A to #&C0. This is the OSFIND (&FFCE) option to open a file for update/random access.
Store A (the OSFIND option number) to the 6502 stack.

Call routine &AD36 to obtain the value after the OPENUP keyword (this should be the filename to open).
If the result of the expression value is not a String value then issue a 'Type mismatch' error.

Call routine &BE2B to store a carriage return character at the end of the SWA (as the OSFIND operation requires the filename to be terminated by a '<cr>' character.

Set X to #&00 and Y to #&06, so that X & Y point to the filename (i.e. the SWA location - &0600).
Retrieve the OSFIND option from the 6502 stack.
Call the Operating System OSFIND routine to perform the file open operation and return the channel number that the file was opened on in A.

Jump to routine &AE18 to set the IWA to the 8-bit value in A (i.e. the channel number of the opened file).
This causes this routine to return the channel number value - which is usually then assigned to a variable by the BASIC program.

Disassembly for the OPENUP routine

AAE7		169 192	A9 C0	LDA#&C0
AAE9	H	072	48	PHA
AAEA	6	032 054 173	20 36 AD	JSR &AD36 Get value / result of expression (if bracketed)
AAED		208 013	D0 0D	BNE 13 --> &AAFC
AAEF	+	032 043 190	20 2B BE	JSR &BE2B Store #&0D (carriage return, '<cr>') at the end of the SWA value
AAF2		162 000	A2 00	LDX#&00
AAF4		160 006	A0 06	LDY#&06
AAF6	h	104	68	PLA
AAF7		032 206 255	20 CE FF	JSR &FFCE OSFIND
AAFA		128 198	80 C6	BRA -58 --> &AAC2 [JMP &AE18 Set IWA to the 8-bit value in A]

AAFC L 076 146 144 4C 92 90 JMP &9092 Type mismatch error

BASIC IV ROM Routines

Page Last Altered: undefined

AABC VPOS

Submitted by Steve Fewell

Description:

Call OSBYTE #&86 to obtain the Text cursor position. After this call, X contains the horizontal position of the cursor (POS) and Y contains the vertical position of the cursor (VPOS).
Set A to Y (VPOS value) and jump to &AE18 to set the IWA to the 1-byte value in A and exit.

Disassembly for the VPOS routine

AABC		169 134	A9 86	LDA#&86
AABE		032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
AAC1		152	98	TYA
AAC2	L	076 024 174	4C 18 AE	JMP &AE18 Set IWA to 8-bit value in A

BASIC IV ROM Routines

Page Last Altered: undefined

AAC9 =PTR

Submitted by Steve Fewell

Description:

Store #&00 on the stack. This is the OSARGS (&FFDA) option to retrieve the file's PTR value.
Call routine &BA4A to check that the next character at the BASIC Text Pointer B location (&0B-&0C) is a '#'-character. If not then issue 'Missing #' error message. Evaluate the expression after the '#' [Type mismatch error if string value found], and set Y to the number found after the '#' character (this is the file channel number).

Call OSARGS (with X [the parameter block] set to the IWA (#&2A) - this is the 4-byte parameter block that the file's PTR value will be returned to.
After the call to OSARGS (&FFDA), the IWA will contain the PTR value for the file that is opened on channel Y.

Call &AA90 to exit with A = #&40 (as the result value is an Integer).

Disassembly for the =PTR routine

AAC9		169 000	A9 00	LDA#&00
AACB	H	072	48	PHA
AACC	J	032 074 186	20 4A BA	JSR &BA4A Check for '#' (PTR B), Set Y to file channel number (PTR B)
AACF	*	162 042	A2 2A	LDX#&2A
AAD1	h	104	68	PLA
AAD2		032 218 255	20 DA FF	JSR &FFDA OSARGS
AAD5		128 185	80 B9	BRA -71 --> &AA90 Exit with A=&40

BASIC IV ROM Routines

Page Last Altered: undefined

AE08 =PAGE

Submitted by Steve Fewell

Description:

This routine will return the value of PAGE.
Set Y (MSB of the address) to the value of location &18 (i.e. the PAGE number).
Set A (LSB of the address) to 0, as pages always start at byte 0 within the page.
Call routine &AE1A to set the IWA to the value of Y * 256 + A.

Disassembly for the =PAGE routine

AE08	164 024	A4 18	LDY &18
AE0A	169 000	A9 00	LDA#&00
AE0C	128 012	80 0C	BRA 12 --> &AE1A Set IWA to 16-bit value

BASIC IV ROM Routines

Page Last Altered: undefined

AE44 =TIME

Submitted by Steve Fewell

Description:

If the next character after the TIME keyword is a dollar sign '$' character then jump to the =TIME$ routine.

Set Y to #&00 and X to #&2A, so that &002A is the first byte of the parameter block for the OSWORD call).
Execute the OSWORD &01 command to obtain the TIME value and place it in the parameter block (the IWA).
Exit with A = #&40 (as the current value is an Integer and is located in the IWA).

Disassembly for the =TIME routine

AE44		200	C8	INY
AE45		177 025	B1 19	LDA (&19),Y
AE47	$	201 036	C9 24	CMP#&24
AE49		240 012	F0 0C	BEQ 12 --> &AE57 TIME$
AE4B	*	162 042	A2 2A	LDX#&2A
AE4D		160 000	A0 00	LDY#&00
AE4F		169 001	A9 01	LDA#&01
AE51		032 241 255	20 F1 FF	JSR &FFF1 OSWORD
AE54	@	169 064	A9 40	LDA#&40
AE56	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

AE57 =TIME$

Submitted by Steve Fewell

Description:

Increment the BASIC Text Pointer B offset (to skip the dollar sign character.
Set Y to #&06 and X to #&00, so that &0600 (the SWA) is the first byte of the OSWORD parameter block.
Zero the first byte of the parameter block (to pass an empty string to the OSWORD call).
Execute the OSWORD &0E command to obtain the TIME$ value and place it in the parameter block (the SWA).
Set the SWA length (byte &36) to #&18 (the fixed length of a TIME$ value).
Exit with A = #&00 (as the current value is a String and is located in the SWA).

Disassembly for the =TIME$ routine

AE57		230 027	E6 1B	INC &1B
AE59		169 014	A9 0E	LDA#&0E
AE5B		162 000	A2 00	LDX#&00
AE5D		160 006	A0 06	LDY#&06
AE5F		156 000 006	9C 00 06	STZ &0600
AE62		032 241 255	20 F1 FF	JSR &FFF1
AE65		169 024	A9 18	LDA#&18
AE67	&	128 038	80 26	BRA 38 --> &AE8F Set SWA length (&36) to A and exit with A = 0

BASIC IV ROM Routines

Page Last Altered: undefined

AE74 RIGHT$(

Submitted by Steve Fewell

Description:

On entry carry flag is set if the RIGHT$ function was called (&AE74); otherwise, if carry is clear then we were entered from the LEFT$ function (&AE73).
Store the entry status (carry flag) to the 6502 processor stack.
Call routine &9D3B to evaluate the expression at BASIC text pointer B and issue a 'Type mismatch' error if a String value was not returned.
If the next non-space character (stored in X) is not a comma then issue 'Missing ,' error; otherwise, increment BASIC text pointer B offset to skip past the comma value.

Push the SWA value to the BASIC Stack, get the Integer result of the expression at BASIC Text pointer B ('Type mismatch' error if value is not numeric) and check for a closing ')' character ('Missing)' error if closing parenthesis not found.

Retrieve the String value (SWA) from the BASIC Stack.
Retrieve the carry flag routine entry setting from the 6502 processor stack.

If carry is not set (LEFT$) then:

* If the Integer value (&2A) is greater than or equal to the length of the String value in the SWA (&36) then exit with A = 0 (as the result is a String value).
* Otherwise, Set the SWA String length (&36) to the Integer value (to reduce the SWA length to the width specified and exit with A = 0 (as the result is a String value).

If carry is set (RIGHT$) then:

* Subtract the Integer value (&2A) from the SWA length (&36) [Note that carry flag is automatically set before subtraction].
* If the subtraction underflowed (the Integer value is greater than of equal to the SWA length) then exit with A = 0 (as the result is a String value).
* Set X to the result (the first character of the result value).
* Set the SWA length (&36) to the Integer value specified (&2A).
* If the new SWA length is 0 then exit.
* Set Y = 0 and keep copying the character at SWA location X to SWA location Y and then incrementing Y and X, and decrementing &2A until &2A is 0 (the required number of characters has been copied).
* Exit with A = 0 (as the result is a String value located in the SWA).

Disassembly for the RIGHT$(routine

AE74		008	08	PHP
AE75	;	032 059 157	20 3B 9D	JSR &9D3B Evaluate expression at BASIC Text pointer B
AE78	E	208 069	D0 45	BNE 69 --> &AEBF [JMP &9092 Type mismatch error]
AE7A	,	224 044	E0 2C	CPX#&2C
AE7C	D	208 068	D0 44	BNE 68 --> &AEC2 [JMP &8EF6 'Missing ,' error]

AE7E		230 027	E6 1B	INC &1B
AE80		032 164 150	20 A4 96	JSR &96A4 Push SWA to stack, get Integer result of expression & check for closing ')'
AE83		032 210 188	20 D2 BC	JSR &BCD2 Pop String (SWA) from the stack
AE86	(	040	28	PLP
AE87		176 011	B0 0B	BCS 11 --> &AE94 Calculate RIGHT$
AE89	*	165 042	A5 2A	LDA &2A
AE8B	6	197 054	C5 36	CMP &36
AE8D		176 002	B0 02	BCS 2 --> &AE91
AE8F	6	133 054	85 36	STA &36
AE91		169 000	A9 00	LDA#&00
AE93	`	096	60	RTS
AE94	6	165 054	A5 36	LDA &36
AE96	*	229 042	E5 2A	SBC &2A
AE98		144 247	90 F7	BCC -9 --> &AE91
AE9A		240 247	F0 F7	BEQ -9 --> &AE93
AE9C		170	AA	TAX
AE9D	*	165 042	A5 2A	LDA &2A
AE9F	6	133 054	85 36	STA &36
AEA1		240 240	F0 F0	BEQ -16 --> &AE93
AEA3		160 000	A0 00	LDY#&00
AEA5		189 000 006	BD 00 06	LDA &0600,X
AEA8		153 000 006	99 00 06	STA &0600,Y
AEAB		232	E8	INX
AEAC		200	C8	INY
AEAD	*	198 042	C6 2A	DEC &2A
AEAF		208 244	D0 F4	BNE -12 --> &AEA5
AEB1		128 222	80 DE	BRA -34 --> &AE91

Disassembly for the Push SWA to stack, get Integer result of expression & check for closing ')' routine

96A4	Q	032 081 188	20 51 BC	JSR &BC51 Push SWA to Stack
96A7				... Extract Integer result of expression & check for ')' ...

BASIC IV ROM Routines

Page Last Altered: undefined

AE29 =LOMEM

Submitted by Steve Fewell

Description:

This routine will return the value of LOMEM in the IWA.
Set A to &00 (The LSB of the LOMEM address).
Set Y to &01 (The MSB of the LOMEM address).
Call routine &AE1A to set the IWA to the value of Y * 256 + A.

Disassembly for the =LOMEM routine

AE29	165 000	A5 00	LDA &00
AE2B	164 001	A4 01	LDY &01
AE2D	128 235	80 EB	BRA -21 --> &AE1A Set IWA to 16-bit value

BASIC IV ROM Routines

Page Last Altered: undefined

AE2F =HIMEM

Submitted by Steve Fewell

Description:

This routine will return the value of HIMEM in the IWA.
Set A to &06 (The LSB of the HIMEM address).
Set Y to &07 (The MSB of the HIMEM address).
Call routine &AE1A to set the IWA to the value of Y * 256 + A.

Disassembly for the =HIMEM routine

AE2F	165 006	A5 06	LDA &06
AE31	164 007	A4 07	LDY &07
AE33	128 229	80 E5	BRA -27 --> &AE1A Set IWA to 16-bit value

BASIC IV ROM Routines

Page Last Altered: undefined

ACB7 ABS

Submitted by Steve Fewell

Description:

Call the Evaluate value routine (&AD36) which will evaluate the next value at BASIC text
pointer B location, or the expression at that location if the value begins with an open bracket ("(").
If the returned value is a String (A = 0) then issue a Type mismatch error as the value is not numeric.

If the returned value is a Floating-Point value then jump to &ACC4 to set the FWA Sign byte to zero.
This will force the FWA's sign to be positive.
Otherwise, continue to the Integer Positive routine (&ACBE), which will make the number in the IWA positive.

Disassembly for the ABS routine

ACB4	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
ACB7	6	032 054 173	20 36 AD	JSR &AD36 Evaluate value
ACBA		240 248	F0 F8	BEQ -8 --> &ACB4
ACBC	0	048 006	30 06	BMI 6 --> &ACC4
...				...
ACC4	d.	100 046	64 2E	STZ &2E
ACC6	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

ADEC ADVAL

Submitted by Steve Fewell

Description:

Call routine &96B4 to obtain the Integer value from the expression after the ADVAL keyword (issue 'Type mismatch' error if a String value was found).
Set X (the OSBYTE parameter) to the byte at location &2A (The LSB of the Integer ADVAL parameter obtained).
Perform the OSBYTE #&80 command to execute the ADVAL function.
On return, X will be the low byte (LSB) of the ADVAL return value and Y will be the high byte (MSB) of the ADVAL return value.
Set A to the value of X (the LSB of the return value of the OSBYTE ADVAL call).
Call routine &AE1A to set the IWA to the 2-byte value of: A (low byte) and Y (high byte).

Disassembly for the ADVAL routine

ADEC		032 180 150	20 B4 96	JSR &96B4 Get Integer value at PTR B
ADEF	*	166 042	A6 2A	LDX &2A
ADF1		169 128	A9 80	LDA#&80
ADF3		032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
ADF6		138	8A	TXA
ADF7	!	128 033	80 21	BRA 33 --> &AE1A Set IWA to 16-bit value

---

BASIC IV ROM Routines

Page Last Altered: undefined

ABB3 ASC

Submitted by Steve Fewell

Description:

Call the Evaluate value routine (&AD36) which will evaluate the next value at BASIC text
pointer B location, or the expression at that location if the value begins with an open bracket ("(").
If the obtained value is not a String (A is not 0) then generate a Type mismatch error as a string is required.

If the length of the String (&36, the length of the SWA) is 0 then set the IWA
to -1 and exit, as a zero length string does not have a valid ASCII code!.

Otherwise, Load A with the ASCII code of the first character and jump to routine &AE18 to set the IWA to the
8-bit value of A and exit.

Disassembly for the ASC routine

ABB3	6	032 054 173	20 36 AD	JSR &AD36 Evaluate value
ABB6		208 020	D0 14	BNE 20 --> &ABCC [JMP &9092 - Type mismatch error]
ABB8	6	165 054	A5 36	LDA &36
ABBA		240 031	F0 1F	BEQ 31 --> &ABDB Set IWA to -1
ABBC		173 000 006	AD 00 06	LDA &0600
ABBF	L	076 024 174	4C 18 AE	JMP &AE18 Set IWA to 8-bit value in A

BASIC IV ROM Routines

Page Last Altered: undefined

Set IWA = TRUE

Submitted by Steve Fewell

Routine: TRUE
Name: Set IWA = -1
Starting Address: &ABDB
Entry criteria: None
Exit: IWA contains -1.

Description:
Sets the IWA to -1 [Note: -1 is where every byte of the IWA equals 255], and sets A to #&40 to show that an Integer number is being processed.

Disassembly for the Set IWA = TRUE routine

ABDB	162 255	A2 FF	LDX#&FF
ABDD			
Store X (#&FF) in every byte of the IWA	...		... (routine in iFALSE) - ABE8

BASIC IV ROM Routines

Page Last Altered: undefined

AAD7 BGET

Submitted by Steve Fewell

Description:

Call routine &BA4A to Check that the next character at BASIC Text Pointer B is '#' (issue 'Missing #' error if not), set Y to the LSB of the number after the '#' (the file channel number).

Call the Operating system OSBGET routine (&FFD7) to carry out the read next byte from the file open at channel number 'Y' operation.
Jump to routine &AE18 to set the IWA to the 8-bit value in A (i.e. the byte that was read from the file).

Disassembly for the BGET routine

AAD7	J	032 074 186	20 4A BA	JSR &BA4A Check for '#' (PTR B), Set Y to file channel number (PTR B)
AADA		032 215 255	20 D7 FF	JSR &FFD7 OSBGET
AADD		128 227	80 E3	BRA -29 --> &AAC2 [JMP &AE18 Set IWA to the 8-bit value in A]

BASIC IV ROM Routines

Page Last Altered: undefined

AE25 COUNT

Submitted by Steve Fewell

Description:

Set A to &1E (The current COUNT value) and call routine &AE18 to set the IWA to the 1-byte value contained in A.

Disassembly for the COUNT routine

AE25	165 030	A5 1E	LDA &1E
AE27	128 239	80 EF	BRA -17 --> &AE18 Set IWA to 8-bit value (in A)

BASIC IV ROM Routines

Page Last Altered: undefined

A9D8 DEG

Submitted by Steve Fewell

Description:

Call the &96DA routine to get the Float value.
Jump to &A9D4 with A = &3D to multiply the FWA by the Floating-Point constant at location &BF3D in the Floating-Point constant table.
The constant value at &BF3D is 57.2957795. This value is 1 / (Pi / 180), or Radan. Multiplying the Float value by this constant gives the value in Degrees.

Disassembly for the DEG routine

A9D8		032 218 150	20 DA 96	JSR &96DA Get and Check Float (conv if Int)
A9DB	=	169 061	A9 3D	LDA#&3D
A9DD		128 245	80 F5	BRA -11 --> &A9D4 Multiply FWA by &BF3D (57.2957795)

BASIC IV ROM Routines

Page Last Altered: undefined

AE35 ERL

Submitted by Steve Fewell

Description:

This routine will return the Line Number of the last Program Line with generated an error (ERL) in the IWA.
Set A to &08 (The LSB of the Program Line Number).
Set Y to &09 (The MSB of the Program Line Number).
Call routine &AE1A to set the IWA to the value of Y * 256 + A.

Disassembly for the ERL routine

AE35	164 009	A4 09	LDY &09
AE37	165 008	A5 08	LDA &08
AE39	128 223	80 DF	BRA -33 --> &AE1A Set IWA to 16-bit value

BASIC IV ROM Routines

Page Last Altered: undefined

AE3B ERR

Submitted by Steve Fewell

Description:

Set A to the byte value at the location pointed to by &FD-&FE. &FD-&FD is the Operating System's pointer to the last BRK (error message) instruction. The first byte of the error message is the Error Number (or ERR). call routine &AE18 to set the IWA to the 1-byte value contained in A.

Disassembly for the ERR routine

AE3B	178 253	B2 FD	LDA (&FD)
AE3D	128 217	80 D9	BRA -39 --> &AE18 Set IWA to 8-bit value (in A)

BASIC IV ROM Routines

Page Last Altered: undefined

AB05 EVAL

Submitted by Steve Fewell

Description:

Call routine &AD36 to obtain the value of the expression after the EVAL keyword.
If the expression result is not 0 (i.e. the result is Numeric & not a String value) then issue 'Type mismatch' error.

Increment the SWA length and store a '<cr>' (carriage return) character at the end of the existing SWA value.
Push the SWA value to the BASIC stack.
Store the current BASIC Text pointer B values (&19, &1A & &1B) to the 6502 stack for temporary storage.

Set BASIC Text Pointer B (&19, &1A) and pointer (&37, &38) to the current address of the BASIC Stack pointer + 1, the BASIC Stack pointer points to the next free location on the stack, so adding the 1 will point to the first character (length) of the SWA value on the stack.

Set location &3B to #&FF (to set the middle of a statement mode).
Zero byte &3C (as a Line number is not expected).
Call the &8DB2 routine to tokenise the String value.

Zero byte &1B (BASIC Text pointer B offset), to reset BASIC Text Pointer B to the start of the tokenised String value.
Call routine &9D3B to evalualate the tokenised string and obtain the result of the String Expression to the SWA.
Call routine &BCE1 to restore the Stack space used by the String value.
Continue to the ASCII-Numeric routine (&AB3A) to retrieve the BASIC Text pointer B values from the 6502 stack and convert the ASCII String result (SWA) to a numeric value (either a Floating-Point value or an Integer value) and exit.

Disassembly for the EVAL routine

AB05	6	032 054 173	20 36 AD	JSR &AD36 Get Value (Keyword, variable, value, open bracket)
AB08	<	208 060	D0 3C	BNE 60 --> &AB46 [JMP &9092 Type mismatch error]
AB0A	6	230 054	E6 36	INC &36
AB0C	6	164 054	A4 36	LDY &36
AB0E		169 013	A9 0D	LDA#&0D
AB10		153 255 005	99 FF 05	STA &05FF,Y
AB13	Q	032 081 188	20 51 BC	JSR &BC51 Push SWA to Stack
AB16		165 025	A5 19	LDA &19

```
AB18   H  072          48        PHA
AB19      165 026      A5 1A     LDA &1A
AB1B   H  072          48        PHA
AB1C      165 027      A5 1B     LDA &1B
AB1E   H  072          48        PHA
AB1F      164 004      A4 04     LDY &04
AB21      166 005      A6 05     LDX &05
AB23      200          C8        INY
AB24      132 025      84 19     STY &19
AB26   7  132 055      84 37     STY &37
AB28      208 001      D0 01     BNE 1 --> &AB2B
AB2A      232          E8        INX
AB2B      134 026      86 1A     STX &1A
AB2D   8  134 056      86 38     STX &38
AB2F      032 031 142  20 1F 8E  JSR &8E1F Set tokenise flags (&3B & &3C) and tokenise the String value
AB32   d  100 027      64 1B     STZ &1B
AB34   ;  032 059 157  20 3B 9D  JSR &9D3B Get the result of the expression at PTR B
AB37      032 225 188  20 E1 BC  JSR &BCE1 Restore Stack Space used by String
AB3A                             ... ASCNUM (Convert ASCII String value to Numeric value) ...
```

BASIC IV ROM Routines

Page Last Altered: undefined

AAC5 [=] EXT

Submitted by Steve Fewell

Description:

Set A to #&02 --> this is the OSARGS (&FFDA) option to obtain the file's EXT value.
Call &AACB (in the =PTR routine) to read the '#' character (issuing 'Missing #' error if not found), read the Integer after the '#' (the file channel number) - store the LSB of this value in Y, Call OSARGS (with X [the parameter block] set to the IWA (#&2A), and exit with A = #&40 (as the result value is an Integer).
After the call to OSARGS (&FFDA), the IWA will contain the EXT value for the file that is opened on channel Y.

Disassembly for the EXT routine

AAC5	169 002	A9 02	LDA#&02
AAC7	128 002	80 02	BRA 2 --> &AACB Get '#', get channel number (in Y) & call OSARGS to return the EXT value in the IWA

BASIC IV ROM Routines

Page Last Altered: undefined

AE3F GET

Submitted by Steve Fewell

Description:

Call OSRDCH operating system routine to wait for a key to be pressed on the keyboard
(if the keyboard buffer is empty) and return in A the next character read from the
Keyboard buffer.
Call the Set 8-bit Integer routine to put the ASCII value of the returned character
into the first byte of the IWA (setting the rest of the IWA to zero),
returning with A = &40 (as result was an Integer).

Disassembly for the GET routine

AE3F	032 224 255	20 E0 FF	JSR &FFE0 OSRDCH
AE42	128 212	80 D4	BRA -44 --> &AE18

BASIC IV ROM Routines

Page Last Altered: undefined

ABC2 INKEY

Submitted by Steve Fewell

Description:

Call routine &AA12 to get the Integer parameter (after the INKEY keyword) and call OSBYTE &81 (Inkey).
&AA12 will set X (LSB) and Y (MSB) to the Inkey Time limit (if <= &7FFF) or Key value to check (if >= &8000) specified in the IWA (&2A (LSB) & &2B (MSB)).

On return from the OSBYTE #&81 call, the following conditions will apply:
If a negative parameter value is specified then Y & X will both equal #&FF if the key specified is being pressed.
If a positive parameter value is specified then Y will equal 0 if a key has been pressed within the time limit, and X will be the ASCII value of the Key that was pressed.

If the OSBYTE #&81 call returns with Y not set to 0 then no key was pressed (if Time Limit) or the specified key was pressed (if a negative - specific key scan - parameter was supplied), so exit with IWA set to -1 and A set to #&40 (result is Integer).
Otherwise, Set A to X (the ASCII code of the character pressed, or #&FF (if specific key scan was done - and the key wasn't pressed) and set the IWA to the 2-byte value of: A (low byte) and Y (high byte) and exit with A=#&40 (result is Integer).

Disassembly for the INKEY routine

ABC2		032 018 170	20 12 AA	JSR &AA12 Get Integer value and call OSBYTE &81 (Inkey)
ABC5		152	98	TYA
ABC6		208 019	D0 13	BNE 19 --> &ABDB Set IWA to TRUE (-1)
ABC8		138	8A	TXA
ABC9	L	076 026 174	4C 1A AE	JMP &AE1A Set IWA to 16-bit value

AA12: Get Integer value and call OSBYTE &81 (Inkey)

AA12		032 180 150	20 B4 96	JSR &96B4 Get Integer value at PTR B
AA15		169 129	A9 81	LDA#&81
AA17	*	166 042	A6 2A	LDX &2A
AA19	+	164 043	A4 2B	LDY &2B

AA1B L 076 244 255 4C F4 FF JMP &FFF4 OSBYTE

BASIC IV ROM Routines

Page Last Altered: undefined

AC36 INSTR(

Submitted by Steve Fewell

Description:

Call &9D3B to obtain the result of the expression at BASIC Text pointer B. This value should be the String value to search within for the search string value.
If the value of the expression is not a String then issue a 'Type mismatch' error, as a string to search value was not found.

If the next non-space character at BASIC Text pointer B is not a comma ',', then issue 'Missing ,' error.
Otherwise, increment BASIC Text pointer B offset (to point to the character after the comma).

Push the SWA String value (the String to search within) to the BASIC Stack.
Call &9D3B to obtain the result of the expression after the comma. This value should be the String value to search for. The INSTR function will search for this string occuring within the first String value specified (the value that is now on the stack).
If the value of the expression is not a String then issue a 'Type mismatch' error, as a search string value was not found.

Set the IWA LSB byte value (&2A) to 1. This is the default search starting position within the String being searched.
Increment the BASIC Text pointer B offset (&1B) to skip the next non-space character, which should be either a comma (if a search starting position is being specified) or a close bracket (to indicate the end of the INSTR statement).

If the next non-space character (the character that was skipped over) was not a ')' (close bracket) then we are not at the end of the INSTR statement yet, so:

* Check that the skipped character was a comma (if not then issue 'Missing ,' error)
* Push the SWA String value (the string to search for) to the BASIC stack
* Get the Integer result of the expression (after the comma) and place the Integer value in the IWA
* Check that the next non-space character is a closing ')' (to terminate the INSTR statement), if not then issue a 'Missing)' error
* Retrieve the second String value (the string to search for) from the BASIC stack.

Set X to the IWA's LSB value (&2A). This is the starting search position within the 'String to search within' value.
If X is zero then set X to 1 (as 0 is an invalid starting position within a String value).
Store X back to the IWA (&2A) location.

Set A to X and then decrement X (storing the new X value in location &2D).
Add the Stack pointer value (in locations &04-&05) to A (the String Starting position before X was decremented, i.e. the Starting position including the String length byte).
Store the result in &37-&38. Now locations &37-&38 point to the required Starting character of the first String value (which is currently located on the BASIC stack).

Set A to the length of the String on the stack (the string to search within). Note: the length of the String on the BASIC Stack is the byte currently pointed to by the stack pointer (&04-&05), i.e. the last byte pushed to the stack.

Subtract the value at location &2D from A (the length of the String on the stack). This will deduct the search starting position (minus 1) that was specified from the length of the String to search within (the string on the BASIC stack).

If the result of the subtraction underflowed then the Search Starting position is greater than the String to search within, so: jump to &AC9F to clear stack space used by the String value & exit with IWA = 0, as the search string (SWA) was not found.

Subtract the length of the SWA (the String to search for) from the previous subtraction result. If the subtraction underflows then the String to search for is larger than the String value to search within, so [&AC9F] restore the BASIC Stack space used by the String and exit with the IWA = 0, as the search value was not found.

Add 1 to A (to obtain the number of searches required).
Store A in location &2B. This is the number of searches that we need to perform before we know whether the search String value was found or not. This value (as calculated above) can be simply calculated as:

Number of Searches (location &2B) = length of String to search within - (starting position -1) - length of search string + 1

Restore the stack space used by the string on the stack (but leave the value of the string where it is, as &37-&38 point to the String's value - which will be accessable as long as nothing else is pushed to the BASIC stack (causing the value to be overwritten)).

[&AC89] Set Y to 0 and X to the length of the Search String (the string value in the SWA).
If X is 0 then the length of the search string is zero, so jump to &AC9A to exit with the IWA = ?&2A (i.e. the search starting position value - as a null String value (of zero bytes long) is always found within another String value (so the INSTRing location for the null string is the starting position of the search.

[&AC8F] Compare the next byte (at offset Y) of the String pointed to by &37-&38 (the first string value, i.e. the String to search within) with the next byte of the SWA (at offset Y).

If the characters match then increment Y (to point to the next character), decrement X (length of search value) and, if X has reached zero (no more characters of the search value to check) we have found a match, so exit with the IWA = ?&2A (i.e. the position of the match within the String value pointed to by &37-&38). Otherwise, jump to &AC8F to check the next character of the search String value against the next character of the &37-&38 String value.

If the characters do not match then try the next character of the String pointed to by &37-&38 as follows:

* [&ACA6] Increment &2A (the current position within the String pointed to by &37-&38)
* Decrement &2B (the number of characters of the &37-&38 String value at which to begin a search)
* If &2B has reached zero, then we have checked for the search String value at all required positions in the &37-&38 String value without finding a match, so exit with IWA = 0 (as end of String value was reached without finding a match).
* Increment the &37-&38 pointer to point to the next character of the String value
* Jump to &AC89 to check for the search SWA value at the new &37-&38 position within the String value

Disassembly for the INSTR routine

AC36	;	032 059 157	20 3B 9D	JSR &9D3B Evaluate expression at BASIC Text pointer B
AC39		208 145	D0 91	BNE -111 --> &ABCC [JMP &9092 - Type mismatch error]
AC3B	,	224 044	E0 2C	CPX#&2C
AC3D		208 024	D0 18	BNE 24 --> &AC57
AC3F		230 027	E6 1B	INC &1B
AC41	Q	032 081 188	20 51 BC	JSR &BC51 Push SWA to Stack
AC44	;	032 059 157	20 3B 9D	JSR &9D3B Evaluate expression at BASIC Text pointer B
AC47		208 131	D0 83	BNE -125 --> &ABCC [JMP &9092 - Type mismatch error]
AC49		169 001	A9 01	LDA#&01
AC4B	*	133 042	85 2A	STA &2A
AC4D		230 027	E6 1B	INC &1B
AC4F	)	224 041	E0 29	CPX#&29
AC51		240 013	F0 0D	BEQ 13 --> &AC60
AC53	,	224 044	E0 2C	CPX#&2C
AC55		240 003	F0 03	BEQ 3 --> &AC5A
AC57	L	076 246 142	4C F6 8E	JMP &8EF6 'Missing ,' error

AC5A		032 164 150	20 A4 96	JSR &96A4 Push SWA to stack, get Integer result of expression & check for closing ')'
AC5D		032 210 188	20 D2 BC	JSR &BCD2 Pop String (SWA) from the stack
AC60	*	166 042	A6 2A	LDX &2A
AC62		208 002	D0 02	BNE 2 --> &AC66
AC64		162 001	A2 01	LDX#&01
AC66	*	134 042	86 2A	STX &2A
AC68		138	8A	TXA
AC69		202	CA	DEX
AC6A	-	134 045	86 2D	STX &2D
AC6C		024	18	CLC
AC6D	e	101 004	65 04	ADC &04
AC6F	7	133 055	85 37	STA &37
AC71		169 000	A9 00	LDA#&00
AC73	e	101 005	65 05	ADC &05
AC75	8	133 056	85 38	STA &38
AC77		178 004	B2 04	LDA (&04)
AC79	8	056	38	SEC
AC7A	-	229 045	E5 2D	SBC &2D
AC7C	!	144 033	90 21	BCC 33 --> &AC9F
AC7E	6	229 054	E5 36	SBC &36
AC80		144 029	90 1D	BCC 29 --> &AC9F
AC82	i	105 000	69 00	ADC#&00
AC84	+	133 043	85 2B	STA &2B
AC86		032 225 188	20 E1 BC	JSR &BCE1 Restore Stack Space used by String (without overwriting the SWA)
AC89		160 000	A0 00	LDY#&00
AC8B	6	166 054	A6 36	LDX &36
AC8D		240 011	F0 0B	BEQ 11 --> &AC9A
AC8F	7	177 055	B1 37	LDA (&37),Y
AC91		217 000 006	D9 00 06	CMP &0600,Y
AC94		208 016	D0 10	BNE 16 --> &ACA6
AC96		200	C8	INY
AC97		202	CA	DEX
AC98		208 245	D0 F5	BNE -11 --> &AC8F
AC9A	*	165 042	A5 2A	LDA &2A
AC9C	L	076 024 174	4C 18 AE	JMP &AE18 Set IWA to the 8-bit value in A
AC9F		032 225 188	20 E1 BC	JSR &BCE1 Restore Stack Space used by String (without overwriting the SWA)
ACA2		169 000	A9 00	LDA#&00
ACA4		128 246	80 F6	BRA -10 --> &AC9C
ACA6	*	230 042	E6 2A	INC &2A
ACA8	+	198 043	C6 2B	DEC &2B
ACAA		240 246	F0 F6	BEQ -10 --> &ACA2
ACAC	7	230 055	E6 37	INC &37
ACAE		208 217	D0 D9	BNE -39 --> &AC89
ACB0	8	230 056	E6 38	INC &38
ACB2		128 213	80 D5	BRA -43 --> &AC89

BASIC IV ROM Routines

Page Last Altered: undefined

AB8A INT

Submitted by Steve Fewell

Description:

Call routine &AD36 to get the Value at the BASIC Text pointer B location.
If the result is a String value then issue a Type mismatch error.
If the result is an Integer value (A = #&40) then exit as the value is already an Integer.

Push the FWA sign byte (6502 processor flags when &2E byte loaded) to the 6502 stack.
Call routine &8275 to move the fractional part of the FWA value out of the FWA and into the FWB mantissa and to convert the FWA value to a two's complement format number (if the FWA sign bit is negative).

Retrieve the FWA sign byte status from the 6502 stack.
If the FWA was positive then copy the FWA mantissa to the IWA and exit with A = #&40 (as the result is an Integer); now the IWA contaions the Integer FWA value.

If the FWA was negative then convert the value to Integer as follows:
* If the FWB Mantissa (bytes &3D to &40) is zero then there is no fractional part, so copy the FWA mantissa to the IWA and exit with A = #&40 (as the result is an Integer).
* Call &82C8 to perform a reverse order complement of the FWA value (to make the value a positive Integer value).
* Call &830D to add 1 to the FWA Mantissa value.
* Call &82C8 to perform a reverse order complement of the FWA value to reverse the previous complement resulting in the two's complement value.
* Copy the FWA Mantissa to the IWA and exit with A = #&40 (as the result is an Integer).

Disassembly for the INT routine

AB8A	6	032 054 173	20 36 AD	JSR &AD36 Evaluate value
AB8D	=	240 061	F0 3D	BEQ 61 --> &ABCC [JMP &9092 'Type mismatch' error]
AB8F	!	016 033	10 21	BPL 33 --> &ABB2 [RTS]
AB91	.	165 046	A5 2E	LDA &2E
AB93		008	08	PHP
AB94	u	032 117 130	20 75 82	JSR &8275 Move fractional part of FWA to FWB
AB97	(	040	28	PLP
AB98		016 019	10 13	BPL 19 --> &ABAD

AB9A	=	165 061	A5 3D	LDA &3D
AB9C	>	005 062	05 3E	ORA &3E
AB9E	?	005 063	05 3F	ORA &3F
ABA0	@	005 064	05 40	ORA &40
ABA2		240 009	F0 09	BEQ 9 --> &ABAD
ABA4		032 200 130	20 C8 82	JSR &82C8 Reverse order complement of FWA
ABA7		032 013 131	20 0D 83	JSR &830D Add 1 to the FWA Mantissa value
ABAA		032 200 130	20 C8 82	JSR &82C8 Reverse order complement of FWA
ABAD		032 198 150	20 C6 96	JSR &96C6 Copy the FWA Mantissa to the IWA
ABB0	@	169 064	A9 40	LDA#&40
ABB2	`	096	60	RTS

Disassembly for the Add 1.0 to the FWA Mantissa value routine

830D	4	230 052	E6 34	INC &34
830F		208 012	D0 0C	BNE 12 --> &831D
8311	3	230 051	E6 33	INC &33
8313		208 008	D0 08	BNE 8 --> &831D
8315	2	230 050	E6 32	INC &32
8317		208 004	D0 04	BNE 4 --> &831D
8319	1	230 049	E6 31	INC &31
831B		240 160	F0 A0	BEQ -96 --> &82BD
831D	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

AE11 LEN

Submitted by Steve Fewell

Description:

Call the Evaluate value routine (&AD36) which will evaluate the next value at BASIC text pointer B location, or the expression at that location if the value begins with an open bracket ("("). If the value obtained is not a String (A isn't 0) then issue a Type mismatch error, as the LENgth can only be obtained for a string value.

Set A to the value in &36 (The SWA length).
Continue to routine &AE18 to set the IWA to the 8-bit value contained in A and exit.

Disassembly for the LEN routine

AE0E	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
AE11	6	032 054 173	20 36 AD	JSR &AD36 Evaluate value
AE14		208 248	D0 F8	BNE -8 --> &AE0E
AE16	6	165 054	A5 36	LDA &36
AE18				...IWA = 8-bit value

BASIC IV ROM Routines

Page Last Altered: undefined

AA93 NOT

Submitted by Steve Fewell

Description:

Call the Evaluate value routine (&96B4) which will evaluate the next value at BASIC text pointer B location, or the expression at that location if the value begins with an open bracket ("(") and convert the result to an Integer value (or generate a Type mismatch error if the value was a String).

Starting at the Most significant byte of the IWA (&2D), EOR (exclusive-OR) each byte of the IWA with &FF (&2D through to &2A). This will reverse the value in the IWA.
Set A to #&40 (meaning that the current value is an Integer) and exit.

Disassembly for the NOT routine

AA93		032 180 150	20 B4 96	JSR &96B4 Get value and convert it to Integer
AA96		162 003	A2 03	LDX#&03
AA98	*	181 042	B5 2A	LDA &2A,X
AA9A	I	073 255	49 FF	EOR#&FF
AA9C	*	149 042	95 2A	STA &2A,X
AA9E		202	CA	DEX
AA9F		016 247	10 F7	BPL -9 --> &AA98
AAA1		128 237	80 ED	BRA -19 --> &AA90 Exit with A=&40 [LDA#&40 : RTS]

BASIC IV ROM Routines

Page Last Altered: undefined

AAE3 OPENOUT

Submitted by Steve Fewell

Description:

Set A to #&80. This is the OSFIND (&FFCE) option to open a file for output access only.
Call routine &AAE9 (part of OPENUP) to obtain the filename and place it in the SWA (or issue 'Type mismatch' error if a String value was not found), Append a '<cr>' character to the end of the SWA (so that the filename is correctly terminated for the OSFIND routine's requirements), Set X and Y to point to the SWA (&0600), call OSFIND (&FFCE) to open the file, and return the opened file's channel number in the IWA.

Disassembly for the OPENOUT routine

AAE3	169 128 A9 80	LDA#&80
AAE5	128 002 80 02	BRA 2 --> &AAE9 Get filename, call OSFIND to open the filename specified in the SWA & set IWA to channel#

BASIC IV ROM Routines

Page Last Altered: undefined

AAFF PI

Submitted by Steve Fewell

Description:

Call the &A8BE routine to load the FWA with the Floating-Point constant at location &BF2E in the Floating-Point constant table.
The constant value at &BF2E is 1.57079633. This value is Pi / 2.
Add 1 to the exponent to multiply this value by 2, this will give Pi as the result.
Exit with the FWA set to Pi (3.14159266).

Disassembly for the PI routine

AAFF		032 190 168	20 BE A8	JSR &A8BE Load FWA with &BF2E value (PI/2)
AB02	0	230 048	E6 30	INC &30
AB04	`	096	60	RTS

Disassembly for A8BE: the Load FWA with &BF2E value (PI/2) routine

A8BE	.	169 046	A9 2E	LDA#&2E
A8C0	L	076 150 168	4C 96 A8	JMP &A896 Set FWA to the FP constant at &BF00 + A

BASIC IV ROM Routines

Page Last Altered: undefined

AC0E POINT(

Submitted by Steve Fewell

Description:

Call routine &96AF to get the Integer value of the expression after the POINT(keyword (issue Type mismatch error if the result is a String value)
Push the Integer to the BASIC Stack. This is the first parameter to the POINT routine - i.e. the X screen position.

Read the comma ',' character, and issue 'Missing ,' error if the next non-space character wasn't a comma.
Call routine &96A7 to obtain the Integer value of the expression after the comma and check for a closing parenthesis ')'.
Issue a 'Syntax error' if the closing bracket wasn't found.
Push this Integer's LSB (&2A) to the 6502 Stack, and the 16-bit Integer's MSB (&2B) to the X index register.
These 2 bytes are the second parameter to the POINT routine - i.e. the Y screen position.

Retrieve the first Integer from the BASIC stack.
Store the Second Integer MSB (from X) to location &2D and store the second Integer LSB (retrieved from 6502 Stack) to location &2C.

Set Y to 0 and X to #&2A -> as &002A is the start of the 5-byte OSWORD parameter block, which contains the following:
* Byte &2A -> The X screen position LSB
* Byte &2B -> The X screen position MSB
* Byte &2C -> The Y screen position LSB
* Byte &2D -> The Y screen position MSB
* Byte &2E -> Reserved for return value

Call the OSWORD &09 command to execute the POINT instruction.
If the return value (byte &2E) is negative then exit with the IWA = -1 (TRUE); otherwise, exit with the IWA = byte &2E (which should be 0, so the IWA will be set to FALSE).

Disassembly for the POINT(routine

AC0E		032 175 150	20 AF 96	JSR &96AF Get Integer result of expression
AC11	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
AC14		032 241 142	20 F1 8E	JSR &8EF1 Check for ',' and issue error if not found
AC17		032 167 150	20 A7 96	JSR &96A7 Extract Integer result of expression & check for ')'

AC1A	*	165 042	A5 2A	LDA &2A
AC1C	H	072	48	PHA
AC1D	+	166 043	A6 2B	LDX &2B
AC1F		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from Stack [popi]
AC22	-	134 045	86 2D	STX &2D
AC24	h	104	68	PLA
AC25	,	133 044	85 2C	STA &2C
AC27		160 000	A0 00	LDY#&00
AC29	*	162 042	A2 2A	LDX#&2A
AC2B		169 009	A9 09	LDA#&09
AC2D		032 241 255	20 F1 FF	JSR &FFF1 OSWORD
AC30	.	165 046	A5 2E	LDA &2E
AC32	0	048 167	30 A7	BMI -89 --> &ABDB Set IWA to TRUE (-1)
AC34		128 214	80 D6	BRA -42 --> &AC0C [Jump to &AE18 Set IWA to 8-bit value]

BASIC IV ROM Routines

Page Last Altered: undefined

ABF5 SGN

Submitted by Steve Fewell

Description:

Call routine &AD36 to obtain the value of the expression after the SGN keyword.
If the result is a String value then issue 'Type mismatch' error.

If the result is a Floating-Point value then obtain the sign (SGN) as follows:

* Call &A3F2 to obtain the sign of the FWA value.
* If &A3F2 returns 0 then exit with the IWA set to 0 (FALSE) and A = #&40 (result is Integer).
* If &A3F2 returns a positive value (1) then exit with the IWA set to 1 and A = #&40 (result is Integer).
* If &A3F2 returns a negative value (-1) then exit with the IWA set to -1 (TRUE) and A = #&40 (result is Integer).

Otherwise the result is an Integer so obtain the sign (SGN) as follows:

* If IWA (bytes &2D-&2A) is zero then exit with the IWA unchanged and A = #&40 (result is Integer).
* If byte &2D (the IWA MSB) is negative (>=#&80) then exit with the IWA set to -1 (TRUE) and A = #&40 (result is Integer).
* Otherwise, the IWA is positive, so exit with the IWA set to 1 and A = #&40 (result is Integer).

Disassembly for the SGN routine

ABEC		032 242 163	20 F2 A3	JSR &A3F2 Obtain Sign of the FWA Floating-Point value
ABEF		240 247	F0 F7	BEQ -9 --> &ABE8 Set IWA to FALSE (0)
ABF1		016 023	10 17	BPL 23 --> &AC0A
ABF3		128 230	80 E6	BRA -26 --> &ABDB Set IWA to TRUE (-1)
ABF5	6	032 054 173	20 36 AD	JSR &AD36 Get value / result of expression (if bracketed)
ABF8		240 210	F0 D2	BEQ -46 --> &ABCC [JMP &9092 - Type mismatch error]
ABFA	0	048 240	30 F0	BMI -16 --> &ABEC
ABFC	-	165 045	A5 2D	LDA &2D
ABFE	,	005 044	05 2C	ORA &2C
AC00	+	005 043	05 2B	ORA &2B
AC02	*	005 042	05 2A	ORA &2A

AC04		240 223	F0 DF	BEQ -33 --> &ABE5 [LDA #&40 : RTS]
AC06	-	165 045	A5 2D	LDA &2D
AC08	0	048 209	30 D1	BMI -47 --> &ABDB Set IWA to TRUE (-1)
AC0A		169 001	A9 01	LDA#&01
AC0C		128 177	80 B1	BRA -79 --> &ABBF [JMP &AE18 Set IWA to the 8-bit value (in A)]

BASIC IV ROM Routines

Page Last Altered: undefined

AAA3 POS

Submitted by Steve Fewell

Description:

Call VPOS routine to do the following:

* Call OSBYTE #&86 to obtain the Text cursor position. After this call, X contains the horizontal position of the cursor (POS) and Y contains the vertical position of the cursor (VPOS).
* Set the IWA to the 1-byte VPOS value (in Y) [bytes &2B-&2D of the IWA will be zero].

Store X (The POS value) in location &2A (now the IWA contains the POS value -> locations &2B-&2D have already been zeroed by the VPOS routine), and exit.

Disassembly for the POS routine

AAA3		032 188 170	20 BC AA	JSR &AABC VPOS
AAA6	*	134 042	86 2A	STX &2A
AAA8	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

A9C8 RAD

Submitted by Steve Fewell

Description:

Call the &96DA routine to get the Float value.
Jump to &A9D4 with A = &38 to multiply the FWA by the Floating-Point
constant at location &BF38 in the Floating-Point constant table.
The constant value at &BF38 is 0.0174532925. This value is Pi / 180, or Degree. Multiplying
the Float value by this constant gives the value in Radians.

Disassembly for the RAD routine

A9C8	032 218 150	20 DA 96	JSR &96DA Get and Check Float (conv if Int)
A9CB	8 169 056	A9 38	LDA#&38
A9CD	128 005	80 05	BRA 5 --> &A9D4 Multiply FWA by &BF38 (0.0174532925)

BASIC IV ROM Routines

Page Last Altered: undefined

AA73 RND

Submitted by Steve Fewell

Description:

If the next character after the 'RND-token' at the BASIC Text Pointer B location is not '(' then set the IWA to the Next Random Number Seed value (from locations &0D-&10), as follows:

* Call &831E to calculate the next Random Number Seed value.
* Set X to #&0D (the start of the Random Number seed value).
* Continue to routine &AA80 to set the IWA to the Integer value starting at the location specified in X (that is locations &0D, &0E, &0F, &10) and exit.

[&AA3B] Increment BASIC Text Pointer B offset value (to point to the next character after the '('-character).
Call routine &96A7 to obtain the Integer value at the BASIC Text Pointer B location (putting the result in the IWA), and then check that the next non-space character is a close bracket ')', if it isn't then issue a 'Syntax error'.

If the Integer value is negative (top bit of IWA MSB (&2D) is set) then set the Random number seed value as follows:

* Set X to #&0D
* Call routine &BDC6 to set locations X => X+3 to the IWA value (that is locations &0D-&10).
* Set location &11 (the fifth byte of the Random Number Seed value) to #&40 and exit.

Otherwise, if the IWA value is less than 256 (i.e. bytes &2D, &2C and &2B are zero) then check the IWA LSB byte value (&2A). This determines whether the IWA value is 0 or 1 (as tested below).

If the IWA value is 0 then Return the previous Random Number Float value (between 0 and 1) as follows:

* [&AA21] Zero the FWA Sign (&2E), Exponent Overflow (&2F) and Mantissa Rounding (&35) bytes
* Set the FWA Exponent (&30) to #&80 - so that the FWA value will be between the range 0.0 to 1.0
* EOR the exponent (#&80) with &0D (to remove any top bit) and store the result as FWA Mantissa byte 4 (&34).
* EOR the FWA Mantissa Byte 4 with &0E and store the result as FWA Mantissa byte 3 (&33).
* EOR the FWA Mantissa Byte 3 with &0F and store the result as FWA Mantissa byte 2 (&32).
* EOR the FWA Mantissa Byte 2 with &10 and store the result as FWA Mantissa byte 1 (&31).
* Call routine &A854 to Normalise the FWA value & round the FWA Mantissa to 4 bytes
* Set A to #&FF (as a Floating-Point result has just been calculated) and exit.

If the IWA value is 1 then Return a Random Number Floating-Point value between 0 and 1, as follows:

* [&AA1E] Call routine &831E to calculate the next Random Number Seed value
* [&AA21] Zero the FWA Sign (&2E), Exponent Overflow (&2F) and Mantissa Rounding (&35) bytes
* Set the FWA Exponent (&30) to #&80 - so that the FWA value will be between the range 0.0 to 1.0

* EOR the exponent (#&80) with &0D (to remove any top bit) and store the result as FWA Mantissa byte 4 (&34).
* EOR the FWA Mantissa Byte 4 with &0E and store the result as FWA Mantissa byte 3 (&33).
* EOR the FWA Mantissa Byte 3 with &0F and store the result as FWA Mantissa byte 2 (&32).
* EOR the FWA Mantissa Byte 2 with &10 and store the result as FWA Mantissa byte 1 (&31).
* Call routine &A854 to Normalise the FWA value & round the FWA Mantissa to 4 bytes
* Set A to #&FF (as a Floating-Point result has just been calculated) and exit.

Otherwise, calculate a Random Integer Number between 1 and the IWA value, as follows:

* [&AA52] Call routine &8185 to convert the Integer value (in the IWA) to a Floating-Point value (in the FWA)
* Call routine &BBFA to push the FWA value to the BASIC Stack
* Call routine &AA1E to set the FWA to a Random Floating-Point value between 0.0 and 1.0 (as described above for RND(1)).
* Retrieve the Float Value from the BASIC Stack (and set argp (&4A, &4B) to point to the packed Floating-Point value)
* Call routine &A6CF to Multiply the Packed Floating-Point value (pointed to by argp) by the FWA value (storing the result in the FWA) [i.e. FWA=argp*FWA]
* Call routine &96C3 to convert the FWA value to an Integer (placing the result in the IWA)
* Now the IWA contains an Integer value between 0 and the original Integer value minus 1. This is because the Random Floating-Point number was between the range 0.0 to 1.0; with low FWA values (e.g. 0.0003) resulting in a zero Integer result
  and high FWA values (e.g. 0.9999) resulting in an Integer result of 1 less than the original Integer value. This is because the FWA value is not rounded up on conversion to Integer (the value is truncated instead, with the fractional part being lost).
* Call routine &BEEF to increment the IWA value. Now the IWA value is in the correct range (i.e. a value between 1 and the original Integer value (that was specified between the parenthesis of the RND command).
* Exit with A = #&40 (to indicate that the current result value is an Integer in the IWA)

Disassembly for the RND routine

AA1E		032 030 131	20 1E 83	JSR &831E Calculate next Random Number Seed value
AA21	d.	100 046	64 2E	STZ &2E
AA23	d/	100 047	64 2F	STZ &2F
AA25	d5	100 053	64 35	STZ &35
AA27		169 128	A9 80	LDA#&80
AA29	0	133 048	85 30	STA &30
AA2B		160 000	A0 00	LDY#&00
AA2D		162 003	A2 03	LDX#&03
AA2F	Y	089 013 000	59 0D 00	EOR &000D,Y
AA32	1	149 049	95 31	STA &31,X
AA34		200	C8	INY
AA35		202	CA	DEX
AA36		016 247	10 F7	BPL -9 --> &AA2F
AA38	LT	076 084 168	4C 54 A8	JMP &A854 Normalise FWA, round FWA Mantissa to 4-bytes & set A=&FF
AA3B		230 027	E6 1B	INC &1B
AA3D		032 167 150	20 A7 96	JSR &96A7 Extract Integer result of expression & check for ')'
AA40	-	165 045	A5 2D	LDA &2D
AA42	0%	048 037	30 25	BMI 37 --> &AA69 Set Random Number Seed value
AA44	,	005 044	05 2C	ORA &2C
AA46	+	005 043	05 2B	ORA &2B

AA48		208 008	D0 08	BNE 8 --> &AA52
AA4A	*	165 042	A5 2A	LDA &2A
AA4C		240 211	F0 D3	BEQ -45 --> &AA21 Get previous Random Floating-Point Number
AA4E		201 001	C9 01	CMP#&01
AA50		240 204	F0 CC	BEQ -52 --> &AA1E Get new Random Floating-Point Number
AA52		032 133 129	20 85 81	JSR &8185 Convert Integer to Float
AA55		032 250 187	20 FA BB	JSR &BBFA Push FWA to Stack
AA58		032 030 170	20 1E AA	JSR &AA1E Get new Random Floating-Point Number
AA5B		032 232 187	20 E8 BB	JSR &BBE8 Pop Float from Stack to argp
AA5E		032 207 166	20 CF A6	JSR &A6CF Floating-Point Multiplication (FWA=argp*FWA)
AA61		032 195 150	20 C3 96	JSR &96C3 Convert Float to Integer
AA64		032 239 190	20 EF BE	JSR &BEEF Increment IWA value
AA67	'	128 039	80 27	BRA 39 --> &AA90 Exit with A=&40
AA69		162 013	A2 0D	LDX#&0D
AA6B		032 198 189	20 C6 BD	JSR &BDC6 Store Integer (IWA) to zero page location
AA6E	@	169 064	A9 40	LDA#&40
AA70		133 017	85 11	STA &11
AA72	`	096	60	RTS
AA73		164 027	A4 1B	LDY &1B
AA75		177 025	B1 19	LDA (&19),Y
AA77	(	201 040	C9 28	CMP#&28
AA79		240 192	F0 C0	BEQ -64 --> &AA3B
AA7B		032 030 131	20 1E 83	JSR &831E Calculate next Random Number Seed value
AA7E		162 013	A2 0D	LDX#&0D
AA80				...Load IWA with Integer from Zero Page Address...

BASIC IV ROM Routines

Page Last Altered: undefined

ADF9 TOP

Submitted by Steve Fewell
This routine is called when the 'TO'-Token is encountered in the program.
However, 'TO' itself is not handled by this routine, instead the 'FOR' routine will check that the
'TO'-Token occurs at the requested position, etc...
The BASIC Keyword 'TOP' does not have it's own token, as it consists of the 'TO' token followed by the letter 'P'.

Firstly, this routine checks whether the character after the 'TO'-Token is a 'P' or not.
If the character is not a 'P' then a 'No such variable error' is generated as the 'TO' BASIC Keyword
is only valid within a 'FOR' statement.

Otherwise, the keyword we have is 'TOP'.
So, increment the BASIC Text Pointer B offset to point to the character after the 'P'.
Set A to &12 (The LSB of the TOP address).
Set Y to &13 (The MSB of the TOP address).
Call routine &AE1A to set the IWA to the value of Y * 256 + A.

Description:

Disassembly for the TOP routine

ADF9		200	C8	INY
ADFA		177 025	B1 19	LDA (&19),Y
ADFC	P	201 080	C9 50	CMP#&50
ADFE		208 140	D0 8C	BNE -116 --> &AD8C No such variable error
AE00		230 027	E6 1B	INC &1B
AE02		165 018	A5 12	LDA &12
AE04		164 019	A4 13	LDY &13
AE06		128 018	80 12	BRA 18 --> &AE1A Set IWA to 16-bit value

BASIC IV ROM Routines

Page Last Altered: undefined

AAA9 USR

Submitted by Steve Fewell

Description:

Call routine &96B4 to obtain the result of the expression after the CALL Keyword and convert the result value to Integer (if it was float, or issue 'Type mismatch' error if the result value was a String). Now the IWA contains the CALL address for the machine code routine that is required to be executed.
br>

Unlike CALL, USR does not provide the facility for any variables to be passed to the machine code routine, so the machine code can now be executed directly by calling routine &9304, which does the following:

* Set A to &040C (the LSB of the C% variable) and shift the bits right 1 place, so that the carry flag is set to the value of the first bit (bit 0) of the &040C value.
* Set A to &0404 (the LSB byte of the A% variable)
* Set X to &0460 (the LSB byte of the X% variable)
* Set Y to &0464 (the LSB byte of the Y% variable)
* jump to the machine code address pointed to by IWA bytes &2A-&2B. The machine code routine will provide the return instruction to return from the &9304 subroutine.

After the routine &9304 has been executed, and the machine code routine has been run, the 6502 register and flag values (containing the 6502 processor state immediately after the machine code routine was executed) are stored in the IWA. The A register is stored in &2A (the LSB byte of the IWA), X is stored in &2B, Y is stored in &2C and the processor flags are stored in &2D (the MSB byte of the IWA).
Next, the decimal flag is cleared (incase is had been set by the machine code routine - as BASIC requires the decimal flag to be clear - otherwise its calculations will be incorrect).
Lastly, as USR is a function, A is set to #&40 (to indicate that the function result is an Integer value - stored in the IWA) and the USR routine exits back to the BASIC expression handler.

Disassembly for the USR routine

AAA9		032 180 150	20 B4 96	JSR &96B4 Get value and convert it to Integer
AAAC		032 004 147	20 04 93	JSR &9304 Execute the machine code routine pointed to by the IWA (&2A-&2B)
AAAF	*	133 042	85 2A	STA &2A
AAB1	+	134 043	86 2B	STX &2B
AAB3	,	132 044	84 2C	STY &2C

```
AAB5      008         08        PHP
AAB6   h  104         68        PLA
AAB7   -  133 045     85 2D     STA &2D
AAB9      216         D8        CLD
AABA      128 212     80 D4     BRA -44 --> &AA90 Exit with A=&40
```

BASIC IV ROM Routines

Page Last Altered: undefined

92BE CALL

Submitted by Steve Fewell

Description:

Call routine &9D2F to get the result of the expression at the BASIC Text pointer A location. This value should be the CALL address.
Call routine &96BC to convert the result to an Integer value (if Float, or issue 'Type mismatch' error if it is a String value).

Push the Integer call address value (IWA) to the BASIC stack.
Store zero in location &0600 (the parameter block used to pass parameters to the machine code routine - &0600 contains the number of variables in the parameter block, which is initalised to 0).
Set Y to 0 (the last used location in the machine code call parameter block).
[&92CC] Store Y on the 6502 stack.
Get the next non-space character from BASIC text pointer B location.

If the next character was a comma ',', a (or another) parameter value is present (and is required to be stored in the machine code call parameter block (&0600). This is done as follows:

* Set Y to &1B (the BASIC Text pointer B offset)
* Call routine &9901 to store Y to &1B (Text pointer B offset), skip any space characters and return the address or the value of the variable denoted by the variable name (located after the comma)
* If routine &9901 returns with the zero flag set then issue 'No such variable' error as a variable name was not found
* Retrieve Y from the 6502 stack, and increment Y (to point to the next free location in the parameter block)
* Store the variable address (from location &2A-&2B) and type (from location &2C) details in the next 3 locations in the machine code call parameter block (starting at offset Y) and increment Y twice
* Increment &0600 (the number of variables in the machine code call parameter block)
* Jump back to &92CC to check for further variable parameters

[&92F1] Otherwise (no comma found), there are no (or no further) variable paremeter details to store in the machine

code call parameter block (located at &0600-&06FF).

Retrieve Y from the 6502 stack (it's value can now be discarded).
Decrement the BASIC Text pointer B offset (&1B) to point to the character after the CALL address (or after the last variable parameter value) and call routine &9B96 to check that the statement is terminated correctly with either a ':', '<cr>' or 'ELSE-token' character ('Syntax error' is issued if the statement is not correctly terminated).

Retrieve the CALL Address value from the BASIC stack (and store the value in the IWA).

Call the &9304 subroutine to execute the machinbe code routine as follows:

* Set A to &040C (the LSB of the C% variable) and shift the bits right 1 place, so that the carry flag is set to the value of the first bit (bit 0) of the &040C value.
* Set A to &0404 (the LSB byte of the A% variable)
* Set X to &0460 (the LSB byte of the X% variable)
* Set Y to &0464 (the LSB byte of the Y% variable)
* jump to the machine code address pointed to by IWA bytes &2A-&2B. The machine code routine will provide the return instruction to return from the &9304 subroutine.

[&92FD] After the routine &9304 has been executed, and the machine code routine has been run, the decimal flag is cleared (as BASIC requires the decimal flag to be clear - otherwise its calculations will be incorrect) and jumps to &9005 to execute the next statement in the BASIC program/command line input.

Disassembly for the CALL routine

92BE	/	032 047 157	20 2F 9D	JSR &9D2F Ptr B = Ptr A & Get result of expression
92C1		032 188 150	20 BC 96	JSR &96BC Check result type (location &27) - if Float then convert to Integer
92C4	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
92C7		156 000 006	9C 00 06	STZ &0600
92CA		160 000	A0 00	LDY#&00
92CC	Z	090	5A	PHY
92CD		032 235 142	20 EB 8E	JSR &8EEB Get next non-space char (PTR B) & compare with ','
92D0		208 031	D0 1F	BNE 31 --> &92F1
92D2		164 027	A4 1B	LDY &1B
92D4		032 001 153	20 01 99	JSR &9901 Store Y to &1B, skip spaces & Evaluate variable/array name & return the address of the value
92D7	(	240 040	F0 28	BEQ 40 --> &9301 'No such variable' error
92D9	z	122	7A	PLY
92DA		200	C8	INY
92DB	*	165 042	A5 2A	LDA &2A
92DD		153 000 006	99 00 06	STA &0600,Y
92E0		200	C8	INY
92E1	+	165 043	A5 2B	LDA &2B

92E3		153 000 006	99 00 06	STA &0600,Y
92E6		200	C8	INY
92E7	,	165 044	A5 2C	LDA &2C
92E9		153 000 006	99 00 06	STA &0600,Y
92EC		238 000 006	EE 00 06	INC &0600
92EF		128 219	80 DB	BRA -37 --> &92CC
92F1	z	122	7A	PLY
92F2		198 027	C6 1B	DEC &1B
92F4		032 150 155	20 96 9B	JSR &9B96 Check for end of Statement (PTR B)
92F7		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from the BASIC Stack [popi]
92FA		032 004 147	20 04 93	JSR &9304 Execute the machine code routine pointed to by the IWA (&2A-&2B)
92FD		216	D8	CLD
92FE	L	076 005 144	4C 05 90	JMP &9005 Continue running program (execute next statement)
9301	L	076 140 173	4C 8C AD	BNE 12 --> &AD8C No such variable error
9304		173 012 004	AD 0C 04	LDA &040C
9307	J	074	4A	LSR A
9308		173 004 004	AD 04 04	LDA &0404
930B	`	174 096 004	AE 60 04	LDX &0460
930E	d	172 100 004	AC 64 04	LDY &0464
9311	l*	108 042 000	6C 2A 00	JMP (&002A)

BASIC IV ROM Routines

Page Last Altered: undefined

AEB3 INKEY$

Submitted by Steve Fewell

Description:

Call routine &AA12 to get the Integer parameter (after the INKEY$ keyword) and call OSBYTE &81 (Inkey).
&AA12 will set X (LSB) and Y (MSB) to the Inkey Time limit (if <= &7FFF) or Key value to check (if >= &8000) specified in the IWA (&2A (LSB) & &2B (MSB)). However, the Key value setting is of no use when using the INKEY$ function.

On return from the OSBYTE #&81 call, the following conditions will apply:
If a negative parameter value is specified then Y & X will both equal #&FF if the key specified is being pressed.
If a positive parameter value is specified then Y will equal 0 if a key has been pressed within the time limit, and X will be the ASCII value of the Key that was pressed.

Set A to X (the ASCII code for the key pressed - if any key was pressed, or #&FF (if a specific key scan was done)).
If the OSBYTE #&81 call returns with Y not set to 0 then no key was pressed (if Time Limit) or the specified key was pressed (if a negative - specific key scan - parameter was supplied), so set the SWA to a null string (set SWA length (&36) to 0) and exit with A = #&00 (as the result is a String value).
Otherwise, call routine &AE6C to Store the ASCII code for the key pressed in location &0600 (the SWA), set the SWA length (&36) to 1, and exit with A = #&00 (the result is a String value).

Disassembly for the INKEY$ routine

AEB3	032 018 170	20 12 AA	JSR &AA12 Get Integer parameter and call OSBYTE &81 (Inkey)
AEB6	138	8A	TXA
AEB7	192 000	C0 00	CPY#&00
AEB9	240 177	F0 B1	BEQ -79 --> &AE6C Set SWA to the 1-byte ASCII character (in A)
AEBB	169 000	A9 00	LDA#&00
AEBD	128 208	80 D0	BRA -48 --> &AE8F Set SWA length (&36) to A and exit with A = 0

BASIC IV ROM Routines

Page Last Altered: undefined

AE73 LEFT$(

Submitted by Steve Fewell

Description:

This routine clears the carry flag (to indicate that the characters should be taken from the left hand side of the String value and calls the RIGHT$ routine to carry out the operation and return the correct subsection of the String.

Disassembly for the LEFT$(routine

AE73	024	18	CLC
AE74			...RIGHT$...

BASIC IV ROM Routines

Page Last Altered: undefined

AEC5 MID$(

Submitted by Steve Fewell

Description:

Call routine &9D3B to evaluate the expression at BASIC text pointer B and issue a 'Type mismatch' error if a String value was not returned.
If the next non-space character (stored in X) is not a comma then issue 'Missing ,' error; otherwise, increment BASIC text pointer B offset to skip past the comma value and push the SWA value to the BASIC stack.

Get the Integer result of the expression at BASIC Text pointer B ('Type mismatch' error if value is not numeric or convert to Integer if Floating-Point value).
Store &2A (the LSB of the Integer value - i.e. the starting point within the SWA value) on the 6502 processor Stack.

Set &2A to value #&FF (i.e. default value for Number of characters is #&FF, or the maximum number of characters (which will return all characters from the Starting point to the end of SWA is reached)).

Increment BASIC text pointer B offset to skip past the next character (which should be either a comma or closing parenthesis).
If the next character (stored in X) was neither a Comma or closing parenthesis, then issue 'Missing ,' error.
If the next character (stored in X) was a Comma (','), then [&96A7] Extract the Integer result (after the comma) and check for a closing parenthesis character (')'). If close bracket was not found then issue 'Syntax error'.

Retrieve the SWA String value from the Stack.
Retrieve the Integer Starting Position from the 6502 Stack and store the value in Y.

If Y is not zero (i.e. not starting at the beginning of the String Value) then:

* Subtract the SWA length (&36) + 1 from the Starting position.
* If the result did not underflow (SWA length is less than the Starting position), so exit with SWA set to a null string (length 0) and exit.
* Decrement Y (The Starting Position value)

Now Y contains the Starting Position - 1, or 0 is we are starting at the first character of the String value. This is needed because the first SWA character starts at 0, whereas the first String character in BASIC starts at 1.
Store the (Starting Position - 1, i.e. the Y value before the above calculation) in X register and location &2C.
Set Y to 0.
Subtract &2C (Starting Position-1) from the SWA length (&36) to obtain the number of characters from the Starting position to the end of the SWA value.
If the result is less than &2A (number of characters to take) then set &2A (number of characters to take) to A (the number of characters in the SWA between the Starting point and the end of the SWA value).

If &2A (Number of characters to take) is zero then set the SWA value to a null string (length 0) and exit.
Otherwise, Keep copying the character at SWA location X (Starting position) to SWA location Y (start of SWA value, initially 0) and then incrementing X and Y until Y is equal to &2A (number of characters to copy).
Set the SWA length (&36) to Y and exit with A = 0 (to indicate that the result is a String value).

Disassembly for the MID$(routine

AEBF	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
AEC2	L	076 246 142	4C F6 8E	JMP &8EF6 'Missing ,' error
AEC5	;	032 059 157	20 3B 9D	JSR &9D3B Evaluate expression at BASIC Text pointer B
AEC8		208 245	D0 F5	BNE -11 --> &AEBF [JMP &9092 Type mismatch error]
AECA	,	224 044	E0 2C	CPX#&2C
AECC		208 244	D0 F4	BNE -12 --> &AEC2 [JMP &8EF6 'Missing ,' error]
AECE	Q	032 081 188	20 51 BC	JSR &BC51 Push SWA to Stack
AED1		230 027	E6 1B	INC &1B
AED3		032 175 150	20 AF 96	JSR &96AF Get expression result & convert it to Integer
AED6	*	165 042	A5 2A	LDA &2A
AED8	H	072	48	PHA
AED9		169 255	A9 FF	LDA#&FF
AEDB	*	133 042	85 2A	STA &2A
AEDD		230 027	E6 1B	INC &1B
AEDF	)	224 041	E0 29	CPX#&29
AEE1		240 007	F0 07	BEQ 7 --> &AEEA
AEE3	,	224 044	E0 2C	CPX#&2C
AEE5		208 219	D0 DB	BNE -37 --> &AEC2 [JMP &8EF6 'Missing ,' error]
AEE7		032 167 150	20 A7 96	JSR &96A7 Extract Integer result of expression & check for ')'
AEEA		032 210 188	20 D2 BC	JSR &BCD2 Pop String (SWA) from the stack
AEED	h	104	68	PLA
AEEE		168	A8	TAY
AEEF		024	18	CLC
AEF0		240 006	F0 06	BEQ 6 --> &AEF8
AEF2	6	229 054	E5 36	SBC &36
AEF4		176 197	B0 C5	BCS -59 --> &AEBB Exit with SWA length (&36) = 0 and A = 0
AEF6		136	88	DEY
AEF7		152	98	TYA
AEF8	,	133 044	85 2C	STA &2C
AEFA		170	AA	TAX
AEFB		160 000	A0 00	LDY#&00
AEFD	6	165 054	A5 36	LDA &36
AEFF	8	056	38	SEC
AF00	,	229 044	E5 2C	SBC &2C
AF02	*	197 042	C5 2A	CMP &2A
AF04		176 002	B0 02	BCS 2 --> &AF08

```
AF06    *  133 042          85 2A       STA &2A
AF08    *  165 042          A5 2A       LDA &2A
AF0A       240 175          F0 AF       BEQ -81 --> &AEBB Exit with SWA length (&36) = 0 and A = 0
AF0C       189 000 006      BD 00 06    LDA &0600,X
AF0F       153 000 006      99 00 06    STA &0600,Y
AF12       200              C8          INY
AF13       232              E8          INX
AF14    *  196 042          C4 2A       CPY &2A
AF16       208 244          D0 F4       BNE -12 --> &AF0C
AF18    6  132 054          84 36       STY &36
AF1A    ^  128 094          80 5E       BRA 94 --> &AF7A   Exit with A = 0
```

BASIC IV ROM Routines

Page Last Altered: undefined

AF1C STR$

Submitted by Steve Fewell

Description:

Set Y to #&FE (meaning that Hexadecimal output of the value is required).
Get the next non-space character after the STR$-token.
If the next charaxcter is not a '~' (tilde), then set Y to 0 (meaning that decimal output of the numeric value is required), and decrement &1B (BASIC Text Pointer B offset) so that the non-space character we obtained will be the first one read by the Get value routine (&AD36).
Store Y to the stack, while we get the value.
Call the Get Value routine (&AD36) to return the value pointed to by the BASIC Test Pointer B.
A single value will be returned. If the value starts with an open bracket ("("), then the expression within the brackets will be evaluated.
If the type of the returned value (A) is 0 (String), then the value is not numeric, so generate a Type mismatch error.

Otherwise, retrieve the byte from the stack and store the value in location &15. The ASCNUM routine will use this value to determine whether the numeric value will be converted to a Hexadecimal String or a Decimal String.
Set Y to the type of the variable returned from the Get value routine (A).

If the 4th byte of @% (&0403) is not zero then the @%-number format settings will be used in the conversion of the numeric value, so call NUMASC routine (at location &A118) to convert the numeric value to an ASCII string (located in the SWA), Set A to 0 (meaning that the Current value is a String) and exit.

If the 4th byte of @% (&0403) is zero then the @%-number format settings will not be used in the conversion of the numeric value, so set &37 to 0 (Output format type = General) and call NUMASC routine (at location &A132) convert the numeric value to an ASCII string (located in the SWA), Set A to 0 (meaning that the Current value is a String) and exit.

Disassembly for the STR$ routine

AF1C		032 213 142	20 D5 8E	JSR &8ED5 Get next non-space character (PTR B)
AF1F		160 255	A0 FF	LDY#&FF
AF21	~	201 126	C9 7E	CMP#&7E '~'
AF23		240 004	F0 04	BEQ 4 --> &AF29

```
AF25      160 000       A0 00       LDY#&00
AF27      198 027       C6 1B       DEC &1B
AF29   Z  090           5A          PHY
AF2A   6  032 054 173   20 36 AD    JSR &AD36 Get value / result of expression (if bracketed)
AF2D      240 021       F0 15       BEQ 21 --> &AF44 Type Mismatch error
AF2F      168           A8          TAY
AF30   h  104           68          PLA
AF31      133 021       85 15       STA &15
AF33      173 003 004   AD 03 04    LDA &0403
AF36      208 007       D0 07       BNE 7 --> &AF3F
AF38   7  133 055       85 37       STA &37
AF3A   2  032 050 161   20 32 A1    JSR &A132 ASCNUM (Convert number to ASCII) [Skip @%-format settings]
AF3D   ;  128 059       80 3B       BRA 59 --> &AF7A [LDA#&00 : RTS]
AF3F      032 024 161   20 18 A1    JSR &A118 ASCNUM (Convert number to ASCII) [With @%-format settings]
AF42   6  128 054       80 36       BRA 54 --> &AF7A [LDA#&00 : RTS]
AF44   L  076 146 144   4C 92 90    JMP &9092 Type mismatch error
```

BASIC IV ROM Routines

Page Last Altered: undefined

AF47 STRING$

Submitted by Steve Fewell

Description:

Call routine &96AF to obtain the result of the expression after the STRING$(keyword and convert the result to Integer (or issue 'Type mismatch' error if the result was a String value).
Push the Integer result to the BASIC stack. This is the number of times that the String value (the second parameter) is required to be repeated.

Call routine &8EF1 to check that the next non-space character is a comma ',' (issuing 'Missing ,' error if it isn't).
Call routine &ADAC to get the result of the expression after the comma and then check for a closing bracket character ')', and issue 'Missing)' error if a closing bracket was not the next non-space character.
If the result of the expression is not a String value then issue 'Type mismatch' error.

Retrieve the Integer value from the BASIC Stack.
Now the IWA contains the number of times to copy the SWA String value and the SWA contains the String value to be duplicated.

If the length of the String value to duplicate (SWA) is zero then exit with A = 0 (as the SWA already contains the result - a null string!
Set Y to the String length (from location &36).
Set A to the IWA's LSB value (location &2A), as only the first byte is taken into account when obtaining the number of copies required.

If A (number of copies) is zero then set the SWA length (location &36) to zero and exit with A = 0, as the result of 'zero copies' is a null String value.
Decrement the IWA value (&2A). If the result is now zero (i.e. the IWA value was 1) then exit with A = 0, as the SWA contains the correct result - a single copy of the String value.

[&AF64] Set X to zero.
Now X points to the start of the SWA String Value (offset from &0600) and Y points to the start of the location within the SWA to copy the duplication of the String value to (i.e. the first free character after the current SWA value, offset from &0600).
[&AF66] Copy the next character from the X offset to the Y offset of the SWA
Increment X to point to the next character of the original SWA's value.
Increment Y to point to the next free character in the 'new' SWA value.
If Y has reached zero then we have reached the end of the SWA storage space (more than 255 characters), so issue the 'String too long' error message.

If X is less than the SWA Length byte (location &36) then jump back to &AF66 to copy the next character of the original String's value.
Otherwise, we have duplicated another copy of the String value and added it to the end of the original SWA value, so decrement the IWA value (location &2A), if &2A is now zero then set the SWA length (location &36) to Y and exit with A = 0; otherwise, jump back to &AF64 to copy the original String value again.

Disassembly for the STRING$ routine

AF47		032 175 150	20 AF 96	JSR &96AF Get expression result & convert it to Integer
AF4A	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
AF4D		032 241 142	20 F1 8E	JSR &8EF1 Check for ',' and issue error if not found
AF50		032 172 173	20 AC AD	JSR &ADAC Evaluate expression & check for ')'
AF53		208 239	D0 EF	BNE -17 --> &AF44 'Type mismatch' error
AF55		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from the BASIC Stack [popi]
AF58	6	164 054	A4 36	LDY &36
AF5A		240 030	F0 1E	BEQ 30 --> &AF7A
AF5C	*	165 042	A5 2A	LDA &2A
AF5E		240 029	F0 1D	BEQ 29 --> &AF7D
AF60	*	198 042	C6 2A	DEC &2A
AF62		240 022	F0 16	BEQ 22 --> &AF7A
AF64		162 000	A2 00	LDX#&00
AF66		189 000 006	BD 00 06	LDA &0600,X
AF69		153 000 006	99 00 06	STA &0600,Y
AF6C		232	E8	INX
AF6D		200	C8	INY
AF6E		240 016	F0 10	BEQ 16 --> &AF80
AF70	6	228 054	E4 36	CPX &36
AF72		144 242	90 F2	BCC -14 --> &AF66
AF74	*	198 042	C6 2A	DEC &2A
AF76		208 236	D0 EC	BNE -20 --> &AF64
AF78	6	132 054	84 36	STY &36
AF7A		169 000	A9 00	LDA#&00
AF7C	`	096	60	RTS
AF7D	6	133 054	85 36	STA &36
AF7F	`	096	60	RTS
AF80	L	076 016 158	4C 10 9E	JMP &9E10 String Too Long error

BASIC IV ROM Routines

Page Last Altered: undefined

ABCF EOF

Submitted by Steve Fewell

Description:

Call routine &BA4A to check that the next character at the BASIC Text Pointer B location (&0B-&0C) is a '#'-character. If not then issue 'Missing #' error message. Evaluate the expression after the '#' [Type mismatch error if string value found], and set Y to the number found after the '#' character (this is the file channel number).

Set X to the file channel number (in A).
Call the Operating System OSBYTE #&7F function to check for the end-of-file for the file currently open on channel number X.
If X is returned as zero then the end-of-file has not been reached, so exit with IWA = 0; otherwise, exit with IWA = TRUE (&FFFFFFFF), as the end-of-file has been reached.

Disassembly for the EOF routine

ABCF	J 032 074 186	20 4A BA	JSR &BA4A Check for '#' (PTR B), Set Y to file channel number (PTR B)
ABD2	170	AA	TAX
ABD3	169 127	A9 7F	LDA#&7F
ABD5	032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
ABD8	138	8A	TXA
ABD9	240 002	F0 02	BEQ 2 --> &ABDD Store X (zero) in every byte of the IWA (so that the IWA = 0)
ABDB			...Set IWA to TRUE (-1)...

BASIC IV ROM Routines

Page Last Altered: undefined

9489 AUTO

Submitted by Steve Fewell

Description:

Call routine &934D to read the Starting Line Number and Line increment parameters (if any were specified) as follows:

* Set the IWA to 10 (the default program starting Line Number)
* Call routine &9B1E to detokenise and extract the line number at BASIC text pointer A location and store it in the IWA. If no line number was found at BASIC text pointer A then the IWA will retain the default value of 10.
* Push the IWA value (the starting line number) to the BASIC stack
* Set the IWA to 10 (the default line number increment)
* If the next non-space character is not comma ',', then no line increment parameter has been specified, so call &9BB0 to check that the statement is terminated correctly (with a ':', '<cr>' or 'ELSE-token' terminator character), [if not then a 'Syntax error' will be issued] and exit routine &934D.
* Otherwise, call routine &9B1E to detokenise and extract the Line number value from the BASIC Text pointer A location and store it in the IWA. The IWA now contains the specified Line increment value.
* If byte &2B of the IWA contains a non-zero value, then the increment value is more than 255, so issue error 'Silly', as the line increment cannot be more than 255.
* If byte &2A (the LSB) of the IWA contains a zero value, then the increment value is zero, so issue error 'Silly', as the line increment cannot be zero.
* Call &9BA6 to skip any space characters and check that the statement is terminated correctly (with a ':', '<cr>' or 'ELSE-token' terminator character), [if not then a 'Syntax error' will be issued] and exit routine &934D.

Store the step size (the line increment value) to the 6502 stack.

Retrieve the Starting Line number from the BASIC Stack (and store in the IWA).

[&BC26] Push the IWA Line Number value to the BASIC stack.
Call routine &A085 to display the Line Number (in the IWA) on the screen.

Call routine &BA74 to use OSWORD 0 to read the input line entered by the user.
If Escape is pressed during routine &BA74 then it will be detected and the 'Escape' error will be issued.
Retrieve the Line Number from the BASIC Stack (and store it in the IWA).

Call routine &8DE7 to tokenise the input line.
Set Y (pointer offset) to 0.
Call routine &BB08 to Add the tokenised input line (at the line number position denoted by the IWA value) in the BASIC program (replacing any previous line of the same number).

Call routine &BBAC to initialise page &7 variables and the program non-resident variable pointer tables (as a program line change means that all current variables should be lost - due to the Stack location changing).

Retrieve the Step size from the 6502 stack and push the value back to the 6502 stack.
Add the step size (line increment value) to the IWA value.
If the IWA &2B value is positive (i.e. less than &8000) then jump back to &9492 for entry of the next program line; otherwise, the maximum line number (32767) has been exceeded, so jump to &8F83 to initialise variable space and prompt for the user to enter another command line input.

Disassembly for the AUTO routine

9489	M	032 077 147	20 4D 93	JSR &934D Read Start line and line increment parameters
948C	*	165 042	A5 2A	LDA &2A
948E	H	072	48	PHA
948F		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from the BASIC Stack [popi]
9492	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
9495		032 133 160	20 85 A0	JSR &A085 Print Line Number on screen (width 5)
9498	t	032 116 186	20 74 BA	JSR &BA74 Prompt for & get the User's input line (storing it in buffer &0700-&07FF)
949B		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from the BASIC Stack [popi]
949E		032 231 141	20 E7 8D	JSR &8DE7 Tokenise the input line
94A1		160 000	A0 00	LDY#&00
94A3		032 008 187	20 08 BB	JSR &BB08 Insert tokenised input line & insert it into the program
94A6		032 172 187	20 AC BB	JSR &BBAC Initialise Page 7 & reset Variable pointers, etc...
94A9	h	104	68	PLA
94AA	H	072	48	PHA

94AB		024	18	CLC
94AC	e*	101 042	65 2A	ADC &2A
94AE	*	133 042	85 2A	STA &2A
94B0		144 224	90 E0	BCC -32 --> &9492
94B2	+	230 043	E6 2B	INC &2B
94B4		016 220	10 DC	BPL -36 --> &9492
94B6	L	076 131 143	4C 83 8F	JMP &8F83 Initialise & prompt for next command line

---

BASIC IV ROM Routines

Page Last Altered: undefined

9384 RENUMBER

Submitted by Steve Fewell

Description:

Call routine &934D to read the Starting Line Number and Line increment parameters (if any were specified) as follows:

* Set the IWA to 10 (the default program starting Line Number)
* Call routine &9B1E to detokenise and extract the line number at BASIC text pointer A location and store it in the IWA. If no line number was found at BASIC text pointer A then the IWA will retain the default value of 10.
* Push the IWA value (the starting line number) to the BASIC stack
* Set the IWA to 10 (the default line number increment)
* If the next non-space character is not comma ',', then no line increment parameter has been specified, so call &9BB0 to check that the statement is terminated correctly (with a ':', '<cr>' or 'ELSE-token' terminator character), [if not then a 'Syntax error' will be issued] and exit routine &934D.
* Otherwise, call routine &9B1E to detokenise and extract the Line number value from the BASIC Text pointer A location and store it in the IWA. The IWA now contains the specified Line increment value.
* If byte &2B of the IWA contains a non-zero value, then the increment value is more than 255, so issue error 'Silly', as the line increment cannot be more than 255.
* If byte &2A (the LSB) of the IWA contains a zero value, then the increment value is zero, so issue error 'Silly', as the line increment cannot be zero.
* Call &9BA6 to skip any space characters and check that the statement is terminated correctly (with a ':', '<cr>' or 'ELSE-token' terminator character), [if not then a 'Syntax error' will be issued] and exit routine &934D.

On return from routine &934D, the Integer on the BASIC stack is the Start Line Number and the IWA contains the step size.

Retrieve the Start Line number from the BASIC stack and store it in locations &39-&3A (bytes &3B-&3C are unused as the Integer should only be 2-bytes wide). This is the renumbered line number counter.
Call routine &BDE5 to check that the program can be read correctly ('Bad program' message is displayed if the program

cannot be correctly read).

Call routine &9373 to set locations &3B-&3C to point to TOP (the end of the program) and to set &37-&38 to point to PAGE+1 (the start of the program ignoring the first '<cr>' character) and to set Y to 1.

Renumber - Phase 1: Store all current line numbers in the program in the free storage space after TOP
Use the free space after TOP (pointed to by &3B-&3C) to store a copy of all of of the Line numbers present in the program (and to issue an error if there is not enough space to store this required informaton) as follows.
[&9392] Read the next character (offset 0) from the &37-&38 pointer.
If the character is negative (end of program reached) then jump to &93C4 (all line numbers are stored, so continue to the main RENUMBER routine).

Store the character (the first byte of the Line Number) at the location pointed to by the &3B-&3C pointer, i.e. the free space after the TOP of the program. Read the next character (offset 1) from the &37-&38 pointer and store this character in the next free space pointed to by &3B-&3C (offset 1). This value is the second byte of the Line Number value.
Add 2 to the &3B-&3C pointer (to point to the next free location after the program end).
Compare the &3B-&3C pointer address with HIMEM (stored in locations &06-&07). If &3B-&3C exceeds (or is equal to) HIMEM then issue 'RENUMBER space' error (and terminate the RENUMBER operation).
Otherwise, call routine &947A to update &37-&38 to point to the start of the next program line.
Jump back to &9392 to copy the next program line number.

Renumber - Phase 2: Replace all line numbers in the program with the renumbered line numbers
[&93C4] Call routine &937B to set &37-&38 to point to PAGE + 1 and to set Y to 1.
[&93C7] Load the first character of the Line's Line number. If the character is negative (end of program reached) then jump to &93E7 to start the next phase of the renumbering.
Replace the Line number on the current line with the next renumbered Line Number (from locations &39-&3A (storing &3A first)).

Add the Step Size (&2A) to the renumbered Line Number counter (&39-&3A (&3A is the MSB)).
Correct the &39-&3A Line Number if it has exceeded 32767 (&7FFF) by resetting bit 7 to 0 (removing the &8000 value from the value). So &39-&3A now contains the next renumbered Line Number.
Call routine &947A to update &37-&38 to point to the start of the next program line.
Jump back to &93C7 to replace the next program line number.

Renumber - Phase 3: Replace all line number references in the program with the correct renumbered version
[&93E7] Set BASIC Text pointer A (&0B-&0C) to PAGE (&18, the start of the program code).
[&93ED] Load the first character (the Line Number) of the program line pointed to by &0B-&0C.
If the character is negative then we have reached the end of the program, so jump to &8F83 to initialise the BASIC variable space (as we have just destroyed any previously defined variables by placing the old line number values after TOP) and prompt for the user's next command line input - as the RENUMBER operation is complete.

Set Y (the line offset value) to 4 (so that the Line Number and line length byte values are ignored).
Zero location &2C (the quote toggle flag).

[&93F7] Read the next byte of the program line pointed to by the address in locations &0B-&0C (offset Y).

If &2C is zero (then we are not inside a quote) so:

* If the character is '#&8D' (Line number token) then jump to &941B to renumber the line number (this routine is described below). This token usually follows a GOSUB, GOTO, THEN or ELSE keyword and is followed by the literal line number
* If the character is '#&F4' [REM-token] then jump to &9412 to skip the rest of the current program line

[&9405] Increment Y (to point to the next character on the line)
If the current character (that was read by &93F7) is a quote (") then EOR the character with &2C (to toggle the flag value).
If the character is not '<cr>' (carriage return) then jump back to &93F7 to check the next character of the program line.
[&9412] Otherwise, skip the line end and move on to the next line by Adding the value at offset 3 of the current line (which is the line length value) to the &0B-&0C pointer [and set the &0A offset to 1] and jump back to &93ED to check the next program line for line numbers to renumber.

Renumber the Line Number specified after the line number token ('#&8D')
[&941B] Now a line number token has been encountered within a program line. This routine will now renumber the line number so that it addresses the correct renumbered line.

Call &9B2A to set the IWA to the Line Number (after the Line number token value -'&8D').
Call routine &9373 to set &3B-&3C to TOP and &37-&38 to PAGE+1.
[&9421] Load the character at the &37-&38 location.

If the character is negative then the end of the program has been reached without finding a matching program Line, so jump to &945C to:
* Print the text "Failed at "
* Store A in &2B (the Line number MSB)
* Store the character pointed to by &0B-&0C (the line number LSB) to location &2A
* Call routine &A081 to display the line number (in the IWA) on the screen - this outputs the line number that contained the failed reference
* Call routine &BA92 to output a new line
* Jump to &9446 to Set Y to &0A (the current position of the program line in the phase 3 search) and jump to &93F7 to continue checking the next character of the program line (for further line number references.

Compare the IWA line number with the next Line Number at the &3B-&3C pointed (i.e. the next line number that was stored after the end of the program).

If the line numbers are not equal then the matching renumbered line has not yet been found so:
* [&944A] Clear the carry flag (so that we don't add an extra 1 to the &37-&38 pointer)
* Call routine &947A to update the &37-&38 pointer to point to the next program line (by adding the line length byte (offset 3 from the current &37-&38 start of line pointer) to the &37-&38 pointer and reset Y to 1.
* Add 2 to the &3B-&3C pointer (so that the pointer points to the next unrenumbered line number stored after the end of the program.
* Jump back to &9421 to check the next line of the program for a match

Otherwise (we have found a match), so store the new renumbered line number (from the &37-&38 pointer location)

in the current program position - to replace the reference with the renumbered reference - as follows:

* Store the Line number MSB ((&37,&38)+1) in location &3D
* Store the Line Number LSB (pointed to by (&37, &38)) in X register
* Set Y = the BASIC Text Pointer A offset - 1
* Set &39-&3A to the current &0B-&0C pointer location (i.e. the line containing the reference)
* Call routine &8D62 to tokenise the Program Line Number (in X and &3D) and store it at the ((&39-&3A)+1) location
* Set Y to the BASIC Text pointer A offset (&0A)
* Jump to &93F7 to continue with the next character of the program line (pointed to by &0B-&0C)

Disassembly for the RENUMBER routine

934D		169 010	A9 0A	LDA#&0A
934F		032 024 174	20 18 AE	JSR &AE18 Set IWA to the 8-bit value in A
9352		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
9355	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
9358		169 010	A9 0A	LDA#&0A
935A		032 024 174	20 18 AE	JSR &AE18 Set IWA to the 8-bit value in A
935D		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
9360		208 014	D0 0E	BNE 14 --> &9370
9362		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
9365	+	165 043	A5 2B	LDA &2B
9367	S	208 083	D0 53	BNE 83 --> &93BC 'Silly' error
9369	*	165 042	A5 2A	LDA &2A
936B	O	240 079	F0 4F	BEQ 79 --> &93BC 'Silly' error
936D	L	076 166 155	4C A6 9B	JMP &9BA6 Skip spaces and check end of Statement
9370	L	076 176 155	4C B0 9B	JMP &9BB0 Check for end of Statement
9373		165 018	A5 12	LDA &12
9375	;	133 059	85 3B	STA &3B
9377		165 019	A5 13	LDA &13
9379	<	133 060	85 3C	STA &3C
937B		165 024	A5 18	LDA &18
937D	8	133 056	85 38	STA &38
937F		160 001	A0 01	LDY#&01
9381	7	132 055	84 37	STY &37
9383	`	096	60	RTS
9384	M	032 077 147	20 4D 93	JSR &934D Read Start line and line increment parameters
9387	9	162 057	A2 39	LDX#&39

```
9389          032 008 189     20 08 BD        JSR &BD08 Pop Integer from Stack to Zero page address
938C          032 229 189     20 E5 BD        JSR &BDE5 Check program can be read correctly ('Bad program' message if not)
938F  s       032 115 147     20 73 93        JSR &9373 Set &3B-&3C to TOP and &37-&38 to PAGE+1
9392  7       178 055         B2 37           LDA (&37)
9394  0.      048 046         30 2E           BMI 46 --> &93C4
9396  ;       146 059         92 3B           STA (&3B)
9398  7       177 055         B1 37           LDA (&37),Y
939A  ;       145 059         91 3B           STA (&3B),Y
939C  8       056             38              SEC
939D          152             98              TYA
939E  e;      101 059         65 3B           ADC &3B
93A0  ;       133 059         85 3B           STA &3B
93A2          144 002         90 02           BCC 2 --> &93A6
93A4  <       230 060         E6 3C           INC &3C
93A6          197 006         C5 06           CMP &06
93A8  <       165 060         A5 3C           LDA &3C
93AA          229 007         E5 07           SBC &07
93AC          176 005         B0 05           BCS 5 --> &93B3 'RENUMBER space' error
93AE  z       032 122 148     20 7A 94        JSR &947A Advance &37-&38 to point to the next Program Line
93B1          128 223         80 DF           BRA -33 --> &9392
93B3                                          ... 'Silly' error...
93BC                                          ... 'RENUMBER space' error...
93C4  {       032 123 147     20 7B 93        JSR &937B Set &37-&38 to PAGE+1
93C7  7       178 055         B2 37           LDA (&37)
93C9  0       048 028         30 1C           BMI 28 --> &93E7
93CB  :       165 058         A5 3A           LDA &3A
93CD  7       146 055         92 37           STA (&37)
93CF  9       165 057         A5 39           LDA &39
93D1  7       145 055         91 37           STA (&37),Y
93D3          024             18              CLC
93D4  9       165 057         A5 39           LDA &39
93D6  e*      101 042         65 2A           ADC &2A
93D8  9       133 057         85 39           STA &39
93DA          169 000         A9 00           LDA#&00
93DC  e:      101 058         65 3A           ADC &3A
93DE  )       041 127         29 7F           AND#&7F
93E0  :       133 058         85 3A           STA &3A
```

93E2	z	032 122 148	20 7A 94	JSR &947A Advance &37-&38 to point to the next Program Line
93E5		128 224	80 E0	BRA -32 --> &93C7
93E7		165 024	A5 18	LDA &18
93E9		133 012	85 0C	STA &0C
93EB	d	100 011	64 0B	STZ &0B
93ED		160 001	A0 01	LDY#&01
93EF		177 011	B1 0B	LDA (&0B),Y
93F1	0g	048 103	30 67	BMI 103 --> &945A
93F3		160 004	A0 04	LDY#&04
93F5	d,	100 044	64 2C	STZ &2C
93F7		177 011	B1 0B	LDA (&0B),Y
93F9	,	166 044	A6 2C	LDX &2C
93FB		208 008	D0 08	BNE 8 --> &9405
93FD		201 141	C9 8D	CMP#&8D
93FF		240 026	F0 1A	BEQ 26 --> &941B
9401		201 244	C9 F4	CMP#&F4
9403		240 013	F0 0D	BEQ 13 --> &9412
9405		200	C8	INY
9406	"	201 034	C9 22	CMP#&22
9408		208 004	D0 04	BNE 4 --> &940E
940A	E,	069 044	45 2C	EOR &2C
940C	,	133 044	85 2C	STA &2C
940E		201 013	C9 0D	CMP#&0D
9410		208 229	D0 E5	BNE -27 --> &93F7
9412		160 003	A0 03	LDY#&03
9414		177 011	B1 0B	LDA (&0B),Y
9416		032 244 155	20 F4 9B	JSR &9BF4 Add A to PTR A pointer (&0B,&0C) & set PTR A (&0A) offset to 1
9419		128 210	80 D2	BRA -46 --> &93ED
941B	*	032 042 155	20 2A 9B	JSR &9B2A Detokenise Line Number at PTR A (without skip spaces or skip #&8D char) & Set IWA to the Line Number value
941E	s	032 115 147	20 73 93	JSR &9373 Set &3B-&3C to TOP and &37-&38 to PAGE+1
9421	7	178 055	B2 37	LDA (&37)
9423	07	048 055	30 37	BMI 55 --> &945C
9425	;	178 059	B2 3B	LDA (&3B)
9427	+	197 043	C5 2B	CMP &2B
9429		208 031	D0 1F	BNE 31 --> &944A
942B	;	177 059	B1 3B	LDA (&3B),Y
942D	*	197 042	C5 2A	CMP &2A

942F		208 025	D0 19	BNE 25 --> &944A
9431	7	177 055	B1 37	LDA (&37),Y
9433	=	133 061	85 3D	STA &3D
9435	7	178 055	B2 37	LDA (&37)
9437		170	AA	TAX
9438		164 010	A4 0A	LDY &0A
943A		136	88	DEY
943B		165 011	A5 0B	LDA &0B
943D	9	133 057	85 39	STA &39
943F		165 012	A5 0C	LDA &0C
9441	:	133 058	85 3A	STA &3A
9443	b	032 098 141	20 62 8D	JSR &8D62 Tokenise program line & store at location specified by &39-&3A (which is &0B-&0C)
9446		164 010	A4 0A	LDY &0A
9448		128 173	80 AD	BRA -83 --> &93F7
944A		024	18	CLC
944B	z	032 122 148	20 7A 94	JSR &947A Advance &37-&38 to point to the next Program Line
944E	;	165 059	A5 3B	LDA &3B
9450	i	105 002	69 02	ADC#&02
9452	;	133 059	85 3B	STA &3B
9454		144 203	90 CB	BCC -53 --> &9421
9456	<	230 060	E6 3C	INC &3C
9458		128 199	80 C7	BRA -57 --> &9421
945A	Z	128 090	80 5A	BRA 90 --> &94B6 [JMP &8F83 Initialise & prompt for next command line]
945C		032 207 190	20 CF BE	JSR &BECF Print following text as a warning message & jump to address specified
945F	Failed at	070 097 105 108 101 100 032 097 116 032	46 61 69 6C 65 64 20 61 74 20	EQUS "Failed at "
9469		177	B1	EQUB &B1
946A		011	0B	EQUB &0B
946B	+	133 043	85 2B	STA &2B
946D		200	C8	INY
946E		177 011	B1 0B	LDA (&0B),Y
9470	*	133 042	85 2A	STA &2A
9472		032 129 160	20 81 A0	JSR &A081 Print Line Number on screen (width 0)
9475		032 146 186	20 92 BA	JSR &BA92 Start new output line
9478		128 204	80 CC	BRA -52 --> &9446
947A		200	C8	INY
947B	7	177 055	B1 37	LDA (&37),Y

947D		160 001	A0 01	LDY#&01
947F	e7	101 055	65 37	ADC &37
9481	7	133 055	85 37	STA &37
9483		144 003	90 03	BCC 3 --> &9488
9485	8	230 056	E6 38	INC &38
9487		024	18	CLC
9488	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

9317 DELETE

Submitted by Steve Fewell

Description:

Call routine &9B1E to detokenise the Line Number at BASIC Text pointer A & Set the IWA to the Line Number value.
If a line number was not found at BASIC text pointer A location (carry flag was clear on return from &9B1E) then issue 'Syntax error', as the DELETE statement is not correctly formed.
Push the IWA value (the first line number) to the BASIC stack.

Call routine &8CE5 to check that the next non-space character is a comma ',', if it isn't then issue 'Syntax error'.
Call routine &9B1E to detokenise the Line Number at BASIC Text pointer A & Set the IWA to the Line Number value.
If a line number was not found at BASIC text pointer A location (carry flag was clear on return from &9B1E) then issue 'Syntax error', as the DELETE statement is not correctly formed.
The IWA now contains the program line number to delete up to.

Call routine &9BA6 to check that the statement is correctly terminated by a ':', '<cr>' or 'ELSE-token' character ('Syntax error' is issued if the statement terminator character was not found).

Store the Line number in the IWA (&2A-&2B) [the line number to delete up to] in locations &39-&3A.
Retrieve the Integer value from the BASIC stack - the IWA will now contain the first ('delete from') Line Number.

[&9337] Call routine &BA98 to remove the Line number specified in the IWA from the program - if the line number specified is not present in the program then no action will be taken by routine &BA98.

Call routine &9BCA to check for an 'Escape key press' condition. If the Escape key is detected as being pressed then the 'Escape' error will be issued and the deletion will be prematurely stopped.

Call routine &BEEF to increment the IWA value by 1.

If the Line number value specified by &39-&3A is greater then the value in the IWA then jump back to &9337 to attempt to remove the Line number specified in the IWA from the program.
Otherwise, the deletion has been completed.

To exit, jump to &8F83 to initialise program variable space and prompt for the user to enter a command line.

Note: Calling &BA98 for every possible line number seems a slow method of deleting. I wonder why this method was chosen.

Disassembly for the DELETE routine

9314	Li	076 105 155	4C 69 9B	JMP &9B69 Syntax error
9317		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
931A		144 248	90 F8	BCC -8 --> &9314
931C	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
931F		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
9322		208 240	D0 F0	BNE -16 --> &9314
9324		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
9327		144 235	90 EB	BCC -21 --> &9314
9329		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
932C	*	165 042	A5 2A	LDA &2A
932E	9	133 057	85 39	STA &39
9330	+	165 043	A5 2B	LDA &2B
9332	:	133 058	85 3A	STA &3A
9334		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from the BASIC Stack [popi]
9337		032 152 186	20 98 BA	JSR &BA98 Remove Line Number (specified in IWA) from Program
933A		032 202 155	20 CA 9B	JSR &9BCA Check for 'Escape key pressed' condition (if so, issue 'Escape' error)
933D		032 239 190	20 EF BE	JSR &BEEF Increment IWA value
9340	9	165 057	A5 39	LDA &39
9342	*	197 042	C5 2A	CMP &2A
9344	:	165 058	A5 3A	LDA &3A
9346	+	229 043	E5 2B	SBC &2B
9348		176 237	B0 ED	BCS -19 --> &9337
934A	L	076 131 143	4C 83 8F	JMP &8F83 Initialise & prompt for next command line

---

BASIC IV ROM Routines

Page Last Altered: undefined

8F20 LOAD

Submitted by Steve Fewell

Description:

Call routine &BDD7 to get the filename after the LOAD keyword (and place it in the SWA), and load the named file (which should be a BASIC program) at the PAGE address and check that the program can be read correctly ('Bad program' message is displayed if not).
Jump to &8F83 to initialise and clear the variable storage memory block and prompt for entry of the next Command Line.

Disassembly for the LOAD routine

8F20		032 215 189	20 D7 BD	JSR &BDD7 Get filename and Load the named file
8F23	^	128 094	80 5E	BRA 94 --> &8F83 Initialise & prompt for next command line

---

BASIC IV ROM Routines

Page Last Altered: undefined

BDD7 Load/Save named file

Submitted by Steve Fewell

Description:

Call routine &BE41 to obtain the filename, and place it in the SWA, add <cr> character to the end of the filename, set (&37-&38) to point to the filename and set &39-&3C to the filename load address (PAGE).

Set X (the parameter block starting address) to #&37.
The parameter block (&0037 - &0048) should now contain the following information:
* &0037-&0038 - Address of filename (the filename is terminated by #&0D).
* &0039-&003C - Load address of the file (LSB first).
* &003D-&0040 - Executeable address of the file (LSB first).
* &0041-&0044 - Start address (if we are saving a file) or the length of file (if not saving) (LSB first).
* &0045-&0048 - End address of data save or file attributes (LSB first).

Call OSFILE (&FFDD) to load the specified file.

When loading, locations &003D-&0048 are left blank.

Disassembly for the Load/Save named file routine

BDD7	A	032 065 190	20 41 BE	JSR &BE41 Get Filename and set parameter block filename and load address
BDDA	d=	100 061	64 3D	STZ &3D
BDDC		160 000	A0 00	LDY#&00
BDDE		169 255	A9 FF	LDA#&FF
BDE0	7	162 055	A2 37	LDX#&37
BDE2		032 221 255	20 DD FF	JSR &FFDD OSFILE
BDE5				... Check program can be read correctly...

BASIC IV ROM Routines

Page Last Altered: undefined

BE41 Get Filename and set parameter block Filename and Load address

Submitted by Steve Fewell

Routine:Get Filename
Name: Get Filename and set parameter block Filename and Load address
Starting Address: &BE41
Entry criteria: BASIC Text Pointer A points to a String expression.
Exit: X and Y contain the High Order address (X = MSB). &3B-&3C = the High Order address.

Description:

[BE41] Call &BE36 to do the following:
* Set BASIC Text Pointer B offset to BASIC Text Pointer A offset.
* Get result of expression at BASIC Text Pointer B.
* Issue 'Type mismatch' error if the result is not a String value.
* Call routine &BE25 to Store a carriage return (#&0D) character at the end of the String, and to set &37-&38 to point the the SWA address (&0600).
* Call &9B91 to check that the Statement is terminated correctly.
This obtains the filename specified and appends a <CR> to the end of the name, so that the filename is in the correct MOS format - ready for an operating system call to OSFILE (&FFDD) [which requires a <cr> to terminate the filename].
The OSFILE parameter block filename address field (&37-&38) is set to point to the filename (located in the SWA).

Store the Ptr A Offset value (in Y) minus 1 in location &39 - this value will be &00, as the Check end of statement routine resets the PTR A offset to 1. This is the load address (LSB) parameter in the OSFILE parameter block.
Store &18 (PAGE -> MSB Byte) in location &3A. This is file load address (MSB) parameter in the OSFILE parameter block.

[BE4B] Read the Machine's High Order address (OSBYTE call #&82).
If no memory extensions are in place, the High order address will be &FFFF.
Store the High Order address in locations &3B-&3C (high byte first).
This sets the High Order location of the Load address in the OSFILE parameter block.
(now, locations &39-&3C contain the complete load address for the file).

Disassembly for the Get Filename and set parameter block Filename and Load address routine

BE33	L	076 146 144	4C 92 90	JMP &9092 Type mismatch error
BE36	/	032 047 157	20 2F 9D	JSR &9D2F Ptr B = Ptr A & Get result of expression
BE39		208 248	D0 F8	BNE -8 --> &BE33

BE3B	%	032 037 190	20 25 BE	JSR &BE25 Store #&0D (Carriage Return) at end of SWA
BE3E	L	076 145 155	4C 91 9B	JMP &9B91 X = A; Check for end of Statement (PTR B)
BE41	6	032 054 190	20 36 BE	JSR &BE36 Get filename
BE44		136	88	DEY
BE45	9	132 057	84 39	STY &39
BE47		165 024	A5 18	LDA &18
BE49	:	133 058	85 3A	STA &3A
BE4B		169 130	A9 82	LDA#&82
BE4D		032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
BE50	;	134 059	86 3B	STX &3B
BE52	<	132 060	84 3C	STY &3C
BE54	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B3DD LISTO

Submitted by Steve Fewell

Description:

This routine is called when the LIST token is executed. This routine checks whether an 'O' character follows the LIST token, and if not, then the LIST routine is executed instead, otherwise LISTO is executed as described below.

Disassembly for the LISTO routine

B3DD		200	C8	INY
B3DE		177 011	B1 0B	LDA (&0B),Y
B3E0	O	201 079	C9 4F	CMP#&4F
B3E2		208 182	D0 B6	BNE -74 --> &B39A
B3E4		230 010	E6 0A	INC &0A
B3E6	o	032 111 146	20 6F 92	JSR &926F
B3E9		032 166 155	20 A6 9B	JSR &9BA6
B3EC	*	165 042	A5 2A	LDA &2A
B3EE		133 031	85 1F	STA &1F
B3F0	L	076 134 143	4C 86 8F	JMP &8F86

BASIC IV ROM Routines

Page Last Altered: undefined

8F00 OLD

Submitted by Steve Fewell

Description:

Call routine &9BA6 to check that the statement terminates with a ':', '<cr>' or 'ELSE' character (Syntax error if not).
Set pointer &37-&38 to point to PAGE (high byte from location &18, and low byte = &00).
Store #&00 to the second byte of the program ((&37-&38) + Y (1)).
Call routine &BDE5 to check that the program at the address PAGE can be read correctly & is a valid program, 'Bad program' if not.
Jump to &8F83 to prompt for the next command line input.
Note: The rest of the command line is ignored even if an 'ELSE' or ':' character terminated the statement.

Disassembly for the OLD routine

8F00		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
8F03		165 024	A5 18	LDA &18
8F05	8	133 056	85 38	STA &38
8F07	d7	100 055	64 37	STZ &37
8F09		169 000	A9 00	LDA#&00
8F0B	7	145 055	91 37	STA (&37),Y
8F0D		032 229 189	20 E5 BD	JSR &BDE5 Check program can be read correctly ('Bad program' message if not)
8F10	q	128 113	80 71	BRA 113 --> &8F83 Initialise & prompt for next command line

---

BASIC IV ROM Routines

Page Last Altered: undefined

BE55 SAVE

Submitted by Steve Fewell

Description:

Call routine &BDE5 to check that the program can be read correctly. If it cannot be read then display the 'Bad program' message and stop processing the SAVE command.

Call the routine &BE41 to obtain the filename (after the SAVE keyword) and append a '<cr>' character to the end of the filename (so that the Operating System can identify the end of the name). It also checks that the SAVE statement has been terminated correctly with an ':', '<cr>' or 'ELSE-token' character (or issue 'Syntax error' if not).
This routine sets locations &37-&38 to point to the filename (i.e. the SWA address - &0600), locations &39-&3A to PAGE (setting the LSB byte to zero), i.e. the file load address, and locations &3B-&3C to the Operating System High order address as ready by OSBYTE call #&82 (this value is usually &FFFF).
This completes the 32-bit load address for the file stored in locations &39-&3C.
On return from routine &BE41, X is the high byte for the Operating System High order address and Y is the low byte for the 16-bit Operating System High order address.

Set &3D-&40 to value #&802B (with bytes &3F-&40 being the 16-bit Operating System High order address returned in X and Y). This is the BASIC ROM access point, and the execution address for the file (i.e. the BASIC program's execution address). This address doesn't, however, seem to have much use as there isn't a way to directly execute a BASIC program from the Operating System / Filing System.
Set &41-&44 to PAGE (&18) (with byte &41 being zero and bytes &43-&44 being the 16-bit Operating System High order address returned in X and Y). This is the start address for the data to save.
Set &45-&48 to TOP (&12-&13) (with bytes &47-&48 being the 16-bit Operating System High order address returned in X and Y). This sets the end address of the data to save.

Set A to #&00 (this is the OSFILE (&FFDD) option for saving a block of memory to a file).
Set X to #&37 and Y to #&00 (this is the address of the parameter block for the OSFILE command - #&0037).

Call the OSFILE Operating System routine (&FFDD) to execute the file save.
Jump to &9005 to execute the next statement on the current command/program line (if any).

Disassembly for the SAVE routine

BE55		032 229 189	20 E5 BD	JSR &BDE5 Check program can be read correctly ('Bad program' message if not)
BE58	A	032 065 190	20 41 BE	JSR &BE41 Get Filename and set parameter block filename and load address
BE5B	?	134 063	86 3F	STX &3F
BE5D	@	132 064	84 40	STY &40
BE5F	C	134 067	86 43	STX &43
BE61	D	132 068	84 44	STY &44
BE63	G	134 071	86 47	STX &47
BE65	H	132 072	84 48	STY &48
BE67	dA	100 065	64 41	STZ &41
BE69		166 018	A6 12	LDX &12
BE6B	E	134 069	86 45	STX &45
BE6D		166 019	A6 13	LDX &13
BE6F	F	134 070	86 46	STX &46
BE71	+	162 043	A2 2B	LDX#&2B
BE73	=	134 061	86 3D	STX &3D
BE75		162 128	A2 80	LDX#&80
BE77	>	134 062	86 3E	STX &3E
BE79		166 024	A6 18	LDX &18
BE7B	B	134 066	86 42	STX &42
BE7D		169 000	A9 00	LDA#&00
BE7F		168	A8	TAY
BE80	7	162 055	A2 37	LDX#&37
BE82		032 221 255	20 DD FF	JSR &FFDD OSFILE
BE85	$	128 036	80 24	BRA 36 --> &BEAB [JMP &9005 Execute next statement]

BASIC IV ROM Routines

Page Last Altered: undefined

B393 EDIT

Submitted by Steve Fewell

Description:

Call &BBAC to Initialise Page &7 and to clear variables, etc...
Set LISTO flag to #&80 (so that the top bit is set (indicating that we are in EDIT mode).
Before the EDITOR application can be caled, the program needs to be detokenised into ASCII-text and stored to memory
This is done by calling the LIST routine with the LISTO flag set to #&80 (meaning that all output will be stored to memory instead of being displayed on the screen). All but the top bit of the LISTO flag is set to 0 to specify that no special LISTO spacing formats are to be applied to the program listing.
So, lastly, call the LIST routine to handle the listing of the program and storing to memory.

Disassembly for the EDIT routine

B389	EDIT 12,2	069 068 073 084 032 049 050 044 050 013	45 44 49 54 20 31 32 2C 32 0D	EQUS "EDIT 12,2" + '<cr>'
B393		032 172 187	20 AC BB	JSR &BBAC Initialise Page 7 & reset Variable pointers, etc...
B396		169 128	A9 80	LDA#&80
B398		133 031	85 1F	STA &1F
B39A				...LIST...

BASIC IV ROM Routines

Page Last Altered: undefined

B39A LIST

Submitted by Steve Fewell

Description:

Zero &3B (FOR counter) & &3C (REPEAT counter).

Obtain the "list from" line number after the LIST token (if specified), as follows:
Set the IWA to zero.
Call &9B1E to set the IWA to the untokenised Line number value based on the tokenised Line Number at BASIC Text pointer A (if any found; otherwise, the IWA will retain the value 0).
Store the processor flags (PHP) on the 6502 stack (for future checking).
Push the IWA value to the BASIC Stack.

Obtain the "list to" line number after the LIST token (if specified), as follows:
Set the IWA to -1 (&FFFFFFFF) (true).
Shift right the 2nd byte of the IWA (byte &2B), so that the first 2 bytes of the IWA contain the maximum Line Number value: &7FFF.
Retrieve the processor flags from the stack and check the carry flag as follows:

- -->If carry is set (the first Line Number was specified) then check for a second (To) line number as follows:
  - * Check whether the next non-space character (after the "from Line number") is a comma ','.
  - * If comma found then check for a "To Line Number":
    - * [&B3C6] Call &9B1E to set the IWA to the untokenised Line number value based on the tokenised Line Number at BASIC Text pointer A (if any found; otherwise, the IWA will retain the value &7FFF).
    - * Store the IWA value ("To Line Number") at locations &31-&34.
    - * If the next non-space character if 'IF-token' then set the BASIC text Pointer A to point to the next non-space character (and set PTR A offset to 1); otherwise, [&B3F3] Check for the end of statement ('Syntax error' if not terminated correctly).
  - * If comma not found then, a single Line has been specified, so:
    - * Retrieve the "from Line Number" IWA value from the BASIC Stack.

* Store the "from Line Number" to the BASIC Stack again (keeping the IWA set to the "From Line Number" value).
* Decrement the BASIC Text pointer A offset.
* [&B3C9] Store the IWA value (the "From Line Number" specified) at locations &31-&34 (the "To Line Number").
* If the next non-space character if 'IF-token' then set the BASIC text Pointer A to point to the next non-space character (and set PTR A offset to 1); otherwise, [&B3F3] Check for the end of statement ('Syntax error' if not terminated correctly).

-->If carry was not set (the first Line Number wasn't specified) then check for a second (To) line number as follows:
* Check whether the next non-space character (after the "from Line number") is a comma ','.
* If comma found then check for a "To Line Number":
  * [&B3C6] Call &9B1E to set the IWA to the untokenised Line number value based on the tokenised Line Number at BASIC Text pointer A (if any found; otherwise, the IWA will retain the value &7FFF).
  * Store the IWA value ("To Line Number") at locations &31-&34.
  * If the next non-space character if 'IF-token' then set the BASIC text Pointer A to point to the next non-space character (and set PTR A offset to 1); otherwise, [&B3F3] Check for the end of statement ('Syntax error' if not terminated correctly).
* If comma not found then:
  * [&B3C4] Decrement the BASIC Text Pointer A offset and jump to &B3C9.
  * [&B3C9] Store the IWA value ("To Line Number" - &7FFF) at locations &31-&34.
  * If the next non-space character if 'IF-token' then set the BASIC text Pointer A to point to the next non-space character (and set PTR A offset to 1); otherwise, [&B3F3] Check for the end of statement ('Syntax error' if not terminated correctly).

Now, BASIC Text pointer A points to the Next command line statement and BASIC text pointer B points to the 'IF' part of the List Statement.
[&B3F6] Set BASIC Text Pointer B (LSB) to BASIC Text Pointer A (LSB).
Call routine &BDE5 to check that the program can be read correctly (display the warning message 'Bad program' if the program is corrupt and cannot be read correctly).

Retrieve the Integer value from the BASIC Stack into the IWA. This is the minimum line number value.
Call routine &80CD to find the first program line that is greater than or equal to the IWA value.
Set text pointer A to the address of the first program line returned by the &80CD routine.

If the line is not equal to the one we want (carry is 0) then:
* [&B40D] Decrement the Y offset value, so that Y = 1
* [&B41F] Set the IWA to the next line number found (the 2-byte line number pointed to by PTR A + offset 1)

Compare the current Line number (in the IWA) with the Maximum line number (bytes &31-&32).

If the current line number is greater than the IWA Line number then we have reached the end of the LIST selection, so:
* Check the LISTO flag (&1F). If the flag is not negative then we are not in 'EDIT' mode, so jump to &8F86 to return to the command line prompt and await the user's next command entry.
* Otherwise, we are in 'EDIT' mode, and now that the program (in ASCII text) has been copied to memory we can load the editor application to enable the user to edit the program text.
* To do this, Set X (LSB) and Y (MSB) to point to address &B389, which contains the text "EDIT 12,2" + CHR$(13), and jump to the OSCLI (Operating System Command Line Interface) routine to execute the command at &B389.
* The BASIC language has now been exited, and will not be reloaded until the EDITOR exits back to the language.

Otherwise, the line number is within the specified range, so continue to display the line (in ASCII text) to the screen (or memory if we are in EDIT mode!).

Zero locations &4C (ignore program text status) and &4D (meets IF condition status).
Store 4 in the BASIC Text pointer A offset and the BASIC text pointer B offset (to skip over the line number).
If &3B (FOR loop counter) is negative (encountered NEXT without a FOR) then zero location &3B.
If &3C (REPEAT loop counter) is negative (encountered UNTIL without a REPEAT) then zero location &3C.

[&B454] Process the LISTO spacing for the current Program Line, as follows:
Read the next character on the program line. If it is '<cr>' then jump to &B491 (Print program line).
If the current character on the program line is 'REM-token' then Store #&F4 in location &4C (ignore program text flag), as the rest of the line is to be ignored as far as the LISTO spacing is concerned.

If the current character on the program line is "" [quote] then EOR #&22 with the current value of location &4C (ignore program text), i.e.: if we were inside a String-literal quote then the EOR would set &4C back to zero (stop ignoring Program text); and if we weren't inside a String-literal then the EOR would set the &4C flag to #&22, specifying that the rest of the text should be ignored until the end of the String-literal is found.

Check the Ignore Program Text flag (location &4C). If it is zero then we are not ignoring text, so set the loop counters as follows:

* If the current character on the program line is 'NEXT-token' then decrement the FOR-loop counter (&3B)
* If the current character on the program line is 'UNTIL-token' then decrement the REPEAT-loop counter (&3C)

[&B476] Check the IF condition:
Set X to &19 (BASIC Text Pointer B LSB) - this location is used to store the current Command line (&700-&7FF) offset.
[&B478] Load the next character (after the IF keyword [if specified]) on the Command line). If it is '<cr>' then the 'IF' condition has been checked (and matches the text at the current location on the Program Line), so store #&0D in location &4D (a non-zero value means that the 'IF' condition has passed), increment &1B (Program Line offset), set Y to the &1B value and jump back to &B454 to continue checking the program line.
Compare each character of the 'IF'-condition text (at &700 + X) with the next character of the Program Line (&0B,&0C) + Y, incrementing X and Y for each character checked. Keep checking [from &B478] until we find a mismatch or until the '<cr>' character of the 'IF'-condition is reached.

If a mismatch is found then do not alter the &4D value (so that it remains as zero - meaning that the 'IF' test wasn't passed), increment &1B (the Program Line offset [position of the current character on the Program Line]), set Y to the &1B value and jump to &B454 to continue checking the program line.

[&B491] Print the program Line:
If the 'IF' condition flag (&4D) is zero then the 'IF' condition does not match the current line, so jump back to &B41C to advance the (&0A, &0B) pointer to the next program line and [&B41F] continue processing with the next program line.

Call the &A085 routine to display the Line Number (in the IWA) on the screen.

The user may specify that none, one or many of the different LISTO spacing options should be used, these are now checked.
Check LISTO option#1 as follows:

Set A to 1 and increment X to 1 [X is 0 after call the &A085 routine] which specifies the number of spaces to print if the current LISTO option is met. And set the carry flag.
Call routine &BDB4 to check whether the LISTO option#1 is set (in the LISTO flag, &1F), if it is then X number of spaces will be printed to the screen.

Check LISTO option#2 as follows:
Set A to 2 and set X to the value of location &3B (the FOR-loop counter) which specifies the number of spaces to print if the current LISTO option is met. And set the carry flag.
Call routine &BDB3 to check whether the LISTO option#2 is set (in the LISTO flag, &1F), if it is then X number of spaces will be printed to the screen.

Check LISTO option#4 as follows:
Set A to 4 and set X to the value of location &3C (the REPEAT-loop counter) which specifies the number of spaces to print if the current LISTO option is met. And set the carry flag.
Call routine &BDB3 to check whether the LISTO option#4 is set (in the LISTO flag, &1F), if it is then X number of spaces will be printed to the screen.

Zero flag &4C (the program text ignore flag).
Reset Y (Program Line offset) to the first character of the Program Line (from location &0A).
[&B4B1] Load the next character of the program line.

If the current character is '<cr>' then the line has been printed, so jump back to &B410 to do the following:

* Output the '<cr>' character to the screen (or to memory if we are in EDIT mode)
* If we are not in EDIT mode then also output a Linefeed character (ASCII &0A) to the screen
* Call &9BBC to increment the pointer (&0A, &0B) to the next program line
* Continue with &B41F to process the next program line

If the current character on the program line is "" [quote] then EOR #&22 with the current value of location &4C (ignore program text), i.e.: if we were inside a String-literal quote then the EOR would set &4C back to zero (stop ignoring Program text); and if we weren't inside a String-literal then the EOR would set the &4C flag to #&22, specifying that the rest of the text should be ignored until the end of the String-literal is found. Output the quote character to the screen, increment Y (to point to the next character) and jump back to &B4B1 to check and output the next character.

[&B4C7] If the ignore program text flag contains a value (&4C) then:

* [&B4C1] Call &BD94 to output the current character to the screen (or to memory if we are in EDIT mode)
* Increment Y (to point to the next character)
* Jump back to &B4B1 to check and output the next character

If the current character on the program line is a Line Number token (#&8D) then:

* Call &9B2A to load the next character and set &2A-&2B to the value of the tokenised Line Number held in the next 3 characters.
* Store the updated offset (Y) back in location &0A (PTR A offset)
* Call &A081 to output the Line number to the screen (or memory if we are in EDIT mode)
* Jump back to &A4DF to Load Y with the updated &0A value and [&B4B1] process the next character

If the current character on the program line is 'FOR-token' then increment the FOR-loop counter (at location &3B).
If the current character on the program line is 'REPEAT-token' then increment the REPEAT-loop counter (at location &3C).
If the current character on the program line is 'REM-token' then set the ignore program text flag (&4C) to #&F4,
causing the rest of the line to be ignored as far as LISTO space formatting is concerned.

Call routine &BD37 to output the current character/token to the screen (or to memory if we are in EDIT mdoe).
Increment Y (to point to the next character on the program line) and jump back to &B4B1 to process the next character.

Disassembly for the LIST - parameters routine

B39A	d;	100 059	64 3B	STZ &3B
B39C	d<	100 060	64 3C	STZ &3C
B39E		032 232 171	20 E8 AB	JSR &ABE8 Set IWA to FALSE (0)
B3A1		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
B3A4		008	08	PHP
B3A5	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
B3A8		032 219 171	20 DB AB	JSR &ABDB Set IWA to TRUE (-1)
B3AB	F+	070 043	46 2B	LSR &2B
B3AD	(	040	28	PLP
B3AE		144 015	90 0F	BCC 15 --> &B3BF
B3B0		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
B3B3		240 017	F0 11	BEQ 17 --> &B3C6
B3B5		032 230 188	20 E6 BC	JSR &BCE6 Retrieve IWA value from the BASIC Stack [popi]
B3B8	&	032 038 188	20 26 BC	JSR &BC26 Push IWA value to the BASIC Stack [pushi]
B3BB		198 010	C6 0A	DEC &0A
B3BD		128 010	80 0A	BRA 10 --> &B3C9
B3BF		032 229 140	20 E5 8C	JSR &8CE5 Compare next non-space [PTR A] character with ','
B3C2		240 002	F0 02	BEQ 2 --> &B3C6
B3C4		198 010	C6 0A	DEC &0A
B3C6		032 030 155	20 1E 9B	JSR &9B1E Detokenise the Line Number at PTR A & Set IWA to the Line Number value
B3C9	1	162 049	A2 31	LDX#&31
B3CB		032 198 189	20 C6 BD	JSR &BDC6 Store Integer (IWA) to zero page location (specified by X)
B3CE		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space character pointed to by Ptr A
B3D1		201 231	C9 E7	CMP#&E7
B3D3		208 030	D0 1E	BNE 30 --> &B3F3

```
B3D5        032 224 142     20 E0 8E    JSR &8EE0 Get next non-space character pointed to by Ptr A
B3D8        032 188 155     20 BC 9B    JSR &9BBC Update BASIC Text pointer A (Add offset value & then reset offset to 1)
B3DB        128 025         80 19       BRA 25 --> &B3F6
```

Disassembly for the LIST - listing routine

```
B3F3        032 176 155 20 B0 9B  BRA 26 --> &9BB0 Check for end of Statement
B3F6        165 011     A5 0B     LDA &0B
B3F8        133 025     85 19     STA &19
B3FA        032 229 189 20 E5 BD  JSR &BDE5 Check program can be read correctly ('Bad program' message if not)
B3FD        032 230 188 20 E6 BC  JSR &BCE6 Retrieve IWA value from the BASIC Stack [popi]
B400        032 205 128 20 CD 80  JSR &80CD Search for Program Line >= the Line Number in the IWA
B403  =     165 061     A5 3D     LDA &3D
B405        133 011     85 0B     STA &0B
B407  >     165 062     A5 3E     LDA &3E
B409        133 012     85 0C     STA &0C
B40B        176 027     B0 1B     BCS 27 --> &B428
B40D        136         88        DEY
B40E        128 015     80 0F     BRA 15 --> &B41F
B410        032 148 189 20 94 BD  JSR &BD94 Output character to the screen
B413  $     036 031     24 1F     BIT &1F
B415  0     048 005     30 05     BMI 5 --> &B41C
B417        169 010     A9 0A     LDA#&0A
B419        032 238 255 20 EE FF  JSR &FFEE   OSWRCH
B41C        032 188 155 20 BC 9B  JSR &9BBC Update BASIC Text pointer A (Add offset value & then reset offset to 1)
B41F        177 011     B1 0B     LDA (&0B),Y
B421  +     133 043     85 2B     STA &2B
B423        200         C8        INY
B424        177 011     B1 0B     LDA (&0B),Y
B426  *     133 042     85 2A     STA &2A
B428  *     165 042     A5 2A     LDA &2A
B42A        024         18        CLC
B42B  1     229 049     E5 31     SBC &31
B42D  +     165 043     A5 2B     LDA &2B
B42F  2     229 050     E5 32     SBC &32
```

```
B431       144 011      90 0B     BCC 11 --> &B43E
B433  $    036 031      24 1F     BIT &1F
B435       016 185      10 B9     BPL -71 --> &B3F0
B437       162 137      A2 89     LDX#&89
B439       160 179      A0 B3     LDY#&B3
B43B  L    076 247 255  4C F7 FF  JMP &FFF7   OSCLI
B43E  dL   100 076      64 4C     STZ &4C
B440  dM   100 077      64 4D     STZ &4D
B442       160 004      A0 04     LDY#&04
B444       132 010      84 0A     STY &0A
B446       132 027      84 1B     STY &1B
B448  $;   036 059      24 3B     BIT &3B
B44A       016 002      10 02     BPL 2 --> &B44E
B44C  d;   100 059      64 3B     STZ &3B
B44E  $<   036 060      24 3C     BIT &3C
B450       016 002      10 02     BPL 2 --> &B454
B452  d<   100 060      64 3C     STZ &3C
B454       177 011      B1 0B     LDA (&0B),Y
B456       201 013      C9 0D     CMP#&0D
B458  7    240 055      F0 37     BEQ 55 --> &B491
B45A       201 244      C9 F4     CMP#&F4
B45C       240 006      F0 06     BEQ 6 --> &B464
B45E  "    201 034      C9 22     CMP#&22
B460       208 004      D0 04     BNE 4 --> &B466
B462  EL   069 076      45 4C     EOR &4C
B464  L    133 076      85 4C     STA &4C
B466  L    166 076      A6 4C     LDX &4C
B468       208 012      D0 0C     BNE 12 --> &B476
B46A       201 237      C9 ED     CMP#&ED
B46C       208 002      D0 02     BNE 2 --> &B470
B46E  ;    198 059      C6 3B     DEC &3B
B470       201 253      C9 FD     CMP#&FD
B472       208 002      D0 02     BNE 2 --> &B476
B474  <    198 060      C6 3C     DEC &3C
B476       166 025      A6 19     LDX &19
B478       189 000 007  BD 00 07  LDA &0700,X
```

B47B		201 013	C9 0D	CMP#&0D
B47D		240 010	F0 0A	BEQ 10 --> &B489
B47F		209 011	D1 0B	CMP (&0B),Y
B481		208 008	D0 08	BNE 8 --> &B48B
B483		200	C8	INY
B484		232	E8	INX
B485		128 241	80 F1	BRA -15 --> &B478
B487		128 135	80 87	BRA -121 --> &B410
B489	M	133 077	85 4D	STA &4D
B48B		230 027	E6 1B	INC &1B
B48D		164 027	A4 1B	LDY &1B
B48F		128 195	80 C3	BRA -61 --> &B454
B491	M	165 077	A5 4D	LDA &4D
B493		240 135	F0 87	BEQ -121 --> &B41C
B495		032 133 160	20 85 A0	JSR &A085 Print Line Number on screen (width 5)
B498		169 001	A9 01	LDA#&01
B49A		232	E8	INX
B49B	8	056	38	SEC
B49C		032 180 189	20 B4 BD	JSR &BDB4 AND value in A with LISTO option (&1F), Output (result divided by 2 + carry*&80) number of spaces
B49F	;	166 059	A6 3B	LDX &3B
B4A1		169 002	A9 02	LDA#&02
B4A3		032 179 189	20 B3 BD	JSR &BDB3 AND value in A with LISTO option (&1F), Output (result divided by 2) number of spaces
B4A6	<	166 060	A6 3C	LDX &3C
B4A8		169 004	A9 04	LDA#&04
B4AA		032 179 189	20 B3 BD	JSR &BDB3 AND value in A with LISTO option (&1F), Output (result divided by 2) number of spaces
B4AD	dL	100 076	64 4C	STZ &4C
B4AF		164 010	A4 0A	LDY &0A
B4B1		177 011	B1 0B	LDA (&0B),Y
B4B3		201 013	C9 0D	CMP#&0D
B4B5		240 208	F0 D0	BEQ -48 --> &B487
B4B7	"	201 034	C9 22	CMP#&22
B4B9		208 012	D0 0C	BNE 12 --> &B4C7
B4BB	EL	069 076	45 4C	EOR &4C

```
B4BD L   133 076       85 4C     STA &4C
B4BF  "  169 034       A9 22     LDA#&22
B4C1     032 148 189   20 94 BD  JSR &BD94 Output character to the screen
B4C4     200           C8        INY
B4C5     128 234       80 EA     BRA -22 --> &B4B1
B4C7  L  166 076       A6 4C     LDX &4C
B4C9     208 246       D0 F6     BNE -10 --> &B4C1
B4CB     201 141       C9 8D     CMP#&8D
B4CD     208 010       D0 0A     BNE 10 --> &B4D9
B4CF  *  032 042 155   20 2A 9B  JSR &9B2A Detokenise Line Number at PTR A (without skip spaces or skip #&8D char) &
                                  Set IWA to the Line Number value
B4D2     132 010       84 0A     STY &0A
B4D4     032 129 160   20 81 A0  JSR &A081 Print Line Number on screen (width 0)
B4D7     128 214       80 D6     BRA -42 --> &B4AF
B4D9     201 227       C9 E3     CMP#&E3
B4DB     208 002       D0 02     BNE 2 --> &B4DF
B4DD ;   230 059       E6 3B     INC &3B
B4DF     201 245       C9 F5     CMP#&F5
B4E1     208 002       D0 02     BNE 2 --> &B4E5
B4E3  <  230 060       E6 3C     INC &3C
B4E5     201 244       C9 F4     CMP#&F4
B4E7     208 002       D0 02     BNE 2 --> &B4EB
B4E9  L  133 076       85 4C     STA &4C
B4EB  7  032 055 189   20 37 BD  JSR &BD37 Output Character/Token on screen
B4EE     200           C8        INY
B4EF     128 192       80 C0     BRA -64 --> &B4B1
```

BASIC IV ROM Routines

Page Last Altered: undefined

9634 PAGE=

Submitted by Steve Fewell

Description:

Call routine &96B9 to Set BASIC Text Pointer B to BASIC Text Pointer A value, check that the next non-space character is an "=", and issue 'Mistake' error if it isn't. Get Result of the expression at BASIC Text Pointer B. Check that the statement terminates correctly (with a '<cr>', ':' or 'ELSE'), (issue 'Syntax error if not terminated correctly), and convert the numeric result value to Integer (or issue 'Type mismatch' error if the value is a String).

Set PAGE (&18) to the Integer result's MSB byte (&2B) (the LSB byte for page (&2A) is assumed to be '&00'.

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the PAGE= routine

9634	032 185 150	20 B9 96	JSR &96B9	Ptr B = Ptr A; Check '='; Get result; Check end of statement & convert result to Integer
9637 +	165 043	A5 2B	LDA &2B	
9639	133 024	85 18	STA &18	
963B L	076 005 144	4C 05 90	JMP &9005	Process next program statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

9679 TIME=

Submitted by Steve Fewell

Description:

If the next character on the command or program line is '$' then call the TIME$ routine (&968E) to assign to TIME$ and not TIME.

Call &96B9: to check for '=' as the next non-space character at BASIC text pointer B location, (& generate 'Syntax error' if another character was found instead); Get the result of the expression at BASIC text pointer B (and convert it to Integer); and check that the BASIC statement finishes after the expression (with a '<cr>', ':' or 'ELSE-token' character. 'Syntax error' is generated if the end of the statement was not found.

Store zero in location &2E (the imaginary byte 5 (MSB) of the IWA value!), as the set TIME OSWORD call (OSWORD 2) requires a 5-byte Integer value.
Set X (LSB) & Y (MSB) to point to the address of the start of the OSWORD parameter block that contains the new TIME value. This parameter block is located at &002A (i.e. the IWA). The first byte at &002A is the LSB byte of the new TIME value and the last byte (&002E) is the MSB byte of the new TIME value.
Set A to #&02, to indicate that we require the 'OSWORD 2' function.
Call OSWORD (&FFF1) to execute the Set TIME command and set the Operating System's Internal counter to the IWA value.

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the TIME= routine

9679		200	C8	INY
967A		177 011	B1 0B	LDA (&0B),Y
967C	$	201 036	C9 24	CMP#&24
967E		240 014	F0 0E	BEQ 14 --> &968E TIME$
9680		032 185 150	20 B9 96	JSR &96B9 Ptr B = Ptr A; Check '='; Get result; Check end of statement & convert result to Integer
9683	d.	100 046	64 2E	STZ &2E
9685	*	162 042	A2 2A	LDX#&2A
9687		160 000	A0 00	LDY#&00
9689		169 002	A9 02	LDA#&02

968B L 076 018 179 4C 12 B3 JMP &B312 [JSR &FFF1: BRA 11 --> &B222 [JMP &9005]] Process next statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

968E TIME$=

Submitted by Steve Fewell

Description:

Call &9B46: to check for '=' as the next non-space character at BASIC text pointer B location, (& generate 'Syntax error' if another character was found instead); Get the result of the expression at BASIC text pointer B; and check that the BASIC statement finishes after the expression (with a '<cr>', ':' or 'ELSE-token' character.
'Syntax error' is generated if the end of the statement was not found.

If the expression result (&27) is not 0 (String) then issue a 'Type mismatch' error as TIME$ must be set to a String value.

Set A to #&0F (the OSWORD call number for the setting of the time-date string).
Set Y to &36 (the SWA Length byte), store this value in location &05FF (i.e. the first byte of the OSWORD parameter block) - location &05FF shouldn't be used by any other BASIC storage.

Set X (LSB) & Y (MSB) to point to the address of the start of the OSWORD parameter block that contains the new TIME$ value. This parameter block is located at &05FF (i.e. the length byte followed by the SWA).
Call OSWORD (&FFF1) to execute the Set TIME command and set the Operating System's Internal counter to the IWA value.

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the TIME$= routine

968E		230 010	E6 0A	INC &0A
9690	F	032 070 155	20 46 9B	JSR &9B46 Ptr B = Ptr A; Check for '=' & get result of expression; check end of statement
9693	'	165 039	A5 27	LDA &27
9695	@	208 064	D0 40	BNE 64 --> &96D7 [JMP &9092 'Type mismatch' error]
9697		169 015	A9 0F	LDA#&0F
9699	6	164 054	A4 36	LDY &36
969B		140 255 005	8C FF 05	STY &05FF
969E		162 255	A2 FF	LDX#&FF
96A0		160 005	A0 05	LDY#&05

96A2	128 231	80 E7	BRA -25 --> &968B [JMP &B312 [JSR &FFF1: BRA 11 --> &B222 [JMP &9005 Process next Statement]]]

BASIC IV ROM Routines

Page Last Altered: undefined

B2EC ENVELOPE

Submitted by Steve Fewell

Description:

Get the result of the expression at BASIC Text Pointer A and convert the result to Integer (if Float), or issue a 'Type mismatch' error (if String value found).
Set X to 13 (as we have 14 parameters to obtain).

Loop for X number of times. For each loop:

* Push the Integer result (next parameter) LSB (&2A) to the Stack.
* Temporarily push X to the stack
* Check for "," (comma) charater (issue "Missing ," error if not found), and get the Integer result of expression at BASIC Text pointer B.
* Retrieve X from the stack.
* Decrement X value.
* If X is not 0 yet then loop again.

Now, we have the first 13 parameters on the stack and the last parameter in the IWA (&2A).

Call routine &9B96 to check that the Statement terminates correctly (if not, then a 'Syntax' error is issued).
Store the last parameter (IWA location &2A) in location &44.
Set X to 12 (Number of parameters left - 1).
Set Y to 8 (OSWORD call number for ENVELOPE).

[&B307] Pop the last Byte from the stack and store in location (&37 + X), i.e. location &43 for the 13th parameter).
Decrement X and keep repeating from &B307 until X is negative.

Now location &37 to location &37 + Original X value + 1 (i.e. &44 for ENVELOPE parameter block) contain the OSWORD call parameters.
Set A to Y value (the number of the OSWORD call).
Set X to #&37 and Y to #&00 (as the start of the parameter block is &0037).
Call the Operating System OSWORD routine (&FFF1) to perform the required operation.
Jump to &9005 to start processing the next Statement (if any).

Disassembly for the ENVELOPE routine

B2EC	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A & convert result to integer
B2EF		162 013	A2 0D	LDX#&0D
B2F1	*	165 042	A5 2A	LDA &2A
B2F3	H	072	48	PHA
B2F4		218	DA	PHX
B2F5		032 172 150	20 AC 96	JSR &96AC Check for ',', get result of expression and convert result to Integer
B2F8		250	FA	PLX
B2F9		202	CA	DEX
B2FA		208 245	D0 F5	BNE -11 --> &B2F1
B2FC		032 150 155	20 96 9B	JSR &9B96 Check for end of Statement (PTR B)
B2FF	*	165 042	A5 2A	LDA &2A
B301	D	133 068	85 44	STA &44
B303		162 012	A2 0C	LDX#&0C
B305		160 008	A0 08	LDY#&08
B307	h	104	68	PLA
B308	7	149 055	95 37	STA &37,X
B30A		202	CA	DEX
B30B		016 250	10 FA	BPL -6 --> &B307
B30D		152	98	TYA
B30E	7	162 055	A2 37	LDX#&37
B310		160 000	A0 00	LDY#&00
B312		032 241 255	20 F1 FF	JSR &FFF1 OSWORD
B315		128 011	80 0B	BRA 11 --> &B322 [JMP &9005] Continue with next Statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

9620 LOMEM=

Submitted by Steve Fewell

Description:

Call routine &96B9 to Set BASIC Text Pointer B to BASIC Text Pointer A value, check that the next non-space character is an "=", and issue 'Mistake' error if it isn't. Get Result of the expression at BASIC Text Pointer B. Check that the statement terminates correctly (with a '<cr>', ':' or 'ELSE'), (issue 'Syntax error if not terminated correctly), and convert the numeric result value to Integer (or issue 'Type mismatch' error if the value is a String).

Set LOMEM (&00-&01) to the Integer result's 2 LSB bytes (&2A-&2B).
Set the Heap/VARTOP (&02-&03) to the LOMEM value.
Note: this will destroy any variables currently on the heap.

Call routine &BBBB to reset the Floating-Point addresses in page &07, and zero any non-resident variables (that is all vaiables, except, @%, A% through to Z%) by clearing page &04 locations &0480 to &04FF.

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the LOMEM= routine

9620		032 185 150	20 B9 96	JSR &96B9 Ptr B = Ptr A; Check '='; Get result; Check end of statement & convert result to Integer
9623	*	165 042	A5 2A	LDA &2A
9625		133 000	85 00	STA &00
9627		133 002	85 02	STA &02
9629	+	165 043	A5 2B	LDA &2B
962B		133 001	85 01	STA &01
962D		133 003	85 03	STA &03
962F		032 187 187	20 BB BB	JSR &BBBB Initialise page &07, clear non-resident variables (locations &0480-&04FF)
9632		128 007	80 07	BRA 7 --> &963B [JMP &9005] Process the next statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

B2C8 SOUND

Submitted by Steve Fewell

Description:

Get the result of the expression at BASIC Text Pointer A and convert the result to Integer (if Float), or issue a 'Type mismatch' error (if String value found).
Set X to 3 (as we have 4 parameters to obtain).

Loop for X number of times. For each loop:

* Push the Integer result (next parameter) LSB 2-bytes (&2A-&2B) to the Stack.
* Temporarily push X to the stack
* Check for "," (comma) charater (issue "Missing ," error if not found), and get the Integer result of expression at BASIC Text pointer B.
* Retrieve X from the stack.
* Decrement X value.
* If X is not 0 yet then loop again.

Now, we have the first 3 parameters on the stack and the last parameter in the IWA (&2A-&2B).

Call routine &9B96 to check that the Statement terminates correctly (if not, then a 'Syntax' error is issued).
Store the last parameter (IWA locations &2A-&2B) in locations &3D-&3E (LSB value (&2A) first).
Set Y to 7 (OSWORD call number for SOUND).
Set X to 5 (Number of parameter bytes left - 1). Each Sound parameter is 2-bytes wide.

Call routine &B307 to do the following:
* Pop the last Byte from the stack and store in location (&37 + X), i.e. location &3C for the MSB of the 3rd SOUND parameter).
* Decrement X and keep repeating from &B307 until X is negative.
* Now location &37 to location &37 + Original X value + 2 (i.e. &3E for SOUND parameter block) contain the OSWORD call parameters.
* Set A to Y value (the number of the OSWORD call).
* Set X to #&37 and Y to #&00 (as the start of the parameter block is &0037).
* Call the Operating System OSWORD routine (&FFF1) to perform the required operation.
* Jump to &9005 to start processing the next Statement (if any).

Disassembly for the SOUND routine

B2C8	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A & convert result to integer
B2CB		162 003	A2 03	LDX#&03
B2CD	*	165 042	A5 2A	LDA &2A

B2CF	H	072	48	PHA
B2D0	+	165 043	A5 2B	LDA &2B
B2D2	H	072	48	PHA
B2D3		218	DA	PHX
B2D4		032 172 150	20 AC 96	JSR &96AC Check for ',', get result of expression and convert result to Integer
B2D7		250	FA	PLX
B2D8		202	CA	DEX
B2D9		208 242	D0 F2	BNE -14 --> &B2CD
B2DB		032 150 155	20 96 9B	JSR &9B96 Check for end of Statement (PTR B)
B2DE	*	165 042	A5 2A	LDA &2A
B2E0	=	133 061	85 3D	STA &3D
B2E2	+	165 043	A5 2B	LDA &2B
B2E4	>	133 062	85 3E	STA &3E
B2E6		160 007	A0 07	LDY#&07
B2E8		162 005	A2 05	LDX#&05
B2EA		128 027	80 1B	BRA 27 --> &B307 Create Parameter Block, call OSWORD & process next Statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

BEBD BPUT

Submitted by Steve Fewell

Description:

Call routine &BA3E to copy BASIC Text pointer A address to BASIC text pointer B, Check that the next character is '#' ('Missing #' error if not), set Y to the number after the '#' (the file channel number).
Store A (the file channel number, as &BA3E sets A to a copy of Y) to the 6502 stack.

Call routine &96AC to check for a comma ',' (issuing an error if not found), and get the Integer result of the expression after the comma (issuing 'Type mismatch' error if a String value was found). The IWA now contains the ASCII code of the character required to be output to the file.

Call &9B96 to check for the end of statement (at BASIC Text Pointer B). If the next character after the BPUT parameter value is not ':', '<cr>' or 'ELSE-token' then issue 'Syntax error', as the statement was not terminated correctly.

Retrieve Y (the channel number) from the 6502 stack.
Set A to the LSB value of the IWA (location &2A; i.e. the ASCII code of the character to output to the file).
Call the Operating system OSBPUT routine (&FFD4) to carry out the file operation.
Jump to &9005 to execute the next BASIC statement in the program/command line.

Disassembly for the BPUT routine

BEBD	>	032 062 186	20 3E BA	JSR &BA3E Copy PTR A to PTR B, check for '#', get file channel number & store it in Y
BEC0	H	072	48	PHA
BEC1		032 172 150	20 AC 96	JSR &96AC Check for ',', get result of expression and convert result to Integer
BEC4		032 150 155	20 96 9B	JSR &9B96 Check for end of Statement (PTR B)

```
BEC7    z  122          7A       PLY
BEC8    *  165 042      A5 2A    LDA &2A
BECA       032 212 255  20 D4 FF JSR &FFD4   OSBPUT (write a single byte to an open file)
BECD       128 220      80 DC    BRA -36 --> &BEAB   [JMP &9005 Execute next statement]
```

BASIC IV ROM Routines

Page Last Altered: undefined

8EFB CHAIN

Submitted by Steve Fewell

Description:

Call routine &BDD7 to get the filename after the CHAIN keyword (and place it in the SWA), and load the named file (which should be a BASIC program) at the PAGE address and check that the program can be read correctly ('Bad program' message is displayed if not).
Jump to &8F15 to execute the 'RUN' keyword and begin running the program.

Disassembly for the CHAIN routine

8EFB	032 215 189	20 D7 BD	JSR &BDD7 Get filename and Load the named file
8EFE	128 021	80 15	BRA 21 --> &8F15 RUN the program

BASIC IV ROM Routines

Page Last Altered: undefined

8F12 RUN

Submitted by Steve Fewell

Description:

Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character, issuing 'Syntax error' if it doesn't.
Call routine &BBAC to clear the run-time variable memory (variable pointer table), set up the page &7 constants &07F0-&07FF, and the Program Start, Stack Pointer, LOMEM and VARTOP information. It also clears the LISTO flag and clears the REPEAT level (&24), FOR level (&26) and GOSUB level (&25) - ready for the execution of the program.

Set BASIC Text pointer A to PAGE (MSB byte from location &18, and LSB byte = #&00).
Jump to &8F97 to tokenise the statement at the Text Pointer A location (should have no affect, as it is already tokenised) and then execute the statement (via routine &901E). As the Text Pointer A is not pointing to the BASIC Command line (location &0700-&07FF), the next program line will be executed after the current line.

Disassembly for the RUN routine

8F12		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
8F15		032 172 187	20 AC BB	JSR &BBAC Initialise Page 7 & reset Variable pointers, etc...
8F18		165 024	A5 18	LDA &18
8F1A		133 012	85 0C	STA &0C
8F1C	d	100 011	64 0B	STZ &0B
8F1E	w	128 119	80 77	BRA 119 --> &8F97 Tokenise and Execute command line

---

BASIC IV ROM Routines

Page Last Altered: undefined

963E CLEAR

Submitted by Steve Fewell

Description:

Check that the statement terminates correctly with a ':', '<cr>' [carriage-return] or 'ELSE' character, issue 'Syntax error' if it does not.

Call routine &BBAC to initialise HIMEM and LOMEM values, and to reset the Floating-Point addresses in page &07, and zero any non-resident variables (that is all vaiables, except, @%, A% through to Z%) by clearing page &04 locations &0480 to &04FF.

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the CLEAR routine

963E	032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
9641	032 172 187	20 AC BB	JSR &BBAC Initialise Page 7 & reset Variable pointers, etc...
9644	128 245	80 F5	BRA -11 --> &963B [JMP &9005] Process the next program statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

BEAE CLOSE

Submitted by Steve Fewell

Description:

Call routine &BA3E to copy BASIC Text pointer A address to BASIC text pointer B, Check that the next character is '#' ('Missing #' error if not), set Y to the number after the '#' (the file channel number).

Call &9B96 to check for the end of statement (at BASIC Text Pointer B). If the next character after the CLOSE parameter value is not ':', '<cr>' or 'ELSE-token' then issue 'Syntax error', as the statement was not terminated correctly.

Set Y to the value of location &2A (i.e. the LSB byte of the channel number). Note: if Y is zero then all open files will be closed.
Set A to zero (indicating to the OSFIND routine that we are closing a file).
Call the Operating system OSFIND routine (&FFCE) to carry out the file operation.
Jump to &9005 to execute the next BASIC statement in the program/command line.

Disassembly for the CLOSE routine

BEAE	>	032 062 186	20 3E BA	JSR &BA3E Copy PTR A to PTR B, check for '#', get file channel number & store it in Y
BEB1		032 150 155	20 96 9B	JSR &9B96 Check for end of Statement (PTR B)
BEB4	*	164 042	A4 2A	LDY &2A
BEB6		169 000	A9 00	LDA#&00
BEB8		032 206 255	20 CE FF	JSR &FFCE OSFIND (Open or close a file)
BEBB		128 238	80 EE	BRA -18 --> &BEAB [JMP &9005 Execute next statement]

BASIC IV ROM Routines

Page Last Altered: undefined

97E0 CLG

Submitted by Steve Fewell

Description:

Call &9BA6 to check that the BASIC Statement is terminated corectly (Issue 'Syntax' error if not).
Perform the CLG (VDU 16) operation by setting A to 16 and calling &97EE to output the character and then continue with executing the next statement (&9005).

Disassembly for the CLG routine

97E0	032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
97E3	169 016	A9 10	LDA#&10
97E5	128 007	80 07	BRA 7 --> &97EE Output ASCII Character & process next Statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

97E7 CLS

Submitted by Steve Fewell

Routine:CLS
Name: CLS (Clear Screen)
Starting Address: &A541
Entry criteria: &A, &B and &C point to input line.

Description:

Call routine &9BA6 to check that the BASIC Statement is terminated corectly.
Zero &1E [COUNT], as after a CLS, no characters will have been printed on line.
Perform the CLS (VDU 12) operation.
Jump to &9005 to continue executing the next statement.

Disassembly for the CLS routine

97E7		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
97EA	d	100 030	64 1E	STZ &1E
97EC		169 012	A9 0C	LDA#&0C
97EE		032 238 255	20 EE FF	JSR &FFEE OSWRCH
97F1	L	076 005 144	4C 05 90	JMP &9005 Execute the next BASIC program/command line statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

97A6 DRAW

Submitted by Steve Fewell

Description:

Set A to #&05. This is the PLOT type (i.e. the first parameter of the VDU 25 command) for the DRAW command.
[&97A8] Push A to the 6502 Stack.
Call &9D2F to get the result of the expression after the 'DRAW' keyword (at the BASIC text Pointer B location).
Call &96BC to convert the result value to Integer (if it is a Floating-Point value), or to issue a 'Type mismatch' error if the result is a String value.
Jump to the PLOT routine to execute the appropriate VDU 25 command for DRAW.

Disassembly for the DRAW routine

97A6		169 005	A9 05	LDA#&05
97A8	H	072	48	PHA
97A9	/	032 047 157	20 2F 9D	JSR &9D2F Ptr B = Ptr A & Get result of expression
97AC		032 188 150	20 BC 96	JSR &96BC Check result type (location &27) - if Float then convert to Integer
97AF		128 009	80 09	BRA 9 --> &97BA Call PLOT routine

BASIC IV ROM Routines

Page Last Altered: undefined

97B1 PLOT

Submitted by Steve Fewell

Description:

Get the Integer result of the expression after the PLOT keyword (at BASIC Text pointer A location).
If a string value was found then issue a 'Type mismatch' error.

Push the LSB of the Integer value (&2A) to the 6502 stack. This is the first PLOT paramater (i.e. the PLOT type).
Note: Only the 1-byte LSB value of the Integer is used - the rest of the Integer value (if supplied) is ignored.
Call routine &96AC to check for a comma ',' (issuing an error if not found), and get the Integer result of the expression after the comma (issuing 'Type mismatch' error if a String value was found).

[&97BA] Push the second Integer PLOT parameter value (in the IWA bytes &2A-&2D) to the BASIC stack.
Call routine &96AC to check for a comma ',' (issuing an error if not found), and get the Integer result of the expression after the comma (issuing 'Type mismatch' error if a String value was found). The IWA now contains the third PLOT parameter.

Call &9B96 to check for the end of statement (at BASIC Text Pointer B). If the next character after the third PLOT parameter value is not ':', '<cr>' or 'ELSE-token' then issue 'Syntax error', as the statement was not terminated correctly.

Send ASCII character #&19 to the OS Write character routine (to start a VDU 25 command).
Retrieve the first PLOT parameter from the 6502 Stack (this is the 1-byte Integer that specifies the PLOT type).
Send the first PLOT parameter (retrieved above) to the OS Write Character routine (the Operating System will take this value as the first parameter to the VDU 25 command.

Retrieve the second PLOT parameter from the BASIC Stack to locations &37-&3A (this is a 4-byte Integer value).
Send byte 1 (LSB) of this second PLOT parameter (location &37) to the OS Write Character routine (the Operating System will take this value as the second parameter to the VDU 25 command.
Send byte 2 of this second PLOT parameter (location &38) to the OS Write Character routine (the Operating System will take this value as the third parameter to the VDU 25 command.
Note: Only the first 16-bits of the 4 byte Integer are used (bytes 3 and 4 are ignored).

Send byte 1 (LSB) of the third PLOT parameter (IWA byte &2A) to the OS Write Character routine (the Operating System will take this value as the forth parameter to the VDU 25 command.
Send byte 2 of the third PLOT parameter (IWA byte &2B) to the OS Write Character routine (the Operating System will take this value as the fifth parameter to the VDU 25 command.
Note: Only the first 16-bits of the 4 byte Integer are used (bytes 3 and 4 are ignored).

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the PLOT routine

97B1	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
97B4	*	165 042	A5 2A	LDA &2A
97B6	H	072	48	PHA
97B7		032 172 150	20 AC 96	JSR &96AC Check for ',', get result of expression and convert result to Integer
97BA	&	032 038 188	20 26 BC	JSR &BC26 Push Integer to Stack
97BD		032 172 150	20 AC 96	JSR &96AC Check for ',', get result of expression and convert result to Integer
97C0		032 150 155	20 96 9B	JSR &9B96 Check for end of Statement (PTR B)
97C3		169 025	A9 19	LDA#&19
97C5		032 238 255	20 EE FF	JSR &FFEE OSWRCH
97C8	h	104	68	PLA
97C9		032 238 255	20 EE FF	JSR &FFEE OSWRCH
97CC		032 006 189	20 06 BD	JSR &BD06 Retrieve Integer from stack to locations &37-&3A
97CF	7	165 055	A5 37	LDA &37
97D1		032 238 255	20 EE FF	JSR &FFEE OSWRCH
97D4	8	165 056	A5 38	LDA &38
97D6		032 238 255	20 EE FF	JSR &FFEE OSWRCH
97D9	@	032 064 152	20 40 98	JSR &9840 Send character in ?&2A to the Output vector
97DC	+	165 043	A5 2B	LDA &2B
97DE		128 014	80 0E	BRA 14 --> &97EE [JSR &FFEE : JMP &9005 Process next program statement]

BASIC IV ROM Routines

Page Last Altered: undefined

8F25 END

Submitted by Steve Fewell

Description:

Call routine &9BA6 to check that the statement terminates correctly with a ':', '<cr>' or 'ELSE-token' character (if not, a 'Syntax error' is issued).
Call routine &BDE5 to check that the program can be read correctly (display a 'Bad program' message if it cannot be read through correctly - i.e. the lines do not terminate with a '<cr>' character

Jump to &8F86 to prompt for (and execute) the user command line input. Note: when command line input is being processed BASIC no longer recognises that a program is/was running until a 'RUN' command is entered.

Disassembly for the END routine

8F25		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
8F28		032 229 189	20 E5 BD	JSR &BDE5 Check program can be read correctly ('Bad program' message if not)
8F2B	Y	128 089	80 59	BRA 89 --> &8F86 Read & execute command line input

---

BASIC IV ROM Routines

Page Last Altered: undefined

9741 GCOL

Submitted by Steve Fewell

Description:

Get the result of the expression at BASIC Text Pointer A, convert the result to Integer (Type mismatch error if String), and Set BASIC Text Pointer A to BASIC Text Pointer B value.
Push &2A (the LSB of the Integer value - i.e. the first GCOL parameter) to the stack.

Call &96AC to check for a comma ('Missing ,' error if not found) and get the integer result at the BASIC text Pointer B location (issuing 'Type mismatch' error if a String value was found).
Call &9B96 to check that the Statement (at BASIC Text Pointer B location) terminates correctly ('Syntax' error is issued if end of statement marker is not found).

Output character 18 (to start a VDU 18 - GCOL command).
Pop the first Integer from the stack and [&979B] output the character (this is the first parameter to the GCOL command).

Next, output the contents of location &2A (The IWA LSB, which contains the second GCOL parameter), this completes the GCOL parameters, and the Operating system now executes the GCOL command.
Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the GCOL routine

9741	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
9744	*	165 042	A5 2A	LDA &2A
9746	H	072	48	PHA
9747		032 172 150	20 AC 96	JSR &96AC Check for ',', get result of expression and convert result to Integer
974A		032 150 155	20 96 9B	JSR &9B96 Check end of Statement (PTR B)
974D		169 018	A9 12	LDA#&12
974F		032 238 255	20 EE FF	JSR &FFEE OSBYTE
9752	h	104	68	PLA
9753	F	128 070	80 46	BRA 70 --> &979B

BASIC IV ROM Routines

Page Last Altered: undefined

975F MODE

Submitted by Steve Fewell

Description:

Get the result of the expression at BASIC Text Pointer A, convert the result to Integer (Type mismatch error if String), and Set BASIC Text Pointer A to BASIC Text Pointer B value.
Call &9BA6 to check that the Statement terminates correctly ('Syntax' error is issued if end of statement marker is not found).

Use OSBYTE #&82 to obtain the Machine high order address.
If the Machine high order address is not &FFFF then jump to &9797 (skip memory restriction tests).

Compare the Stack Pointer (&04-&05) with HIMEM (&06-&07), if they are not equal (i.e. we have items on Stack, i.e. as the result of local variables inside a PROC/FN call) then issue 'Bad MODE' error, as BASIC will not allow the screen mode to be altered if there are items on the Stack (as the stack could then get corrupted).
Note: as the stack contains the return details for a PROC/FN call, MODE cannot be issued from within a proedure or function, even if no local variables have been declared.

Use OSBYTE #&85 to read the top of User RAM for the specified screen mode (&2A) [X=address LSB & Y=address MSB].
Compare the 'Top of User RAM for the specified Mode' with VARTOP (i.e. the heap, or list of declared variables).
If the 'Top of User RAM for the specified Mode' would fall below the VARTOP value, then the program variables would be corrupted if the Mode was changed - so we are not allowed to change to that mode, and a 'Bad MODE' error is issued.

Compare the 'Top of User RAM for the specified Mode' with TOP (i.e. the end of the program).
If the 'Top of User RAM for the specified Mode' would fall below the TOP value, then the part of the program would be corrupted if the Mode was changed - so we are not allowed to change to that mode, and a 'Bad MODE' error is issued.

Set HIMEM (&06-&07) and the Stack Pointer to the 'Top of User RAM for the specified Mode' value.
Zero COUNT (&1E), as after a mode change, the screen is clear, and no characters will have been output on the current line yet.
Output special character 22 (the operating system's Mode change operator (VDU 22)).
Next, output the contents of location &2A (The IWA LSB, which contains the MODE number), this supplies the mode parameter to the VDU 22 call, and allows the operating system to execute a MODE change to the MODE specified in &2A.
Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the MODE routine

```
975F  o  032 111 146   20 6F 92   JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
9762     032 166 155   20 A6 9B   JSR &9BA6 Check end of Statement
9765  K  032 075 190   20 4B BE   JSR &BE4B Read High Order address (on return X = MSB, Y = LSB)
9768     232           E8         INX
9769  ,  208 044       D0 2C      BNE 44 --> &9797
976B     200           C8         INY
976C  )  208 041       D0 29      BNE 41 --> &9797
976E     165 004       A5 04      LDA &04
9770     197 006       C5 06      CMP &06
9772     208 197       D0 C5      BNE -59 --> &9739 Bad MODE error
9774     165 005       A5 05      LDA &05
9776     197 007       C5 07      CMP &07
9778     208 191       D0 BF      BNE -65 --> &9739 Bad MODE error
977A  *  166 042       A6 2A      LDX &2A
977C     169 133       A9 85      LDA#&85
977E     032 244 255   20 F4 FF   JSR &FFF4 OSBYTE
9781     228 002       E4 02      CPX &02
9783     152           98         TYA
9784     229 003       E5 03      SBC &03
9786     144 177       90 B1      BCC -79 --> &9739 Bad MODE error
9788     228 018       E4 12      CPX &12
978A     152           98         TYA
978B     229 019       E5 13      SBC &13
978D     144 170       90 AA      BCC -86 --> &9739 Bad MODE error
978F     134 006       86 06      STX &06
9791     134 004       86 04      STX &04
9793     132 007       84 07      STY &07
9795     132 005       84 05      STY &05
9797  d  100 030       64 1E      STZ &1E
9799     169 022       A9 16      LDA#&16
979B     032 238 255   20 EE FF   JSR &FFEE OSWRCH
979E  *  165 042       A5 2A      LDA &2A
97A0  L  128 076       80 4C      BRA 76 --> &97EE Output ASCII Character & process next Statement
```

BASIC IV ROM Routines

Page Last Altered: undefined

97A2 MOVE

Submitted by Steve Fewell

Description:

Set A to #&05. This is the PLOT type (i.e. the first parameter of the VDU 25 command) for the MOVE command.
Call &97A8 (the DRAW keyword routine) to Push A to the 6502 stack, obtain the Integer value after the 'MOVE' keyword and jump to the PLOT routine to execute the appropriate VDU 25 command for MOVE.

Disassembly for the MOVE routine

97A2	169 004	A9 04	LDA#&04
97A4	128 002	80 02	BRA 2 --> &97A8 DRAW (get parameters and call PLOT)

BASIC IV ROM Routines

Page Last Altered: undefined

980D VDU

Submitted by Steve Fewell

Description:

[&980D] Get the next non-space character (from BASIC Text Pointer A) after the 'VDU' keyword.
[&9810] If end of VDU statement found (i.e. the character is ':', '<cr>' [carriage-return] or 'ELSE-token') then jump to &9000 to execute the next command line or program statement.

If not end of statement found, then decrement BASIC Text pointer A Offset (&0A), and call routine &926F to get the result of the expression at BASIC Text Pointer A position and convert the result to Integer (or issue 'Syntax Error' if a String value was found).

Call routine &9840 to output the ASCII Code in the IWA LSB byte (location &2A) to the screen.

Get the next non-space character. If the character is ',' [comma] then jump back to &980D to check for any further VDU codes.
If the character is ';' [semi-colon] then [&9808] Output the ASCII code in location &2B (the MSB byte of the 16-bit value in the IWA) to the screen, and then continue with &980D (to check for further VDU codes to output).
If the character is '|' [vertical-bar] then output 9 ASCII-0 [Nul] bytes to complete any pending VDU statements and then jump back to &980D to check for further VDU codes to output.

If the character isn't ',', ';' or '|' then jump back to &9810 to check for end of statement and output further VDU codes (if end of statement isn't found).
This enables VDU calls such as 'W%=67: VDU 65W%66|68&66(69)' to be accepted and display the result: 'ACBDfE'! And additionally allows for spaces to be used instead of commas.

Disassembly for the VDU routine

9808	+	165 043	A5 2B	LDA &2B
980A		032 238 255	20 EE FF	JSR &FFEE OSWRCH
980D		032 224 142	20 E0 8E	JSR &8EE0 Get next non-space char (PTR A)
9810	:	201 058	C9 3A	CMP#&3A
9812	)	240 041	F0 29	BEQ 41 --> &983D [JMP &9000 Check end of statement & execute next statement]
9814		201 013	C9 0D	CMP#&0D
9816	%	240 037	F0 25	BEQ 37 --> &983D [JMP &9000 Check end of statement & execute next statement]

9818 201 139 C9 8B CMP#&8B
981A ! 240 033 F0 21 BEQ 33 --> &983D [JMP &9000 Check end of statement & execute next statement]
981C 198 010 C6 0A DEC &0A
981E o 032 111 146 20 6F 92 JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
9821 @ 032 064 152 20 40 98 JSR &9840 Send character in ?&2A to the Output vector
9824 032 229 140 20 E5 8C JSR &8CE5 Compare next non-space [PTR A] character with ','
9827 240 228 F0 E4 BEQ -28 --> &980D
9829 ; 201 059 C9 3B CMP#&3B
982B 240 219 F0 DB BEQ -37 --> &9808
982D | 201 124 C9 7C CMP#&7C
982F 208 223 D0 DF BNE -33 --> &9810
9831 169 000 A9 00 LDA#&00
9833 160 009 A0 09 LDY#&09
9835 032 238 255 20 EE FF JSR &FFEE OSBYTE
9838 136 88 DEY
9839 208 250 D0 FA BNE -6 --> &9835
983B 128 208 80 D0 BRA -48 --> &980D
983D L 076 000 144 4C 00 90 JMP &9000 Check end of statement & execute the next statement on the Command or Program Line

---

BASIC IV ROM Routines

Page Last Altered: undefined

97F4 REPORT

Submitted by Steve Fewell

Description:

Call &9BA6 to check that the Statement terminates correctly ('Syntax' error is issued if end of statement marker is not found).
Call routine &BA92 to print a carriage return and begin a new output line - so that the Error Message is shown on a new line.

Load each character of the string pointed to by the Operating System's 'Last Error' pointer [&FD-&FE] and, for each character, if it isn't 0 (end of String), call routine &BD37 to output the Token (in ASCII-text), or [ASCII] Character (if the value isn't a token).
When a ASCII Character with a code of 0 is encountered, or after 255 characters have been output, jump to &9005 to start processing the next program statement.

Disassembly for the REPORT routine

97F4		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
97F7		032 146 186	20 92 BA	JSR &BA92 Start new output line
97FA		160 001	A0 01	LDY#&01
97FC		177 253	B1 FD	LDA (&FD),Y
97FE		240 241	F0 F1	BEQ -15 --> &97F1
9800	7	032 055 189	20 37 BD	JSR &BD37 Output Character/Token on screen
9803		200	C8	INY
9804		208 246	D0 F6	BNE -10 --> &97FC
9806		128 233	80 E9	BRA -23 --> &97F1 [JMP &9005] Execute next statement

---

BASIC IV ROM Routines

Page Last Altered: undefined

9086 STOP

Submitted by Steve Fewell

Description:

Calls 9BA6 to check that the BASIC statement is correct and ends correctly
Syntax Error is given if anything else is found on the line after the STOP keyword.

Issues the STOP error [BRK]. The BASIC error handler will stop the program running and perform other functions required after an error is generated.

Disassembly for the STOP routine

9086	032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
9089	000	00	... STOP error

BASIC IV ROM Routines

Page Last Altered: undefined

9755 COLOUR

Submitted by Steve Fewell

Description:

Get the result of the expression at BASIC Text Pointer A, convert the result to Integer (Type mismatch error if String), and Set BASIC Text Pointer A to BASIC Text Pointer B value.

Call &9BA6 to check that the Statement (at BASIC Text Pointer A location) terminates correctly ('Syntax' error is issued if end of statement marker (':', '<cr>' or 'ELSE') is not found).

Output character 17 (to start a VDU 17 - COLOUR command).
[&979B] Next, output the contents of location &2A (The IWA LSB, which contains the COLOUR number), this completes the single-parameter COLOUR command, and the Operating System now executes the Colour command.

Lastly, jump to &9005 to start processing the next program statement.

Disassembly for the COLOUR routine

9755	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A convert result to integer
9758		032 166 155	20 A6 9B	JSR &9BA6 Check end of Statement
975B		169 017	A9 11	LDA#&11
975D	<	128 060	80 3C	BRA 60 --> &979B

BASIC IV ROM Routines

Page Last Altered: undefined

B317 WIDTH

Submitted by Steve Fewell

Description:

Get the result of the expression at BASIC Text Pointer A and convert the result to Integer (if Float), or issue a 'Type mismatch' error (if String value found).
Call routine &9B96 to check that the Statement terminates correctly (if not, then a 'Syntax' error is issued).
Set location &23 (WIDTH) to the Integer LSB Value (&2A) minus 1.
Jump to &9005 to start processing the next Statement.

Disassembly for the WIDTH routine

B317	o	032 111 146	20 6F 92	JSR &926F Evaluate Expression at BASIC Text pointer A & convert result to integer
B31A		032 150 155	20 96 9B	JSR &9B96 Check for end of Statement (PTR B)
B31D	*	164 042	A4 2A	LDY &2A
B31F		136	88	DEY
B320	#	132 035	84 23	STY &23
B322	L	076 005 144	4C 05 90	JMP &9005 Process next Statement

BASIC IV ROM Routines

Page Last Altered: undefined

BE87 OSCLI

Submitted by Steve Fewell

Description:

[BE87] Call &BE36 to do the following:
* Set BASIC Text Pointer B offset to BASIC Text Pointer A offset.
* Get result of expression at BASIC Text Pointer B.
* Issue 'Type mismatch' error if the result is not a String value.
* Call routine &BE25 to Store a carriage return (#&0D) character at the end of the String, and to set &37-&38 to point the the SWA address (&0600).
* Call &9B91 to check that the Statement is terminated correctly.

This obtains the ASCII String specified after the OSCLI keyword and appends a '<CR>' character to the end of the SWA value so that the string is in the correct format to be processed by the Operating System as a MOS command - ready for an operating system call to OSCLI (&FFF7) [which requires a <cr> to terminate the command String].

Set X (LSB - &00) and Y (MSB - &06) to the SWA address (&0600). This is the address of the command to send to the operating system command line interpreter routine (OSCLI).
Call OSCLI (&FFF7) to execute the Operating System ('*') command.
Jump to &9005 to execute the next BASIC command/statement.

Disassembly for the OSCLI routine

BE87	6 032 054 190	20 36 BE	JSR &BE36 Obtain string value and append an '<cr>' character to the string
BE8A	162 000	A2 00	LDX#&00
BE8C	160 006	A0 06	LDY#&06
BE8E	032 247 255	20 F7 FF	JSR &FFF7 OSCLI
BE91	128 024	80 18	BRA 24 --> &BEAB [JMP &9005 Execute next statement]

BASIC IV ROM Routines

Page Last Altered: undefined

98D1 '?' and '!' address peek/poke operators

Submitted by Steve Fewell

Routine:!addrop
Name: '?' and '!' address peek/poke operators
Starting Address: &98D1 for '!', or &98D3 (with A=0) for '?'
Entry criteria: PTR B points to next character in command line.
Exit: The IWA contains the address (&2A, &2B).
&2C contains 4 (if '!') or 0 (if '?') to indicate the variable return type at the address.
Carry is clear.

Description:

Push the Variable type (either 0 (for '?') or 4 (for '!')) to the Stack.
Increment the Text pointer B offset, to point to the next character of the command.
Get the integer value after the '?' or '!' operator and place the value in the IWA.
Note: As '!' and '?' are 'peek/poke' operators, the expression handler is not used, as the address value for the 'peek/poke' operation has the highest precedence.
Pop the variable type from the Stack and set &2C to the variable type (0 for '?' or 4 for '!').
Clear the carry flag and exit with A = #&FF and the IWA containing the address.

Disassembly for the '!' and '?' address peek/poke operators routine

98D1		169 004	A9 04	LDA#&04
98D3	H	072	48	PHA
98D4		230 027	E6 1B	INC &1B
98D6		032 180 150	20 B4 96	JSR &96B4 Get Integer value at PTR B
98D9	L	076 206 153	4C CE 99	JMP &99CE Set &2C and A=#&FF & exit with carry clear (numerical)

BASIC IV ROM Routines

Page Last Altered: undefined

98DC '$' address peek/poke operator

Submitted by Steve Fewell

Routine:$addrop
Name: '$' address peek/poke operator
Starting Address: &98DC
Entry criteria: PTR B points to next character in command line.
Exit: The IWA contains the address (&2A, &2B).
&2C contains #&80 to indicate that a string is the variable type at the address.
Carry is set.

Description:

Increment the Text pointer B offset, to point to the next character of the command.
Get the integer value after the '$' operator and place the value in the IWA.
Note: As '$' is a 'peek/poke' operator, the expression handler is not used, as the address value for the 'peek/poke' operation has the highest precedence.
BASIC does not allow an '$' function to be applied to a zero-page address, so if the addess MSB (in &2B) is zero then generate the '$ range' error.
Otherwise, set &2C (the variable type at the address) to #&80 (to indicate a String), set the carry flag (as the return value is a String) and exit.

Disassembly for the '$' address peek/poke operator routine

98DC		230 027	E6 1B	INC &1B
98DE		032 180 150	20 B4 96	JSR &96B4 Get Integer value at PTR B
98E1	+	165 043	A5 2B	LDA &2B
98E3		240 006	F0 06	BEQ 6 --> #&98EB $ range error
98E5		169 128	A9 80	LDA#&80
98E7	,	133 044	85 2C	STA &2C
98E9	8	056	38	SEC
98EA	`	096	60	RTS

BASIC IV ROM Routines

Page Last Altered: undefined

B278 BASIC error handler

Submitted by Steve Fewell

Description:

This code is executed everytime a BRK instruction (i.e. a BASIC, or OS, error message is issued), as the address in the BRK vector (&202-&203) is set to point to this code when the BASIC language is initiated.
Therefore, this routine will be executed when an error occurs within BASIC (i.e. 'Type mismatch', 'Mistake', 'Escape' and 'Missing #') or when an error occurs in the OS (or another ROM), i.e. 'Bad String', 'Bad command'.

On entry locations &FD-&FE point to the byte after the BRK instruction (i.e. the error number) and this number will be followed by the actual error message.

Set the OPT flag (location &28) to #&FF - ?
Reset the stack pointer to #&FF to discard any stack storage / routine return addresses that are no longer required (there aren't required as the Error handler will pass control to the ON ERROR code - and ignore the current position in the program).

Set X and Y to zero (as required by the OSBYTE calls below).
Call the Operating System OSBYTE call &DA to abandon any expected items that were still to be read by the VDU queue.
Call the Operating System OSBYTE call &7E to acknowledge the detection of an escape condition (this flushes the buffers and closes any open EXEC files).

Call routine &B237 to obtain the ERL value as follows:

* If BASIC Text pointer A offset (location &0A) is not zero then decrement it (so it points to the last character read)
* Call routine &9BBC to add the BASIC Text pointer A offset value (&0A) to the pointer address (&0B-&0C) and then reset the offset (&0A) to 1.
* Zero the ERL value (&08-&09) so that the default ERL value for the error condition is zero.
* Set &37-&38 to point to PAGE (the start of the program).
* If the BASIC Text Pointer MSB (&0C) value is #&07 then we are not executing a program (only the command line input) so exit the &B237 routine
* Set X to the BASIC Text pointer A LSB and Y to the BASIC Text pointer A MSB
* Next, obtain the Program ERL number as follows:
* [&B251] Read the next byte of the program at the location pointed to by &37-&38 & increment the pointer
* If the byte that was read was a return character ('<cr>') then:
    * Compare the &37-&38 pointer address with X and Y (the BASIC text pointer A address)
    * If the &37-&38 pointer has exceeded the X & Y position then exit the &B237 routine
    * Read the next character at the &37-&38 pointer and increment the pointer
    * ORA the character with #&00 (to set the flags) and if the negative flag is set then the end of program has been reached, so exit the &B237 routine (as we don't have an ERL number).
    * Store the character in location &09 - this is the MSB of the Error Line number (ERL)
    * Read the next character at the &37-&38 pointer and increment the pointer
    * Store the character in location &08 - this is the LSB of the Error Line number (ERL)
    * Read the next character at the &37-&38 pointer and increment the pointer
* If the BASIC Text pointer A address (in registers X and Y) is greater than the &37-&38 pointer then jump back to &BC51 to keep looking for the ERL value.

Store zero at location &20 (to turn off TRACE).
If the byte at the location pointed to by &FD-&FE (i.e. the error message number) is zero then the error is not trapable (i.e. it is 'No room' or 'STOP'), so call &B2A6 to reset the 'ON ERROR' pointer to the BASIC default (which displays the error message and ERL number and ENDs the program).

[&B296] Set BASIC text pointer A (&0B-&0C) to the BASIC code pointed to at the address of the 'ON ERROR' pointer (&16-&17) and zero the BASIC text pointer A offset (&0A).

Call routine &BBCF to initialise the BASIC environment - this sets the program start address, the Stack pointer, the *EDIT mode and the REPEAT/FOR/GOSUB control information.
Jump to &900B to execute the BASIC statement beginning at the BASIC text pointer A location.

Disassembly for the BASIC error handler routine

B237		164 010	A4 0A	LDY &0A	
B239		240 001	F0 01	BEQ 1 --> &B23C	
B23B		136	88	DEY	
B23C		032 188 155	20 BC 9B	JSR &9BBC Update BASIC Text pointer A (Add offset value & then reset offset to 1)	
B23F	d	100 008	64 08	STZ &08	
B241	d	100 009	64 09	STZ &09	
B243		166 024	A6 18	LDX &18	
B245	8	134 056	86 38	STX &38	
B247	d7	100 055	64 37	STZ &37	
B249		164 012	A4 0C	LDY &0C	
B24B		192 007	C0 07	CPY#&07	
B24D	(	240 040	F0 28	BEQ 40 --> &B277	
B24F		166 011	A6 0B	LDX &0B	
B251		032 160 141	20 A0 8D	JSR &8DA0 Read character pointed to by &37-&38 and then increment the &37-&38 pointer	
B254		201 013	C9 0D	CMP#&0D	
B256		208 024	D0 18	BNE 24 --> &B270	
B258	7	228 055	E4 37	CPX &37	
B25A		152	98	TYA	
B25B	8	229 056	E5 38	SBC &38	
B25D		144 024	90 18	BCC 24 --> &B277	
B25F		032 160 141	20 A0 8D	JSR &8DA0 Read character pointed to by &37-&38 and then increment the &37-&38 pointer	
B262		009 000	09 00	ORA#&00	
B264	0	048 017	30 11	BMI 17 --> &B277	
B266		133 009	85 09	STA &09	
B268		032 160 141	20 A0 8D	JSR &8DA0 Read character pointed to by &37-&38 and then increment the &37-&38 pointer	
B26B		133 008	85 08	STA &08	
B26D		032 160 141	20 A0 8D	JSR &8DA0	JSR &8DA0 Read character pointed to by &37-&38 and then increment the &37-&38 pointer
B270	7	228 055	E4 37	CPX &37	
B272		152	98	TYA	
B273	8	229 056	E5 38	SBC &38	
B275		176 218	B0 DA	BCS -38 --> &B251	
B277	`	096	60	RTS	
B278		162 255	A2 FF	LDX#&FF	

B27A	(	134 040	86 28	STX &28
B27C		154	9A	TXS
B27D		232	E8	INX
B27E		160 000	A0 00	LDY#&00
B280		169 218	A9 DA	LDA#&DA
B282		032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
B285	~	169 126	A9 7E	LDA#&7E
B287		032 244 255	20 F4 FF	JSR &FFF4 OSBYTE
B28A	7	032 055 178	20 37 B2	JSR &B237 Obtain the ERL value
B28D	d	100 032	64 20	STZ &20
B28F		178 253	B2 FD	LDA (&FD)
B291		208 003	D0 03	BNE 3 --> &B296
B293		032 166 178	20 A6 B2	JSR &B2A6 Reset 'ON ERROR' pointer to BASIC's default error handling routine
B296		165 022	A5 16	LDA &16
B298		133 011	85 0B	STA &0B
B29A		165 023	A5 17	LDA &17
B29C		133 012	85 0C	STA &0C
B29E	d	100 010	64 0A	STZ &0A
B2A0		032 207 187	20 CF BB	JSR &BBCF Initialise Program start address, Stack pointer, *EDIT mode & REPEAT/FOR/GOSUB levels
B2A3	L	076 011 144	4C 0B 90	JMP &900B Process next BASIC program statement

---

BASIC IV ROM Routines

Memory usage

Submitted by Steve Fewell

Page Last Altered: undefined

This page contains a listing of the main memory locations used by the BASIC ROM. Some of the addresses listed on this page will have multiple uses - in this case, only the most common usage will be given.

The listing is in address order (zero page first):

Address (in hex)	Description of usage
00	LOMEM [Pointer to start of BASIC variables] LSB
01	LOMEM [Pointer to start of BASIC variables] MSB
02	VARTOP [Pointer to end of BASIC variables] LSB
03	VARTOP [Pointer to end of BASIC variables] MSB
04	BASIC Stack Pointer [Pointer to latest entry in BASIC Stack] LSB
05	BASIC Stack Pointer [Pointer to latest entry in BASIC Stack] MSB
06	HIMEM [Pointer to start of memory-mapped screen area] LSB
07	HIMEM [Pointer to start of memory-mapped screen area] MSB
08	ERL [Address of the BASIC line in which the last error occurred] LSB
09	ERL [Address of the BASIC line in which the last error occurred] MSB

0A	BASIC Text Pointer offset (offset to the current byte being processed from the address pointed to by (&B, &C))
0B	BASIC Text Pointer LSB [Pointer to start of BASIC text line]
0C	BASIC Text Pointer MSB [Pointer to start of BASIC text line]
0D	Random Number working area
0E	Random Number working area
0F	Random Number working area
10	Random Number working area
11	Random Number working area
12	TOP [Pointer to the end of the BASIC program - not including variables] LSB
13	TOP [Pointer to the end of the BASIC program - not including variables] MSB
14	PRINT BYTES - The number of bytes to include in a print output field
15	PRINT FLAG - 0 = Decimal, N = Hexadecimal
16	Pointer to BASIC Error routine code [either the default, or the code which occurs after an ON ERROR statement] LSB
17	Pointer to BASIC Error routine code [either the default, or the code which occurs after an ON ERROR statement] MSB The default value of (&16, &17) is &B2AF.
18	PAGE (DIV by 256). The page number where the BASIC program starts. The page offset is always 00.
19	BASIC Second Text Pointer LSB [Pointer to start of BASIC text line]
1A	BASIC Second Text Pointer MSB [Pointer to start of BASIC text line]
1B	BASIC Second Text Pointer offset (offset to the current byte being processed from the address pointed to by (&19, &1A))
1C	BASIC PROGRAM START [Pointer to start of BASIC program] LSB (when program is not running) / Next DATA Location LSB (when program is running)
1D	BASIC PROGRAM START [Pointer to start of BASIC program] MSB (when program is not running) / Next DATA Location MSB (when program is running)
1E	COUNT - The number of bytes printed since the last new line
1F	LISTO flag (0 = off, 1 = a space after each line number, 2 = Indent FOR, 4 = Indent REPEAT), the top bit is also set for *EDIT mode
20	TRACE flag (0 = off, 1 = on)
21	Maximum TRACE LINE number (LSB)
22	Maximum TRACE LINE number (MSB)
23	WIDTH
24	Current REPEAT level

25	Current GOSUB level
26	15* Current FOR level
27	Variable type (00 = byte, 04 = integer, 05 = floating-point, &81 = String, &A4 = Function, &F2 = Procedure)
28	OPT flag (0 - list, 1 - errors, 2 = relocate)
29	
2A	IWA - Integer working area -> LSB
2B	IWA - Integer working area -> byte 2
2C	IWA - Integer working area -> byte 3
2D	IWA - Integer working area -> MSB
2E	FWA - Floating-point working area A -> Sign byte
2F	FWA - Floating-point working area A -> Exponent overflow byte
30	FWA - Floating-point working area A -> Exponent byte
31	FWA - Floating-point working area A -> Mantissa byte 1
32	FWA - Floating-point working area A -> Mantissa byte 2
33	FWA - Floating-point working area A -> Mantissa byte 3
34	FWA - Floating-point working area A -> Mantissa byte 4
35	FWA - Floating-point working area A -> Mantissa rounding byte
36	Current length of String in the String working area buffer (&600 - &6FF)
37	Pointer to next BASIC character [LSB] Also: Pointer to Integer variable.
38	Pointer to next BASIC character [MSB] Also: Pointer to Integer variable.
39	
3A	
3B	FWB - Floating-point working area B -> Sign byte
3C	FWB - Floating-point working area B -> Exponent byte
3D	FWB - Floating-point working area B -> Mantissa byte 1
3E	FWB - Floating-point working area B -> Mantissa byte 2
3F	FWB - Floating-point working area B -> Mantissa byte 3
40	FWB - Floating-point working area B -> Mantissa byte 4
41	FWB - Floating-point working area B -> Mantissa rounding byte
42	

43	
44	
45	
46	
47	Number of decimal points found in number [used by ASCII to binary number routine]
48	
49	
4A	argp - pointer to Floating-Point variable LSB
4B	argp - pointer to Floating-Point variable MSB
4C	
4D	
4E	
4F	
50	
51	
52	
53	
54	
55	
56	
57	
58	
59	
5A	
5B	
5C	
5D	
5E	
5F	
60	

61
62
63
64
65
66
67
68
69
6A
6B
6C
6D
6E
6F
70
71
72
73
74
75
76
77
78
79
7A
7B
7C
7D
7E

7F	
...	
F9	
FA	
FB	
FC	
FD	MOS Error Message pointer LSB [Error Number, Error String]
FE	MOS Error Message pointer MSB [Error Number, Error String]
FF	MOS Error Byte (Top bit set if error occurred?)
...	
	The following locations contain the values that BASIC assigns to any MOS vectors:
0202-0203	BASIC assigns the address &B278 to the BRK vector, as this points to the BASIC ROM code which BASIC wants to be executed every time an error condition occurs (generated by a BRK command in assembly language).
...	
	The following locations contain the values of the resident integer variables:
400-403	@% value LSB first
404-407	A% value LSB first
408-40B	B% value LSB first
40C-40F	C% value LSB first
410-413	D% value LSB first
414-417	E% value LSB first
418-41B	F% value LSB first
41C-41F	G% value LSB first
420-423	H% value LSB first
424-427	I% value LSB first
428-42B	J% value LSB first
42C-42F	K% value LSB first
430-433	L% value LSB first
434-437	M% value LSB first

438-43B	N% value LSB first
43C-43F	O% value LSB first
440-443	P% value LSB first
444-447	Q% value LSB first
448-44B	R% value LSB first
44C-44F	S% value LSB first
450-453	T% value LSB first
454-457	U% value LSB first
458-45B	V% value LSB first
45C-45F	W% value LSB first
460-463	X% value LSB first
464-467	Y% value LSB first
468-46B	Z% value LSB first
	The following locations contain the values of some temporary Floating-Point variables, in packed (variable) form:
46C-470	Temporary Floating-Point variable area 1
471-475	Temporary Floating-Point variable area 2
476-47A	Temporary Floating-Point variable area 3
47B-47F	Temporary Floating-Point variable area 4

The following locations contain the variable pointer tables.
[from Colin Pharo] There is a variable look-up table for each character with which a variable name in BASIC can start. Each pair of addresses is a pointer (low byte, high byte) to the variables whose names start with that particular character.

The "variable look-up" table pointed to by each set of 2-bytes is stored in the variable area above the program TOP and has the following format:
* 2-byte pointer to the next variable (Low byte, high byte) [&0000 specifies the last variable in the table]
* The name of the variable (without the first character - which is already known, as it was used to find this table)
* End of name marker -> &00
* Variable details/variable contents.

The variable details/variable contents have the following formats:
* for Integer variables -->The value of the integer (in 4-byte format)
* for Floating-Point variables -->The value of the Floating-Point number (in 5-byte packed floating-point format)
* for String variables --> A pointer to the value of the string [Low byte, high byte], followed by the maximum size allocated to the string

	[1 byte], and the current size of the string [1 byte]. Note: The String value is stored without the &00 terminator byte. * for Arrays --> 2 * The number of dimensions + 1 [1 byte], The number of elements in the first dimension, The number of elements in the second dimension. Followed by the contents of each array element (in the format described above [That is 4-bytes for Integer, 5-bytes for Floating-point number or 4-bytes for String]).
480-481	Variable Pointer table address [Low byte, high byte] for variables beginning with @
482-483	Variable Pointer table address [Low byte, high byte] for variables beginning with A
484-485	Variable Pointer table address [Low byte, high byte] for variables beginning with B
486-487	Variable Pointer table address [Low byte, high byte] for variables beginning with C
488-489	Variable Pointer table address [Low byte, high byte] for variables beginning with D
48A-48B	Variable Pointer table address [Low byte, high byte] for variables beginning with E
48C-48D	Variable Pointer table address [Low byte, high byte] for variables beginning with F
48E-48F	Variable Pointer table address [Low byte, high byte] for variables beginning with G
490-491	Variable Pointer table address [Low byte, high byte] for variables beginning with H
492-493	Variable Pointer table address [Low byte, high byte] for variables beginning with I
494-495	Variable Pointer table address [Low byte, high byte] for variables beginning with J
496-497	Variable Pointer table address [Low byte, high byte] for variables beginning with K
498-499	Variable Pointer table address [Low byte, high byte] for variables beginning with L
49A-49B	Variable Pointer table address [Low byte, high byte] for variables beginning with M
49C-49D	Variable Pointer table address [Low byte, high byte] for variables beginning with N
49E-49F	Variable Pointer table address [Low byte, high byte] for variables beginning with O
4A0-4A1	Variable Pointer table address [Low byte, high byte] for variables beginning with P
4A2-4A3	Variable Pointer table address [Low byte, high byte] for variables beginning with Q
4A4-4A5	Variable Pointer table address [Low byte, high byte] for variables beginning with R
4A6-4A7	Variable Pointer table address [Low byte, high byte] for variables beginning with S
4A8-4A9	Variable Pointer table address [Low byte, high byte] for variables beginning with T
4AA-4AB	Variable Pointer table address [Low byte, high byte] for variables beginning with U
4AC-4AD	Variable Pointer table address [Low byte, high byte] for variables beginning with V
4AE-4AF	Variable Pointer table address [Low byte, high byte] for variables beginning with W
4B0-4B1	Variable Pointer table address [Low byte, high byte] for variables beginning with X
4B2-4B3	Variable Pointer table address [Low byte, high byte] for variables beginning with Y
4B4-4B5	Variable Pointer table address [Low byte, high byte] for variables beginning with Z

4B6-4B7	Variable Pointer table address [Low byte, high byte] for variables beginning with [
4B8-4B9	Variable Pointer table address [Low byte, high byte] for variables beginning with \
4BA-4BB	Variable Pointer table address [Low byte, high byte] for variables beginning with]
4BC-4BD	Variable Pointer table address [Low byte, high byte] for variables beginning with ^
4BE-4BF	Variable Pointer table address [Low byte, high byte] for variables beginning with _
4C0-4C1	Variable Pointer table address [Low byte, high byte] for variables beginning with '
4C2-4C3	Variable Pointer table address [Low byte, high byte] for variables beginning with a
4C4-4C5	Variable Pointer table address [Low byte, high byte] for variables beginning with b
4C6-4C7	Variable Pointer table address [Low byte, high byte] for variables beginning with c
4C8-4C9	Variable Pointer table address [Low byte, high byte] for variables beginning with d
4CA-4CB	Variable Pointer table address [Low byte, high byte] for variables beginning with e
4CC-4CD	Variable Pointer table address [Low byte, high byte] for variables beginning with f
4CE-4CF	Variable Pointer table address [Low byte, high byte] for variables beginning with g
4D0-4D1	Variable Pointer table address [Low byte, high byte] for variables beginning with h
4D2-4D3	Variable Pointer table address [Low byte, high byte] for variables beginning with i
4D4-4D5	Variable Pointer table address [Low byte, high byte] for variables beginning with j
4D6-4D7	Variable Pointer table address [Low byte, high byte] for variables beginning with k
4D8-4D9	Variable Pointer table address [Low byte, high byte] for variables beginning with l
4DA-4DB	Variable Pointer table address [Low byte, high byte] for variables beginning with m
4DC-4DD	Variable Pointer table address [Low byte, high byte] for variables beginning with n
4DE-4DF	Variable Pointer table address [Low byte, high byte] for variables beginning with o
4E0-4E1	Variable Pointer table address [Low byte, high byte] for variables beginning with p
4E2-4E3	Variable Pointer table address [Low byte, high byte] for variables beginning with q
4E4-4E5	Variable Pointer table address [Low byte, high byte] for variables beginning with r
4E6-4E7	Variable Pointer table address [Low byte, high byte] for variables beginning with s
4E8-4E9	Variable Pointer table address [Low byte, high byte] for variables beginning with t
4EA-4EB	Variable Pointer table address [Low byte, high byte] for variables beginning with u
4EC-4ED	Variable Pointer table address [Low byte, high byte] for variables beginning with v
4EE-4EF	Variable Pointer table address [Low byte, high byte] for variables beginning with w
4F0-4F1	Variable Pointer table address [Low byte, high byte] for variables beginning with x

4F2-4F3	Variable Pointer table address [Low byte, high byte] for variables beginning with y
4F4-4F5	Variable Pointer table address [Low byte, high byte] for variables beginning with z
4F6-4F7	Variable Pointer table address [Low byte, high byte] for Procedures
4F8-4F9	Variable Pointer table address [Low byte, high byte] for Functions
04FA	
04FB	
04FC	
04FD	
04FE	
04FF	
500-527	The REPEAT Start address Stack (&0500-&0513 is the start address LSB stack and &0514-&0527 is the start address MSB stack)
	The FOR Loop Stack A maximum of 10 FOR loop details can be stored. Each FOR block is 15 bytes long. The 1st FOR block (&0528-&0536) has the following layout: Byte 1 (&0528) is the variable location - LSB. Byte 2 (&0529) is the variable location - MSB Byte 3 (&052A) is the variable type Byte 4 (&052B) - Byte 8 (&052F) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&0530) - Byte 13 (&0534) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&0535) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&0536) is the program location of the start of the first statement within the FOR loop - MSB The 2nd FOR block (&0537-&0545) has the following layout: Byte 1 (&0537) is the variable location - LSB. Byte 2 (&0538) is the variable location - MSB Byte 3 (&0539) is the variable type Byte 4 (&053A) - Byte 8 (&053E) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&053F) - Byte 13 (&0543) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&0544) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&0545) is the program location of the start of the first statement within the FOR loop - MSB The 3rd FOR block (&0546-&0554) has the following layout: Byte 1 (&0546) is the variable location - LSB. Byte 2 (&0547) is the variable location - MSB

528-5BD	Byte 3 (&0548) is the variable type Byte 4 (&0549) - Byte 8 (&054D) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&054E) - Byte 13 (&0552) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&0553) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&0554) is the program location of the start of the first statement within the FOR loop - MSB The 4th FOR block (&0555-&0563) has the following layout: Byte 1 (&0555) is the variable location - LSB. Byte 2 (&0556) is the variable location - MSB Byte 3 (&0557) is the variable type Byte 4 (&0558) - Byte 8 (&055C) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&055D) - Byte 13 (&0561) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&0562) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&0563) is the program location of the start of the first statement within the FOR loop - MSB The 5th FOR block (&0564-&0572) has the following layout: Byte 1 (&0564) is the variable location - LSB. Byte 2 (&0565) is the variable location - MSB Byte 3 (&0566) is the variable type Byte 4 (&0567) - Byte 8 (&056B) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&056C) - Byte 13 (&0570) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&0571) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&0572) is the program location of the start of the first statement within the FOR loop - MSB The 6th FOR block (&0573-&0581) has the following layout: Byte 1 (&0573) is the variable location - LSB. Byte 2 (&0574) is the variable location - MSB Byte 3 (&0575) is the variable type Byte 4 (&0576) - Byte 8 (&057A) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&057B) - Byte 13 (&057F) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&0580) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&0581) is the program location of the start of the first statement within the FOR loop - MSB The 7th FOR block (&0582-&0590) has the following layout: Byte 1 (&0582) is the variable location - LSB. Byte 2 (&0583) is the variable location - MSB Byte 3 (&0584) is the variable type Byte 4 (&0585) - Byte 8 (&0589) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&058A) - Byte 13 (&058E) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&058F) is the program location of the start of the first statement within the FOR loop - LSB

	Byte 15 (&0590) is the program location of the start of the first statement within the FOR loop - MSB The 8th FOR block (&0591-&059F) has the following layout: Byte 1 (&0591) is the variable location - LSB. Byte 2 (&0592) is the variable location - MSB Byte 3 (&0593) is the variable type Byte 4 (&0594) - Byte 8 (&0598) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&0599) - Byte 13 (&059D) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&059E) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&059F) is the program location of the start of the first statement within the FOR loop - MSB The 9th FOR block (&05A0-&05AE) has the following layout: Byte 1 (&05A0) is the variable location - LSB. Byte 2 (&05A1) is the variable location - MSB Byte 3 (&05A2) is the variable type Byte 4 (&05A3) - Byte 8 (&05A7) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&05A8) - Byte 13 (&05AC) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&05AD) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&05AE) is the program location of the start of the first statement within the FOR loop - MSB The 10th FOR block (&05AF-&05BD) has the following layout: Byte 1 (&05AF) is the variable location - LSB. Byte 2 (&05B0) is the variable location - MSB Byte 3 (&05B1) is the variable type Byte 4 (&05B2) - Byte 8 (&05B6) is the STEP value (Float (5-byte) / Integer (4-byte)) Byte 9 (&05B7) - Byte 13 (&05BB) is the TO value (Float (5-byte) / Integer (4-byte)) Byte 14 (&05BC) is the program location of the start of the first statement within the FOR loop - LSB Byte 15 (&05BD) is the program location of the start of the first statement within the FOR loop - MSB
5BE-5CB	Unused These 14 bytes are unused.
5CC-5FF	The GOSUB Return address Stack (&05CC-&05E5 is the return address LSB stack and &05E6-&05FF is the return address MSB stack)
600-6FF	SWA - String working area/buffer [also: call parameter block]
700-7EF	BASIC Line Input buffer

	The following address are pointers [Low Byte, High byte] to specific floating-point routines in the BASIC ROM. BASIC provides these addresses as fixed external pointers to these often used routines within the BASIC ROM. The details in addresses &07F0 to &07FF are exactly the same as those in the BASIC4 ROM locations &BF14 to &BF23.
7F0-7F1	Contains &A7B8 which is the address of the Floating-Point SQR function [FWA = SQR(FWA)]
7F2-7F3	Contains &A5EE which is the address of the Floating-Point / function [FWA = {argp}/ FWA]
7F4-7F5	Contains &A6A6 which is the address of the Floating-Point * function [FWA = {argp}* FWA]
7F6-7F7	Contains &A68D which is the address of the Floating-Point + function [FWA = {argp}+ FWA]
7F8-7F9	Contains &ACCA which is the address of the Floating-Point Compliment function [FWA = -FWA]
7FA-7FB	Contains &A541 which is the address of the Floating-Point Load function [FWA = {argp}]
7FC-7FD	Contains &A519 which is the address of the Floating-Point Store function [{argp}= FWA]
07FE	Contains &4A which is the start of the zero page location of argp (a pointer to a floating-point variable)
07FF	Contains &2E which is the start of the zero page location for the FWA (Floating-Point working area A)

BASIC IV ROM Routines

Disassembly 8000 to 9000

8000		201 001	C9 01	CMP#&01
8002	'	240 039	F0 27	BEQ 39 --> &802B
8004	`	096	60	RTS
8005		234	EA	NOP
8006	`	096	60	RTS
8007	B	014 004 066	0E 04 42	ASL &4204
800A	AS	065 083	41 53	EOR (&53,X)
800C	IC	073 067	49 43	EOR#&43
800E		000	00	BRK
800F	(	040	28	PLP
8010	C	067	43	xxx Invalid Code
8011	)1	041 049	29 31	AND#&31
8013	984	057 056 052	39 38 34	AND &3438,Y
8016	Ac	032 065 099	20 41 63	JSR &6341
8019	o	111	6F	xxx Invalid Code

801A	rn	114 110	72 6E	ADC (&6E)
801C		010	0A	ASL A
801D		013 000 000	0D 00 00	ORA &0000
8020		128 000	80 00	BRA 0 --> &8022
8022		000	00	BRK
8023		000	00	BRK
8024		003	03	xxx Invalid Code
8025	'	039	27	xxx Invalid Code
8026		001 010	01 0A	ORA (&0A,X)
8028	d	100 232	64 E8	STZ &E8
802A		016	10	xxx Invalid Code
802B	%	037 017	25 11	AND &11
802D		005 013	05 0D	ORA &0D
802F		005 014	05 0E	ORA &0E
8031		005 015	05 0F	ORA &0F
8033		005 016	05 10	ORA &10
8035		208 012	D0 0C	BNE 12 --> &8043
8037	A	169 065	A9 41	LDA#&41
8039		133 013	85 0D	STA &0D
803B	I	073 019	49 13	EOR#&13
803D		133 014	85 0E	STA &0E
803F	I	073 005	49 05	EOR#&05
8041		133 015	85 0F	STA &0F
8043		169 132	A9 84	LDA#&84
8045		032 244 255	20 F4 FF	JSR &FFF4
8048		134 006	86 06	STX &06
804A		132 007	84 07	STY &07
804C	:	058	3A	DEC A
804D		032 244 255	20 F4 FF	JSR &FFF4
8050		132 024	84 18	STY &18
8052	d	100 031	64 1F	STZ &1F
8054		156 002 004	9C 02 04	STZ &0402
8057		156 003 004	9C 03 04	STZ &0403
805A		162 255	A2 FF	LDX#&FF
805C	#	134 035	86 23	STX &23

```
805E        162 010       A2 0A       LDX#&0A
8060        142 000 004   8E 00 04    STX &0400
8063        202           CA          DEX
8064        142 001 004   8E 01 04    STX &0401
8067  x     169 120       A9 78       LDA#&78
8069        141 002 002   8D 02 02    STA &0202
806C        169 178       A9 B2       LDA#&B2
806E        141 003 002   8D 03 02    STA &0203
8071  X     088           58          CLI
8072  L-    076 045 143   4C 2D 8F    JMP &8F2D
8075  9     132 057       84 39       STY &39
8077        160 001       A0 01       LDY#&01
8079  7     177 055       B1 37       LDA (&37),Y
807B        160 246       A0 F6       LDY#&F6
807D        201 242       C9 F2       CMP#&F2
807F        240 012       F0 0C       BEQ 12 --> &808D
8081        160 248       A0 F8       LDY#&F8
8083        128 008       80 08       BRA 8 --> &808D
8085  9     132 057       84 39       STY &39
8087        160 001       A0 01       LDY#&01
8089  7     177 055       B1 37       LDA (&37),Y
808B        010           0A          ASL A
808C        168           A8          TAY
808D        185 001 004   B9 01 04    LDA &0401,Y
8090  :     240 058       F0 3A       BEQ 58 --> &80CC
8092  +     133 043       85 2B       STA &2B
8094        185 000 004   B9 00 04    LDA &0400,Y
8097        128 011       80 0B       BRA 11 --> &80A4
8099        160 001       A0 01       LDY#&01
809B  *     177 042       B1 2A       LDA (&2A),Y
809D  -     240 045       F0 2D       BEQ 45 --> &80CC
809F        168           A8          TAY
80A0  *     178 042       B2 2A       LDA (&2A)
80A2  +     132 043       84 2B       STY &2B
80A4  *     133 042       85 2A       STA &2A
```

```
80A6        160 002     A0 02     LDY#&02
80A8  *     177 042     B1 2A     LDA (&2A),Y
80AA        208 010     D0 0A     BNE 10 --> &80B6
80AC  9     196 057     C4 39     CPY &39
80AE        208 233     D0 E9     BNE -23 --> &8099
80B0        128 017     80 11     BRA 17 --> &80C3
80B2  *     177 042     B1 2A     LDA (&2A),Y
80B4        240 227     F0 E3     BEQ -29 --> &8099
80B6  7     209 055     D1 37     CMP (&37),Y
80B8        208 223     D0 DF     BNE -33 --> &8099
80BA        200         C8        INY
80BB  9     196 057     C4 39     CPY &39
80BD        208 243     D0 F3     BNE -13 --> &80B2
80BF  *     177 042     B1 2A     LDA (&2A),Y
80C1        208 214     D0 D6     BNE -42 --> &8099
80C3        152         98        TYA
80C4  e*    101 042     65 2A     ADC &2A
80C6  *     133 042     85 2A     STA &2A
80C8        144 002     90 02     BCC 2 --> &80CC
80CA  +     230 043     E6 2B     INC &2B
80CC  `     096         60        RTS
80CD  d=    100 061     64 3D     STZ &3D
80CF        165 024     A5 18     LDA &18
80D1  >     133 062     85 3E     STA &3E
80D3        160 001     A0 01     LDY#&01
80D5  =     177 061     B1 3D     LDA (&3D),Y
80D7  +     197 043     C5 2B     CMP &2B
80D9        176 014     B0 0E     BCS 14 --> &80E9
80DB        160 003     A0 03     LDY#&03
80DD  =     177 061     B1 3D     LDA (&3D),Y
80DF  e=    101 061     65 3D     ADC &3D
80E1  =     133 061     85 3D     STA &3D
80E3        144 238     90 EE     BCC -18 --> &80D3
80E5  >     230 062     E6 3E     INC &3E
80E7        128 234     80 EA     BRA -22 --> &80D3
```

80E9		208 010	D0 0A	BNE 10 --> &80F5
80EB		200	C8	INY
80EC	=	177 061	B1 3D	LDA (&3D),Y
80EE	*	197 042	C5 2A	CMP &2A
80F0		144 233	90 E9	BCC -23 --> &80DB
80F2		208 001	D0 01	BNE 1 --> &80F5
80F4	`	096	60	RTS
80F5		160 002	A0 02	LDY#&02
80F7		024	18	CLC
80F8	`	096	60	RTS
80F9		032 190 150	20 BE 96	JSR &96BE
80FC	-	165 045	A5 2D	LDA &2D
80FE	H	072	48	PHA
80FF		032 190 172	20 BE AC	JSR &ACBE
8102		032 015 160	20 0F A0	JSR &A00F
8105	'	134 039	86 27	STX &27
8107		032 190 150	20 BE 96	JSR &96BE
810A	h	104	68	PLA
810B	8	133 056	85 38	STA &38
810D	E-	069 045	45 2D	EOR &2D
810F	7	133 055	85 37	STA &37
8111		032 190 172	20 BE AC	JSR &ACBE
8114	9	162 057	A2 39	LDX#&39
8116		032 008 189	20 08 BD	JSR &BD08
8119	d=	100 061	64 3D	STZ &3D
811B	d>	100 062	64 3E	STZ &3E
811D	d?	100 063	64 3F	STZ &3F
811F	d@	100 064	64 40	STZ &40
8121	-	165 045	A5 2D	LDA &2D
8123	*	005 042	05 2A	ORA &2A
8125	+	005 043	05 2B	ORA &2B
8127	,	005 044	05 2C	ORA &2C
8129	G	240 071	F0 47	BEQ 71 --> &8172
812B		160 032	A0 20	LDY#&20
812D		136	88	DEY

812E	A	240 065	F0 41	BEQ 65 --> &8171
8130	9	006 057	06 39	ASL &39
8132	&:	038 058	26 3A	ROL &3A
8134	&;	038 059	26 3B	ROL &3B
8136	&<	038 060	26 3C	ROL &3C
8138		016 243	10 F3	BPL -13 --> &812D
813A	&9	038 057	26 39	ROL &39
813C	&:	038 058	26 3A	ROL &3A
813E	&;	038 059	26 3B	ROL &3B
8140	&<	038 060	26 3C	ROL &3C
8142	&=	038 061	26 3D	ROL &3D
8144	&>	038 062	26 3E	ROL &3E
8146	&?	038 063	26 3F	ROL &3F
8148	&@	038 064	26 40	ROL &40
814A	8	056	38	SEC
814B	=	165 061	A5 3D	LDA &3D
814D	*	229 042	E5 2A	SBC &2A
814F	H	072	48	PHA
8150	>	165 062	A5 3E	LDA &3E
8152	+	229 043	E5 2B	SBC &2B
8154	H	072	48	PHA
8155	?	165 063	A5 3F	LDA &3F
8157	,	229 044	E5 2C	SBC &2C
8159		170	AA	TAX
815A	@	165 064	A5 40	LDA &40
815C	-	229 045	E5 2D	SBC &2D
815E		144 012	90 0C	BCC 12 --> &816C
8160	@	133 064	85 40	STA &40
8162	?	134 063	86 3F	STX &3F
8164	h	104	68	PLA
8165	>	133 062	85 3E	STA &3E
8167	h	104	68	PLA
8168	=	133 061	85 3D	STA &3D
816A		176 002	B0 02	BCS 2 --> &816E
816C	h	104	68	PLA

816D	h	104	68	PLA
816E		136	88	DEY
816F		208 201	D0 C9	BNE -55 --> &813A
8171	`	096	60	RTS
8172		000	00	BRK
8173		018	12	EQUB &12
8174	D	068	44	"D"
8175	i	105	69	"i"
8176	v	118	76	"v"
8177	i	105	69	"i"
8178	s	115	73	"s"
8179	i	105	69	"i"
817A	o	111	6F	"o"
817B	n	110	6E	"n"
817C		032	20	" "
817D	b	098	62	"b"
817E	y	121	79	"y"
817F		032	20	" "
8180	z	122	7A	"z"
8181	e	101	65	"e"
8182	r	114	72	"r"
8183	o	111	6F	"o"
8184		000	00	EQUB &00
8185	d5	100 053	64 35	STZ &35
8187	d/	100 047	64 2F	STZ &2F
8189	-	165 045	A5 2D	LDA &2D
818B	.	133 046	85 2E	STA &2E
818D		016 005	10 05	BPL 5 --> &8194
818F		032 222 172	20 DE AC	JSR &ACDE
8192	-	165 045	A5 2D	LDA &2D
8194	&	208 038	D0 26	BNE 38 --> &81BC
8196	d4	100 052	64 34	STZ &34
8198	,	165 044	A5 2C	LDA &2C
819A		208 020	D0 14	BNE 20 --> &81B0
819C	d3	100 051	64 33	STZ &33

819E	+	165 043	A5 2B	LDA &2B
81A0		208 006	D0 06	BNE 6 --> &81A8
81A2	d2	100 050	64 32	STZ &32
81A4	*	165 042	A5 2A	LDA &2A
81A6	8	128 056	80 38	BRA 56 --> &81E0
81A8	*	164 042	A4 2A	LDY &2A
81AA	2	132 050	84 32	STY &32
81AC		160 144	A0 90	LDY#&90
81AE	2	128 050	80 32	BRA 50 --> &81E2
81B0	+	164 043	A4 2B	LDY &2B
81B2	2	132 050	84 32	STY &32
81B4	*	164 042	A4 2A	LDY &2A
81B6	3	132 051	84 33	STY &33
81B8		160 152	A0 98	LDY#&98
81BA	&	128 038	80 26	BRA 38 --> &81E2
81BC	,	164 044	A4 2C	LDY &2C
81BE	2	132 050	84 32	STY &32
81C0	+	164 043	A4 2B	LDY &2B
81C2	3	132 051	84 33	STY &33
81C4	*	164 042	A4 2A	LDY &2A
81C6	4	132 052	84 34	STY &34
81C8		160 160	A0 A0	LDY#&A0
81CA		128 022	80 16	BRA 22 --> &81E2
81CC	d.	100 046	64 2E	STZ &2E
81CE	d0	100 048	64 30	STZ &30
81D0	d/	100 047	64 2F	STZ &2F
81D2	d1	100 049	64 31	STZ &31
81D4	`	096	60	RTS
81D5		032 180 166	20 B4 A6	JSR &A6B4
81D8		168	A8	TAY
81D9		016 005	10 05	BPL 5 --> &81E0
81DB	.	133 046	85 2E	STA &2E
81DD	I	073 255	49 FF	EOR#&FF
81DF		026	1A	INC A
81E0		160 136	A0 88	LDY#&88

81E2		009 000	09 00	ORA#&00
81E4	0	048 012	30 0C	BMI 12 --> &81F2
81E6		240 228	F0 E4	BEQ -28 --> &81CC
81E8		136	88	DEY
81E9	4	006 052	06 34	ASL &34
81EB	&3	038 051	26 33	ROL &33
81ED	&2	038 050	26 32	ROL &32
81EF	*	042	2A	ROL A
81F0		016 246	10 F6	BPL -10 --> &81E8
81F2	1	133 049	85 31	STA &31
81F4	0	132 048	84 30	STY &30
81F6	`	096	60	RTS
81F7	1	165 049	A5 31	LDA &31
81F9	0	048 217	30 D9	BMI -39 --> &81D4
81FB	-	208 045	D0 2D	BNE 45 --> &822A
81FD	2	005 050	05 32	ORA &32
81FF	3	005 051	05 33	ORA &33
8201	4	005 052	05 34	ORA &34
8203	5	005 053	05 35	ORA &35
8205		240 197	F0 C5	BEQ -59 --> &81CC
8207	0	165 048	A5 30	LDA &30
8209	2	164 050	A4 32	LDY &32
820B	1	132 049	84 31	STY &31
820D	3	164 051	A4 33	LDY &33
820F	2	132 050	84 32	STY &32
8211	4	164 052	A4 34	LDY &34
8213	3	132 051	84 33	STY &33
8215	5	164 053	A4 35	LDY &35
8217	4	132 052	84 34	STY &34
8219	d5	100 053	64 35	STZ &35
821B	8	056	38	SEC
821C		233 008	E9 08	SBC#&08
821E		176 002	B0 02	BCS 2 --> &8222
8220	/	198 047	C6 2F	DEC &2F
8222	1	164 049	A4 31	LDY &31

8224		240 227	F0 E3	BEQ -29 --> &8209
8226	0	048 023	30 17	BMI 23 --> &823F
8228		128 002	80 02	BRA 2 --> &822C
822A	0	165 048	A5 30	LDA &30
822C		024	18	CLC
822D		233 000	E9 00	SBC#&00
822F		176 002	B0 02	BCS 2 --> &8233
8231	/	198 047	C6 2F	DEC &2F
8233	5	006 053	06 35	ASL &35
8235	&4	038 052	26 34	ROL &34
8237	&3	038 051	26 33	ROL &33
8239	&2	038 050	26 32	ROL &32
823B	&1	038 049	26 31	ROL &31
823D		016 238	10 EE	BPL -18 --> &822D
823F	0	133 048	85 30	STA &30
8241	`	096	60	RTS
8242	0	165 048	A5 30	LDA &30
8244	,	016 044	10 2C	BPL 44 --> &8272
8246	1	164 049	A4 31	LDY &31
8248	4	240 052	F0 34	BEQ 52 --> &827E
824A	F1	070 049	46 31	LSR &31
824C	f2	102 050	66 32	ROR &32
824E	f3	102 051	66 33	ROR &33
8250	f4	102 052	66 34	ROR &34
8252		026	1A	INC A
8253	h	240 104	F0 68	BEQ 104 --> &82BD
8255		201 160	C9 A0	CMP#&A0
8257	g	176 103	B0 67	BCS 103 --> &82C0
8259		201 153	C9 99	CMP#&99
825B		176 237	B0 ED	BCS -19 --> &824A
825D	i	105 008	69 08	ADC#&08
825F	3	164 051	A4 33	LDY &33
8261	4	132 052	84 34	STY &34
8263	2	164 050	A4 32	LDY &32
8265	3	132 051	84 33	STY &33

8267	1	164 049	A4 31	LDY &31
8269	2	132 050	84 32	STY &32
826B	d1	100 049	64 31	STZ &31
826D		128 230	80 E6	BRA -26 --> &8255
826F		032 011 164	20 0B A4	JSR &A40B
8272	L	076 180 166	4C B4 A6	JMP &A6B4
8275	0	165 048	A5 30	LDA &30
8277		016 246	10 F6	BPL -10 --> &826F
8279	p	032 112 165	20 70 A5	JSR &A570
827C	1	164 049	A4 31	LDY &31
827E	D	240 068	F0 44	BEQ 68 --> &82C4
8280	F1	070 049	46 31	LSR &31
8282	f2	102 050	66 32	ROR &32
8284	f3	102 051	66 33	ROR &33
8286	f4	102 052	66 34	ROR &34
8288	f=	102 061	66 3D	ROR &3D
828A	f>	102 062	66 3E	ROR &3E
828C	f?	102 063	66 3F	ROR &3F
828E	f@	102 064	66 40	ROR &40
8290		026	1A	INC A
8291	*	240 042	F0 2A	BEQ 42 --> &82BD
8293		201 160	C9 A0	CMP#&A0
8295	)	176 041	B0 29	BCS 41 --> &82C0
8297		201 153	C9 99	CMP#&99
8299		176 229	B0 E5	BCS -27 --> &8280
829B	i	105 008	69 08	ADC#&08
829D	?	164 063	A4 3F	LDY &3F
829F	@	132 064	84 40	STY &40
82A1	>	164 062	A4 3E	LDY &3E
82A3	?	132 063	84 3F	STY &3F
82A5	=	164 061	A4 3D	LDY &3D
82A7	>	132 062	84 3E	STY &3E
82A9	4	164 052	A4 34	LDY &34
82AB	=	132 061	84 3D	STY &3D
82AD	3	164 051	A4 33	LDY &33

82AF	4	132 052	84 34	STY &34
82B1	2	164 050	A4 32	LDY &32
82B3	3	132 051	84 33	STY &33
82B5	1	164 049	A4 31	LDY &31
82B7	2	132 050	84 32	STY &32
82B9	d1	100 049	64 31	STZ &31
82BB		128 214	80 D6	BRA -42 --> &8293
82BD	L	076 197 166	4C C5 A6	JMP &A6C5
82C0		208 251	D0 FB	BNE -5 --> &82BD
82C2	0	133 048	85 30	STA &30
82C4	.	165 046	A5 2E	LDA &2E
82C6		016 023	10 17	BPL 23 --> &82DF
82C8	8	056	38	SEC
82C9		160 000	A0 00	LDY#&00
82CB		152	98	TYA
82CC	4	229 052	E5 34	SBC &34
82CE	4	133 052	85 34	STA &34
82D0		152	98	TYA
82D1	3	229 051	E5 33	SBC &33
82D3	3	133 051	85 33	STA &33
82D5		152	98	TYA
82D6	2	229 050	E5 32	SBC &32
82D8	2	133 050	85 32	STA &32
82DA		152	98	TYA
82DB	1	229 049	E5 31	SBC &31
82DD	1	133 049	85 31	STA &31
82DF	`	096	60	RTS
82E0	0	165 048	A5 30	LDA &30
82E2	0	048 005	30 05	BMI 5 --> &82E9
82E4	dI	100 073	64 49	STZ &49
82E6	L	076 242 163	4C F2 A3	JMP &A3F2
82E9	u	032 117 130	20 75 82	JSR &8275
82EC	4	165 052	A5 34	LDA &34
82EE	I	133 073	85 49	STA &49
82F0	S	032 083 131	20 53 83	JSR &8353

82F3		169 128	A9 80	LDA#&80
82F5	0	133 048	85 30	STA &30
82F7	1	166 049	A6 31	LDX &31
82F9		016 015	10 0F	BPL 15 --> &830A
82FB	E.	069 046	45 2E	EOR &2E
82FD	.	133 046	85 2E	STA &2E
82FF		016 004	10 04	BPL 4 --> &8305
8301	I	230 073	E6 49	INC &49
8303		128 002	80 02	BRA 2 --> &8307
8305	I	198 073	C6 49	DEC &49
8307		032 200 130	20 C8 82	JSR &82C8
830A	L	076 247 129	4C F7 81	JMP &81F7
830D	4	230 052	E6 34	INC &34
830F		208 012	D0 0C	BNE 12 --> &831D
8311	3	230 051	E6 33	INC &33
8313		208 008	D0 08	BNE 8 --> &831D
8315	2	230 050	E6 32	INC &32
8317		208 004	D0 04	BNE 4 --> &831D
8319	1	230 049	E6 31	INC &31
831B		240 160	F0 A0	BEQ -96 --> &82BD
831D	`	096	60	RTS
831E		160 004	A0 04	LDY#&04
8320	f	102 017	66 11	ROR &11
8322		165 016	A5 10	LDA &10
8324		170	AA	TAX
8325	j	106	6A	ROR A
8326		133 017	85 11	STA &11
8328		165 015	A5 0F	LDA &0F
832A		133 016	85 10	STA &10
832C	J	074	4A	LSR A
832D	E	069 014	45 0E	EOR &0E
832F	)	041 015	29 0F	AND#&0F
8331	E	069 014	45 0E	EOR &0E
8333	j	106	6A	ROR A
8334	j	106	6A	ROR A

8335	j	106	6A	ROR A
8336	j	106	6A	ROR A
8337	E	069 017	45 11	EOR &11
8339		134 017	86 11	STX &11
833B		166 014	A6 0E	LDX &0E
833D		134 015	86 0F	STX &0F
833F		166 013	A6 0D	LDX &0D
8341		134 014	86 0E	STX &0E
8343		133 013	85 0D	STA &0D
8345		136	88	DEY
8346		208 216	D0 D8	BNE -40 --> &8320
8348	`	096	60	RTS
8349	;	165 059	A5 3B	LDA &3B
834B	.	133 046	85 2E	STA &2E
834D	d/	100 047	64 2F	STZ &2F
834F	<	165 060	A5 3C	LDA &3C
8351	0	133 048	85 30	STA &30
8353	=	165 061	A5 3D	LDA &3D
8355	1	133 049	85 31	STA &31
8357	>	165 062	A5 3E	LDA &3E
8359	2	133 050	85 32	STA &32
835B	?	165 063	A5 3F	LDA &3F
835D	3	133 051	85 33	STA &33
835F	@	165 064	A5 40	LDA &40
8361	4	133 052	85 34	STA &34
8363	A	165 065	A5 41	LDA &41
8365	5	133 053	85 35	STA &35
8367	`	096	60	RTS
8368	1	165 049	A5 31	LDA &31
836A		240 221	F0 DD	BEQ -35 --> &8349
836C	8	056	38	SEC
836D	0	165 048	A5 30	LDA &30
836F	<	229 060	E5 3C	SBC &3C
8371	o	240 111	F0 6F	BEQ 111 --> &83E2
8373	4	144 052	90 34	BCC 52 --> &83A9

8375	%	201 037	C9 25	CMP#&25
8377		176 238	B0 EE	BCS -18 --> &8367
8379		168	A8	TAY
837A	)8	041 056	29 38	AND#&38
837C		240 023	F0 17	BEQ 23 --> &8395
837E	8	056	38	SEC
837F	@	166 064	A6 40	LDX &40
8381	A	134 065	86 41	STX &41
8383	?	166 063	A6 3F	LDX &3F
8385	@	134 064	86 40	STX &40
8387	>	166 062	A6 3E	LDX &3E
8389	?	134 063	86 3F	STX &3F
838B	=	166 061	A6 3D	LDX &3D
838D	>	134 062	86 3E	STX &3E
838F	d=	100 061	64 3D	STZ &3D
8391		233 008	E9 08	SBC#&08
8393		208 234	D0 EA	BNE -22 --> &837F
8395		152	98	TYA
8396	)	041 007	29 07	AND#&07
8398	H	240 072	F0 48	BEQ 72 --> &83E2
839A	F=	070 061	46 3D	LSR &3D
839C	f>	102 062	66 3E	ROR &3E
839E	f?	102 063	66 3F	ROR &3F
83A0	f@	102 064	66 40	ROR &40
83A2	fA	102 065	66 41	ROR &41
83A4	:	058	3A	DEC A
83A5		208 243	D0 F3	BNE -13 --> &839A
83A7	9	128 057	80 39	BRA 57 --> &83E2
83A9	I	073 255	49 FF	EOR#&FF
83AB		026	1A	INC A
83AC	%	201 037	C9 25	CMP#&25
83AE		176 153	B0 99	BCS -103 --> &8349
83B0	<	164 060	A4 3C	LDY &3C
83B2	0	132 048	84 30	STY &30
83B4		168	A8	TAY

83B5	)8	041 056	29 38	AND#&38
83B7		240 023	F0 17	BEQ 23 --> &83D0
83B9	8	056	38	SEC
83BA	4	166 052	A6 34	LDX &34
83BC	5	134 053	86 35	STX &35
83BE	3	166 051	A6 33	LDX &33
83C0	4	134 052	86 34	STX &34
83C2	2	166 050	A6 32	LDX &32
83C4	3	134 051	86 33	STX &33
83C6	1	166 049	A6 31	LDX &31
83C8	2	134 050	86 32	STX &32
83CA	d1	100 049	64 31	STZ &31
83CC		233 008	E9 08	SBC#&08
83CE		208 234	D0 EA	BNE -22 --> &83BA
83D0		152	98	TYA
83D1	)	041 007	29 07	AND#&07
83D3		240 013	F0 0D	BEQ 13 --> &83E2
83D5	F1	070 049	46 31	LSR &31
83D7	f2	102 050	66 32	ROR &32
83D9	f3	102 051	66 33	ROR &33
83DB	f4	102 052	66 34	ROR &34
83DD	f5	102 053	66 35	ROR &35
83DF	:	058	3A	DEC A
83E0		208 243	D0 F3	BNE -13 --> &83D5
83E2	.	165 046	A5 2E	LDA &2E
83E4	E;	069 059	45 3B	EOR &3B
83E6	0	048 004	30 04	BMI 4 --> &83EC
83E8		024	18	CLC
83E9	LG	076 071 164	4C 47 A4	JMP &A447
83EC	1	165 049	A5 31	LDA &31
83EE	=	197 061	C5 3D	CMP &3D
83F0		208 027	D0 1B	BNE 27 --> &840D
83F2	2	165 050	A5 32	LDA &32
83F4	>	197 062	C5 3E	CMP &3E
83F6		208 021	D0 15	BNE 21 --> &840D

83F8	3	165 051	A5 33	LDA &33
83FA	?	197 063	C5 3F	CMP &3F
83FC		208 015	D0 0F	BNE 15 --> &840D
83FE	4	165 052	A5 34	LDA &34
8400	@	197 064	C5 40	CMP &40
8402		208 009	D0 09	BNE 9 --> &840D
8404	5	165 053	A5 35	LDA &35
8406	A	197 065	C5 41	CMP &41
8408		208 003	D0 03	BNE 3 --> &840D
840A	L	076 180 166	4C B4 A6	JMP &A6B4
840D	&	176 038	B0 26	BCS 38 --> &8435
840F	;	165 059	A5 3B	LDA &3B
8411	.	133 046	85 2E	STA &2E
8413	8	056	38	SEC
8414	A	165 065	A5 41	LDA &41
8416	5	229 053	E5 35	SBC &35
8418	5	133 053	85 35	STA &35
841A	@	165 064	A5 40	LDA &40
841C	4	229 052	E5 34	SBC &34
841E	4	133 052	85 34	STA &34
8420	?	165 063	A5 3F	LDA &3F
8422	3	229 051	E5 33	SBC &33
8424	3	133 051	85 33	STA &33
8426	>	165 062	A5 3E	LDA &3E
8428	2	229 050	E5 32	SBC &32
842A	2	133 050	85 32	STA &32
842C	=	165 061	A5 3D	LDA &3D
842E	1	229 049	E5 31	SBC &31
8430	1	133 049	85 31	STA &31
8432	L	076 249 129	4C F9 81	JMP &81F9
8435	5	165 053	A5 35	LDA &35
8437	A	229 065	E5 41	SBC &41
8439	5	133 053	85 35	STA &35
843B	4	165 052	A5 34	LDA &34
843D	@	229 064	E5 40	SBC &40

843F	4	133 052	85 34	STA &34
8441	3	165 051	A5 33	LDA &33
8443	?	229 063	E5 3F	SBC &3F
8445	3	133 051	85 33	STA &33
8447	2	165 050	A5 32	LDA &32
8449	>	229 062	E5 3E	SBC &3E
844B	2	133 050	85 32	STA &32
844D	1	165 049	A5 31	LDA &31
844F	=	229 061	E5 3D	SBC &3D
8451	1	133 049	85 31	STA &31
8453	L	076 249 129	4C F9 81	JMP &81F9
8456	AN	065 078	41 4E	EOR (&4E,X)
8458	D	068	44	xxx Invalid Code
8459		128 000	80 00	BRA 0 --> &845B
845B	AB	065 066	41 42	EOR (&42,X)
845D	S	083	53	xxx Invalid Code
845E		148 000	94 00	STY &00,X
8460	AC	065 067	41 43	EOR (&43,X)
8462	S	083	53	xxx Invalid Code
8463		149 000	95 00	STA &00,X
8465	AD	065 068	41 44	EOR (&44,X)
8467	VA	086 065	56 41	LSR &41,X
8469	L	076 150 000	4C 96 00	JMP &0096
846C	AS	065 083	41 53	EOR (&53,X)
846E	C	067	43	xxx Invalid Code
846F		151	97	xxx Invalid Code
8470		000	00	BRK
8471	AS	065 083	41 53	EOR (&53,X)
8473	N	078 152 000	4E 98 00	LSR &0098
8476	AT	065 084	41 54	EOR (&54,X)
8478	N	078 153 000	4E 99 00	LSR &0099
847B	AU	065 085	41 55	EOR (&55,X)
847D	T	084	54	xxx Invalid Code
847E	O	079	4F	xxx Invalid Code
847F		198 016	C6 10	DEC &10

8481	B	066	42	xxx Invalid Code
8482	G	071	47	xxx Invalid Code
8483	ET	069 084	45 54	EOR &54
8485		154	9A	TXS
8486	B	001 066	01 42	ORA (&42,X)
8488	PU	080 085	50 55	BVC 85 --> &84DF
848A	T	084	54	xxx Invalid Code
848B		213 003	D5 03	CMP &03,X
848D	C	067	43	xxx Invalid Code
848E	O	079	4F	xxx Invalid Code
848F	LOU	076 079 085	4C 4F 55	JMP &554F
8492	R	082 251	52 FB	EOR (&FB)
8494		002	02	xxx Invalid Code
8495	C	067	43	xxx Invalid Code
8496	AL	065 076	41 4C	EOR (&4C,X)
8498	L	076 214 002	4C D6 02	JMP &02D6
849B	C	067	43	xxx Invalid Code
849C	H	072	48	PHA
849D	AI	065 073	41 49	EOR (&49,X)
849F	N	078 215 002	4E D7 02	LSR &02D7
84A2	C	067	43	xxx Invalid Code
84A3	H	072	48	PHA
84A4	R$	082 036	52 24	EOR (&24)
84A6	C	189 000 067	BD 00 43	LDA &4300,X
84A9	LEA	076 069 065	4C 45 41	JMP &4145
84AC	R	082 216	52 D8	EOR (&D8)
84AE	C	001 067	01 43	ORA (&43,X)
84B0	LOS	076 079 083	4C 4F 53	JMP &534F
84B3	E	069 217	45 D9	EOR &D9
84B5		003	03	xxx Invalid Code
84B6	C	067	43	xxx Invalid Code
84B7	LG	076 071 218	4C 47 DA	JMP &DA47
84BA	C	001 067	01 43	ORA (&43,X)
84BC	LS	076 083 219	4C 53 DB	JMP &DB53
84BF	C	001 067	01 43	ORA (&43,X)

84C1	O	079	4F	xxx Invalid Code
84C2	S	083	53	xxx Invalid Code
84C3		155	9B	xxx Invalid Code
84C4		000	00	BRK
84C5	C	067	43	xxx Invalid Code
84C6	O	079	4F	xxx Invalid Code
84C7	UN	085 078	55 4E	EOR &4E,X
84C9	T	084	54	xxx Invalid Code
84CA	C	156 001 067	9C 01 43	STZ &4301
84CD	O	079	4F	xxx Invalid Code
84CE	LOR	076 079 082	4C 4F 52	JMP &524F
84D1		251	FB	xxx Invalid Code
84D2		002	02	xxx Invalid Code
84D3	D	068	44	xxx Invalid Code
84D4	AT	065 084	41 54	EOR (&54,X)
84D6	A	065 220	41 DC	EOR (&DC,X)
84D8	DE	032 068 069	20 44 45	JSR &4544
84DB	G	071	47	xxx Invalid Code
84DC	D	157 000 068	9D 00 44	STA &4400,X
84DF	EF	069 070	45 46	EOR &46
84E1	D	221 000 068	DD 00 44	CMP &4400,X
84E4	EL	069 076	45 4C	EOR &4C
84E6	ET	069 084	45 54	EOR &54
84E8	E	069 199	45 C7	EOR &C7
84EA	D	016 068	10 44	BPL 68 --> &8530
84EC	IV	073 086	49 56	EOR#&56
84EE		129 000	81 00	STA (&00,X)
84F0	D	068	44	xxx Invalid Code
84F1	IM	073 077	49 4D	EOR#&4D
84F3	D	222 002 068	DE 02 44	DEC &4402,X
84F6	RA	082 065	52 41	EOR (&41)
84F8	W	087	57	xxx Invalid Code
84F9		223	DF	xxx Invalid Code
84FA		002	02	xxx Invalid Code
84FB	EN	069 078	45 4E	EOR &4E

84FD	D	068	44	xxx Invalid Code
84FE	PR	080 082	50 52	BVC 82 --> &8552
8500	O	079	4F	xxx Invalid Code
8501	C	067	43	xxx Invalid Code
8502		225 001	E1 01	SBC (&01,X)
8504	EN	069 078	45 4E	EOR &4E
8506	D	068	44	xxx Invalid Code
8507		224 001	E0 01	CPX#&01
8509	EN	069 078	45 4E	EOR &4E
850B	VE	086 069	56 45	LSR &45,X
850D	LOP	076 079 080	4C 4F 50	JMP &504F
8510	E	069 226	45 E2	EOR &E2
8512		002	02	xxx Invalid Code
8513	EL	069 076	45 4C	EOR &4C
8515	S	083	53	xxx Invalid Code
8516	E	069 139	45 8B	EOR &8B
8518	E	020 069	14 45	TRB &45
851A	VA	086 065	56 41	LSR &41,X
851C	L	076 160 000	4C A0 00	JMP &00A0
851F	ER	069 082	45 52	EOR &52
8521	L	076 158 001	4C 9E 01	JMP &019E
8524	ER	069 082	45 52	EOR &52
8526	RO	082 079	52 4F	EOR (&4F)
8528	R	082 133	52 85	EOR (&85)
852A	E	004 069	04 45	TSB &45
852C	O	079	4F	xxx Invalid Code
852D	F	070 197	46 C5	LSR &C5
852F	E	001 069	01 45	ORA (&45,X)
8531	O	079	4F	xxx Invalid Code
8532	R	082 130	52 82	EOR (&82)
8534		000	00	BRK
8535	ER	069 082	45 52	EOR &52
8537	R	082 159	52 9F	EOR (&9F)
8539	E	001 069	01 45	ORA (&45,X)
853B	X	088	58	CLI

853C	P	080 161	50 A1	BVC -95 --> &84DF
853E		000	00	BRK
853F	EX	069 088	45 58	EOR &58
8541	T	084	54	xxx Invalid Code
8542		162 001	A2 01	LDX#&01
8544	ED	069 068	45 44	EOR &44
8546	IT	073 084	49 54	EOR#&54
8548		206	CE	xxx Invalid Code
8549	F	016 070	10 46	BPL 70 --> &8591
854B	O	079	4F	xxx Invalid Code
854C	R	082 227	52 E3	EOR (&E3)
854E		002	02	xxx Invalid Code
854F	FA	070 065	46 41	LSR &41
8551	LSE	076 083 069	4C 53 45	JMP &4553
8554		163	A3	xxx Invalid Code
8555	F	001 070	01 46	ORA (&46,X)
8557	N	078 164 008	4E A4 08	LSR &08A4
855A	G	071	47	xxx Invalid Code
855B	O	079	4F	xxx Invalid Code
855C	T	084	54	xxx Invalid Code
855D	O	079	4F	xxx Invalid Code
855E		229 018	E5 12	SBC &12
8560	G	071	47	xxx Invalid Code
8561	ET	069 084	45 54	EOR &54
8563	$	036 190	24 BE	BIT &BE
8565		000	00	BRK
8566	G	071	47	xxx Invalid Code
8567	ET	069 084	45 54	EOR &54
8569		165 000	A5 00	LDA &00
856B	G	071	47	xxx Invalid Code
856C	O	079	4F	xxx Invalid Code
856D	S	083	53	xxx Invalid Code
856E	UB	085 066	55 42	EOR &42,X
8570		228 018	E4 12	CPX &12
8572	G	071	47	xxx Invalid Code

8573	C	067	43	xxx Invalid Code
8574	O	079	4F	xxx Invalid Code
8575	L	076 230 002	4C E6 02	JMP &02E6
8578	H	072	48	PHA
8579	IM	073 077	49 4D	EOR#&4D
857B	EM	069 077	45 4D	EOR &4D
857D		147	93	xxx Invalid Code
857E	C	067	43	xxx Invalid Code
857F	IN	073 078	49 4E	EOR#&4E
8581	PU	080 085	50 55	BVC 85 --> &85D8
8583	T	084	54	xxx Invalid Code
8584		232	E8	INX
8585		002	02	xxx Invalid Code
8586	IF	073 070	49 46	EOR#&46
8588		231	E7	xxx Invalid Code
8589		002	02	xxx Invalid Code
858A	IN	073 078	49 4E	EOR#&4E
858C	K	075	4B	xxx Invalid Code
858D	EY	069 089	45 59	EOR &59
858F	$	036 191	24 BF	BIT &BF
8591		000	00	BRK
8592	IN	073 078	49 4E	EOR#&4E
8594	K	075	4B	xxx Invalid Code
8595	EY	069 089	45 59	EOR &59
8597		166 000	A6 00	LDX &00
8599	IN	073 078	49 4E	EOR#&4E
859B	T	084	54	xxx Invalid Code
859C		168	A8	TAY
859D		000	00	BRK
859E	IN	073 078	49 4E	EOR#&4E
85A0	S	083	53	xxx Invalid Code
85A1	T	084	54	xxx Invalid Code
85A2	R(	082 040	52 28	EOR (&28)
85A4		167	A7	xxx Invalid Code
85A5		000	00	BRK

85A6	LIS	076 073 083	4C 49 53	JMP &5349
85A9	T	084	54	xxx Invalid Code
85AA		201 016	C9 10	CMP#&10
85AC	LIN	076 073 078	4C 49 4E	JMP &4E49
85AF	E	069 134	45 86	EOR &86
85B1		000	00	BRK
85B2	LOA	076 079 065	4C 4F 41	JMP &414F
85B5	D	068	44	xxx Invalid Code
85B6		200	C8	INY
85B7		002	02	xxx Invalid Code
85B8	LOM	076 079 077	4C 4F 4D	JMP &4D4F
85BB	EM	069 077	45 4D	EOR &4D
85BD	C	146 067	92 43	STA (&43)
85BF	LOC	076 079 067	4C 4F 43	JMP &434F
85C2	AL	065 076	41 4C	EOR (&4C,X)
85C4		234	EA	NOP
85C5		002	02	xxx Invalid Code
85C6	LEF	076 069 070	4C 45 46	JMP &4645
85C9	T	084	54	xxx Invalid Code
85CA	$(	036 040	24 28	BIT &28
85CC		192 000	C0 00	CPY#&00
85CE	LEN	076 069 078	4C 45 4E	JMP &4E45
85D1		169 000	A9 00	LDA#&00
85D3	LET	076 069 084	4C 45 54	JMP &5445
85D6		233 004	E9 04	SBC#&04
85D8	LOG	076 079 071	4C 4F 47	JMP &474F
85DB		171	AB	xxx Invalid Code
85DC		000	00	BRK
85DD	LN	076 078 170	4C 4E AA	JMP &AA4E
85E0		000	00	BRK
85E1	MID	077 073 068	4D 49 44	EOR &4449
85E4	$(	036 040	24 28	BIT &28
85E6		193 000	C1 00	CMP (&00,X)
85E8	MOD	077 079 068	4D 4F 44	EOR &444F
85EB	E	069 235	45 EB	EOR &EB

85ED		002	02	xxx Invalid Code
85EE	MOD	077 079 068	4D 4F 44	EOR &444F
85F1		131	83	xxx Invalid Code
85F2		000	00	BRK
85F3	MOV	077 079 086	4D 4F 56	EOR &564F
85F6	E	069 236	45 EC	EOR &EC
85F8		002	02	xxx Invalid Code
85F9	NEX	078 069 088	4E 45 58	LSR &5845
85FC	T	084	54	xxx Invalid Code
85FD	N	237 002 078	ED 02 4E	SBC &4E02
8600	EW	069 087	45 57	EOR &57
8602		202	CA	DEX
8603	N	001 078	01 4E	ORA (&4E,X)
8605	O	079	4F	xxx Invalid Code
8606	T	084	54	xxx Invalid Code
8607	O	172 000 079	AC 00 4F	LDY &4F00
860A	LD	076 068 203	4C 44 CB	JMP &CB44
860D	O	001 079	01 4F	ORA (&4F,X)
860F	N	078 238 002	4E EE 02	LSR &02EE
8612	O	079	4F	xxx Invalid Code
8613	FF	070 070	46 46	LSR &46
8615		135	87	xxx Invalid Code
8616		000	00	BRK
8617	O	079	4F	xxx Invalid Code
8618	R	082 132	52 84	EOR (&84)
861A		000	00	BRK
861B	O	079	4F	xxx Invalid Code
861C	PE	080 069	50 45	BVC 69 --> &8663
861E	NIN	078 073 078	4E 49 4E	LSR &4E49
8621	O	142 000 079	8E 00 4F	STX &4F00
8624	PE	080 069	50 45	BVC 69 --> &866B
8626	NOU	078 079 085	4E 4F 55	LSR &554F
8629	T	084	54	xxx Invalid Code
862A	O	174 000 079	AE 00 4F	LDX &4F00
862D	PE	080 069	50 45	BVC 69 --> &8674

862F	NUP	078 085 080	4E 55 50	LSR &5055
8632	O	173 000 079	AD 00 4F	LDA &4F00
8635	S	083	53	xxx Invalid Code
8636	C	067	43	xxx Invalid Code
8637	LI	076 073 255	4C 49 FF	JMP &FF49
863A		002	02	xxx Invalid Code
863B	PR	080 082	50 52	BVC 82 --> &868F
863D	IN	073 078	49 4E	EOR#&4E
863F	T	084	54	xxx Invalid Code
8640		241 002	F1 02	SBC (&02),Y
8642	PA	080 065	50 41	BVC 65 --> &8685
8644	G	071	47	xxx Invalid Code
8645	E	069 144	45 90	EOR &90
8647	C	067	43	xxx Invalid Code
8648	PT	080 084	50 54	BVC 84 --> &869E
864A	R	082 143	52 8F	EOR (&8F)
864C	C	067	43	xxx Invalid Code
864D	PI	080 073	50 49	BVC 73 --> &8698
864F		175	AF	xxx Invalid Code
8650	P	001 080	01 50	ORA (&50,X)
8652	LOT	076 079 084	4C 4F 54	JMP &544F
8655		240 002	F0 02	BEQ 2 --> &8659
8657	PO	080 079	50 4F	BVC 79 --> &86A8
8659	IN	073 078	49 4E	EOR#&4E
865B	T	084	54	xxx Invalid Code
865C	(	040	28	PLP
865D		176 000	B0 00	BCS 0 --> &865F
865F	PR	080 082	50 52	BVC 82 --> &86B3
8661	O	079	4F	xxx Invalid Code
8662	C	067	43	xxx Invalid Code
8663		242 010	F2 0A	SBC (&0A)
8665	PO	080 079	50 4F	BVC 79 --> &86B6
8667	S	083	53	xxx Invalid Code
8668		177 001	B1 01	LDA (&01),Y
866A	RE	082 069	52 45	EOR (&45)

866C	T	084	54	xxx Invalid Code
866D	UR	085 082	55 52	EOR &52,X
866F	N	078 248 001	4E F8 01	LSR &01F8
8672	RE	082 069	52 45	EOR (&45)
8674	PE	080 069	50 45	BVC 69 --> &86BB
8676	AT	065 084	41 54	EOR (&54,X)
8678		245 000	F5 00	SBC &00,X
867A	RE	082 069	52 45	EOR (&45)
867C	PO	080 079	50 4F	BVC 79 --> &86CD
867E	RT	082 084	52 54	EOR (&54)
8680		246 001	F6 01	INC &01,X
8682	RE	082 069	52 45	EOR (&45)
8684	AD	065 068	41 44	EOR (&44,X)
8686		243	F3	xxx Invalid Code
8687		002	02	xxx Invalid Code
8688	RE	082 069	52 45	EOR (&45)
868A	M	077 244 032	4D F4 20	EOR &20F4
868D	RU	082 085	52 55	EOR (&55)
868F	N	078 249 001	4E F9 01	LSR &01F9
8692	RA	082 065	52 41	EOR (&41)
8694	D	068	44	xxx Invalid Code
8695		178 000	B2 00	LDA (&00)
8697	RE	082 069	52 45	EOR (&45)
8699	S	083	53	xxx Invalid Code
869A	T	084	54	xxx Invalid Code
869B	O	079	4F	xxx Invalid Code
869C	RE	082 069	52 45	EOR (&45)
869E		247	F7	xxx Invalid Code
869F	R	018 082	12 52	ORA (&52)
86A1	IG	073 071	49 47	EOR#&47
86A3	H	072	48	PHA
86A4	T	084	54	xxx Invalid Code
86A5	$(	036 040	24 28	BIT &28
86A7		194	C2	xxx Invalid Code
86A8		000	00	BRK

86A9	RN	082 078	52 4E	EOR (&4E)
86AB	D	068	44	xxx Invalid Code
86AC		179	B3	xxx Invalid Code
86AD	R	001 082	01 52	ORA (&52,X)
86AF	EN	069 078	45 4E	EOR &4E
86B1	UM	085 077	55 4D	EOR &4D,X
86B3	B	066	42	xxx Invalid Code
86B4	ER	069 082	45 52	EOR &52
86B6	S	204 016 083	CC 10 53	CPY &5310
86B9	T	084	54	xxx Invalid Code
86BA	EP	069 080	45 50	EOR &50
86BC		136	88	DEY
86BD		000	00	BRK
86BE	S	083	53	xxx Invalid Code
86BF	AV	065 086	41 56	EOR (&56,X)
86C1	E	069 205	45 CD	EOR &CD
86C3		002	02	xxx Invalid Code
86C4	S	083	53	xxx Invalid Code
86C5	G	071	47	xxx Invalid Code
86C6	N	078 180 000	4E B4 00	LSR &00B4
86C9	S	083	53	xxx Invalid Code
86CA	IN	073 078	49 4E	EOR#&4E
86CC		181 000	B5 00	LDA &00,X
86CE	S	083	53	xxx Invalid Code
86CF	QR	081 082	51 52	EOR (&52),Y
86D1		182 000	B6 00	LDX &00,Y
86D3	S	083	53	xxx Invalid Code
86D4	PC	080 067	50 43	BVC 67 --> &8719
86D6		137 000	89 00	BIT#&00
86D8	S	083	53	xxx Invalid Code
86D9	T	084	54	xxx Invalid Code
86DA	R$	082 036	52 24	EOR (&24)
86DC		195	C3	xxx Invalid Code
86DD		000	00	BRK
86DE	S	083	53	xxx Invalid Code

86DF	T	084	54	xxx Invalid Code
86E0	RI	082 073	52 49	EOR (&49)
86E2	NG$	078 071 036	4E 47 24	LSR &2447
86E5	(	040	28	PLP
86E6		196 000	C4 00	CPY &00
86E8	S	083	53	xxx Invalid Code
86E9	O	079	4F	xxx Invalid Code
86EA	UN	085 078	55 4E	EOR &4E,X
86EC	D	068	44	xxx Invalid Code
86ED		212	D4	xxx Invalid Code
86EE		002	02	xxx Invalid Code
86EF	S	083	53	xxx Invalid Code
86F0	T	084	54	xxx Invalid Code
86F1	O	079	4F	xxx Invalid Code
86F2	P	080 250	50 FA	BVC -6 --> &86EE
86F4	T	001 084	01 54	ORA (&54,X)
86F6	AN	065 078	41 4E	EOR (&4E,X)
86F8		183	B7	xxx Invalid Code
86F9		000	00	BRK
86FA	T	084	54	xxx Invalid Code
86FB	H	072	48	PHA
86FC	EN	069 078	45 4E	EOR &4E
86FE	T	140 020 084	8C 14 54	STY &5414
8701	O	079	4F	xxx Invalid Code
8702		184	B8	CLV
8703		000	00	BRK
8704	T	084	54	xxx Invalid Code
8705	AB	065 066	41 42	EOR (&42,X)
8707	(	040	28	PLP
8708		138	8A	TXA
8709		000	00	BRK
870A	T	084	54	xxx Invalid Code
870B	RA	082 065	52 41	EOR (&41)
870D	C	067	43	xxx Invalid Code
870E	E	069 252	45 FC	EOR &FC

8710	T	018 084	12 54	ORA (&54)
8712	IM	073 077	49 4D	EOR#&4D
8714	E	069 145	45 91	EOR &91
8716	C	067	43	xxx Invalid Code
8717	T	084	54	xxx Invalid Code
8718	RU	082 085	52 55	EOR (&55)
871A	E	069 185	45 B9	EOR &B9
871C	U	001 085	01 55	ORA (&55,X)
871E	NTI	078 084 073	4E 54 49	LSR &4954
8721	L	076 253 002	4C FD 02	JMP &02FD
8724	US	085 083	55 53	EOR &53,X
8726	R	082 186	52 BA	EOR (&BA)
8728		000	00	BRK
8729	VD	086 068	56 44	LSR &44,X
872B	U	085 239	55 EF	EOR &EF,X
872D		002	02	xxx Invalid Code
872E	VA	086 065	56 41	LSR &41,X
8730	L	076 187 000	4C BB 00	JMP &00BB
8733	VP	086 080	56 50	LSR &50,X
8735	O	079	4F	xxx Invalid Code
8736	S	083	53	xxx Invalid Code
8737	W	188 001 087	BC 01 57	LDY &5701,X
873A	ID	073 068	49 44	EOR#&44
873C	T	084	54	xxx Invalid Code
873D	H	072	48	PHA
873E	P	254 002 080	FE 02 50	INC &5002,X
8741	AG	065 071	41 47	EOR (&47,X)
8743	E	069 208	45 D0	EOR &D0
8745		000	00	BRK
8746	PT	080 084	50 54	BVC 84 --> &879C
8748	R	082 207	52 CF	EOR (&CF)
874A		000	00	BRK
874B	T	084	54	xxx Invalid Code
874C	IM	073 077	49 4D	EOR#&4D
874E	E	069 209	45 D1	EOR &D1

8750		000	00	BRK
8751	LOM	076 079 077	4C 4F 4D	JMP &4D4F
8754	EM	069 077	45 4D	EOR &4D
8756		210 000	D2 00	CMP (&00)
8758	H	072	48	PHA
8759	IM	073 077	49 4D	EOR#&4D
875B	EM	069 077	45 4D	EOR &4D
875D		211	D3	xxx Invalid Code
875E		000	00	BRK
875F	Mis	077 105 115	4D 69 73	EOR &7369
8762	s	115	73	xxx Invalid Code
8763	in	105 110	69 6E	ADC#&6E
8765	g	103	67	xxx Invalid Code
8766		032 141 000	20 8D 00	JSR &008D
8769		223	DF	xxx Invalid Code
876A		170	AA	TAX
876B		201 170	C9 AA	CMP#&AA
876D		008	08	PHP
876E	D	174 068 174	AE 44 AE	LDX &AE44
8771	)	041 174	29 AE	AND#&AE
8773	/	047	2F	xxx Invalid Code
8774		174 183 172	AE B7 AC	LDX &ACB7
8777		156 168 236	9C A8 EC	STZ &ECA8
877A		173 179 171	AD B3 AB	LDA &ABB3
877D		161 168	A1 A8	LDA (&A8,X)
877F		195	C3	xxx Invalid Code
8780		168	A8	TAY
8781		215	D7	xxx Invalid Code
8782		170	AA	TAX
8783	%	014 169 037	0E A9 25	ASL &25A9
8786		174 216 169	AE D8 A9	LDX &A9D8
8789	5	053 174	35 AE	AND &AE,X
878B	;	059	3B	xxx Invalid Code
878C		174 005 171	AE 05 AB	LDX &AB05
878F		223	DF	xxx Invalid Code

8790		169 197	A9 C5	LDA#&C5
8792		170	AA	TAX
8793		232	E8	INX
8794		171	AB	xxx Invalid Code
8795		023	17	xxx Invalid Code
8796	?	176 063	B0 3F	BCS 63 --> &87D7
8798		174 194 171	AE C2 AB	LDX &ABC2
879B	6	054 172	36 AC	ROL &AC,X
879D		138	8A	TXA
879E		171	AB	xxx Invalid Code
879F		017 174	11 AE	ORA (&AE),Y
87A1	F	070 167	46 A7	LSR &A7
87A3		207	CF	xxx Invalid Code
87A4		169 147	A9 93	LDA#&93
87A6		170	AA	TAX
87A7		231	E7	xxx Invalid Code
87A8		170	AA	TAX
87A9		227	E3	xxx Invalid Code
87AA		170	AA	TAX
87AB		255	FF	xxx Invalid Code
87AC		170	AA	TAX
87AD		014 172 163	0E AC A3	ASL &A3AC
87B0		170	AA	TAX
87B1		200	C8	INY
87B2	s	169 115	A9 73	LDA#&73
87B4		170	AA	TAX
87B5		245 171	F5 AB	SBC &AB,X
87B7		013 169 181	0D A9 B5	ORA &B5A9
87BA		167	A7	xxx Invalid Code
87BB		155	9B	xxx Invalid Code
87BC		165 249	A5 F9	LDA &F9
87BE		173 219 171	AD DB AB	LDA &ABDB
87C1		169 170	A9 AA	LDA#&AA
87C3	I	073 171	49 AB	EOR#&AB
87C5	/	188 170 047	BC AA 2F	LDY &2FAA,X

87C8	i	178 105	B2 69	LDA (&69)
87CA		174 179 174	AE B3 AE	LDX &AEB3
87CD	s	115	73	xxx Invalid Code
87CE		174 197 174	AE C5 AE	LDX &AEC5
87D1	t	116 174	74 AE	STZ &AE,X
87D3	G	028 175 071	1C AF 47	TRB &47AF
87D6		175	AF	xxx Invalid Code
87D7		207	CF	xxx Invalid Code
87D8		171	AB	xxx Invalid Code
87D9		137 148	89 94	BIT#&94
87DB		023	17	xxx Invalid Code
87DC		147	93	xxx Invalid Code
87DD		032 143 221	20 8F DD	JSR &DD8F
87E0		179	B3	xxx Invalid Code
87E1	}	125 143 000	7D 8F 00	ADC &008F,X
87E4		143	8F	xxx Invalid Code
87E5		132 147	84 93	STY &93
87E7	U	085 190	55 BE	EOR &BE,X
87E9		147	93	xxx Invalid Code
87EA		179	B3	xxx Invalid Code
87EB		151	97	xxx Invalid Code
87EC	4	190 052 150	BE 34 96	LDX &9634,Y
87EF	y	121 150 032	79 96 20	ADC &2096,Y
87F2		150 015	96 0F	STX &0F,Y
87F4		150 200	96 C8	STX &C8,Y
87F6		178 189	B2 BD	LDA (&BD)
87F8		190 190 146	BE BE 92	LDX &92BE,Y
87FB		251	FB	xxx Invalid Code
87FC	>	142 062 150	8E 3E 96	STX &963E
87FF		174 190 224	AE BE E0	LDX &E0BE
8802		151	97	xxx Invalid Code
8803		231	E7	xxx Invalid Code
8804		151	97	xxx Invalid Code
8805		174 143 174	AE 8F AE	LDX &AE8F
8808		143	8F	xxx Invalid Code

```
8809    4       052 149         34 95       BIT &95,X
880B            166 151         A6 97       LDX &97
880D    %       037 143         25 8F       AND &8F
880F            154             9A          TXS
8810            155             9B          xxx Invalid Code
8811            236 178 024     EC B2 18    CPX &18B2
8814            182 217         B6 D9       LDX &D9,Y
8816            182 029         B6 1D       LDX &1D,Y
8818            183             B7          xxx Invalid Code
8819    A       065 151         41 97       EOR (&97,X)
881B            008             08          PHP
881C            156 182 184     9C B6 B8    STZ &B8B6
881F    J       074             4A          LSR A
8820            144 003         90 03       BCC 3 --> &8825
8822            151             97          xxx Invalid Code
8823    _       095             5F          xxx Invalid Code
8824            151             97          xxx Invalid Code
8825            162 151         A2 97       LDX#&97
8827            241 180         F1 B4       SBC (&B4),Y
8829    [       091             5B          xxx Invalid Code
882A            183             B7          xxx Invalid Code
882B            013 152 177     0D 98 B1    ORA &B198
882E            151             97          xxx Invalid Code
882F            141 145 228     8D 91 E4    STA &E491
8832    }       150 125         96 7D       STX &7D,Y
8834            185 174 143     B9 AE 8F    LDA &8FAE,Y
8837    X       088             58          CLI
8838            186             BA          TSX
8839            244             F4          xxx Invalid Code
883A            151             97          xxx Invalid Code
883B    M       077 185 007     4D B9 07    EOR &07B9
883E            183             B7          xxx Invalid Code
883F            018 143         12 8F       ORA (&8F)
8841            134 144         86 90       STX &90
8843    U       085 151         55 97       EOR &97,X
```

8845	F	070 150	46 96	LSR &96
8847		023	17	xxx Invalid Code
8848		186	BA	TSX
8849		023	17	xxx Invalid Code
884A		179	B3	xxx Invalid Code
884B		135	87	xxx Invalid Code
884C	K	190 075 131	BE 4B 83	LDX &834B,Y
884F		132 137	84 89	STY &89
8851		150 184	96 B8	STX &B8,Y
8853		185 216 217	B9 D8 D9	LDA &D9D8,Y
8856		240 001	F0 01	BEQ 1 --> &8859
8858		016 129	10 81	BPL -127 --> &87DB
885A		144 137	90 89	BCC -119 --> &87E5
885C		147	93	xxx Invalid Code
885D		163	A3	xxx Invalid Code
885E		164 169	A4 A9	LDY &A9
8860	8	056	38	SEC
8861	9x	057 120 001	39 78 01	AND &0178,Y
8864		019	13	xxx Invalid Code
8865	!	033 161	21 A1	AND (&A1,X)
8867		193 025	C1 19	CMP (&19,X)
8869		024	18	CLC
886A	c	153 152 099	99 98 63	STA &6398,Y
886D	s	115	73	xxx Invalid Code
886E		177 169	B1 A9	LDA (&A9),Y
8870		197 012	C5 0C	CMP &0C
8872		195	C3	xxx Invalid Code
8873		211	D3	xxx Invalid Code
8874	A	065 196	41 C4	EOR (&C4,X)
8876	A	242 065	F2 41	SBC (&41)
8878		131	83	xxx Invalid Code
8879		176 129	B0 81	BCS -127 --> &87FC
887B	C	067	43	xxx Invalid Code
887C	lr	108 114 236	6C 72 EC	JMP (&EC72)
887F		242 163	F2 A3	SBC (&A3)

8881		195	C3	xxx Invalid Code
8882		146 154	92 9A	STA (&9A)
8884		024	18	CLC
8885	bB	025 098 066	19 62 42	ORA &4262,Y
8888	4	052 176	34 B0	BIT &B0,X
888A	r	114 152	72 98	ADC (&98)
888C		153 129 152	99 81 98	STA &9881,Y
888F	5	153 020 053	99 14 35	STA &3514,Y
8892		010	0A	ASL A
8893		013 013 013	0D 0D 0D	ORA &0D0D
8896		013 016 016	0D 10 10	ORA &1010
8899	%%	037 037	25 25	AND &25
889B	9AA	057 065 065	39 41 41	AND &4141,Y
889E	AA	065 065	41 41	EOR (&41,X)
88A0	J	074	4A	LSR A
88A1	J	074	4A	LSR A
88A2	LLL	076 076 076	4C 4C 4C	JMP &4C4C
88A5	PP	080 080	50 50	BVC 80 --> &88F7
88A7	RS	082 083	52 53	EOR (&53)
88A9	S	083	53	xxx Invalid Code
88AA	S	083	53	xxx Invalid Code
88AB	%	016 037	10 25	BPL 37 --> &88D2
88AD	AA	065 065	41 41	EOR (&41,X)
88AF	AA	065 065	41 41	EOR (&41,X)
88B1		008	08	PHP
88B2		008	08	PHP
88B3		008	08	PHP
88B4		009 009	09 09	ORA#&09
88B6		010	0A	ASL A
88B7		010	0A	ASL A
88B8		010	0A	ASL A
88B9		010	0A	ASL A
88BA		005 021	05 15	ORA &15
88BC	>	062 004 013	3E 04 0D	ROL &0D04,X
88BF	0L	048 076	30 4C	BMI 76 --> &890D

88C1	2	006 050	06 32	ASL &32
88C3	II	073 073	49 49	EOR#&49
88C5	%	016 037	10 25	BPL 37 --> &88EC
88C7	N	013 078 014	0D 4E 0E	ORA &0E4E
88CA	RR	014 082 082	0E 52 52	ASL &5252
88CD	)	009 041	09 29	ORA#&29
88CF	*	042	2A	ROL A
88D0	00	048 048	30 30	BMI 48 --> &8902
88D2	NNN	078 078 078	4E 4E 4E	LSR &4E4E
88D5	>	062 022 000	3E 16 00	ROL &0016,X
88D8		024	18	CLC
88D9		216	D8	CLD
88DA	X	088	58	CLI
88DB		184	B8	CLV
88DC		202	CA	DEX
88DD		136	88	DEY
88DE		232	E8	INX
88DF		200	C8	INY
88E0		234	EA	NOP
88E1	H	072	48	PHA
88E2		008	08	PHP
88E3	h	104	68	PLA
88E4	(	040	28	PLP
88E5	@	064	40	RTI
88E6	`	096	60	RTS
88E7	8	056	38	SEC
88E8		248	F8	SED
88E9	x	120	78	SEI
88EA		170	AA	TAX
88EB		168	A8	TAY
88EC		186	BA	TSX
88ED		138	8A	TXA
88EE		154	9A	TXS
88EF		152	98	TYA
88F0	:	058	3A	DEC A

88F1		026	1A	INC A
88F2	Z	090	5A	PHY
88F3		218	DA	PHX
88F4	z	122	7A	PLY
88F5		250	FA	PLX
88F6		144 176	90 B0	BCC -80 --> &88A8
88F8	0	240 048	F0 30	BEQ 48 --> &892A
88FA		208 016	D0 10	BNE 16 --> &890C
88FC	Pp	080 112	50 70	BVC 112 --> &896E
88FE	!	128 033	80 21	BRA 33 --> &8921
8900	A	065 001	41 01	EOR (&01,X)
8902	a	097 193 161	61 C1 A1	ADC (&A1C1,X)
8905		225 006	E1 06	SBC (&06,X)
8907	F&	070 038	46 26	LSR &26
8909	f	102 198	66 C6	ROR &C6
890B		230 156	E6 9C	INC &9C
890D		156 224 192	9C E0 C0	STZ &C0E0
8910		000	00	BRK
8911	$	016 036	10 24	BPL 36 --> &8937
8913	L	076 032 162	4C 20 A2	JMP &A220
8916		160 129	A0 81	LDY#&81
8918		134 132	86 84	STX &84
891A	:	058	3A	DEC A
891B	(	133 040	85 28	STA &28
891D	L	076 011 144	4C 0B 90	JMP &900B
8920		032 224 142	20 E0 8E	JSR &8EE0
8923	I]	073 093	49 5D	EOR#&5D
8925		240 243	F0 F3	BEQ -13 --> &891A
8927		032 188 155	20 BC 9B	JSR &9BBC
892A		198 010	C6 0A	DEC &0A
892C		032 235 137	20 EB 89	JSR &89EB
892F		198 010	C6 0A	DEC &0A
8931	(	165 040	A5 28	LDA &28
8933	J	074	4A	LSR A
8934	x	144 120	90 78	BCC 120 --> &89AE

```
8936        165 030        A5 1E       LDA &1E
8938   i    105 004        69 04       ADC#&04
893A   ?    133 063        85 3F       STA &3F
893C   8    165 056        A5 38       LDA &38
893E   l    032 108 189     20 6C BD    JSR &BD6C
8941   7    165 055        A5 37       LDA &37
8943        032 143 189     20 8F BD    JSR &BD8F
8946        162 252        A2 FC       LDX#&FC
8948   9    164 057        A4 39       LDY &39
894A        016 002        10 02       BPL 2 --> &894E
894C   6    164 054        A4 36       LDY &36
894E   8    132 056        84 38       STY &38
8950        240 025        F0 19       BEQ 25 --> &896B
8952        160 000        A0 00       LDY#&00
8954        232            E8          INX
8955        208 010        D0 0A       BNE 10 --> &8961
8957        032 146 186    20 92 BA    JSR &BA92
895A   ?    166 063        A6 3F       LDX &3F
895C        032 191 189    20 BF BD    JSR &BDBF
895F        162 253        A2 FD       LDX#&FD
8961   :    177 058        B1 3A       LDA (&3A),Y
8963        032 143 189    20 8F BD    JSR &BD8F
8966        200            C8          INY
8967   8    198 056        C6 38       DEC &38
8969        208 233        D0 E9       BNE -23 --> &8954
896B        138            8A          TXA
896C        168            A8          TAY
896D        200            C8          INY
896E        240 007        F0 07       BEQ 7 --> &8977
8970        162 003        A2 03       LDX#&03
8972        032 191 189    20 BF BD    JSR &BDBF
8975        128 246        80 F6       BRA -10 --> &896D
8977        162 010        A2 0A       LDX#&0A
8979        178 011        B2 0B       LDA (&0B)
897B   .    201 046        C9 2E       CMP#&2E
```

```
897D        208 015       D0 0F       BNE 15 --> &898E
897F    7   032 055 189   20 37 BD     JSR &BD37
8982        202           CA          DEX
8983        208 002       D0 02       BNE 2 --> &8987
8985        162 001       A2 01       LDX#&01
8987        200           C8          INY
8988        177 011       B1 0B       LDA (&0B),Y
898A    N   196 078       C4 4E       CPY &4E
898C        208 241       D0 F1       BNE -15 --> &897F
898E        032 191 189   20 BF BD     JSR &BDBF
8991        136           88          DEY
8992        200           C8          INY
8993        209 011       D1 0B       CMP (&0B),Y
8995        240 251       F0 FB       BEQ -5 --> &8992
8997        177 011       B1 0B       LDA (&0B),Y
8999    :   201 058       C9 3A       CMP#&3A
899B        240 010       F0 0A       BEQ 10 --> &89A7
899D        201 013       C9 0D       CMP#&0D
899F        240 010       F0 0A       BEQ 10 --> &89AB
89A1    7   032 055 189   20 37 BD     JSR &BD37
89A4        200           C8          INY
89A5        128 240       80 F0       BRA -16 --> &8997
89A7        196 010       C4 0A       CPY &0A
89A9        144 246       90 F6       BCC -10 --> &89A1
89AB        032 146 186   20 92 BA     JSR &BA92
89AE        164 010       A4 0A       LDY &0A
89B0        136           88          DEY
89B1        200           C8          INY
89B2        177 011       B1 0B       LDA (&0B),Y
89B4    :   201 058       C9 3A       CMP#&3A
89B6        240 004       F0 04       BEQ 4 --> &89BC
89B8        201 013       C9 0D       CMP#&0D
89BA        208 245       D0 F5       BNE -11 --> &89B1
89BC        032 168 155   20 A8 9B     JSR &9BA8
89BF        178 011       B2 0B       LDA (&0B)
```

89C1	:	201 058	C9 3A	CMP#&3A
89C3		240 012	F0 0C	BEQ 12 --> &89D1
89C5		165 012	A5 0C	LDA &0C
89C7		201 007	C9 07	CMP#&07
89C9		208 003	D0 03	BNE 3 --> &89CE
89CB	L	076 134 143	4C 86 8F	JMP &8F86
89CE		032 222 155	20 DE 9B	JSR &9BDE
89D1	L	076 032 137	4C 20 89	JMP &8920
89D4		032 174 152	20 AE 98	JSR &98AE
89D7	\	240 092	F0 5C	BEQ 92 --> &8A35
89D9	Z	176 090	B0 5A	BCS 90 --> &8A35
89DB	C	032 067 188	20 43 BC	JSR &BC43
89DE		032 132 173	20 84 AD	JSR &AD84
89E1	'	133 039	85 27	STA &27
89E3	+	032 043 179	20 2B B3	JSR &B32B
89E6	u	032 117 146	20 75 92	JSR &9275
89E9	N	132 078	84 4E	STY &4E
89EB		032 224 142	20 E0 8E	JSR &8EE0
89EE		160 000	A0 00	LDY#&00
89F0	d=	100 061	64 3D	STZ &3D
89F2	:	201 058	C9 3A	CMP#&3A
89F4	h	240 104	F0 68	BEQ 104 --> &8A5E
89F6		201 013	C9 0D	CMP#&0D
89F8	d	240 100	F0 64	BEQ 100 --> &8A5E
89FA	\	201 092	C9 5C	CMP#&5C
89FC	`	240 096	F0 60	BEQ 96 --> &8A5E
89FE	.	201 046	C9 2E	CMP#&2E
8A00		240 210	F0 D2	BEQ -46 --> &89D4
8A02		198 010	C6 0A	DEC &0A
8A04		162 003	A2 03	LDX#&03
8A06		164 010	A4 0A	LDY &0A
8A08		230 010	E6 0A	INC &0A
8A0A		177 011	B1 0B	LDA (&0B),Y
8A0C	0*	048 042	30 2A	BMI 42 --> &8A38
8A0E		201 032	C9 20	CMP#&20

8A10		240 016	F0 10	BEQ 16 --> &8A22
8A12		160 005	A0 05	LDY#&05
8A14		010	0A	ASL A
8A15		010	0A	ASL A
8A16		010	0A	ASL A
8A17		010	0A	ASL A
8A18	&=	038 061	26 3D	ROL &3D
8A1A	&>	038 062	26 3E	ROL &3E
8A1C		136	88	DEY
8A1D		208 248	D0 F8	BNE -8 --> &8A17
8A1F		202	CA	DEX
8A20		208 228	D0 E4	BNE -28 --> &8A06
8A22	E	162 069	A2 45	LDX#&45
8A24	=	165 061	A5 3D	LDA &3D
8A26	L	221 076 136	DD 4C 88	CMP &884C,X
8A29		208 007	D0 07	BNE 7 --> &8A32
8A2B		188 145 136	BC 91 88	LDY &8891,X
8A2E	>	196 062	C4 3E	CPY &3E
8A30	!	240 033	F0 21	BEQ 33 --> &8A53
8A32		202	CA	DEX
8A33		208 241	D0 F1	BNE -15 --> &8A26
8A35	Li	076 105 155	4C 69 9B	JMP &9B69
8A38	)	162 041	A2 29	LDX#&29
8A3A		201 128	C9 80	CMP#&80
8A3C		240 021	F0 15	BEQ 21 --> &8A53
8A3E		232	E8	INX
8A3F		201 130	C9 82	CMP#&82
8A41		240 016	F0 10	BEQ 16 --> &8A53
8A43		232	E8	INX
8A44		201 132	C9 84	CMP#&84
8A46		208 237	D0 ED	BNE -19 --> &8A35
8A48		230 010	E6 0A	INC &0A
8A4A		200	C8	INY
8A4B		177 011	B1 0B	LDA (&0B),Y
8A4D	)	041 223	29 DF	AND#&DF

8A4F	A	201 065	C9 41	CMP#&41
8A51		208 226	D0 E2	BNE -30 --> &8A35
8A53		189 214 136	BD D6 88	LDA &88D6,X
8A56	)	133 041	85 29	STA &29
8A58		160 001	A0 01	LDY#&01
8A5A		224 032	E0 20	CPX#&20
8A5C	H	176 072	B0 48	BCS 72 --> &8AA6
8A5E	@	173 064 004	AD 40 04	LDA &0440
8A61	7	133 055	85 37	STA &37
8A63	9	132 057	84 39	STY &39
8A65	(	166 040	A6 28	LDX &28
8A67		224 004	E0 04	CPX#&04
8A69	A	174 065 004	AE 41 04	LDX &0441
8A6C	8	134 056	86 38	STX &38
8A6E		144 006	90 06	BCC 6 --> &8A76
8A70	<	173 060 004	AD 3C 04	LDA &043C
8A73	=	174 061 004	AE 3D 04	LDX &043D
8A76	:	133 058	85 3A	STA &3A
8A78	;	134 059	86 3B	STX &3B
8A7A		152	98	TYA
8A7B	(	240 040	F0 28	BEQ 40 --> &8AA5
8A7D		016 004	10 04	BPL 4 --> &8A83
8A7F	6	164 054	A4 36	LDY &36
8A81	"	240 034	F0 22	BEQ 34 --> &8AA5
8A83		136	88	DEY
8A84	)	185 041 000	B9 29 00	LDA &0029,Y
8A87	$9	036 057	24 39	BIT &39
8A89		016 003	10 03	BPL 3 --> &8A8E
8A8B		185 000 006	B9 00 06	LDA &0600,Y
8A8E	:	145 058	91 3A	STA (&3A),Y
8A90	@	238 064 004	EE 40 04	INC &0440
8A93		208 003	D0 03	BNE 3 --> &8A98
8A95	A	238 065 004	EE 41 04	INC &0441
8A98		144 008	90 08	BCC 8 --> &8AA2
8A9A	<	238 060 004	EE 3C 04	INC &043C

8A9D		208 003	D0 03	BNE 3 --> &8AA2
8A9F	=	238 061 004	EE 3D 04	INC &043D
8AA2		152	98	TYA
8AA3		208 222	D0 DE	BNE -34 --> &8A83
8AA5	`	096	60	RTS
8AA6	)	224 041	E0 29	CPX#&29
8AA8	<	176 060	B0 3C	BCS 60 --> &8AE6
8AAA	o	032 111 146	20 6F 92	JSR &926F
8AAD		024	18	CLC
8AAE	*	165 042	A5 2A	LDA &2A
8AB0	@	237 064 004	ED 40 04	SBC &0440
8AB3		168	A8	TAY
8AB4	+	165 043	A5 2B	LDA &2B
8AB6	A	237 065 004	ED 41 04	SBC &0441
8AB9		192 001	C0 01	CPY#&01
8ABB		136	88	DEY
8ABC		233 000	E9 00	SBC#&00
8ABE		240 027	F0 1B	BEQ 27 --> &8ADB
8AC0		026	1A	INC A
8AC1		208 003	D0 03	BNE 3 --> &8AC6
8AC3		152	98	TYA
8AC4	0	048 025	30 19	BMI 25 --> &8ADF
8AC6	(	165 040	A5 28	LDA &28
8AC8	)	041 002	29 02	AND#&02
8ACA		240 018	F0 12	BEQ 18 --> &8ADE
8ACC		000	00	BRK
8ACD		001	01	EQUB &01
8ACE	O	079	4F	ORA (&4F,X)
8ACF	ut	117 116	75 74	ADC &74,X
8AD1	of	032 111 102	20 6F 66	JSR &666F
8AD4	ra	032 114 097	20 72 61	JSR &6172
8AD7	nge	110 103 101	6E 67 65	ROR &6567
8ADA		000	00	EQUB &00
8ADB		152	98	TYA
8ADC	0	048 232	30 E8	BMI -24 --> &8AC6

8ADE		168	A8	TAY
8ADF	*	132 042	84 2A	STY &2A
8AE1		160 002	A0 02	LDY#&02
8AE3	L^	076 094 138	4C 5E 8A	JMP &8A5E
8AE6	0	224 048	E0 30	CPX#&30
8AE8		176 022	B0 16	BCS 22 --> &8B00
8AEA		032 223 140	20 DF 8C	JSR &8CDF
8AED		208 024	D0 18	BNE 24 --> &8B07
8AEF		032 204 140	20 CC 8C	JSR &8CCC
8AF2	o	032 111 146	20 6F 92	JSR &926F
8AF5	+	165 043	A5 2B	LDA &2B
8AF7		240 232	F0 E8	BEQ -24 --> &8AE1
8AF9		000	00	BRK
8AFA		002	02	EQUB &02
8AFB	B	066	42	xxx Invalid Code
8AFC	yte	121 116 101	79 74 65	ADC &6574,Y
8AFF		000	00	EQUB &00
8B00	A	224 065	E0 41	CPX#&41
8B02	c	208 099	D0 63	BNE 99 --> &8B67
8B04		032 224 142	20 E0 8E	JSR &8EE0
8B07	(	201 040	C9 28	CMP#&28
8B09	9	208 057	D0 39	BNE 57 --> &8B44
8B0B	o	032 111 146	20 6F 92	JSR &926F
8B0E		032 224 142	20 E0 8E	JSR &8EE0
8B11	)	201 041	C9 29	CMP#&29
8B13		208 023	D0 17	BNE 23 --> &8B2C
8B15		032 201 140	20 C9 8C	JSR &8CC9
8B18		032 229 140	20 E5 8C	JSR &8CE5
8B1B		240 004	F0 04	BEQ 4 --> &8B21
8B1D	)	230 041	E6 29	INC &29
8B1F		128 212	80 D4	BRA -44 --> &8AF5
8B21		032 224 142	20 E0 8E	JSR &8EE0
8B24	)	041 223	29 DF	AND#&DF
8B26	Y	201 089	C9 59	CMP#&59
8B28		240 203	F0 CB	BEQ -53 --> &8AF5

8B2A		128 016	80 10	BRA 16 --> &8B3C
8B2C	,	201 044	C9 2C	CMP#&2C
8B2E		208 012	D0 0C	BNE 12 --> &8B3C
8B30		032 215 140	20 D7 8C	JSR &8CD7
8B33		208 007	D0 07	BNE 7 --> &8B3C
8B35		032 224 142	20 E0 8E	JSR &8EE0
8B38	)	201 041	C9 29	CMP#&29
8B3A		240 185	F0 B9	BEQ -71 --> &8AF5
8B3C		000	00	BRK
8B3D		003	03	EQUB &03
8B3E	In	073 110	49 6E	EOR#&6E
8B40	de	100 101	64 65	STZ &65
8B42	x	120	78	SEI
8B43		000	00	EQUB &00
8B44	m	032 109 146	20 6D 92	JSR &926D
8B47		032 229 140	20 E5 8C	JSR &8CE5
8B4A		208 018	D0 12	BNE 18 --> &8B5E
8B4C		032 201 140	20 C9 8C	JSR &8CC9
8B4F		032 215 140	20 D7 8C	JSR &8CD7
8B52		240 010	F0 0A	BEQ 10 --> &8B5E
8B54	Y	201 089	C9 59	CMP#&59
8B56		208 228	D0 E4	BNE -28 --> &8B3C
8B58		032 204 140	20 CC 8C	JSR &8CCC
8B5B	L	076 254 139	4C FE 8B	JMP &8BFE
8B5E		032 207 140	20 CF 8C	JSR &8CCF
8B61	+	165 043	A5 2B	LDA &2B
8B63		208 243	D0 F3	BNE -13 --> &8B58
8B65		128 144	80 90	BRA -112 --> &8AF7
8B67	6	224 054	E0 36	CPX#&36
8B69	6	176 054	B0 36	BCS 54 --> &8BA1
8B6B		032 224 142	20 E0 8E	JSR &8EE0
8B6E	)	041 223	29 DF	AND#&DF
8B70	A	201 065	C9 41	CMP#&41
8B72		240 018	F0 12	BEQ 18 --> &8B86
8B74	m	032 109 146	20 6D 92	JSR &926D

8B77		032 229 140	20 E5 8C	JSR &8CE5
8B7A		208 229	D0 E5	BNE -27 --> &8B61
8B7C		032 201 140	20 C9 8C	JSR &8CC9
8B7F		032 215 140	20 D7 8C	JSR &8CD7
8B82		240 221	F0 DD	BEQ -35 --> &8B61
8B84		128 182	80 B6	BRA -74 --> &8B3C
8B86		200	C8	INY
8B87		177 011	B1 0B	LDA (&0B),Y
8B89		032 132 141	20 84 8D	JSR &8D84
8B8C		176 230	B0 E6	BCS -26 --> &8B74
8B8E		160 022	A0 16	LDY#&16
8B90	4	224 052	E0 34	CPX#&34
8B92		144 006	90 06	BCC 6 --> &8B9A
8B94		208 002	D0 02	BNE 2 --> &8B98
8B96	6	160 054	A0 36	LDY#&36
8B98	)	132 041	84 29	STY &29
8B9A		032 207 140	20 CF 8C	JSR &8CCF
8B9D		160 001	A0 01	LDY#&01
8B9F	_	128 095	80 5F	BRA 95 --> &8C00
8BA1	8	224 056	E0 38	CPX#&38
8BA3	%	176 037	B0 25	BCS 37 --> &8BCA
8BA5	o	032 111 146	20 6F 92	JSR &926F
8BA8		160 003	A0 03	LDY#&03
8BAA		162 001	A2 01	LDX#&01
8BAC	+	165 043	A5 2B	LDA &2B
8BAE		208 007	D0 07	BNE 7 --> &8BB7
8BB0		162 015	A2 0F	LDX#&0F
8BB2	d	169 100	A9 64	LDA#&64
8BB4	)	133 041	85 29	STA &29
8BB6		136	88	DEY
8BB7	Z	090	5A	PHY
8BB8		032 229 140	20 E5 8C	JSR &8CE5
8BBB		208 010	D0 0A	BNE 10 --> &8BC7
8BBD		032 215 140	20 D7 8C	JSR &8CD7
8BC0		208 194	D0 C2	BNE -62 --> &8B84

8BC2		138	8A	TXA
8BC3	e)	101 041	65 29	ADC &29
8BC5	)	133 041	85 29	STA &29
8BC7	z	122	7A	PLY
8BC8	6	128 054	80 36	BRA 54 --> &8C00
8BCA	<	224 060	E0 3C	CPX#&3C
8BCC		176 028	B0 1C	BCS 28 --> &8BEA
8BCE	:	224 058	E0 3A	CPX#&3A
8BD0		176 007	B0 07	BCS 7 --> &8BD9
8BD2		032 223 140	20 DF 8C	JSR &8CDF
8BD5		240 016	F0 10	BEQ 16 --> &8BE7
8BD7		198 010	C6 0A	DEC &0A
8BD9	o	032 111 146	20 6F 92	JSR &926F
8BDC		128 128	80 80	BRA -128 --> &8B5E
8BDE		032 223 140	20 DF 8C	JSR &8CDF
8BE1		208 145	D0 91	BNE -111 --> &8B74
8BE3		160 137	A0 89	LDY#&89
8BE5	)	132 041	84 29	STY &29
8BE7	L	076 242 138	4C F2 8A	JMP &8AF2
8BEA		240 242	F0 F2	BEQ -14 --> &8BDE
8BEC	>	224 062	E0 3E	CPX#&3E
8BEE		240 011	F0 0B	BEQ 11 --> &8BFB
8BF0	7	176 055	B0 37	BCS 55 --> &8C29
8BF2		032 224 142	20 E0 8E	JSR &8EE0
8BF5	(	201 040	C9 28	CMP#&28
8BF7		240 010	F0 0A	BEQ 10 --> &8C03
8BF9		198 010	C6 0A	DEC &0A
8BFB	o	032 111 146	20 6F 92	JSR &926F
8BFE		160 003	A0 03	LDY#&03
8C00	L^	076 094 138	4C 5E 8A	JMP &8A5E
8C03		032 201 140	20 C9 8C	JSR &8CC9
8C06		032 201 140	20 C9 8C	JSR &8CC9
8C09	o	032 111 146	20 6F 92	JSR &926F
8C0C		032 224 142	20 E0 8E	JSR &8EE0
8C0F	)	201 041	C9 29	CMP#&29

8C11		240 235	F0 EB	BEQ -21 --> &8BFE
8C13	,	201 044	C9 2C	CMP#&2C
8C15		208 015	D0 0F	BNE 15 --> &8C26
8C17		032 201 140	20 C9 8C	JSR &8CC9
8C1A		032 215 140	20 D7 8C	JSR &8CD7
8C1D		208 007	D0 07	BNE 7 --> &8C26
8C1F		032 224 142	20 E0 8E	JSR &8EE0
8C22	)	201 041	C9 29	CMP#&29
8C24		240 216	F0 D8	BEQ -40 --> &8BFE
8C26	L<	076 060 139	4C 3C 8B	JMP &8B3C
8C29	D	224 068	E0 44	CPX#&44
8C2B	M	176 077	B0 4D	BCS 77 --> &8C7A
8C2D	=	165 061	A5 3D	LDA &3D
8C2F	I	073 001	49 01	EOR#&01
8C31	)	041 031	29 1F	AND#&1F
8C33	H	072	48	PHA
8C34	A	224 065	E0 41	CPX#&41
8C36	!	176 033	B0 21	BCS 33 --> &8C59
8C38		032 223 140	20 DF 8C	JSR &8CDF
8C3B		208 003	D0 03	BNE 3 --> &8C40
8C3D	h	104	68	PLA
8C3E		128 167	80 A7	BRA -89 --> &8BE7
8C40	m	032 109 146	20 6D 92	JSR &926D
8C43	h	104	68	PLA
8C44	7	133 055	85 37	STA &37
8C46		032 229 140	20 E5 8C	JSR &8CE5
8C49		208 145	D0 91	BNE -111 --> &8BDC
8C4B		032 224 142	20 E0 8E	JSR &8EE0
8C4E	)	041 031	29 1F	AND#&1F
8C50	7	197 055	C5 37	CMP &37
8C52		208 210	D0 D2	BNE -46 --> &8C26
8C54		032 201 140	20 C9 8C	JSR &8CC9
8C57		128 131	80 83	BRA -125 --> &8BDC
8C59	o	032 111 146	20 6F 92	JSR &926F
8C5C	h	104	68	PLA

8C5D	7	133 055	85 37	STA &37
8C5F		032 229 140	20 E5 8C	JSR &8CE5
8C62		208 019	D0 13	BNE 19 --> &8C77
8C64		032 224 142	20 E0 8E	JSR &8EE0
8C67	)	041 031	29 1F	AND#&1F
8C69	7	197 055	C5 37	CMP &37
8C6B		208 185	D0 B9	BNE -71 --> &8C26
8C6D		032 201 140	20 C9 8C	JSR &8CC9
8C70	+	165 043	A5 2B	LDA &2B
8C72		240 003	F0 03	BEQ 3 --> &8C77
8C74	L	076 249 138	4C F9 8A	JMP &8AF9
8C77	La	076 097 139	4C 61 8B	JMP &8B61
8C7A		208 011	D0 0B	BNE 11 --> &8C87
8C7C	o	032 111 146	20 6F 92	JSR &926F
8C7F	*	165 042	A5 2A	LDA &2A
8C81	(	133 040	85 28	STA &28
8C83		160 000	A0 00	LDY#&00
8C85	*	128 042	80 2A	BRA 42 --> &8CB1
8C87		162 001	A2 01	LDX#&01
8C89		164 010	A4 0A	LDY &0A
8C8B		230 010	E6 0A	INC &0A
8C8D		177 011	B1 0B	LDA (&0B),Y
8C8F	)	041 223	29 DF	AND#&DF
8C91	B	201 066	C9 42	CMP#&42
8C93		240 018	F0 12	BEQ 18 --> &8CA7
8C95		232	E8	INX
8C96	W	201 087	C9 57	CMP#&57
8C98		240 013	F0 0D	BEQ 13 --> &8CA7
8C9A		162 004	A2 04	LDX#&04
8C9C	D	201 068	C9 44	CMP#&44
8C9E		240 007	F0 07	BEQ 7 --> &8CA7
8CA0	S	201 083	C9 53	CMP#&53
8CA2		240 019	F0 13	BEQ 19 --> &8CB7
8CA4	Li	076 105 155	4C 69 9B	JMP &9B69
8CA7		218	DA	PHX

8CA8	o	032 111 146	20 6F 92	JSR &926F
8CAB	)	162 041	A2 29	LDX#&29
8CAD		032 198 189	20 C6 BD	JSR &BDC6
8CB0	z	122	7A	PLY
8CB1	L^	076 094 138	4C 5E 8A	JMP &8A5E
8CB4	L	076 146 144	4C 92 90	JMP &9092
8CB7	(	165 040	A5 28	LDA &28
8CB9	H	072	48	PHA
8CBA	/	032 047 157	20 2F 9D	JSR &9D2F
8CBD		208 245	D0 F5	BNE -11 --> &8CB4
8CBF	h	104	68	PLA
8CC0	(	133 040	85 28	STA &28
8CC2	u	032 117 146	20 75 92	JSR &9275
8CC5		160 255	A0 FF	LDY#&FF
8CC7		128 232	80 E8	BRA -24 --> &8CB1
8CC9		032 204 140	20 CC 8C	JSR &8CCC
8CCC		032 207 140	20 CF 8C	JSR &8CCF
8CCF	)	165 041	A5 29	LDA &29
8CD1		024	18	CLC
8CD2	i	105 004	69 04	ADC#&04
8CD4	)	133 041	85 29	STA &29
8CD6	`	096	60	RTS
8CD7		032 224 142	20 E0 8E	JSR &8EE0
8CDA	)	041 223	29 DF	AND#&DF
8CDC	X	201 088	C9 58	CMP#&58
8CDE	`	096	60	RTS
8CDF		032 224 142	20 E0 8E	JSR &8EE0
8CE2	#	201 035	C9 23	CMP#&23
8CE4	`	096	60	RTS
8CE5		032 224 142	20 E0 8E	JSR &8EE0
8CE8	,	201 044	C9 2C	CMP#&2C
8CEA	`	096	60	RTS
8CEB	7	146 055	92 37	STA (&37)
8CED		024	18	CLC
8CEE		152	98	TYA

```
8CEF    e7      101 055         65 37       ADC &37
8CF1    9       133 057         85 39       STA &39
8CF3            160 000         A0 00       LDY#&00
8CF5            152             98          TYA
8CF6    e8      101 056         65 38       ADC &38
8CF8    :       133 058         85 3A       STA &3A
8CFA            200             C8          INY
8CFB    9       177 057         B1 39       LDA (&39),Y
8CFD    7       145 055         91 37       STA (&37),Y
8CFF            201 013         C9 0D       CMP#&0D
8D01            208 247         D0 F7       BNE -9 --> &8CFA
8D03    `       096             60          RTS
8D04    )       041 015         29 0F       AND#&0F
8D06    =       133 061         85 3D       STA &3D
8D08            162 000         A2 00       LDX#&00
8D0A            160 000         A0 00       LDY#&00
8D0C            200             C8          INY
8D0D    7       177 055         B1 37       LDA (&37),Y
8D0F            032 148 141     20 94 8D    JSR &8D94
8D12    .       144 046         90 2E       BCC 46 --> &8D42
8D14    )       041 015         29 0F       AND#&0F
8D16    H       072             48          PHA
8D17    >       134 062         86 3E       STX &3E
8D19    =       165 061         A5 3D       LDA &3D
8D1B            010             0A          ASL A
8D1C    &>      038 062         26 3E       ROL &3E
8D1E    0       048 031         30 1F       BMI 31 --> &8D3F
8D20            010             0A          ASL A
8D21    &>      038 062         26 3E       ROL &3E
8D23    0       048 026         30 1A       BMI 26 --> &8D3F
8D25    e=      101 061         65 3D       ADC &3D
8D27    =       133 061         85 3D       STA &3D
8D29            138             8A          TXA
8D2A    e>      101 062         65 3E       ADC &3E
8D2C    =       006 061         06 3D       ASL &3D
```

8D2E	*	042	2A	ROL A
8D2F	0	048 014	30 0E	BMI 14 --> &8D3F
8D31		176 012	B0 0C	BCS 12 --> &8D3F
8D33		170	AA	TAX
8D34	h	104	68	PLA
8D35	e=	101 061	65 3D	ADC &3D
8D37	=	133 061	85 3D	STA &3D
8D39		144 209	90 D1	BCC -47 --> &8D0C
8D3B		232	E8	INX
8D3C		016 206	10 CE	BPL -50 --> &8D0C
8D3E	H	072	48	PHA
8D3F	h	104	68	PLA
8D40	8	056	38	SEC
8D41	`	096	60	RTS
8D42		136	88	DEY
8D43		169 141	A9 8D	LDA#&8D
8D45		032 235 140	20 EB 8C	JSR &8CEB
8D48	7	165 055	A5 37	LDA &37
8D4A	9	133 057	85 39	STA &39
8D4C	8	165 056	A5 38	LDA &38
8D4E	:	133 058	85 3A	STA &3A
8D50		032 162 141	20 A2 8D	JSR &8DA2
8D53		032 162 141	20 A2 8D	JSR &8DA2
8D56		032 162 141	20 A2 8D	JSR &8DA2
8D59	9	177 057	B1 39	LDA (&39),Y
8D5B	7	145 055	91 37	STA (&37),Y
8D5D		136	88	DEY
8D5E		208 249	D0 F9	BNE -7 --> &8D59
8D60		160 003	A0 03	LDY#&03
8D62		138	8A	TXA
8D63	@	009 064	09 40	ORA#&40
8D65	9	145 057	91 39	STA (&39),Y
8D67		136	88	DEY
8D68	=	165 061	A5 3D	LDA &3D
8D6A	)?	041 063	29 3F	AND#&3F

8D6C	@	009 064	09 40	ORA#&40
8D6E	9	145 057	91 39	STA (&39),Y
8D70		136	88	DEY
8D71	?	169 063	A9 3F	LDA#&3F
8D73	=	020 061	14 3D	TRB &3D
8D75		138	8A	TXA
8D76	)	041 192	29 C0	AND#&C0
8D78	J	074	4A	LSR A
8D79	J	074	4A	LSR A
8D7A	=	005 061	05 3D	ORA &3D
8D7C	J	074	4A	LSR A
8D7D	J	074	4A	LSR A
8D7E	IT	073 084	49 54	EOR#&54
8D80	9	145 057	91 39	STA (&39),Y
8D82		024	18	CLC
8D83	`	096	60	RTS
8D84	{	201 123	C9 7B	CMP#&7B
8D86		176 250	B0 FA	BCS -6 --> &8D82
8D88	_	201 095	C9 5F	CMP#&5F
8D8A		176 014	B0 0E	BCS 14 --> &8D9A
8D8C	[	201 091	C9 5B	CMP#&5B
8D8E		176 242	B0 F2	BCS -14 --> &8D82
8D90	A	201 065	C9 41	CMP#&41
8D92		176 006	B0 06	BCS 6 --> &8D9A
8D94	:	201 058	C9 3A	CMP#&3A
8D96		176 234	B0 EA	BCS -22 --> &8D82
8D98	0	201 048	C9 30	CMP#&30
8D9A	`	096	60	RTS
8D9B	.	201 046	C9 2E	CMP#&2E
8D9D		208 245	D0 F5	BNE -11 --> &8D94
8D9F	`	096	60	RTS
8DA0	7	178 055	B2 37	LDA (&37)
8DA2	7	230 055	E6 37	INC &37
8DA4	9	208 057	D0 39	BNE 57 --> &8DDF
8DA6	8	230 056	E6 38	INC &38

8DA8	`	096	60	RTS
8DA9		032 162 141	20 A2 8D	JSR &8DA2
8DAC	7	178 055	B2 37	LDA (&37)
8DAE	`	096	60	RTS
8DAF		032 162 141	20 A2 8D	JSR &8DA2
8DB2	7	178 055	B2 37	LDA (&37)
8DB4		201 013	C9 0D	CMP#&0D
8DB6	'	240 039	F0 27	BEQ 39 --> &8DDF
8DB8		201 032	C9 20	CMP#&20
8DBA		240 243	F0 F3	BEQ -13 --> &8DAF
8DBC	&	201 038	C9 26	CMP#&26
8DBE		208 016	D0 10	BNE 16 --> &8DD0
8DC0		032 169 141	20 A9 8D	JSR &8DA9
8DC3		032 148 141	20 94 8D	JSR &8D94
8DC6		176 248	B0 F8	BCS -8 --> &8DC0
8DC8	A	201 065	C9 41	CMP#&41
8DCA		144 230	90 E6	BCC -26 --> &8DB2
8DCC	G	201 071	C9 47	CMP#&47
8DCE		144 240	90 F0	BCC -16 --> &8DC0
8DD0	"	201 034	C9 22	CMP#&22
8DD2		208 012	D0 0C	BNE 12 --> &8DE0
8DD4		032 169 141	20 A9 8D	JSR &8DA9
8DD7	"	201 034	C9 22	CMP#&22
8DD9		240 212	F0 D4	BEQ -44 --> &8DAF
8DDB		201 013	C9 0D	CMP#&0D
8DDD		208 245	D0 F5	BNE -11 --> &8DD4
8DDF	`	096	60	RTS
8DE0	:	201 058	C9 3A	CMP#&3A
8DE2		208 009	D0 09	BNE 9 --> &8DED
8DE4		032 162 141	20 A2 8D	JSR &8DA2
8DE7	d;	100 059	64 3B	STZ &3B
8DE9	d<	100 060	64 3C	STZ &3C
8DEB		128 197	80 C5	BRA -59 --> &8DB2
8DED	,	201 044	C9 2C	CMP#&2C
8DEF		240 190	F0 BE	BEQ -66 --> &8DAF

```
8DF1    *       201 042         C9 2A       CMP#&2A
8DF3            208 012         D0 0C       BNE 12 --> &8E01
8DF5    ;       165 059         A5 3B       LDA &3B
8DF7            240 230         F0 E6       BEQ -26 --> &8DDF
8DF9            162 255         A2 FF       LDX#&FF
8DFB    ;       134 059         86 3B       STX &3B
8DFD    d<      100 060         64 3C       STZ &3C
8DFF            128 174         80 AE       BRA -82 --> &8DAF
8E01    .       201 046         C9 2E       CMP#&2E
8E03            240 014         F0 0E       BEQ 14 --> &8E13
8E05            032 148 141     20 94 8D    JSR &8D94
8E08    ,       144 044         90 2C       BCC 44 --> &8E36
8E0A    <       166 060         A6 3C       LDX &3C
8E0C            240 005         F0 05       BEQ 5 --> &8E13
8E0E            032 004 141     20 04 8D    JSR &8D04
8E11            144 156         90 9C       BCC -100 --> &8DAF
8E13    7       178 055         B2 37       LDA (&37)
8E15            032 155 141     20 9B 8D    JSR &8D9B
8E18            144 005         90 05       BCC 5 --> &8E1F
8E1A            032 162 141     20 A2 8D    JSR &8DA2
8E1D            128 244         80 F4       BRA -12 --> &8E13
8E1F            162 255         A2 FF       LDX#&FF
8E21    ;       134 059         86 3B       STX &3B
8E23            128 196         80 C4       BRA -60 --> &8DE9
8E25            032 132 141     20 84 8D    JSR &8D84
8E28            144 207         90 CF       BCC -49 --> &8DF9
8E2A    7       178 055         B2 37       LDA (&37)
8E2C            032 132 141     20 84 8D    JSR &8D84
8E2F            144 238         90 EE       BCC -18 --> &8E1F
8E31            032 162 141     20 A2 8D    JSR &8DA2
8E34            128 244         80 F4       BRA -12 --> &8E2A
8E36    A       201 065         C9 41       CMP#&41
8E38            144 191         90 BF       BCC -65 --> &8DF9
8E3A    X       201 088         C9 58       CMP#&58
8E3C            176 231         B0 E7       BCS -25 --> &8E25
```

8E3E	V	162 086	A2 56	LDX#&56
8E40	9	134 057	86 39	STX &39
8E42		162 132	A2 84	LDX#&84
8E44	:	134 058	86 3A	STX &3A
8E46		160 000	A0 00	LDY#&00
8E48	9	210 057	D2 39	CMP (&39)
8E4A		144 222	90 DE	BCC -34 --> &8E2A
8E4C		208 015	D0 0F	BNE 15 --> &8E5D
8E4E		200	C8	INY
8E4F	9	177 057	B1 39	LDA (&39),Y
8E51	01	048 049	30 31	BMI 49 --> &8E84
8E53	7	209 055	D1 37	CMP (&37),Y
8E55		240 247	F0 F7	BEQ -9 --> &8E4E
8E57	7	177 055	B1 37	LDA (&37),Y
8E59	.	201 046	C9 2E	CMP#&2E
8E5B		240 011	F0 0B	BEQ 11 --> &8E68
8E5D		200	C8	INY
8E5E	9	177 057	B1 39	LDA (&39),Y
8E60		016 251	10 FB	BPL -5 --> &8E5D
8E62		201 254	C9 FE	CMP#&FE
8E64		208 015	D0 0F	BNE 15 --> &8E75
8E66		176 194	B0 C2	BCS -62 --> &8E2A
8E68		200	C8	INY
8E69	9	177 057	B1 39	LDA (&39),Y
8E6B	0	048 023	30 17	BMI 23 --> &8E84
8E6D	9	230 057	E6 39	INC &39
8E6F		208 248	D0 F8	BNE -8 --> &8E69
8E71	:	230 058	E6 3A	INC &3A
8E73		128 244	80 F4	BRA -12 --> &8E69
8E75	8	056	38	SEC
8E76		200	C8	INY
8E77		152	98	TYA
8E78	e9	101 057	65 39	ADC &39
8E7A	9	133 057	85 39	STA &39
8E7C		144 002	90 02	BCC 2 --> &8E80

8E7E	:	230 058	E6 3A	INC &3A
8E80	7	178 055	B2 37	LDA (&37)
8E82		128 194	80 C2	BRA -62 --> &8E46
8E84		170	AA	TAX
8E85		200	C8	INY
8E86	9	177 057	B1 39	LDA (&39),Y
8E88	=	133 061	85 3D	STA &3D
8E8A		136	88	DEY
8E8B	J	074	4A	LSR A
8E8C		144 007	90 07	BCC 7 --> &8E95
8E8E	7	177 055	B1 37	LDA (&37),Y
8E90		032 132 141	20 84 8D	JSR &8D84
8E93		176 149	B0 95	BCS -107 --> &8E2A
8E95		138	8A	TXA
8E96	$=	036 061	24 3D	BIT &3D
8E98	P	080 006	50 06	BVC 6 --> &8EA0
8E9A	;	166 059	A6 3B	LDX &3B
8E9C		208 002	D0 02	BNE 2 --> &8EA0
8E9E	i@	105 064	69 40	ADC#&40
8EA0		136	88	DEY
8EA1		032 235 140	20 EB 8C	JSR &8CEB
8EA4		162 255	A2 FF	LDX#&FF
8EA6	=	165 061	A5 3D	LDA &3D
8EA8	J	074	4A	LSR A
8EA9	J	074	4A	LSR A
8EAA		144 004	90 04	BCC 4 --> &8EB0
8EAC	;	134 059	86 3B	STX &3B
8EAE	d<	100 060	64 3C	STZ &3C
8EB0	J	074	4A	LSR A
8EB1		144 004	90 04	BCC 4 --> &8EB7
8EB3	d;	100 059	64 3B	STZ &3B
8EB5	d<	100 060	64 3C	STZ &3C
8EB7	J	074	4A	LSR A
8EB8		144 016	90 10	BCC 16 --> &8ECA
8EBA	H	072	48	PHA

```
8EBB        160 001         A0 01       LDY#&01
8EBD   7    177 055         B1 37       LDA (&37),Y
8EBF        032 132 141     20 84 8D    JSR &8D84
8EC2        144 005         90 05       BCC 5 --> &8EC9
8EC4        032 162 141     20 A2 8D    JSR &8DA2
8EC7        128 244         80 F4       BRA -12 --> &8EBD
8EC9   h    104             68          PLA
8ECA   J    074             4A          LSR A
8ECB        144 002         90 02       BCC 2 --> &8ECF
8ECD   <    134 060         86 3C       STX &3C
8ECF   J    074             4A          LSR A
8ED0        176 013         B0 0D       BCS 13 --> &8EDF
8ED2   L    076 175 141     4C AF 8D    JMP &8DAF
8ED5        164 027         A4 1B       LDY &1B
8ED7        230 027         E6 1B       INC &1B
8ED9        177 025         B1 19       LDA (&19),Y
8EDB        201 032         C9 20       CMP#&20
8EDD        240 246         F0 F6       BEQ -10 --> &8ED5
8EDF   `    096             60          RTS
8EE0        164 010         A4 0A       LDY &0A
8EE2        230 010         E6 0A       INC &0A
8EE4        177 011         B1 0B       LDA (&0B),Y
8EE6        201 032         C9 20       CMP#&20
8EE8        240 246         F0 F6       BEQ -10 --> &8EE0
8EEA   `    096             60          RTS
8EEB        032 213 142     20 D5 8E    JSR &8ED5
8EEE   ,    201 044         C9 2C       CMP#&2C
8EF0   `    096             60          RTS
8EF1        032 235 142     20 EB 8E    JSR &8EEB
8EF4        240 244         F0 F4       BEQ -12 --> &8EEA
8EF6        000             00          BRK
8EF7        005             05          EQUB &05
8EF8        141             8D          xxx Invalid Code
8EF9   ,    044             2C          xxx Invalid Code
8EFA        000             00          EQUB &00
```

```
8EFB        032 215 189     20 D7 BD    JSR &BDD7
8EFE        128 021         80 15       BRA 21 --> &8F15
8F00        032 166 155     20 A6 9B    JSR &9BA6
8F03        165 024         A5 18       LDA &18
8F05    8   133 056         85 38       STA &38
8F07    d7  100 055         64 37       STZ &37
8F09        169 000         A9 00       LDA#&00
8F0B    7   145 055         91 37       STA (&37),Y
8F0D        032 229 189     20 E5 BD    JSR &BDE5
8F10    q   128 113         80 71       BRA 113 --> &8F83
8F12        032 166 155     20 A6 9B    JSR &9BA6
8F15        032 172 187     20 AC BB    JSR &BBAC
8F18        165 024         A5 18       LDA &18
8F1A        133 012         85 0C       STA &0C
8F1C    d   100 011         64 0B       STZ &0B
8F1E    w   128 119         80 77       BRA 119 --> &8F97
8F20        032 215 189     20 D7 BD    JSR &BDD7
8F23    ^   128 094         80 5E       BRA 94 --> &8F83
8F25        032 166 155     20 A6 9B    JSR &9BA6
8F28        032 229 189     20 E5 BD    JSR &BDE5
8F2B    Y   128 089         80 59       BRA 89 --> &8F86
8F2D        169 242         A9 F2       LDA#&F2
8F2F        032 024 174     20 18 AE    JSR &AE18
8F32        032 226 190     20 E2 BE    JSR &BEE2
8F35        170             AA          TAX
8F36        032 226 190     20 E2 BE    JSR &BEE2
8F39    +   133 043         85 2B       STA &2B
8F3B    *   134 042         86 2A       STX &2A
8F3D        162 020         A2 14       LDX#&14
8F3F        202             CA          DEX
8F40    >   240 062         F0 3E       BEQ 62 --> &8F80
8F42        032 226 190     20 E2 BE    JSR &BEE2
8F45        201 013         C9 0D       CMP#&0D
8F47    7   240 055         F0 37       BEQ 55 --> &8F80
8F49    @   201 064         C9 40       CMP#&40
```

8F4B		208 242	D0 F2	BNE -14 --> &8F3F
8F4D		032 226 190	20 E2 BE	JSR &BEE2
8F50		201 013	C9 0D	CMP#&0D
8F52	,	208 044	D0 2C	BNE 44 --> &8F80
8F54		032 254 190	20 FE BE	JSR &BEFE
8F57		160 000	A0 00	LDY#&00
8F59	d	100 011	64 0B	STZ &0B
8F5B		169 007	A9 07	LDA#&07
8F5D		133 012	85 0C	STA &0C
8F5F		178 000	B2 00	LDA (&00)
8F61		240 032	F0 20	BEQ 32 --> &8F83
8F63		145 011	91 0B	STA (&0B),Y
8F65		200	C8	INY
8F66		240 024	F0 18	BEQ 24 --> &8F80
8F68		230 000	E6 00	INC &00
8F6A		208 002	D0 02	BNE 2 --> &8F6E
8F6C		230 001	E6 01	INC &01
8F6E		201 013	C9 0D	CMP#&0D
8F70		208 237	D0 ED	BNE -19 --> &8F5F
8F72		165 001	A5 01	LDA &01
8F74		197 007	C5 07	CMP &07
8F76		176 011	B0 0B	BCS 11 --> &8F83
8F78		032 235 186	20 EB BA	JSR &BAEB
8F7B		128 218	80 DA	BRA -38 --> &8F57
8F7D		032 166 155	20 A6 9B	JSR &9BA6
8F80		032 254 190	20 FE BE	JSR &BEFE
8F83		032 172 187	20 AC BB	JSR &BBAC
8F86		160 007	A0 07	LDY#&07
8F88		132 012	84 0C	STY &0C
8F8A	d	100 011	64 0B	STZ &0B
8F8C		032 166 178	20 A6 B2	JSR &B2A6
8F8F	>	169 062	A9 3E	LDA#&3E
8F91		032 238 255	20 EE FF	JSR &FFEE
8F94	t	032 116 186	20 74 BA	JSR &BA74
8F97		162 255	A2 FF	LDX#&FF

8F99		154	9A	TXS
8F9A		032 166 178	20 A6 B2	JSR &B2A6
8F9D		032 235 186	20 EB BA	JSR &BAEB
8FA0		176 225	B0 E1	BCS -31 --> &8F83
8FA2	z	128 122	80 7A	BRA 122 --> &901E
8FA4		032 188 155	20 BC 9B	JSR &9BBC
8FA7		166 011	A6 0B	LDX &0B
8FA9		164 012	A4 0C	LDY &0C
8FAB		032 247 255	20 F7 FF	JSR &FFF7
8FAE		169 013	A9 0D	LDA#&0D
8FB0		164 010	A4 0A	LDY &0A
8FB2		136	88	DEY
8FB3		200	C8	INY
8FB4		209 011	D1 0B	CMP (&0B),Y
8FB6		208 251	D0 FB	BNE -5 --> &8FB3
8FB8		032 188 155	20 BC 9B	JSR &9BBC
8FBB		128 004	80 04	BRA 4 --> &8FC1
8FBD		201 013	C9 0D	CMP#&0D
8FBF		208 237	D0 ED	BNE -19 --> &8FAE
8FC1		165 012	A5 0C	LDA &0C
8FC3		201 007	C9 07	CMP#&07
8FC5		240 191	F0 BF	BEQ -65 --> &8F86
8FC7		160 001	A0 01	LDY#&01
8FC9		177 011	B1 0B	LDA (&0B),Y
8FCB	0	048 185	30 B9	BMI -71 --> &8F86
8FCD		166 032	A6 20	LDX &20
8FCF		240 010	F0 0A	BEQ 10 --> &8FDB
8FD1	+	133 043	85 2B	STA &2B
8FD3		200	C8	INY
8FD4		177 011	B1 0B	LDA (&0B),Y
8FD6	*	133 042	85 2A	STA &2A
8FD8	K	032 075 156	20 4B 9C	JSR &9C4B
8FDB		160 004	A0 04	LDY#&04
8FDD		132 010	84 0A	STY &0A
8FDF	,	128 044	80 2C	BRA 44 --> &900D

8FE1		169 003	A9 03	LDA#&03
8FE3	(	133 040	85 28	STA &28
8FE5	L	076 032 137	4C 20 89	JMP &8920
8FE8	L	076 147 190	4C 93 BE	JMP &BE93
8FEB		164 010	A4 0A	LDY &0A
8FED		136	88	DEY
8FEE		177 011	B1 0B	LDA (&0B),Y
8FF0	*	201 042	C9 2A	CMP#&2A
8FF2		240 176	F0 B0	BEQ -80 --> &8FA4
8FF4	[	201 091	C9 5B	CMP#&5B
8FF6		240 233	F0 E9	BEQ -23 --> &8FE1
8FF8		201 162	C9 A2	CMP#&A2
8FFA		240 236	F0 EC	BEQ -20 --> &8FE8
8FFC	=	201 061	C9 3D	CMP#&3D
8FFE	`	240 096	F0 60	BEQ 96 --> &9060
9000		198 010	C6 0A	DEC &0A

```
8000      201 001     C9 01     CMP#&01
8002  '   240 039     F0 27     BEQ 39   --> &802B
8004 `    096         60        RTS
8005      234         EA        NOP
8006 `    096         60        RTS
8007   B  014 004 066 0E 04 42  ASL &4204
800A AS   065 083     41 53     EOR (&53,X)
800C IC   073 067     49 43     EOR#&43
800E      000         00        BRK
800F (    040         28        PLP
8010 C    067         43        xxx Invalid Code
8011 )1   041 049     29 31     AND#&31
8013 984  057 056 052 39 38 34  AND &3438,Y
8016  Ac  032 065 099 20 41 63  JSR &6341
8019 o    111         6F        xxx Invalid Code
801A rn   114 110     72 6E     ADC (&6E)
801C      010         0A        ASL A
801D      013 000 000 0D 00 00  ORA &0000
8020      128 000     80 00     BRA 0   --> &8022
8022      000         00        BRK
8023      000         00        BRK
8024      003         03        xxx Invalid Code
8025 '    039         27        xxx Invalid Code
8026      001 010     01 0A     ORA (&0A,X)
8028 d    100 232     64 E8     STZ &E8
802A      016         10        xxx Invalid Code
802B %    037 017     25 11     AND &11
802D      005 013     05 0D     ORA &0D
802F      005 014     05 0E     ORA &0E
8031      005 015     05 0F     ORA &0F
8033      005 016     05 10     ORA &10
8035      208 012     D0 0C     BNE 12   --> &8043
8037  A   169 065     A9 41     LDA#&41
8039      133 013     85 0D     STA &0D
803B I    073 019     49 13     EOR#&13
803D      133 014     85 0E     STA &0E
803F I    073 005     49 05     EOR#&05
8041      133 015     85 0F     STA &0F
8043      169 132     A9 84     LDA#&84
8045      032 244 255 20 F4 FF  JSR &FFF4
8048      134 006     86 06     STX &06
804A      132 007     84 07     STY &07
```

```
804C :   058        3A       DEC A
804D     032 244 255 20 F4 FF  JSR &FFF4
8050     132 024     84 18    STY &18
8052 d   100 031     64 1F    STZ &1F
8054     156 002 004 9C 02 04  STZ &0402
8057     156 003 004 9C 03 04  STZ &0403
805A     162 255     A2 FF    LDX#&FF
805C  #  134 035     86 23    STX &23
805E     162 010     A2 0A    LDX#&0A
8060     142 000 004 8E 00 04  STX &0400
8063     202         CA       DEX
8064     142 001 004 8E 01 04  STX &0401
8067  x  169 120     A9 78    LDA#&78
8069     141 002 002 8D 02 02  STA &0202
806C     169 178     A9 B2    LDA#&B2
806E     141 003 002 8D 03 02  STA &0203
8071 X   088         58       CLI
8072 L-  076 045 143 4C 2D 8F  JMP &8F2D
8075  9  132 057     84 39    STY &39
8077     160 001     A0 01    LDY#&01
8079  7  177 055     B1 37    LDA (&37),Y
807B     160 246     A0 F6    LDY#&F6
807D     201 242     C9 F2    CMP#&F2
807F     240 012     F0 0C    BEQ 12   --> &808D
8081     160 248     A0 F8    LDY#&F8
8083     128 008     80 08    BRA 8   --> &808D
8085  9  132 057     84 39    STY &39
8087     160 001     A0 01    LDY#&01
8089  7  177 055     B1 37    LDA (&37),Y
808B     010         0A       ASL A
808C     168         A8       TAY
808D     185 001 004 B9 01 04  LDA &0401,Y
8090  :  240 058     F0 3A    BEQ 58   --> &80CC
8092  +  133 043     85 2B    STA &2B
8094     185 000 004 B9 00 04  LDA &0400,Y
8097     128 011     80 0B    BRA 11   --> &80A4
8099     160 001     A0 01    LDY#&01
809B  *  177 042     B1 2A    LDA (&2A),Y
809D  -  240 045     F0 2D    BEQ 45   --> &80CC
809F     168         A8       TAY
80A0  *  178 042     B2 2A    LDA (&2A)
80A2  +  132 043     84 2B    STY &2B
80A4  *  133 042     85 2A    STA &2A
```

```
80A6      160 002     A0 02     LDY#&02
80A8  *   177 042     B1 2A     LDA (&2A),Y
80AA      208 010     D0 0A     BNE 10   --> &80B6
80AC  9   196 057     C4 39     CPY &39
80AE      208 233     D0 E9     BNE -23   --> &8099
80B0      128 017     80 11     BRA 17   --> &80C3
80B2  *   177 042     B1 2A     LDA (&2A),Y
80B4      240 227     F0 E3     BEQ -29   --> &8099
80B6  7   209 055     D1 37     CMP (&37),Y
80B8      208 223     D0 DF     BNE -33   --> &8099
80BA      200         C8        INY
80BB  9   196 057     C4 39     CPY &39
80BD      208 243     D0 F3     BNE -13   --> &80B2
80BF  *   177 042     B1 2A     LDA (&2A),Y
80C1      208 214     D0 D6     BNE -42   --> &8099
80C3      152         98        TYA
80C4 e*   101 042     65 2A     ADC &2A
80C6  *   133 042     85 2A     STA &2A
80C8      144 002     90 02     BCC 2   --> &80CC
80CA  +   230 043     E6 2B     INC &2B
80CC `    096         60        RTS
80CD d=   100 061     64 3D     STZ &3D
80CF      165 024     A5 18     LDA &18
80D1  >   133 062     85 3E     STA &3E
80D3      160 001     A0 01     LDY#&01
80D5  =   177 061     B1 3D     LDA (&3D),Y
80D7  +   197 043     C5 2B     CMP &2B
80D9      176 014     B0 0E     BCS 14   --> &80E9
80DB      160 003     A0 03     LDY#&03
80DD  =   177 061     B1 3D     LDA (&3D),Y
80DF e=   101 061     65 3D     ADC &3D
80E1  =   133 061     85 3D     STA &3D
80E3      144 238     90 EE     BCC -18   --> &80D3
80E5  >   230 062     E6 3E     INC &3E
80E7      128 234     80 EA     BRA -22   --> &80D3
80E9      208 010     D0 0A     BNE 10   --> &80F5
80EB      200         C8        INY
80EC  =   177 061     B1 3D     LDA (&3D),Y
80EE  *   197 042     C5 2A     CMP &2A
80F0      144 233     90 E9     BCC -23   --> &80DB
80F2      208 001     D0 01     BNE 1   --> &80F5
80F4 `    096         60        RTS
80F5      160 002     A0 02     LDY#&02
```

```
80F7      024         18        CLC
80F8 `    096         60        RTS
80F9      032 190 150 20 BE 96  JSR &96BE
80FC  -   165 045     A5 2D     LDA &2D
80FE H    072         48        PHA
80FF      032 190 172 20 BE AC  JSR &ACBE
8102      032 015 160 20 0F A0  JSR &A00F
8105  '   134 039     86 27     STX &27
8107      032 190 150 20 BE 96  JSR &96BE
810A h    104         68        PLA
810B  8   133 056     85 38     STA &38
810D E-   069 045     45 2D     EOR &2D
810F  7   133 055     85 37     STA &37
8111      032 190 172 20 BE AC  JSR &ACBE
8114  9   162 057     A2 39     LDX#&39
8116      032 008 189 20 08 BD  JSR &BD08
8119 d=   100 061     64 3D     STZ &3D
811B d>   100 062     64 3E     STZ &3E
811D d?   100 063     64 3F     STZ &3F
811F d@   100 064     64 40     STZ &40
8121  -   165 045     A5 2D     LDA &2D
8123  *   005 042     05 2A     ORA &2A
8125  +   005 043     05 2B     ORA &2B
8127  ,   005 044     05 2C     ORA &2C
8129  G   240 071     F0 47     BEQ 71   --> &8172
812B      160 032     A0 20     LDY#&20
812D      136         88        DEY
812E  A   240 065     F0 41     BEQ 65   --> &8171
8130  9   006 057     06 39     ASL &39
8132 &:   038 058     26 3A     ROL &3A
8134 &;   038 059     26 3B     ROL &3B
8136 &<   038 060     26 3C     ROL &3C
8138      016 243     10 F3     BPL -13   --> &812D
813A &9   038 057     26 39     ROL &39
813C &:   038 058     26 3A     ROL &3A
813E &;   038 059     26 3B     ROL &3B
8140 &<   038 060     26 3C     ROL &3C
8142 &=   038 061     26 3D     ROL &3D
8144 &>   038 062     26 3E     ROL &3E
8146 &?   038 063     26 3F     ROL &3F
8148 &@   038 064     26 40     ROL &40
814A 8    056         38        SEC
814B  =   165 061     A5 3D     LDA &3D
```

```
814D  *   229 042     E5 2A     SBC &2A
814F H    072         48        PHA
8150  >   165 062     A5 3E     LDA &3E
8152  +   229 043     E5 2B     SBC &2B
8154 H    072         48        PHA
8155  ?   165 063     A5 3F     LDA &3F
8157  ,   229 044     E5 2C     SBC &2C
8159      170         AA        TAX
815A  @   165 064     A5 40     LDA &40
815C  -   229 045     E5 2D     SBC &2D
815E      144 012     90 0C     BCC 12   --> &816C
8160  @   133 064     85 40     STA &40
8162  ?   134 063     86 3F     STX &3F
8164 h    104         68        PLA
8165  >   133 062     85 3E     STA &3E
8167 h    104         68        PLA
8168  =   133 061     85 3D     STA &3D
816A      176 002     B0 02     BCS 2   --> &816E
816C h    104         68        PLA
816D h    104         68        PLA
816E      136         88        DEY
816F      208 201     D0 C9     BNE -55   --> &813A
8171 `    096         60        RTS
8172      000         00        BRK
8173      018         12        EQUB &12
8174 D    068         44        "D"
8175 i    105         69        "i"
8176 v    118         76        "v"
8177 i    105         69        "i"
8178 s    115         73        "s"
8179 i    105         69        "i"
817A o    111         6F        "o"
817B n    110         6E        "n"
817C      032         20        " "
817D b    098         62        "b"
817E y    121         79        "y"
817F      032         20        " "
8180 z    122         7A        "z"
8181 e    101         65        "e"
8182 r    114         72        "r"
8183 o    111         6F        "o"
8184      000         00        EQUB &00
8185 d5   100 053     64 35     STZ &35
```

```
8187 d/   100 047     64 2F     STZ &2F
8189  -   165 045     A5 2D     LDA &2D
818B  .   133 046     85 2E     STA &2E
818D      016 005     10 05     BPL 5   --> &8194
818F      032 222 172 20 DE AC  JSR &ACDE
8192  -   165 045     A5 2D     LDA &2D
8194  &   208 038     D0 26     BNE 38   --> &81BC
8196 d4   100 052     64 34     STZ &34
8198  ,   165 044     A5 2C     LDA &2C
819A      208 020     D0 14     BNE 20   --> &81B0
819C d3   100 051     64 33     STZ &33
819E  +   165 043     A5 2B     LDA &2B
81A0      208 006     D0 06     BNE 6   --> &81A8
81A2 d2   100 050     64 32     STZ &32
81A4  *   165 042     A5 2A     LDA &2A
81A6  8   128 056     80 38     BRA 56   --> &81E0
81A8  *   164 042     A4 2A     LDY &2A
81AA  2   132 050     84 32     STY &32
81AC      160 144     A0 90     LDY#&90
81AE  2   128 050     80 32     BRA 50   --> &81E2
81B0  +   164 043     A4 2B     LDY &2B
81B2  2   132 050     84 32     STY &32
81B4  *   164 042     A4 2A     LDY &2A
81B6  3   132 051     84 33     STY &33
81B8      160 152     A0 98     LDY#&98
81BA  &   128 038     80 26     BRA 38   --> &81E2
81BC  ,   164 044     A4 2C     LDY &2C
81BE  2   132 050     84 32     STY &32
81C0  +   164 043     A4 2B     LDY &2B
81C2  3   132 051     84 33     STY &33
81C4  *   164 042     A4 2A     LDY &2A
81C6  4   132 052     84 34     STY &34
81C8      160 160     A0 A0     LDY#&A0
81CA      128 022     80 16     BRA 22   --> &81E2
81CC d.   100 046     64 2E     STZ &2E
81CE d0   100 048     64 30     STZ &30
81D0 d/   100 047     64 2F     STZ &2F
81D2 d1   100 049     64 31     STZ &31
81D4 `    096         60        RTS
81D5      032 180 166 20 B4 A6  JSR &A6B4
81D8      168         A8        TAY
81D9      016 005     10 05     BPL 5   --> &81E0
81DB  .   133 046     85 2E     STA &2E
```

```
81DD I    073 255     49 FF     EOR#&FF
81DF      026         1A        INC A
81E0      160 136     A0 88     LDY#&88
81E2      009 000     09 00     ORA#&00
81E4 0    048 012     30 0C     BMI 12   --> &81F2
81E6      240 228     F0 E4     BEQ -28   --> &81CC
81E8      136         88        DEY
81E9  4   006 052     06 34     ASL &34
81EB &3   038 051     26 33     ROL &33
81ED &2   038 050     26 32     ROL &32
81EF *    042         2A        ROL A
81F0      016 246     10 F6     BPL -10   --> &81E8
81F2  1   133 049     85 31     STA &31
81F4  0   132 048     84 30     STY &30
81F6 `    096         60        RTS
81F7  1   165 049     A5 31     LDA &31
81F9 0    048 217     30 D9     BMI -39   --> &81D4
81FB  -   208 045     D0 2D     BNE 45   --> &822A
81FD  2   005 050     05 32     ORA &32
81FF  3   005 051     05 33     ORA &33
8201  4   005 052     05 34     ORA &34
8203  5   005 053     05 35     ORA &35
8205      240 197     F0 C5     BEQ -59   --> &81CC
8207  0   165 048     A5 30     LDA &30
8209  2   164 050     A4 32     LDY &32
820B  1   132 049     84 31     STY &31
820D  3   164 051     A4 33     LDY &33
820F  2   132 050     84 32     STY &32
8211  4   164 052     A4 34     LDY &34
8213  3   132 051     84 33     STY &33
8215  5   164 053     A4 35     LDY &35
8217  4   132 052     84 34     STY &34
8219 d5   100 053     64 35     STZ &35
821B 8    056         38        SEC
821C      233 008     E9 08     SBC#&08
821E      176 002     B0 02     BCS 2   --> &8222
8220  /   198 047     C6 2F     DEC &2F
8222  1   164 049     A4 31     LDY &31
8224      240 227     F0 E3     BEQ -29   --> &8209
8226 0    048 023     30 17     BMI 23   --> &823F
8228      128 002     80 02     BRA 2   --> &822C
822A  0   165 048     A5 30     LDA &30
822C      024         18        CLC
```

```
822D      233 000     E9 00     SBC#&00
822F      176 002     B0 02     BCS 2   --> &8233
8231  /   198 047     C6 2F     DEC &2F
8233  5   006 053     06 35     ASL &35
8235 &4   038 052     26 34     ROL &34
8237 &3   038 051     26 33     ROL &33
8239 &2   038 050     26 32     ROL &32
823B &1   038 049     26 31     ROL &31
823D      016 238     10 EE     BPL -18   --> &822D
823F  0   133 048     85 30     STA &30
8241 `    096         60        RTS
8242  0   165 048     A5 30     LDA &30
8244  ,   016 044     10 2C     BPL 44   --> &8272
8246  1   164 049     A4 31     LDY &31
8248  4   240 052     F0 34     BEQ 52   --> &827E
824A F1   070 049     46 31     LSR &31
824C f2   102 050     66 32     ROR &32
824E f3   102 051     66 33     ROR &33
8250 f4   102 052     66 34     ROR &34
8252      026         1A        INC A
8253  h   240 104     F0 68     BEQ 104   --> &82BD
8255      201 160     C9 A0     CMP#&A0
8257  g   176 103     B0 67     BCS 103   --> &82C0
8259      201 153     C9 99     CMP#&99
825B      176 237     B0 ED     BCS -19   --> &824A
825D i    105 008     69 08     ADC#&08
825F  3   164 051     A4 33     LDY &33
8261  4   132 052     84 34     STY &34
8263  2   164 050     A4 32     LDY &32
8265  3   132 051     84 33     STY &33
8267  1   164 049     A4 31     LDY &31
8269  2   132 050     84 32     STY &32
826B d1   100 049     64 31     STZ &31
826D      128 230     80 E6     BRA -26   --> &8255
826F      032 011 164 20 0B A4  JSR &A40B
8272 L    076 180 166 4C B4 A6  JMP &A6B4
8275  0   165 048     A5 30     LDA &30
8277      016 246     10 F6     BPL -10   --> &826F
8279  p   032 112 165 20 70 A5  JSR &A570
827C  1   164 049     A4 31     LDY &31
827E  D   240 068     F0 44     BEQ 68   --> &82C4
8280 F1   070 049     46 31     LSR &31
8282 f2   102 050     66 32     ROR &32
```

```
8284 f3   102 051     66 33     ROR &33
8286 f4   102 052     66 34     ROR &34
8288 f=   102 061     66 3D     ROR &3D
828A f>   102 062     66 3E     ROR &3E
828C f?   102 063     66 3F     ROR &3F
828E f@   102 064     66 40     ROR &40
8290      026         1A        INC A
8291  *   240 042     F0 2A     BEQ 42   --> &82BD
8293      201 160     C9 A0     CMP#&A0
8295  )   176 041     B0 29     BCS 41   --> &82C0
8297      201 153     C9 99     CMP#&99
8299      176 229     B0 E5     BCS -27   --> &8280
829B i    105 008     69 08     ADC#&08
829D  ?   164 063     A4 3F     LDY &3F
829F  @   132 064     84 40     STY &40
82A1  >   164 062     A4 3E     LDY &3E
82A3  ?   132 063     84 3F     STY &3F
82A5  =   164 061     A4 3D     LDY &3D
82A7  >   132 062     84 3E     STY &3E
82A9  4   164 052     A4 34     LDY &34
82AB  =   132 061     84 3D     STY &3D
82AD  3   164 051     A4 33     LDY &33
82AF  4   132 052     84 34     STY &34
82B1  2   164 050     A4 32     LDY &32
82B3  3   132 051     84 33     STY &33
82B5  1   164 049     A4 31     LDY &31
82B7  2   132 050     84 32     STY &32
82B9 d1   100 049     64 31     STZ &31
82BB      128 214     80 D6     BRA -42   --> &8293
82BD L    076 197 166 4C C5 A6  JMP &A6C5
82C0      208 251     D0 FB     BNE -5   --> &82BD
82C2  0   133 048     85 30     STA &30
82C4  .   165 046     A5 2E     LDA &2E
82C6      016 023     10 17     BPL 23   --> &82DF
82C8 8    056         38        SEC
82C9      160 000     A0 00     LDY#&00
82CB      152         98        TYA
82CC  4   229 052     E5 34     SBC &34
82CE  4   133 052     85 34     STA &34
82D0      152         98        TYA
82D1  3   229 051     E5 33     SBC &33
82D3  3   133 051     85 33     STA &33
82D5      152         98        TYA
```

```
82D6  2   229 050     E5 32     SBC &32
82D8  2   133 050     85 32     STA &32
82DA      152         98        TYA
82DB  1   229 049     E5 31     SBC &31
82DD  1   133 049     85 31     STA &31
82DF `    096         60        RTS
82E0  0   165 048     A5 30     LDA &30
82E2 0    048 005     30 05     BMI 5   --> &82E9
82E4 dI   100 073     64 49     STZ &49
82E6 L    076 242 163 4C F2 A3  JMP &A3F2
82E9  u   032 117 130 20 75 82  JSR &8275
82EC  4   165 052     A5 34     LDA &34
82EE  I   133 073     85 49     STA &49
82F0  S   032 083 131 20 53 83  JSR &8353
82F3      169 128     A9 80     LDA#&80
82F5  0   133 048     85 30     STA &30
82F7  1   166 049     A6 31     LDX &31
82F9      016 015     10 0F     BPL 15   --> &830A
82FB E.   069 046     45 2E     EOR &2E
82FD  .   133 046     85 2E     STA &2E
82FF      016 004     10 04     BPL 4   --> &8305
8301  I   230 073     E6 49     INC &49
8303      128 002     80 02     BRA 2   --> &8307
8305  I   198 073     C6 49     DEC &49
8307      032 200 130 20 C8 82  JSR &82C8
830A L    076 247 129 4C F7 81  JMP &81F7
830D  4   230 052     E6 34     INC &34
830F      208 012     D0 0C     BNE 12   --> &831D
8311  3   230 051     E6 33     INC &33
8313      208 008     D0 08     BNE 8   --> &831D
8315  2   230 050     E6 32     INC &32
8317      208 004     D0 04     BNE 4   --> &831D
8319  1   230 049     E6 31     INC &31
831B      240 160     F0 A0     BEQ -96   --> &82BD
831D `    096         60        RTS
831E      160 004     A0 04     LDY#&04
8320 f    102 017     66 11     ROR &11
8322      165 016     A5 10     LDA &10
8324      170         AA        TAX
8325 j    106         6A        ROR A
8326      133 017     85 11     STA &11
8328      165 015     A5 0F     LDA &0F
832A      133 016     85 10     STA &10
```

```
832C J    074         4A        LSR A
832D E    069 014     45 0E     EOR &0E
832F )    041 015     29 0F     AND#&0F
8331 E    069 014     45 0E     EOR &0E
8333 j    106         6A        ROR A
8334 j    106         6A        ROR A
8335 j    106         6A        ROR A
8336 j    106         6A        ROR A
8337 E    069 017     45 11     EOR &11
8339      134 017     86 11     STX &11
833B      166 014     A6 0E     LDX &0E
833D      134 015     86 0F     STX &0F
833F      166 013     A6 0D     LDX &0D
8341      134 014     86 0E     STX &0E
8343      133 013     85 0D     STA &0D
8345      136         88        DEY
8346      208 216     D0 D8     BNE -40   --> &8320
8348 `    096         60        RTS
8349 ;    165 059     A5 3B     LDA &3B
834B .    133 046     85 2E     STA &2E
834D d/   100 047     64 2F     STZ &2F
834F <    165 060     A5 3C     LDA &3C
8351 0    133 048     85 30     STA &30
8353 =    165 061     A5 3D     LDA &3D
8355 1    133 049     85 31     STA &31
8357 >    165 062     A5 3E     LDA &3E
8359 2    133 050     85 32     STA &32
835B ?    165 063     A5 3F     LDA &3F
835D 3    133 051     85 33     STA &33
835F @    165 064     A5 40     LDA &40
8361 4    133 052     85 34     STA &34
8363 A    165 065     A5 41     LDA &41
8365 5    133 053     85 35     STA &35
8367 `    096         60        RTS
8368 1    165 049     A5 31     LDA &31
836A      240 221     F0 DD     BEQ -35   --> &8349
836C 8    056         38        SEC
836D 0    165 048     A5 30     LDA &30
836F <    229 060     E5 3C     SBC &3C
8371 o    240 111     F0 6F     BEQ 111   --> &83E2
8373 4    144 052     90 34     BCC 52    --> &83A9
8375 %    201 037     C9 25     CMP#&25
8377      176 238     B0 EE     BCS -18   --> &8367
```

```
8379      168         A8        TAY
837A )8   041 056     29 38     AND#&38
837C      240 023     F0 17     BEQ 23   --> &8395
837E 8    056         38        SEC
837F  @   166 064     A6 40     LDX &40
8381  A   134 065     86 41     STX &41
8383  ?   166 063     A6 3F     LDX &3F
8385  @   134 064     86 40     STX &40
8387  >   166 062     A6 3E     LDX &3E
8389  ?   134 063     86 3F     STX &3F
838B  =   166 061     A6 3D     LDX &3D
838D  >   134 062     86 3E     STX &3E
838F d=   100 061     64 3D     STZ &3D
8391      233 008     E9 08     SBC#&08
8393      208 234     D0 EA     BNE -22   --> &837F
8395      152         98        TYA
8396 )    041 007     29 07     AND#&07
8398  H   240 072     F0 48     BEQ 72   --> &83E2
839A F=   070 061     46 3D     LSR &3D
839C f>   102 062     66 3E     ROR &3E
839E f?   102 063     66 3F     ROR &3F
83A0 f@   102 064     66 40     ROR &40
83A2 fA   102 065     66 41     ROR &41
83A4 :    058         3A        DEC A
83A5      208 243     D0 F3     BNE -13   --> &839A
83A7  9   128 057     80 39     BRA 57   --> &83E2
83A9 I    073 255     49 FF     EOR#&FF
83AB      026         1A        INC A
83AC  %   201 037     C9 25     CMP#&25
83AE      176 153     B0 99     BCS -103   --> &8349
83B0  <   164 060     A4 3C     LDY &3C
83B2  0   132 048     84 30     STY &30
83B4      168         A8        TAY
83B5 )8   041 056     29 38     AND#&38
83B7      240 023     F0 17     BEQ 23   --> &83D0
83B9 8    056         38        SEC
83BA  4   166 052     A6 34     LDX &34
83BC  5   134 053     86 35     STX &35
83BE  3   166 051     A6 33     LDX &33
83C0  4   134 052     86 34     STX &34
83C2  2   166 050     A6 32     LDX &32
83C4  3   134 051     86 33     STX &33
83C6  1   166 049     A6 31     LDX &31
```

```
83C8  2   134 050     86 32     STX &32
83CA d1   100 049     64 31     STZ &31
83CC      233 008     E9 08     SBC#&08
83CE      208 234     D0 EA     BNE -22   --> &83BA
83D0      152         98        TYA
83D1 )    041 007     29 07     AND#&07
83D3      240 013     F0 0D     BEQ 13   --> &83E2
83D5 F1   070 049     46 31     LSR &31
83D7 f2   102 050     66 32     ROR &32
83D9 f3   102 051     66 33     ROR &33
83DB f4   102 052     66 34     ROR &34
83DD f5   102 053     66 35     ROR &35
83DF :    058         3A        DEC A
83E0      208 243     D0 F3     BNE -13   --> &83D5
83E2  .   165 046     A5 2E     LDA &2E
83E4 E;   069 059     45 3B     EOR &3B
83E6 0    048 004     30 04     BMI 4   --> &83EC
83E8      024         18        CLC
83E9 LG   076 071 164 4C 47 A4  JMP &A447
83EC  1   165 049     A5 31     LDA &31
83EE  =   197 061     C5 3D     CMP &3D
83F0      208 027     D0 1B     BNE 27   --> &840D
83F2  2   165 050     A5 32     LDA &32
83F4  >   197 062     C5 3E     CMP &3E
83F6      208 021     D0 15     BNE 21   --> &840D
83F8  3   165 051     A5 33     LDA &33
83FA  ?   197 063     C5 3F     CMP &3F
83FC      208 015     D0 0F     BNE 15   --> &840D
83FE  4   165 052     A5 34     LDA &34
8400  @   197 064     C5 40     CMP &40
8402      208 009     D0 09     BNE 9   --> &840D
8404  5   165 053     A5 35     LDA &35
8406  A   197 065     C5 41     CMP &41
8408      208 003     D0 03     BNE 3   --> &840D
840A L    076 180 166 4C B4 A6  JMP &A6B4
840D  &   176 038     B0 26     BCS 38   --> &8435
840F  ;   165 059     A5 3B     LDA &3B
8411  .   133 046     85 2E     STA &2E
8413 8    056         38        SEC
8414  A   165 065     A5 41     LDA &41
8416  5   229 053     E5 35     SBC &35
8418  5   133 053     85 35     STA &35
841A  @   165 064     A5 40     LDA &40
```

```
841C  4   229 052     E5 34     SBC &34
841E  4   133 052     85 34     STA &34
8420  ?   165 063     A5 3F     LDA &3F
8422  3   229 051     E5 33     SBC &33
8424  3   133 051     85 33     STA &33
8426  >   165 062     A5 3E     LDA &3E
8428  2   229 050     E5 32     SBC &32
842A  2   133 050     85 32     STA &32
842C  =   165 061     A5 3D     LDA &3D
842E  1   229 049     E5 31     SBC &31
8430  1   133 049     85 31     STA &31
8432 L    076 249 129 4C F9 81  JMP &81F9
8435  5   165 053     A5 35     LDA &35
8437  A   229 065     E5 41     SBC &41
8439  5   133 053     85 35     STA &35
843B  4   165 052     A5 34     LDA &34
843D  @   229 064     E5 40     SBC &40
843F  4   133 052     85 34     STA &34
8441  3   165 051     A5 33     LDA &33
8443  ?   229 063     E5 3F     SBC &3F
8445  3   133 051     85 33     STA &33
8447  2   165 050     A5 32     LDA &32
8449  >   229 062     E5 3E     SBC &3E
844B  2   133 050     85 32     STA &32
844D  1   165 049     A5 31     LDA &31
844F  =   229 061     E5 3D     SBC &3D
8451  1   133 049     85 31     STA &31
8453 L    076 249 129 4C F9 81  JMP &81F9
8456 AN   065 078     41 4E     EOR (&4E,X)
8458 D    068         44        xxx Invalid Code
8459      128 000     80 00     BRA 0   --> &845B
845B AB   065 066     41 42     EOR (&42,X)
845D S    083         53        xxx Invalid Code
845E      148 000     94 00     STY &00,X
8460 AC   065 067     41 43     EOR (&43,X)
8462 S    083         53        xxx Invalid Code
8463      149 000     95 00     STA &00,X
8465 AD   065 068     41 44     EOR (&44,X)
8467 VA   086 065     56 41     LSR &41,X
8469 L    076 150 000 4C 96 00  JMP &0096
846C AS   065 083     41 53     EOR (&53,X)
846E C    067         43        xxx Invalid Code
846F      151         97        xxx Invalid Code
```

```
8470      000         00        BRK
8471 AS   065 083     41 53     EOR (&53,X)
8473 N    078 152 000 4E 98 00  LSR &0098
8476 AT   065 084     41 54     EOR (&54,X)
8478 N    078 153 000 4E 99 00  LSR &0099
847B AU   065 085     41 55     EOR (&55,X)
847D T    084         54        xxx Invalid Code
847E O    079         4F        xxx Invalid Code
847F      198 016     C6 10     DEC &10
8481 B    066         42        xxx Invalid Code
8482 G    071         47        xxx Invalid Code
8483 ET   069 084     45 54     EOR &54
8485      154         9A        TXS
8486  B   001 066     01 42     ORA (&42,X)
8488 PU   080 085     50 55     BVC 85   --> &84DF
848A T    084         54        xxx Invalid Code
848B      213 003     D5 03     CMP &03,X
848D C    067         43        xxx Invalid Code
848E O    079         4F        xxx Invalid Code
848F LOU  076 079 085 4C 4F 55  JMP &554F
8492 R    082 251     52 FB     EOR (&FB)
8494      002         02        xxx Invalid Code
8495 C    067         43        xxx Invalid Code
8496 AL   065 076     41 4C     EOR (&4C,X)
8498 L    076 214 002 4C D6 02  JMP &02D6
849B C    067         43        xxx Invalid Code
849C H    072         48        PHA
849D AI   065 073     41 49     EOR (&49,X)
849F N    078 215 002 4E D7 02  LSR &02D7
84A2 C    067         43        xxx Invalid Code
84A3 H    072         48        PHA
84A4 R$   082 036     52 24     EOR (&24)
84A6   C  189 000 067 BD 00 43  LDA &4300,X
84A9 LEA  076 069 065 4C 45 41  JMP &4145
84AC R    082 216     52 D8     EOR (&D8)
84AE  C   001 067     01 43     ORA (&43,X)
84B0 LOS  076 079 083 4C 4F 53  JMP &534F
84B3 E    069 217     45 D9     EOR &D9
84B5      003         03        xxx Invalid Code
84B6 C    067         43        xxx Invalid Code
84B7 LG   076 071 218 4C 47 DA  JMP &DA47
84BA  C   001 067     01 43     ORA (&43,X)
84BC LS   076 083 219 4C 53 DB  JMP &DB53
```

```
84BF  C   001 067     01 43     ORA (&43,X)
84C1 O    079         4F        xxx Invalid Code
84C2 S    083         53        xxx Invalid Code
84C3      155         9B        xxx Invalid Code
84C4      000         00        BRK
84C5 C    067         43        xxx Invalid Code
84C6 O    079         4F        xxx Invalid Code
84C7 UN   085 078     55 4E     EOR &4E,X
84C9 T    084         54        xxx Invalid Code
84CA   C  156 001 067 9C 01 43  STZ &4301
84CD O    079         4F        xxx Invalid Code
84CE LOR  076 079 082 4C 4F 52  JMP &524F
84D1      251         FB        xxx Invalid Code
84D2      002         02        xxx Invalid Code
84D3 D    068         44        xxx Invalid Code
84D4 AT   065 084     41 54     EOR (&54,X)
84D6 A    065 220     41 DC     EOR (&DC,X)
84D8  DE  032 068 069 20 44 45  JSR &4544
84DB G    071         47        xxx Invalid Code
84DC   D  157 000 068 9D 00 44  STA &4400,X
84DF EF   069 070     45 46     EOR &46
84E1   D  221 000 068 DD 00 44  CMP &4400,X
84E4 EL   069 076     45 4C     EOR &4C
84E6 ET   069 084     45 54     EOR &54
84E8 E    069 199     45 C7     EOR &C7
84EA  D   016 068     10 44     BPL 68   --> &8530
84EC IV   073 086     49 56     EOR#&56
84EE      129 000     81 00     STA (&00,X)
84F0 D    068         44        xxx Invalid Code
84F1 IM   073 077     49 4D     EOR#&4D
84F3   D  222 002 068 DE 02 44  DEC &4402,X
84F6 RA   082 065     52 41     EOR (&41)
84F8 W    087         57        xxx Invalid Code
84F9      223         DF        xxx Invalid Code
84FA      002         02        xxx Invalid Code
84FB EN   069 078     45 4E     EOR &4E
84FD D    068         44        xxx Invalid Code
84FE PR   080 082     50 52     BVC 82   --> &8552
8500 O    079         4F        xxx Invalid Code
8501 C    067         43        xxx Invalid Code
8502      225 001     E1 01     SBC (&01,X)
8504 EN   069 078     45 4E     EOR &4E
8506 D    068         44        xxx Invalid Code
```

```
8507      224 001     E0 01     CPX#&01
8509 EN   069 078     45 4E     EOR &4E
850B VE   086 069     56 45     LSR &45,X
850D LOP  076 079 080 4C 4F 50  JMP &504F
8510 E    069 226     45 E2     EOR &E2
8512      002         02        xxx Invalid Code
8513 EL   069 076     45 4C     EOR &4C
8515 S    083         53        xxx Invalid Code
8516 E    069 139     45 8B     EOR &8B
8518  E   020 069     14 45     TRB &45
851A VA   086 065     56 41     LSR &41,X
851C L    076 160 000 4C A0 00  JMP &00A0
851F ER   069 082     45 52     EOR &52
8521 L    076 158 001 4C 9E 01  JMP &019E
8524 ER   069 082     45 52     EOR &52
8526 RO   082 079     52 4F     EOR (&4F)
8528 R    082 133     52 85     EOR (&85)
852A  E   004 069     04 45     TSB &45
852C O    079         4F        xxx Invalid Code
852D F    070 197     46 C5     LSR &C5
852F  E   001 069     01 45     ORA (&45,X)
8531 O    079         4F        xxx Invalid Code
8532 R    082 130     52 82     EOR (&82)
8534      000         00        BRK
8535 ER   069 082     45 52     EOR &52
8537 R    082 159     52 9F     EOR (&9F)
8539  E   001 069     01 45     ORA (&45,X)
853B X    088         58        CLI
853C P    080 161     50 A1     BVC -95   --> &84DF
853E      000         00        BRK
853F EX   069 088     45 58     EOR &58
8541 T    084         54        xxx Invalid Code
8542      162 001     A2 01     LDX#&01
8544 ED   069 068     45 44     EOR &44
8546 IT   073 084     49 54     EOR#&54
8548      206         CE        xxx Invalid Code
8549  F   016 070     10 46     BPL 70   --> &8591
854B O    079         4F        xxx Invalid Code
854C R    082 227     52 E3     EOR (&E3)
854E      002         02        xxx Invalid Code
854F FA   070 065     46 41     LSR &41
8551 LSE  076 083 069 4C 53 45  JMP &4553
8554      163         A3        xxx Invalid Code
```

```
8555  F   001 070     01 46     ORA (&46,X)
8557 N    078 164 008 4E A4 08  LSR &08A4
855A G     071          47         xxx Invalid Code
855B O     079          4F         xxx Invalid Code
855C T     084          54         xxx Invalid Code
855D O     079          4F         xxx Invalid Code
855E       229 018     E5 12     SBC &12
8560 G     071          47         xxx Invalid Code
8561 ET    069 084     45 54     EOR &54
8563 $     036 190     24 BE     BIT &BE
8565       000          00         BRK
8566 G     071          47         xxx Invalid Code
8567 ET    069 084     45 54     EOR &54
8569       165 000     A5 00     LDA &00
856B G     071          47         xxx Invalid Code
856C O     079          4F         xxx Invalid Code
856D S     083          53         xxx Invalid Code
856E UB    085 066     55 42     EOR &42,X
8570       228 018     E4 12     CPX &12
8572 G     071          47         xxx Invalid Code
8573 C     067          43         xxx Invalid Code
8574 O     079          4F         xxx Invalid Code
8575 L     076 230 002 4C E6 02  JMP &02E6
8578 H     072          48         PHA
8579 IM    073 077     49 4D     EOR#&4D
857B EM    069 077     45 4D     EOR &4D
857D       147          93         xxx Invalid Code
857E C     067          43         xxx Invalid Code
857F IN    073 078     49 4E     EOR#&4E
8581 PU    080 085     50 55     BVC 85   --> &85D8
8583 T     084          54         xxx Invalid Code
8584       232          E8         INX
8585       002          02         xxx Invalid Code
8586 IF    073 070     49 46     EOR#&46
8588       231          E7         xxx Invalid Code
8589       002          02         xxx Invalid Code
858A IN    073 078     49 4E     EOR#&4E
858C K     075          4B         xxx Invalid Code
858D EY    069 089     45 59     EOR &59
858F $     036 191     24 BF     BIT &BF
8591       000          00         BRK
8592 IN    073 078     49 4E     EOR#&4E
8594 K     075          4B         xxx Invalid Code
```

```
8595 EY   069 089     45 59     EOR &59
8597      166 000     A6 00     LDX &00
8599 IN   073 078     49 4E     EOR#&4E
859B T    084         54        xxx Invalid Code
859C      168         A8        TAY
859D      000         00        BRK
859E IN   073 078     49 4E     EOR#&4E
85A0 S    083         53        xxx Invalid Code
85A1 T    084         54        xxx Invalid Code
85A2 R(   082 040     52 28     EOR (&28)
85A4      167         A7        xxx Invalid Code
85A5      000         00        BRK
85A6 LIS  076 073 083 4C 49 53  JMP &5349
85A9 T    084         54        xxx Invalid Code
85AA      201 016     C9 10     CMP#&10
85AC LIN  076 073 078 4C 49 4E  JMP &4E49
85AF E    069 134     45 86     EOR &86
85B1      000         00        BRK
85B2 LOA  076 079 065 4C 4F 41  JMP &414F
85B5 D    068         44        xxx Invalid Code
85B6      200         C8        INY
85B7      002         02        xxx Invalid Code
85B8 LOM  076 079 077 4C 4F 4D  JMP &4D4F
85BB EM   069 077     45 4D     EOR &4D
85BD  C   146 067     92 43     STA (&43)
85BF LOC  076 079 067 4C 4F 43  JMP &434F
85C2 AL   065 076     41 4C     EOR (&4C,X)
85C4      234         EA        NOP
85C5      002         02        xxx Invalid Code
85C6 LEF  076 069 070 4C 45 46  JMP &4645
85C9 T    084         54        xxx Invalid Code
85CA $(   036 040     24 28     BIT &28
85CC      192 000     C0 00     CPY#&00
85CE LEN  076 069 078 4C 45 4E  JMP &4E45
85D1      169 000     A9 00     LDA#&00
85D3 LET  076 069 084 4C 45 54  JMP &5445
85D6      233 004     E9 04     SBC#&04
85D8 LOG  076 079 071 4C 4F 47  JMP &474F
85DB      171         AB        xxx Invalid Code
85DC      000         00        BRK
85DD LN   076 078 170 4C 4E AA  JMP &AA4E
85E0      000         00        BRK
85E1 MID  077 073 068 4D 49 44  EOR &4449
```

```
85E4 $(   036 040     24 28     BIT &28
85E6      193 000     C1 00     CMP (&00,X)
85E8 MOD  077 079 068 4D 4F 44  EOR &444F
85EB E    069 235     45 EB     EOR &EB
85ED      002         02        xxx Invalid Code
85EE MOD  077 079 068 4D 4F 44  EOR &444F
85F1      131         83        xxx Invalid Code
85F2      000         00        BRK
85F3 MOV  077 079 086 4D 4F 56  EOR &564F
85F6 E    069 236     45 EC     EOR &EC
85F8      002         02        xxx Invalid Code
85F9 NEX  078 069 088 4E 45 58  LSR &5845
85FC T    084         54        xxx Invalid Code
85FD   N  237 002 078 ED 02 4E  SBC &4E02
8600 EW   069 087     45 57     EOR &57
8602      202         CA        DEX
8603  N   001 078     01 4E     ORA (&4E,X)
8605 O    079         4F        xxx Invalid Code
8606 T    084         54        xxx Invalid Code
8607   O  172 000 079 AC 00 4F  LDY &4F00
860A LD   076 068 203 4C 44 CB  JMP &CB44
860D  O   001 079     01 4F     ORA (&4F,X)
860F N    078 238 002 4E EE 02  LSR &02EE
8612 O    079         4F        xxx Invalid Code
8613 FF   070 070     46 46     LSR &46
8615      135         87        xxx Invalid Code
8616      000         00        BRK
8617 O    079         4F        xxx Invalid Code
8618 R    082 132     52 84     EOR (&84)
861A      000         00        BRK
861B O    079         4F        xxx Invalid Code
861C PE   080 069     50 45     BVC 69   --> &8663
861E NIN  078 073 078 4E 49 4E  LSR &4E49
8621   O  142 000 079 8E 00 4F  STX &4F00
8624 PE   080 069     50 45     BVC 69   --> &866B
8626 NOU  078 079 085 4E 4F 55  LSR &554F
8629 T    084         54        xxx Invalid Code
862A   O  174 000 079 AE 00 4F  LDX &4F00
862D PE   080 069     50 45     BVC 69   --> &8674
862F NUP  078 085 080 4E 55 50  LSR &5055
8632   O  173 000 079 AD 00 4F  LDA &4F00
8635 S    083         53        xxx Invalid Code
8636 C    067         43        xxx Invalid Code
```

```
8637 LI   076 073 255 4C 49 FF  JMP &FF49
863A      002         02        xxx Invalid Code
863B PR   080 082     50 52     BVC 82   --> &868F
863D IN   073 078     49 4E     EOR#&4E
863F T    084         54        xxx Invalid Code
8640      241 002     F1 02     SBC (&02),Y
8642 PA   080 065     50 41     BVC 65   --> &8685
8644 G    071         47        xxx Invalid Code
8645 E    069 144     45 90     EOR &90
8647 C    067         43        xxx Invalid Code
8648 PT   080 084     50 54     BVC 84   --> &869E
864A R    082 143     52 8F     EOR (&8F)
864C C    067         43        xxx Invalid Code
864D PI   080 073     50 49     BVC 73   --> &8698
864F      175         AF        xxx Invalid Code
8650  P   001 080     01 50     ORA (&50,X)
8652 LOT  076 079 084 4C 4F 54  JMP &544F
8655      240 002     F0 02     BEQ 2    --> &8659
8657 PO   080 079     50 4F     BVC 79   --> &86A8
8659 IN   073 078     49 4E     EOR#&4E
865B T    084         54        xxx Invalid Code
865C (    040         28        PLP
865D      176 000     B0 00     BCS 0    --> &865F
865F PR   080 082     50 52     BVC 82   --> &86B3
8661 O    079         4F        xxx Invalid Code
8662 C    067         43        xxx Invalid Code
8663      242 010     F2 0A     SBC (&0A)
8665 PO   080 079     50 4F     BVC 79   --> &86B6
8667 S    083         53        xxx Invalid Code
8668      177 001     B1 01     LDA (&01),Y
866A RE   082 069     52 45     EOR (&45)
866C T    084         54        xxx Invalid Code
866D UR   085 082     55 52     EOR &52,X
866F N    078 248 001 4E F8 01  LSR &01F8
8672 RE   082 069     52 45     EOR (&45)
8674 PE   080 069     50 45     BVC 69   --> &86BB
8676 AT   065 084     41 54     EOR (&54,X)
8678      245 000     F5 00     SBC &00,X
867A RE   082 069     52 45     EOR (&45)
867C PO   080 079     50 4F     BVC 79   --> &86CD
867E RT   082 084     52 54     EOR (&54)
8680      246 001     F6 01     INC &01,X
8682 RE   082 069     52 45     EOR (&45)
```

```
8684 AD   065 068     41 44     EOR (&44,X)
8686      243         F3        xxx Invalid Code
8687      002         02        xxx Invalid Code
8688 RE   082 069     52 45     EOR (&45)
868A M    077 244 032 4D F4 20  EOR &20F4
868D RU   082 085     52 55     EOR (&55)
868F N    078 249 001 4E F9 01  LSR &01F9
8692 RA   082 065     52 41     EOR (&41)
8694 D    068         44        xxx Invalid Code
8695      178 000     B2 00     LDA (&00)
8697 RE   082 069     52 45     EOR (&45)
8699 S    083         53        xxx Invalid Code
869A T    084         54        xxx Invalid Code
869B O    079         4F        xxx Invalid Code
869C RE   082 069     52 45     EOR (&45)
869E      247         F7        xxx Invalid Code
869F  R   018 082     12 52     ORA (&52)
86A1 IG   073 071     49 47     EOR#&47
86A3 H    072         48        PHA
86A4 T    084         54        xxx Invalid Code
86A5 $(   036 040     24 28     BIT &28
86A7      194         C2        xxx Invalid Code
86A8      000         00        BRK
86A9 RN   082 078     52 4E     EOR (&4E)
86AB D    068         44        xxx Invalid Code
86AC      179         B3        xxx Invalid Code
86AD  R   001 082     01 52     ORA (&52,X)
86AF EN   069 078     45 4E     EOR &4E
86B1 UM   085 077     55 4D     EOR &4D,X
86B3 B    066         42        xxx Invalid Code
86B4 ER   069 082     45 52     EOR &52
86B6   S  204 016 083 CC 10 53  CPY &5310
86B9 T    084         54        xxx Invalid Code
86BA EP   069 080     45 50     EOR &50
86BC      136         88        DEY
86BD      000         00        BRK
86BE S    083         53        xxx Invalid Code
86BF AV   065 086     41 56     EOR (&56,X)
86C1 E    069 205     45 CD     EOR &CD
86C3      002         02        xxx Invalid Code
86C4 S    083         53        xxx Invalid Code
86C5 G    071         47        xxx Invalid Code
86C6 N    078 180 000 4E B4 00  LSR &00B4
```

```
86C9 S    083         53        xxx Invalid Code
86CA IN   073 078     49 4E     EOR#&4E
86CC      181 000     B5 00     LDA &00,X
86CE S    083         53        xxx Invalid Code
86CF QR   081 082     51 52     EOR (&52),Y
86D1      182 000     B6 00     LDX &00,Y
86D3 S    083         53        xxx Invalid Code
86D4 PC   080 067     50 43     BVC 67   --> &8719
86D6      137 000     89 00     BIT#&00
86D8 S    083         53        xxx Invalid Code
86D9 T    084         54        xxx Invalid Code
86DA R$   082 036     52 24     EOR (&24)
86DC      195         C3        xxx Invalid Code
86DD      000         00        BRK
86DE S    083         53        xxx Invalid Code
86DF T    084         54        xxx Invalid Code
86E0 RI   082 073     52 49     EOR (&49)
86E2 NG$  078 071 036 4E 47 24  LSR &2447
86E5 (    040         28        PLP
86E6      196 000     C4 00     CPY &00
86E8 S    083         53        xxx Invalid Code
86E9 O    079         4F        xxx Invalid Code
86EA UN   085 078     55 4E     EOR &4E,X
86EC D    068         44        xxx Invalid Code
86ED      212         D4        xxx Invalid Code
86EE      002         02        xxx Invalid Code
86EF S    083         53        xxx Invalid Code
86F0 T    084         54        xxx Invalid Code
86F1 O    079         4F        xxx Invalid Code
86F2 P    080 250     50 FA     BVC -6   --> &86EE
86F4  T   001 084     01 54     ORA (&54,X)
86F6 AN   065 078     41 4E     EOR (&4E,X)
86F8      183         B7        xxx Invalid Code
86F9      000         00        BRK
86FA T    084         54        xxx Invalid Code
86FB H    072         48        PHA
86FC EN   069 078     45 4E     EOR &4E
86FE   T  140 020 084 8C 14 54  STY &5414
8701 O    079         4F        xxx Invalid Code
8702      184         B8        CLV
8703      000         00        BRK
8704 T    084         54        xxx Invalid Code
8705 AB   065 066     41 42     EOR (&42,X)
```

```
8707 (    040         28        PLP
8708      138         8A        TXA
8709      000         00        BRK
870A T    084         54        xxx Invalid Code
870B RA   082 065     52 41     EOR (&41)
870D C    067         43        xxx Invalid Code
870E E    069 252     45 FC     EOR &FC
8710  T   018 084     12 54     ORA (&54)
8712 IM   073 077     49 4D     EOR#&4D
8714 E    069 145     45 91     EOR &91
8716 C    067         43        xxx Invalid Code
8717 T    084         54        xxx Invalid Code
8718 RU   082 085     52 55     EOR (&55)
871A E    069 185     45 B9     EOR &B9
871C  U   001 085     01 55     ORA (&55,X)
871E NTI  078 084 073 4E 54 49  LSR &4954
8721 L    076 253 002 4C FD 02  JMP &02FD
8724 US   085 083     55 53     EOR &53,X
8726 R    082 186     52 BA     EOR (&BA)
8728      000         00        BRK
8729 VD   086 068     56 44     LSR &44,X
872B U    085 239     55 EF     EOR &EF,X
872D      002         02        xxx Invalid Code
872E VA   086 065     56 41     LSR &41,X
8730 L    076 187 000 4C BB 00  JMP &00BB
8733 VP   086 080     56 50     LSR &50,X
8735 O    079         4F        xxx Invalid Code
8736 S    083         53        xxx Invalid Code
8737   W  188 001 087 BC 01 57  LDY &5701,X
873A ID   073 068     49 44     EOR#&44
873C T    084         54        xxx Invalid Code
873D H    072         48        PHA
873E   P  254 002 080 FE 02 50  INC &5002,X
8741 AG   065 071     41 47     EOR (&47,X)
8743 E    069 208     45 D0     EOR &D0
8745      000         00        BRK
8746 PT   080 084     50 54     BVC 84   --> &879C
8748 R    082 207     52 CF     EOR (&CF)
874A      000         00        BRK
874B T    084         54        xxx Invalid Code
874C IM   073 077     49 4D     EOR#&4D
874E E    069 209     45 D1     EOR &D1
8750      000         00        BRK
```

8751 LOM 076 079 077 4C 4F 4D JMP &4D4F
8754 EM   069 077   45 4D   EOR &4D
8756   210 000   D2 00   CMP (&00)
8758 H   072   48   PHA
8759 IM   073 077   49 4D   EOR#&4D
875B EM   069 077   45 4D   EOR &4D
875D   211   D3   xxx Invalid Code
875E   000   00   BRK
875F Mis 077 105 115 4D 69 73 EOR &7369
8762 s   115   73   xxx Invalid Code
8763 in   105 110   69 6E   ADC#&6E
8765 g   103   67   xxx Invalid Code
8766   032 141 000 20 8D 00 JSR &008D
8769   223   DF   xxx Invalid Code
876A   170   AA   TAX
876B   201 170   C9 AA   CMP#&AA
876D   008   08   PHP
876E D   174 068 174 AE 44 AE LDX &AE44
8771)   041 174   29 AE   AND#&AE
8773 /   047   2F   xxx Invalid Code
8774   174 183 172 AE B7 AC LDX &ACB7
8777   156 168 236 9C A8 EC STZ &ECA8
877A   173 179 171 AD B3 AB LDA &ABB3
877D   161 168   A1 A8   LDA (&A8,X)
877F   195   C3   xxx Invalid Code
8780   168   A8   TAY
8781   215   D7   xxx Invalid Code
8782   170   AA   TAX
8783   % 014 169 037 0E A9 25 ASL &25A9
8786   174 216 169 AE D8 A9 LDX &A9D8
8789 5   053 174   35 AE   AND &AE,X
878B ;   059   3B   xxx Invalid Code
878C   174 005 171 AE 05 AB LDX &AB05
878F   223   DF   xxx Invalid Code
8790   169 197   A9 C5   LDA#&C5
8792   170   AA   TAX
8793   232   E8   INX
8794   171   AB   xxx Invalid Code
8795   023   17   xxx Invalid Code
8796 ?   176 063   B0 3F   BCS 63   --> &87D7
8798   174 194 171 AE C2 AB LDX &ABC2
879B 6   054 172   36 AC   ROL &AC,X
879D   138   8A   TXA

879E 171 AB xxx Invalid Code
879F 017 174 11 AE ORA (&AE),Y
87A1 F 070 167 46 A7 LSR &A7
87A3 207 CF xxx Invalid Code
87A4 169 147 A9 93 LDA#&93
87A6 170 AA TAX
87A7 231 E7 xxx Invalid Code
87A8 170 AA TAX
87A9 227 E3 xxx Invalid Code
87AA 170 AA TAX
87AB 255 FF xxx Invalid Code
87AC 170 AA TAX
87AD 014 172 163 0E AC A3 ASL &A3AC
87B0 170 AA TAX
87B1 200 C8 INY
87B2 s 169 115 A9 73 LDA#&73
87B4 170 AA TAX
87B5 245 171 F5 AB SBC &AB,X
87B7 013 169 181 0D A9 B5 ORA &B5A9
87BA 167 A7 xxx Invalid Code
87BB 155 9B xxx Invalid Code
87BC 165 249 A5 F9 LDA &F9
87BE 173 219 171 AD DB AB LDA &ABDB
87C1 169 170 A9 AA LDA#&AA
87C3 I 073 171 49 AB EOR#&AB
87C5 / 188 170 047 BC AA 2F LDY &2FAA,X
87C8 i 178 105 B2 69 LDA (&69)
87CA 174 179 174 AE B3 AE LDX &AEB3
87CD s 115 73 xxx Invalid Code
87CE 174 197 174 AE C5 AE LDX &AEC5
87D1 t 116 174 74 AE STZ &AE,X
87D3 G 028 175 071 1C AF 47 TRB &47AF
87D6 175 AF xxx Invalid Code
87D7 207 CF xxx Invalid Code
87D8 171 AB xxx Invalid Code
87D9 137 148 89 94 BIT#&94
87DB 023 17 xxx Invalid Code
87DC 147 93 xxx Invalid Code
87DD 032 143 221 20 8F DD JSR &DD8F
87E0 179 B3 xxx Invalid Code
87E1 } 125 143 000 7D 8F 00 ADC &008F,X
87E4 143 8F xxx Invalid Code
87E5 132 147 84 93 STY &93

```
87E7 U    085 190     55 BE     EOR &BE,X
87E9      147         93        xxx Invalid Code
87EA      179         B3        xxx Invalid Code
87EB      151         97        xxx Invalid Code
87EC  4   190 052 150 BE 34 96  LDX &9634,Y
87EF y    121 150 032 79 96 20  ADC &2096,Y
87F2      150 015     96 0F     STX &0F,Y
87F4      150 200     96 C8     STX &C8,Y
87F6      178 189     B2 BD     LDA (&BD)
87F8      190 190 146 BE BE 92  LDX &92BE,Y
87FB      251         FB        xxx Invalid Code
87FC  >   142 062 150 8E 3E 96  STX &963E
87FF      174 190 224 AE BE E0  LDX &E0BE
8802      151         97        xxx Invalid Code
8803      231         E7        xxx Invalid Code
8804      151         97        xxx Invalid Code
8805      174 143 174 AE 8F AE  LDX &AE8F
8808      143         8F        xxx Invalid Code
8809 4    052 149     34 95     BIT &95,X
880B      166 151     A6 97     LDX &97
880D %    037 143     25 8F     AND &8F
880F      154         9A        TXS
8810      155         9B        xxx Invalid Code
8811      236 178 024 EC B2 18  CPX &18B2
8814      182 217     B6 D9     LDX &D9,Y
8816      182 029     B6 1D     LDX &1D,Y
8818      183         B7        xxx Invalid Code
8819 A    065 151     41 97     EOR (&97,X)
881B      008         08        PHP
881C      156 182 184 9C B6 B8  STZ &B8B6
881F J    074         4A        LSR A
8820      144 003     90 03     BCC 3   --> &8825
8822      151         97        xxx Invalid Code
8823 _    095         5F        xxx Invalid Code
8824      151         97        xxx Invalid Code
8825      162 151     A2 97     LDX#&97
8827      241 180     F1 B4     SBC (&B4),Y
8829 [    091         5B        xxx Invalid Code
882A      183         B7        xxx Invalid Code
882B      013 152 177 0D 98 B1  ORA &B198
882E      151         97        xxx Invalid Code
882F      141 145 228 8D 91 E4  STA &E491
8832  }   150 125     96 7D     STX &7D,Y
```

```
8834      185 174 143 B9 AE 8F  LDA &8FAE,Y
8837 X    088         58        CLI
8838      186         BA        TSX
8839      244         F4        xxx Invalid Code
883A      151         97        xxx Invalid Code
883B M    077 185 007 4D B9 07  EOR &07B9
883E      183         B7        xxx Invalid Code
883F      018 143     12 8F     ORA (&8F)
8841      134 144     86 90     STX &90
8843 U    085 151     55 97     EOR &97,X
8845 F    070 150     46 96     LSR &96
8847      023         17        xxx Invalid Code
8848      186         BA        TSX
8849      023         17        xxx Invalid Code
884A      179         B3        xxx Invalid Code
884B      135         87        xxx Invalid Code
884C  K   190 075 131 BE 4B 83  LDX &834B,Y
884F      132 137     84 89     STY &89
8851      150 184     96 B8     STX &B8,Y
8853      185 216 217 B9 D8 D9  LDA &D9D8,Y
8856      240 001     F0 01     BEQ 1   --> &8859
8858      016 129     10 81     BPL -127   --> &87DB
885A      144 137     90 89     BCC -119   --> &87E5
885C      147         93        xxx Invalid Code
885D      163         A3        xxx Invalid Code
885E      164 169     A4 A9     LDY &A9
8860 8    056         38        SEC
8861 9x   057 120 001 39 78 01  AND &0178,Y
8864      019         13        xxx Invalid Code
8865 !    033 161     21 A1     AND (&A1,X)
8867      193 025     C1 19     CMP (&19,X)
8869      024         18        CLC
886A   c  153 152 099 99 98 63  STA &6398,Y
886D s    115         73        xxx Invalid Code
886E      177 169     B1 A9     LDA (&A9),Y
8870      197 012     C5 0C     CMP &0C
8872      195         C3        xxx Invalid Code
8873      211         D3        xxx Invalid Code
8874 A    065 196     41 C4     EOR (&C4,X)
8876  A   242 065     F2 41     SBC (&41)
8878      131         83        xxx Invalid Code
8879      176 129     B0 81     BCS -127   --> &87FC
887B C    067         43        xxx Invalid Code
```

```
887C lr   108 114 236 6C 72 EC  JMP (&EC72)
887F      242 163     F2 A3     SBC (&A3)
8881      195         C3        xxx Invalid Code
8882      146 154     92 9A     STA (&9A)
8884      024         18        CLC
8885  bB  025 098 066 19 62 42  ORA &4262,Y
8888 4    052 176     34 B0     BIT &B0,X
888A r    114 152     72 98     ADC (&98)
888C      153 129 152 99 81 98  STA &9881,Y
888F   5  153 020 053 99 14 35  STA &3514,Y
8892      010         0A        ASL A
8893      013 013 013 0D 0D 0D  ORA &0D0D
8896      013 016 016 0D 10 10  ORA &1010
8899 %%   037 037     25 25     AND &25
889B 9AA  057 065 065 39 41 41  AND &4141,Y
889E AA   065 065     41 41     EOR (&41,X)
88A0 J    074         4A        LSR A
88A1 J    074         4A        LSR A
88A2 LLL  076 076 076 4C 4C 4C  JMP &4C4C
88A5 PP   080 080     50 50     BVC 80   --> &88F7
88A7 RS   082 083     52 53     EOR (&53)
88A9 S    083         53        xxx Invalid Code
88AA S    083         53        xxx Invalid Code
88AB  %   016 037     10 25     BPL 37   --> &88D2
88AD AA   065 065     41 41     EOR (&41,X)
88AF AA   065 065     41 41     EOR (&41,X)
88B1      008         08        PHP
88B2      008         08        PHP
88B3      008         08        PHP
88B4      009 009     09 09     ORA#&09
88B6      010         0A        ASL A
88B7      010         0A        ASL A
88B8      010         0A        ASL A
88B9      010         0A        ASL A
88BA      005 021     05 15     ORA &15
88BC >    062 004 013 3E 04 0D  ROL &0D04,X
88BF 0L   048 076     30 4C     BMI 76   --> &890D
88C1  2   006 050     06 32     ASL &32
88C3 II   073 073     49 49     EOR#&49
88C5  %   016 037     10 25     BPL 37   --> &88EC
88C7  N   013 078 014 0D 4E 0E  ORA &0E4E
88CA  RR  014 082 082 0E 52 52  ASL &5252
88CD  )   009 041     09 29     ORA#&29
```

```
88CF *    042        2A       ROL A
88D0 00   048 048    30 30    BMI 48   --> &8902
88D2 NNN  078 078 078 4E 4E 4E  LSR &4E4E
88D5 >    062 022 000 3E 16 00  ROL &0016,X
88D8      024        18       CLC
88D9      216        D8       CLD
88DA X    088        58       CLI
88DB      184        B8       CLV
88DC      202        CA       DEX
88DD      136        88       DEY
88DE      232        E8       INX
88DF      200        C8       INY
88E0      234        EA       NOP
88E1 H    072        48       PHA
88E2      008        08       PHP
88E3 h    104        68       PLA
88E4 (    040        28       PLP
88E5 @    064        40       RTI
88E6 `    096        60       RTS
88E7 8    056        38       SEC
88E8      248        F8       SED
88E9 x    120        78       SEI
88EA      170        AA       TAX
88EB      168        A8       TAY
88EC      186        BA       TSX
88ED      138        8A       TXA
88EE      154        9A       TXS
88EF      152        98       TYA
88F0 :    058        3A       DEC A
88F1      026        1A       INC A
88F2 Z    090        5A       PHY
88F3      218        DA       PHX
88F4 z    122        7A       PLY
88F5      250        FA       PLX
88F6      144 176    90 B0    BCC -80  --> &88A8
88F8  0   240 048    F0 30    BEQ 48   --> &892A
88FA      208 016    D0 10    BNE 16   --> &890C
88FC Pp   080 112    50 70    BVC 112  --> &896E
88FE  !   128 033    80 21    BRA 33   --> &8921
8900 A    065 001    41 01    EOR (&01,X)
8902 a    097 193 161 61 C1 A1  ADC (&A1C1,X)
8905      225 006    E1 06    SBC (&06,X)
8907 F&   070 038    46 26    LSR &26
```

```
8909 f    102 198     66 C6     ROR &C6
890B      230 156     E6 9C     INC &9C
890D      156 224 192 9C E0 C0  STZ &C0E0
8910      000         00        BRK
8911  $   016 036     10 24     BPL 36   --> &8937
8913 L    076 032 162 4C 20 A2  JMP &A220
8916      160 129     A0 81     LDY#&81
8918      134 132     86 84     STX &84
891A :    058         3A        DEC A
891B  (   133 040     85 28     STA &28
891D L    076 011 144 4C 0B 90  JMP &900B
8920      032 224 142 20 E0 8E  JSR &8EE0
8923 I]   073 093     49 5D     EOR#&5D
8925      240 243     F0 F3     BEQ -13   --> &891A
8927      032 188 155 20 BC 9B  JSR &9BBC
892A      198 010     C6 0A     DEC &0A
892C      032 235 137 20 EB 89  JSR &89EB
892F      198 010     C6 0A     DEC &0A
8931  (   165 040     A5 28     LDA &28
8933 J    074         4A        LSR A
8934  x   144 120     90 78     BCC 120   --> &89AE
8936      165 030     A5 1E     LDA &1E
8938 i    105 004     69 04     ADC#&04
893A  ?   133 063     85 3F     STA &3F
893C  8   165 056     A5 38     LDA &38
893E  l   032 108 189 20 6C BD  JSR &BD6C
8941  7   165 055     A5 37     LDA &37
8943      032 143 189 20 8F BD  JSR &BD8F
8946      162 252     A2 FC     LDX#&FC
8948  9   164 057     A4 39     LDY &39
894A      016 002     10 02     BPL 2   --> &894E
894C  6   164 054     A4 36     LDY &36
894E  8   132 056     84 38     STY &38
8950      240 025     F0 19     BEQ 25   --> &896B
8952      160 000     A0 00     LDY#&00
8954      232         E8        INX
8955      208 010     D0 0A     BNE 10   --> &8961
8957      032 146 186 20 92 BA  JSR &BA92
895A  ?   166 063     A6 3F     LDX &3F
895C      032 191 189 20 BF BD  JSR &BDBF
895F      162 253     A2 FD     LDX#&FD
8961  :   177 058     B1 3A     LDA (&3A),Y
8963      032 143 189 20 8F BD  JSR &BD8F
```

```
8966      200        C8       INY
8967  8   198 056    C6 38    DEC &38
8969      208 233    D0 E9    BNE -23   --> &8954
896B      138        8A       TXA
896C      168        A8       TAY
896D      200        C8       INY
896E      240 007    F0 07    BEQ 7   --> &8977
8970      162 003    A2 03    LDX#&03
8972      032 191 189 20 BF BD  JSR &BDBF
8975      128 246    80 F6    BRA -10   --> &896D
8977      162 010    A2 0A    LDX#&0A
8979      178 011    B2 0B    LDA (&0B)
897B  .   201 046    C9 2E    CMP#&2E
897D      208 015    D0 0F    BNE 15   --> &898E
897F  7   032 055 189 20 37 BD  JSR &BD37
8982      202        CA       DEX
8983      208 002    D0 02    BNE 2   --> &8987
8985      162 001    A2 01    LDX#&01
8987      200        C8       INY
8988      177 011    B1 0B    LDA (&0B),Y
898A  N   196 078    C4 4E    CPY &4E
898C      208 241    D0 F1    BNE -15   --> &897F
898E      032 191 189 20 BF BD  JSR &BDBF
8991      136        88       DEY
8992      200        C8       INY
8993      209 011    D1 0B    CMP (&0B),Y
8995      240 251    F0 FB    BEQ -5   --> &8992
8997      177 011    B1 0B    LDA (&0B),Y
8999  :   201 058    C9 3A    CMP#&3A
899B      240 010    F0 0A    BEQ 10   --> &89A7
899D      201 013    C9 0D    CMP#&0D
899F      240 010    F0 0A    BEQ 10   --> &89AB
89A1  7   032 055 189 20 37 BD  JSR &BD37
89A4      200        C8       INY
89A5      128 240    80 F0    BRA -16   --> &8997
89A7      196 010    C4 0A    CPY &0A
89A9      144 246    90 F6    BCC -10   --> &89A1
89AB      032 146 186 20 92 BA  JSR &BA92
89AE      164 010    A4 0A    LDY &0A
89B0      136        88       DEY
89B1      200        C8       INY
89B2      177 011    B1 0B    LDA (&0B),Y
89B4  :   201 058    C9 3A    CMP#&3A
```

```
89B6      240 004     F0 04     BEQ 4   --> &89BC
89B8      201 013     C9 0D     CMP#&0D
89BA      208 245     D0 F5     BNE -11   --> &89B1
89BC      032 168 155 20 A8 9B  JSR &9BA8
89BF      178 011     B2 0B     LDA (&0B)
89C1  :   201 058     C9 3A     CMP#&3A
89C3      240 012     F0 0C     BEQ 12   --> &89D1
89C5      165 012     A5 0C     LDA &0C
89C7      201 007     C9 07     CMP#&07
89C9      208 003     D0 03     BNE 3   --> &89CE
89CB L    076 134 143 4C 86 8F  JMP &8F86
89CE      032 222 155 20 DE 9B  JSR &9BDE
89D1 L    076 032 137 4C 20 89  JMP &8920
89D4      032 174 152 20 AE 98  JSR &98AE
89D7  \   240 092     F0 5C     BEQ 92   --> &8A35
89D9  Z   176 090     B0 5A     BCS 90   --> &8A35
89DB  C   032 067 188 20 43 BC  JSR &BC43
89DE      032 132 173 20 84 AD  JSR &AD84
89E1  '   133 039     85 27     STA &27
89E3  +   032 043 179 20 2B B3  JSR &B32B
89E6  u   032 117 146 20 75 92  JSR &9275
89E9  N   132 078     84 4E     STY &4E
89EB      032 224 142 20 E0 8E  JSR &8EE0
89EE      160 000     A0 00     LDY#&00
89F0 d=   100 061     64 3D     STZ &3D
89F2  :   201 058     C9 3A     CMP#&3A
89F4  h   240 104     F0 68     BEQ 104   --> &8A5E
89F6      201 013     C9 0D     CMP#&0D
89F8  d   240 100     F0 64     BEQ 100   --> &8A5E
89FA  \   201 092     C9 5C     CMP#&5C
89FC  `   240 096     F0 60     BEQ 96   --> &8A5E
89FE  .   201 046     C9 2E     CMP#&2E
8A00      240 210     F0 D2     BEQ -46   --> &89D4
8A02      198 010     C6 0A     DEC &0A
8A04      162 003     A2 03     LDX#&03
8A06      164 010     A4 0A     LDY &0A
8A08      230 010     E6 0A     INC &0A
8A0A      177 011     B1 0B     LDA (&0B),Y
8A0C 0*   048 042     30 2A     BMI 42   --> &8A38
8A0E      201 032     C9 20     CMP#&20
8A10      240 016     F0 10     BEQ 16   --> &8A22
8A12      160 005     A0 05     LDY#&05
8A14      010         0A        ASL A
```

```
8A15      010         0A        ASL A
8A16      010         0A        ASL A
8A17      010         0A        ASL A
8A18 &=   038 061     26 3D     ROL &3D
8A1A &>   038 062     26 3E     ROL &3E
8A1C      136         88        DEY
8A1D      208 248     D0 F8     BNE -8   --> &8A17
8A1F      202         CA        DEX
8A20      208 228     D0 E4     BNE -28   --> &8A06
8A22  E   162 069     A2 45     LDX#&45
8A24  =   165 061     A5 3D     LDA &3D
8A26  L   221 076 136 DD 4C 88  CMP &884C,X
8A29      208 007     D0 07     BNE 7   --> &8A32
8A2B      188 145 136 BC 91 88  LDY &8891,X
8A2E  >   196 062     C4 3E     CPY &3E
8A30  !   240 033     F0 21     BEQ 33   --> &8A53
8A32      202         CA        DEX
8A33      208 241     D0 F1     BNE -15   --> &8A26
8A35 Li   076 105 155 4C 69 9B  JMP &9B69
8A38  )   162 041     A2 29     LDX#&29
8A3A      201 128     C9 80     CMP#&80
8A3C      240 021     F0 15     BEQ 21   --> &8A53
8A3E      232         E8        INX
8A3F      201 130     C9 82     CMP#&82
8A41      240 016     F0 10     BEQ 16   --> &8A53
8A43      232         E8        INX
8A44      201 132     C9 84     CMP#&84
8A46      208 237     D0 ED     BNE -19   --> &8A35
8A48      230 010     E6 0A     INC &0A
8A4A      200         C8        INY
8A4B      177 011     B1 0B     LDA (&0B),Y
8A4D )    041 223     29 DF     AND#&DF
8A4F  A   201 065     C9 41     CMP#&41
8A51      208 226     D0 E2     BNE -30   --> &8A35
8A53      189 214 136 BD D6 88  LDA &88D6,X
8A56  )   133 041     85 29     STA &29
8A58      160 001     A0 01     LDY#&01
8A5A      224 032     E0 20     CPX#&20
8A5C  H   176 072     B0 48     BCS 72   --> &8AA6
8A5E  @   173 064 004 AD 40 04  LDA &0440
8A61  7   133 055     85 37     STA &37
8A63  9   132 057     84 39     STY &39
8A65  (   166 040     A6 28     LDX &28
```

```
8A67      224 004     E0 04     CPX#&04
8A69  A   174 065 004 AE 41 04  LDX &0441
8A6C  8   134 056     86 38     STX &38
8A6E      144 006     90 06     BCC 6   --> &8A76
8A70  <   173 060 004 AD 3C 04  LDA &043C
8A73  =   174 061 004 AE 3D 04  LDX &043D
8A76  :   133 058     85 3A     STA &3A
8A78  ;   134 059     86 3B     STX &3B
8A7A      152         98        TYA
8A7B  (   240 040     F0 28     BEQ 40   --> &8AA5
8A7D      016 004     10 04     BPL 4   --> &8A83
8A7F  6   164 054     A4 36     LDY &36
8A81  "   240 034     F0 22     BEQ 34   --> &8AA5
8A83      136         88        DEY
8A84  )   185 041 000 B9 29 00  LDA &0029,Y
8A87 $9   036 057     24 39     BIT &39
8A89      016 003     10 03     BPL 3   --> &8A8E
8A8B      185 000 006 B9 00 06  LDA &0600,Y
8A8E  :   145 058     91 3A     STA (&3A),Y
8A90  @   238 064 004 EE 40 04  INC &0440
8A93      208 003     D0 03     BNE 3   --> &8A98
8A95  A   238 065 004 EE 41 04  INC &0441
8A98      144 008     90 08     BCC 8   --> &8AA2
8A9A  <   238 060 004 EE 3C 04  INC &043C
8A9D      208 003     D0 03     BNE 3   --> &8AA2
8A9F  =   238 061 004 EE 3D 04  INC &043D
8AA2      152         98        TYA
8AA3      208 222     D0 DE     BNE -34   --> &8A83
8AA5 `    096         60        RTS
8AA6  )   224 041     E0 29     CPX#&29
8AA8  <   176 060     B0 3C     BCS 60   --> &8AE6
8AAA  o   032 111 146 20 6F 92  JSR &926F
8AAD      024         18        CLC
8AAE  *   165 042     A5 2A     LDA &2A
8AB0  @   237 064 004 ED 40 04  SBC &0440
8AB3      168         A8        TAY
8AB4  +   165 043     A5 2B     LDA &2B
8AB6  A   237 065 004 ED 41 04  SBC &0441
8AB9      192 001     C0 01     CPY#&01
8ABB      136         88        DEY
8ABC      233 000     E9 00     SBC#&00
8ABE      240 027     F0 1B     BEQ 27   --> &8ADB
8AC0      026         1A        INC A
```

```
8AC1      208 003     D0 03     BNE 3   --> &8AC6
8AC3      152         98        TYA
8AC4 0    048 025     30 19     BMI 25   --> &8ADF
8AC6  (   165 040     A5 28     LDA &28
8AC8 )    041 002     29 02     AND#&02
8ACA      240 018     F0 12     BEQ 18   --> &8ADE
8ACC      000         00        BRK
8ACD      001         01        EQUB &01
8ACE O    079         4F        ORA (&4F,X)
8ACF ut   117 116     75 74     ADC &74,X
8AD1  of  032 111 102 20 6F 66  JSR &666F
8AD4  ra  032 114 097 20 72 61  JSR &6172
8AD7 nge  110 103 101 6E 67 65  ROR &6567
8ADA      000         00        EQUB &00
8ADB      152         98        TYA
8ADC 0    048 232     30 E8     BMI -24   --> &8AC6
8ADE      168         A8        TAY
8ADF  *   132 042     84 2A     STY &2A
8AE1      160 002     A0 02     LDY#&02
8AE3 L^   076 094 138 4C 5E 8A  JMP &8A5E
8AE6  0   224 048     E0 30     CPX#&30
8AE8      176 022     B0 16     BCS 22   --> &8B00
8AEA      032 223 140 20 DF 8C  JSR &8CDF
8AED      208 024     D0 18     BNE 24   --> &8B07
8AEF      032 204 140 20 CC 8C  JSR &8CCC
8AF2  o   032 111 146 20 6F 92  JSR &926F
8AF5  +   165 043     A5 2B     LDA &2B
8AF7      240 232     F0 E8     BEQ -24   --> &8AE1
8AF9      000         00        BRK
8AFA      002         02        EQUB &02
8AFB B    066         42        xxx Invalid Code
8AFC yte  121 116 101 79 74 65  ADC &6574,Y
8AFF      000         00        EQUB &00
8B00  A   224 065     E0 41     CPX#&41
8B02  c   208 099     D0 63     BNE 99   --> &8B67
8B04      032 224 142 20 E0 8E  JSR &8EE0
8B07  (   201 040     C9 28     CMP#&28
8B09  9   208 057     D0 39     BNE 57   --> &8B44
8B0B  o   032 111 146 20 6F 92  JSR &926F
8B0E      032 224 142 20 E0 8E  JSR &8EE0
8B11  )   201 041     C9 29     CMP#&29
8B13      208 023     D0 17     BNE 23   --> &8B2C
8B15      032 201 140 20 C9 8C  JSR &8CC9
```

```
8B18      032 229 140 20 E5 8C  JSR &8CE5
8B1B      240 004     F0 04     BEQ 4   --> &8B21
8B1D  )   230 041     E6 29     INC &29
8B1F      128 212     80 D4     BRA -44   --> &8AF5
8B21      032 224 142 20 E0 8E  JSR &8EE0
8B24 )    041 223     29 DF     AND#&DF
8B26  Y   201 089     C9 59     CMP#&59
8B28      240 203     F0 CB     BEQ -53   --> &8AF5
8B2A      128 016     80 10     BRA 16   --> &8B3C
8B2C  ,   201 044     C9 2C     CMP#&2C
8B2E      208 012     D0 0C     BNE 12   --> &8B3C
8B30      032 215 140 20 D7 8C  JSR &8CD7
8B33      208 007     D0 07     BNE 7   --> &8B3C
8B35      032 224 142 20 E0 8E  JSR &8EE0
8B38  )   201 041     C9 29     CMP#&29
8B3A      240 185     F0 B9     BEQ -71   --> &8AF5
8B3C      000         00        BRK
8B3D      003         03        EQUB &03
8B3E In   073 110     49 6E     EOR#&6E
8B40 de   100 101     64 65     STZ &65
8B42 x    120         78        SEI
8B43      000         00        EQUB &00
8B44  m   032 109 146 20 6D 92  JSR &926D
8B47      032 229 140 20 E5 8C  JSR &8CE5
8B4A      208 018     D0 12     BNE 18   --> &8B5E
8B4C      032 201 140 20 C9 8C  JSR &8CC9
8B4F      032 215 140 20 D7 8C  JSR &8CD7
8B52      240 010     F0 0A     BEQ 10   --> &8B5E
8B54  Y   201 089     C9 59     CMP#&59
8B56      208 228     D0 E4     BNE -28   --> &8B3C
8B58      032 204 140 20 CC 8C  JSR &8CCC
8B5B L    076 254 139 4C FE 8B  JMP &8BFE
8B5E      032 207 140 20 CF 8C  JSR &8CCF
8B61  +   165 043     A5 2B     LDA &2B
8B63      208 243     D0 F3     BNE -13   --> &8B58
8B65      128 144     80 90     BRA -112   --> &8AF7
8B67  6   224 054     E0 36     CPX#&36
8B69  6   176 054     B0 36     BCS 54   --> &8BA1
8B6B      032 224 142 20 E0 8E  JSR &8EE0
8B6E )    041 223     29 DF     AND#&DF
8B70  A   201 065     C9 41     CMP#&41
8B72      240 018     F0 12     BEQ 18   --> &8B86
8B74  m   032 109 146 20 6D 92  JSR &926D
```

```
8B77      032 229 140 20 E5 8C  JSR &8CE5
8B7A      208 229     D0 E5     BNE -27   --> &8B61
8B7C      032 201 140 20 C9 8C  JSR &8CC9
8B7F      032 215 140 20 D7 8C  JSR &8CD7
8B82      240 221     F0 DD     BEQ -35   --> &8B61
8B84      128 182     80 B6     BRA -74   --> &8B3C
8B86      200         C8        INY
8B87      177 011     B1 0B     LDA (&0B),Y
8B89      032 132 141 20 84 8D  JSR &8D84
8B8C      176 230     B0 E6     BCS -26   --> &8B74
8B8E      160 022     A0 16     LDY#&16
8B90  4   224 052     E0 34     CPX#&34
8B92      144 006     90 06     BCC 6   --> &8B9A
8B94      208 002     D0 02     BNE 2   --> &8B98
8B96  6   160 054     A0 36     LDY#&36
8B98  )   132 041     84 29     STY &29
8B9A      032 207 140 20 CF 8C  JSR &8CCF
8B9D      160 001     A0 01     LDY#&01
8B9F  _   128 095     80 5F     BRA 95   --> &8C00
8BA1  8   224 056     E0 38     CPX#&38
8BA3  %   176 037     B0 25     BCS 37   --> &8BCA
8BA5  o   032 111 146 20 6F 92  JSR &926F
8BA8      160 003     A0 03     LDY#&03
8BAA      162 001     A2 01     LDX#&01
8BAC  +   165 043     A5 2B     LDA &2B
8BAE      208 007     D0 07     BNE 7   --> &8BB7
8BB0      162 015     A2 0F     LDX#&0F
8BB2  d   169 100     A9 64     LDA#&64
8BB4  )   133 041     85 29     STA &29
8BB6      136         88        DEY
8BB7 Z    090         5A        PHY
8BB8      032 229 140 20 E5 8C  JSR &8CE5
8BBB      208 010     D0 0A     BNE 10   --> &8BC7
8BBD      032 215 140 20 D7 8C  JSR &8CD7
8BC0      208 194     D0 C2     BNE -62   --> &8B84
8BC2      138         8A        TXA
8BC3 e)   101 041     65 29     ADC &29
8BC5  )   133 041     85 29     STA &29
8BC7 z    122         7A        PLY
8BC8  6   128 054     80 36     BRA 54   --> &8C00
8BCA  <   224 060     E0 3C     CPX#&3C
8BCC      176 028     B0 1C     BCS 28   --> &8BEA
8BCE  :   224 058     E0 3A     CPX#&3A
```

```
8BD0      176 007     B0 07     BCS 7   --> &8BD9
8BD2      032 223 140 20 DF 8C  JSR &8CDF
8BD5      240 016     F0 10     BEQ 16   --> &8BE7
8BD7      198 010     C6 0A     DEC &0A
8BD9  o   032 111 146 20 6F 92  JSR &926F
8BDC      128 128     80 80     BRA 128   --> &8C5E
8BDE      032 223 140 20 DF 8C  JSR &8CDF
8BE1      208 145     D0 91     BNE -111   --> &8B74
8BE3      160 137     A0 89     LDY#&89
8BE5  )   132 041     84 29     STY &29
8BE7 L    076 242 138 4C F2 8A  JMP &8AF2
8BEA      240 242     F0 F2     BEQ -14   --> &8BDE
8BEC  >   224 062     E0 3E     CPX#&3E
8BEE      240 011     F0 0B     BEQ 11   --> &8BFB
8BF0  7   176 055     B0 37     BCS 55   --> &8C29
8BF2      032 224 142 20 E0 8E  JSR &8EE0
8BF5  (   201 040     C9 28     CMP#&28
8BF7      240 010     F0 0A     BEQ 10   --> &8C03
8BF9      198 010     C6 0A     DEC &0A
8BFB  o   032 111 146 20 6F 92  JSR &926F
8BFE      160 003     A0 03     LDY#&03
8C00 L^   076 094 138 4C 5E 8A  JMP &8A5E
8C03      032 201 140 20 C9 8C  JSR &8CC9
8C06      032 201 140 20 C9 8C  JSR &8CC9
8C09  o   032 111 146 20 6F 92  JSR &926F
8C0C      032 224 142 20 E0 8E  JSR &8EE0
8C0F  )   201 041     C9 29     CMP#&29
8C11      240 235     F0 EB     BEQ -21   --> &8BFE
8C13  ,   201 044     C9 2C     CMP#&2C
8C15      208 015     D0 0F     BNE 15   --> &8C26
8C17      032 201 140 20 C9 8C  JSR &8CC9
8C1A      032 215 140 20 D7 8C  JSR &8CD7
8C1D      208 007     D0 07     BNE 7   --> &8C26
8C1F      032 224 142 20 E0 8E  JSR &8EE0
8C22  )   201 041     C9 29     CMP#&29
8C24      240 216     F0 D8     BEQ -40   --> &8BFE
8C26 L<   076 060 139 4C 3C 8B  JMP &8B3C
8C29  D   224 068     E0 44     CPX#&44
8C2B  M   176 077     B0 4D     BCS 77   --> &8C7A
8C2D  =   165 061     A5 3D     LDA &3D
8C2F I    073 001     49 01     EOR#&01
8C31 )    041 031     29 1F     AND#&1F
8C33 H    072         48        PHA
```

```
8C34  A   224 065     E0 41     CPX#&41
8C36  !   176 033     B0 21     BCS 33   --> &8C59
8C38      032 223 140 20 DF 8C  JSR &8CDF
8C3B      208 003     D0 03     BNE 3   --> &8C40
8C3D h    104         68        PLA
8C3E      128 167     80 A7     BRA -89   --> &8BE7
8C40  m   032 109 146 20 6D 92  JSR &926D
8C43 h    104         68        PLA
8C44  7   133 055     85 37     STA &37
8C46      032 229 140 20 E5 8C  JSR &8CE5
8C49      208 145     D0 91     BNE -111   --> &8BDC
8C4B      032 224 142 20 E0 8E  JSR &8EE0
8C4E )    041 031     29 1F     AND#&1F
8C50  7   197 055     C5 37     CMP &37
8C52      208 210     D0 D2     BNE -46   --> &8C26
8C54      032 201 140 20 C9 8C  JSR &8CC9
8C57      128 131     80 83     BRA -125   --> &8BDC
8C59  o   032 111 146 20 6F 92  JSR &926F
8C5C h    104         68        PLA
8C5D  7   133 055     85 37     STA &37
8C5F      032 229 140 20 E5 8C  JSR &8CE5
8C62      208 019     D0 13     BNE 19   --> &8C77
8C64      032 224 142 20 E0 8E  JSR &8EE0
8C67 )    041 031     29 1F     AND#&1F
8C69  7   197 055     C5 37     CMP &37
8C6B      208 185     D0 B9     BNE -71   --> &8C26
8C6D      032 201 140 20 C9 8C  JSR &8CC9
8C70  +   165 043     A5 2B     LDA &2B
8C72      240 003     F0 03     BEQ 3   --> &8C77
8C74 L    076 249 138 4C F9 8A  JMP &8AF9
8C77 La   076 097 139 4C 61 8B  JMP &8B61
8C7A      208 011     D0 0B     BNE 11   --> &8C87
8C7C  o   032 111 146 20 6F 92  JSR &926F
8C7F  *   165 042     A5 2A     LDA &2A
8C81  (   133 040     85 28     STA &28
8C83      160 000     A0 00     LDY#&00
8C85  *   128 042     80 2A     BRA 42   --> &8CB1
8C87      162 001     A2 01     LDX#&01
8C89      164 010     A4 0A     LDY &0A
8C8B      230 010     E6 0A     INC &0A
8C8D      177 011     B1 0B     LDA (&0B),Y
8C8F )    041 223     29 DF     AND#&DF
8C91  B   201 066     C9 42     CMP#&42
```

```
8C93      240 018     F0 12     BEQ 18   --> &8CA7
8C95      232         E8        INX
8C96  W   201 087     C9 57     CMP#&57
8C98      240 013     F0 0D     BEQ 13   --> &8CA7
8C9A      162 004     A2 04     LDX#&04
8C9C  D   201 068     C9 44     CMP#&44
8C9E      240 007     F0 07     BEQ 7    --> &8CA7
8CA0  S   201 083     C9 53     CMP#&53
8CA2      240 019     F0 13     BEQ 19   --> &8CB7
8CA4 Li   076 105 155 4C 69 9B  JMP &9B69
8CA7      218         DA        PHX
8CA8  o   032 111 146 20 6F 92  JSR &926F
8CAB  )   162 041     A2 29     LDX#&29
8CAD      032 198 189 20 C6 BD  JSR &BDC6
8CB0 z    122         7A        PLY
8CB1 L^   076 094 138 4C 5E 8A  JMP &8A5E
8CB4 L    076 146 144 4C 92 90  JMP &9092
8CB7  (   165 040     A5 28     LDA &28
8CB9 H    072         48        PHA
8CBA  /   032 047 157 20 2F 9D  JSR &9D2F
8CBD      208 245     D0 F5     BNE -11  --> &8CB4
8CBF h    104         68        PLA
8CC0  (   133 040     85 28     STA &28
8CC2  u   032 117 146 20 75 92  JSR &9275
8CC5      160 255     A0 FF     LDY#&FF
8CC7      128 232     80 E8     BRA -24  --> &8CB1
8CC9      032 204 140 20 CC 8C  JSR &8CCC
8CCC      032 207 140 20 CF 8C  JSR &8CCF
8CCF  )   165 041     A5 29     LDA &29
8CD1      024         18        CLC
8CD2 i    105 004     69 04     ADC#&04
8CD4  )   133 041     85 29     STA &29
8CD6 `    096         60        RTS
8CD7      032 224 142 20 E0 8E  JSR &8EE0
8CDA )    041 223     29 DF     AND#&DF
8CDC  X   201 088     C9 58     CMP#&58
8CDE `    096         60        RTS
8CDF      032 224 142 20 E0 8E  JSR &8EE0
8CE2  #   201 035     C9 23     CMP#&23
8CE4 `    096         60        RTS
8CE5      032 224 142 20 E0 8E  JSR &8EE0
8CE8  ,   201 044     C9 2C     CMP#&2C
8CEA `    096         60        RTS
```

```
8CEB  7   146 055     92 37     STA (&37)
8CED      024         18        CLC
8CEE      152         98        TYA
8CEF e7   101 055     65 37     ADC &37
8CF1  9   133 057     85 39     STA &39
8CF3      160 000     A0 00     LDY#&00
8CF5      152         98        TYA
8CF6 e8   101 056     65 38     ADC &38
8CF8  :   133 058     85 3A     STA &3A
8CFA      200         C8        INY
8CFB  9   177 057     B1 39     LDA (&39),Y
8CFD  7   145 055     91 37     STA (&37),Y
8CFF      201 013     C9 0D     CMP#&0D
8D01      208 247     D0 F7     BNE -9   --> &8CFA
8D03 `    096         60        RTS
8D04 )    041 015     29 0F     AND#&0F
8D06  =   133 061     85 3D     STA &3D
8D08      162 000     A2 00     LDX#&00
8D0A      160 000     A0 00     LDY#&00
8D0C      200         C8        INY
8D0D  7   177 055     B1 37     LDA (&37),Y
8D0F      032 148 141 20 94 8D  JSR &8D94
8D12  .   144 046     90 2E     BCC 46   --> &8D42
8D14 )    041 015     29 0F     AND#&0F
8D16 H    072         48        PHA
8D17  >   134 062     86 3E     STX &3E
8D19  =   165 061     A5 3D     LDA &3D
8D1B      010         0A        ASL A
8D1C &>   038 062     26 3E     ROL &3E
8D1E 0    048 031     30 1F     BMI 31   --> &8D3F
8D20      010         0A        ASL A
8D21 &>   038 062     26 3E     ROL &3E
8D23 0    048 026     30 1A     BMI 26   --> &8D3F
8D25 e=   101 061     65 3D     ADC &3D
8D27  =   133 061     85 3D     STA &3D
8D29      138         8A        TXA
8D2A e>   101 062     65 3E     ADC &3E
8D2C  =   006 061     06 3D     ASL &3D
8D2E *    042         2A        ROL A
8D2F 0    048 014     30 0E     BMI 14   --> &8D3F
8D31      176 012     B0 0C     BCS 12   --> &8D3F
8D33      170         AA        TAX
8D34 h    104         68        PLA
```

```
8D35 e=   101 061     65 3D     ADC &3D
8D37  =   133 061     85 3D     STA &3D
8D39      144 209     90 D1     BCC -47   --> &8D0C
8D3B      232         E8        INX
8D3C      016 206     10 CE     BPL -50   --> &8D0C
8D3E H    072         48        PHA
8D3F h    104         68        PLA
8D40 8    056         38        SEC
8D41 `    096         60        RTS
8D42      136         88        DEY
8D43      169 141     A9 8D     LDA#&8D
8D45      032 235 140 20 EB 8C  JSR &8CEB
8D48  7   165 055     A5 37     LDA &37
8D4A  9   133 057     85 39     STA &39
8D4C  8   165 056     A5 38     LDA &38
8D4E  :   133 058     85 3A     STA &3A
8D50      032 162 141 20 A2 8D  JSR &8DA2
8D53      032 162 141 20 A2 8D  JSR &8DA2
8D56      032 162 141 20 A2 8D  JSR &8DA2
8D59  9   177 057     B1 39     LDA (&39),Y
8D5B  7   145 055     91 37     STA (&37),Y
8D5D      136         88        DEY
8D5E      208 249     D0 F9     BNE -7   --> &8D59
8D60      160 003     A0 03     LDY#&03
8D62      138         8A        TXA
8D63  @   009 064     09 40     ORA#&40
8D65  9   145 057     91 39     STA (&39),Y
8D67      136         88        DEY
8D68  =   165 061     A5 3D     LDA &3D
8D6A )?   041 063     29 3F     AND#&3F
8D6C  @   009 064     09 40     ORA#&40
8D6E  9   145 057     91 39     STA (&39),Y
8D70      136         88        DEY
8D71  ?   169 063     A9 3F     LDA#&3F
8D73  =   020 061     14 3D     TRB &3D
8D75      138         8A        TXA
8D76 )    041 192     29 C0     AND#&C0
8D78 J    074         4A        LSR A
8D79 J    074         4A        LSR A
8D7A  =   005 061     05 3D     ORA &3D
8D7C J    074         4A        LSR A
8D7D J    074         4A        LSR A
8D7E IT   073 084     49 54     EOR#&54
```

```
8D80  9   145 057     91 39     STA (&39),Y
8D82      024         18        CLC
8D83 `    096         60        RTS
8D84  {   201 123     C9 7B     CMP#&7B
8D86      176 250     B0 FA     BCS -6   --> &8D82
8D88  _   201 095     C9 5F     CMP#&5F
8D8A      176 014     B0 0E     BCS 14   --> &8D9A
8D8C  [   201 091     C9 5B     CMP#&5B
8D8E      176 242     B0 F2     BCS -14   --> &8D82
8D90  A   201 065     C9 41     CMP#&41
8D92      176 006     B0 06     BCS 6   --> &8D9A
8D94  :   201 058     C9 3A     CMP#&3A
8D96      176 234     B0 EA     BCS -22   --> &8D82
8D98  0   201 048     C9 30     CMP#&30
8D9A `    096         60        RTS
8D9B  .   201 046     C9 2E     CMP#&2E
8D9D      208 245     D0 F5     BNE -11   --> &8D94
8D9F `    096         60        RTS
8DA0  7   178 055     B2 37     LDA (&37)
8DA2  7   230 055     E6 37     INC &37
8DA4  9   208 057     D0 39     BNE 57   --> &8DDF
8DA6  8   230 056     E6 38     INC &38
8DA8 `    096         60        RTS
8DA9      032 162 141 20 A2 8D  JSR &8DA2
8DAC  7   178 055     B2 37     LDA (&37)
8DAE `    096         60        RTS
8DAF      032 162 141 20 A2 8D  JSR &8DA2
8DB2  7   178 055     B2 37     LDA (&37)
8DB4      201 013     C9 0D     CMP#&0D
8DB6  '   240 039     F0 27     BEQ 39   --> &8DDF
8DB8      201 032     C9 20     CMP#&20
8DBA      240 243     F0 F3     BEQ -13   --> &8DAF
8DBC  &   201 038     C9 26     CMP#&26
8DBE      208 016     D0 10     BNE 16   --> &8DD0
8DC0      032 169 141 20 A9 8D  JSR &8DA9
8DC3      032 148 141 20 94 8D  JSR &8D94
8DC6      176 248     B0 F8     BCS -8   --> &8DC0
8DC8  A   201 065     C9 41     CMP#&41
8DCA      144 230     90 E6     BCC -26   --> &8DB2
8DCC  G   201 071     C9 47     CMP#&47
8DCE      144 240     90 F0     BCC -16   --> &8DC0
8DD0  "   201 034     C9 22     CMP#&22
8DD2      208 012     D0 0C     BNE 12   --> &8DE0
```

```
8DD4      032 169 141 20 A9 8D  JSR &8DA9
8DD7  "   201 034     C9 22     CMP#&22
8DD9      240 212     F0 D4     BEQ -44   --> &8DAF
8DDB      201 013     C9 0D     CMP#&0D
8DDD      208 245     D0 F5     BNE -11   --> &8DD4
8DDF `    096         60        RTS
8DE0  :   201 058     C9 3A     CMP#&3A
8DE2      208 009     D0 09     BNE 9   --> &8DED
8DE4      032 162 141 20 A2 8D  JSR &8DA2
8DE7 d;   100 059     64 3B     STZ &3B
8DE9 d<   100 060     64 3C     STZ &3C
8DEB      128 197     80 C5     BRA -59   --> &8DB2
8DED  ,   201 044     C9 2C     CMP#&2C
8DEF      240 190     F0 BE     BEQ -66   --> &8DAF
8DF1  *   201 042     C9 2A     CMP#&2A
8DF3      208 012     D0 0C     BNE 12   --> &8E01
8DF5  ;   165 059     A5 3B     LDA &3B
8DF7      240 230     F0 E6     BEQ -26   --> &8DDF
8DF9      162 255     A2 FF     LDX#&FF
8DFB  ;   134 059     86 3B     STX &3B
8DFD d<   100 060     64 3C     STZ &3C
8DFF      128 174     80 AE     BRA -82   --> &8DAF
8E01  .   201 046     C9 2E     CMP#&2E
8E03      240 014     F0 0E     BEQ 14   --> &8E13
8E05      032 148 141 20 94 8D  JSR &8D94
8E08  ,   144 044     90 2C     BCC 44   --> &8E36
8E0A  <   166 060     A6 3C     LDX &3C
8E0C      240 005     F0 05     BEQ 5   --> &8E13
8E0E      032 004 141 20 04 8D  JSR &8D04
8E11      144 156     90 9C     BCC -100   --> &8DAF
8E13  7   178 055     B2 37     LDA (&37)
8E15      032 155 141 20 9B 8D  JSR &8D9B
8E18      144 005     90 05     BCC 5   --> &8E1F
8E1A      032 162 141 20 A2 8D  JSR &8DA2
8E1D      128 244     80 F4     BRA -12   --> &8E13
8E1F      162 255     A2 FF     LDX#&FF
8E21  ;   134 059     86 3B     STX &3B
8E23      128 196     80 C4     BRA -60   --> &8DE9
8E25      032 132 141 20 84 8D  JSR &8D84
8E28      144 207     90 CF     BCC -49   --> &8DF9
8E2A  7   178 055     B2 37     LDA (&37)
8E2C      032 132 141 20 84 8D  JSR &8D84
8E2F      144 238     90 EE     BCC -18   --> &8E1F
```

```
8E31      032 162 141 20 A2 8D  JSR &8DA2
8E34      128 244     80 F4     BRA -12   --> &8E2A
8E36  A   201 065     C9 41     CMP#&41
8E38      144 191     90 BF     BCC -65   --> &8DF9
8E3A  X   201 088     C9 58     CMP#&58
8E3C      176 231     B0 E7     BCS -25   --> &8E25
8E3E  V   162 086     A2 56     LDX#&56
8E40  9   134 057     86 39     STX &39
8E42      162 132     A2 84     LDX#&84
8E44  :   134 058     86 3A     STX &3A
8E46      160 000     A0 00     LDY#&00
8E48  9   210 057     D2 39     CMP (&39)
8E4A      144 222     90 DE     BCC -34   --> &8E2A
8E4C      208 015     D0 0F     BNE 15   --> &8E5D
8E4E      200         C8        INY
8E4F  9   177 057     B1 39     LDA (&39),Y
8E51 01   048 049     30 31     BMI 49   --> &8E84
8E53  7   209 055     D1 37     CMP (&37),Y
8E55      240 247     F0 F7     BEQ -9   --> &8E4E
8E57  7   177 055     B1 37     LDA (&37),Y
8E59  .   201 046     C9 2E     CMP#&2E
8E5B      240 011     F0 0B     BEQ 11   --> &8E68
8E5D      200         C8        INY
8E5E  9   177 057     B1 39     LDA (&39),Y
8E60      016 251     10 FB     BPL -5   --> &8E5D
8E62      201 254     C9 FE     CMP#&FE
8E64      208 015     D0 0F     BNE 15   --> &8E75
8E66      176 194     B0 C2     BCS -62   --> &8E2A
8E68      200         C8        INY
8E69  9   177 057     B1 39     LDA (&39),Y
8E6B 0    048 023     30 17     BMI 23   --> &8E84
8E6D  9   230 057     E6 39     INC &39
8E6F      208 248     D0 F8     BNE -8   --> &8E69
8E71  :   230 058     E6 3A     INC &3A
8E73      128 244     80 F4     BRA -12   --> &8E69
8E75 8    056         38        SEC
8E76      200         C8        INY
8E77      152         98        TYA
8E78 e9   101 057     65 39     ADC &39
8E7A  9   133 057     85 39     STA &39
8E7C      144 002     90 02     BCC 2   --> &8E80
8E7E  :   230 058     E6 3A     INC &3A
8E80  7   178 055     B2 37     LDA (&37)
```

```
8E82      128 194     80 C2     BRA -62   --> &8E46
8E84      170         AA        TAX
8E85      200         C8        INY
8E86  9   177 057     B1 39     LDA (&39),Y
8E88  =   133 061     85 3D     STA &3D
8E8A      136         88        DEY
8E8B J    074         4A        LSR A
8E8C      144 007     90 07     BCC 7   --> &8E95
8E8E  7   177 055     B1 37     LDA (&37),Y
8E90      032 132 141 20 84 8D  JSR &8D84
8E93      176 149     B0 95     BCS -107   --> &8E2A
8E95      138         8A        TXA
8E96 $=   036 061     24 3D     BIT &3D
8E98 P    080 006     50 06     BVC 6   --> &8EA0
8E9A  ;   166 059     A6 3B     LDX &3B
8E9C      208 002     D0 02     BNE 2   --> &8EA0
8E9E i@   105 064     69 40     ADC#&40
8EA0      136         88        DEY
8EA1      032 235 140 20 EB 8C  JSR &8CEB
8EA4      162 255     A2 FF     LDX#&FF
8EA6  =   165 061     A5 3D     LDA &3D
8EA8 J    074         4A        LSR A
8EA9 J    074         4A        LSR A
8EAA      144 004     90 04     BCC 4   --> &8EB0
8EAC  ;   134 059     86 3B     STX &3B
8EAE d<   100 060     64 3C     STZ &3C
8EB0 J    074         4A        LSR A
8EB1      144 004     90 04     BCC 4   --> &8EB7
8EB3 d;   100 059     64 3B     STZ &3B
8EB5 d<   100 060     64 3C     STZ &3C
8EB7 J    074         4A        LSR A
8EB8      144 016     90 10     BCC 16   --> &8ECA
8EBA H    072         48        PHA
8EBB      160 001     A0 01     LDY#&01
8EBD  7   177 055     B1 37     LDA (&37),Y
8EBF      032 132 141 20 84 8D  JSR &8D84
8EC2      144 005     90 05     BCC 5   --> &8EC9
8EC4      032 162 141 20 A2 8D  JSR &8DA2
8EC7      128 244     80 F4     BRA -12   --> &8EBD
8EC9 h    104         68        PLA
8ECA J    074         4A        LSR A
8ECB      144 002     90 02     BCC 2   --> &8ECF
8ECD  <   134 060     86 3C     STX &3C
```

```
8ECF J    074        4A       LSR A
8ED0      176 013    B0 0D    BCS 13   --> &8EDF
8ED2 L    076 175 141 4C AF 8D  JMP &8DAF
8ED5      164 027    A4 1B    LDY &1B
8ED7      230 027    E6 1B    INC &1B
8ED9      177 025    B1 19    LDA (&19),Y
8EDB      201 032    C9 20    CMP#&20
8EDD      240 246    F0 F6    BEQ -10   --> &8ED5
8EDF `    096        60       RTS
8EE0      164 010    A4 0A    LDY &0A
8EE2      230 010    E6 0A    INC &0A
8EE4      177 011    B1 0B    LDA (&0B),Y
8EE6      201 032    C9 20    CMP#&20
8EE8      240 246    F0 F6    BEQ -10   --> &8EE0
8EEA `    096        60       RTS
8EEB      032 213 142 20 D5 8E  JSR &8ED5
8EEE ,    201 044    C9 2C    CMP#&2C
8EF0 `    096        60       RTS
8EF1      032 235 142 20 EB 8E  JSR &8EEB
8EF4      240 244    F0 F4    BEQ -12   --> &8EEA
8EF6      000        00       BRK
8EF7      005        05       EQUB &05
8EF8      141        8D       xxx Invalid Code
8EF9 ,    044        2C       xxx Invalid Code
8EFA      000        00       EQUB &00
8EFB      032 215 189 20 D7 BD  JSR &BDD7
8EFE      128 021    80 15    BRA 21   --> &8F15
8F00      032 166 155 20 A6 9B  JSR &9BA6
8F03      165 024    A5 18    LDA &18
8F05  8   133 056    85 38    STA &38
8F07 d7   100 055    64 37    STZ &37
8F09      169 000    A9 00    LDA#&00
8F0B  7   145 055    91 37    STA (&37),Y
8F0D      032 229 189 20 E5 BD  JSR &BDE5
8F10  q   128 113    80 71    BRA 113   --> &8F83
8F12      032 166 155 20 A6 9B  JSR &9BA6
8F15      032 172 187 20 AC BB  JSR &BBAC
8F18      165 024    A5 18    LDA &18
8F1A      133 012    85 0C    STA &0C
8F1C d    100 011    64 0B    STZ &0B
8F1E  w   128 119    80 77    BRA 119   --> &8F97
8F20      032 215 189 20 D7 BD  JSR &BDD7
8F23  ^   128 094    80 5E    BRA 94   --> &8F83
```

```
8F25     032 166 155 20 A6 9B  JSR &9BA6
8F28     032 229 189 20 E5 BD  JSR &BDE5
8F2B  Y   128 089     80 59     BRA 89   --> &8F86
8F2D      169 242     A9 F2     LDA#&F2
8F2F      032 024 174 20 18 AE  JSR &AE18
8F32      032 226 190 20 E2 BE  JSR &BEE2
8F35      170         AA        TAX
8F36      032 226 190 20 E2 BE  JSR &BEE2
8F39  +   133 043     85 2B     STA &2B
8F3B  *   134 042     86 2A     STX &2A
8F3D      162 020     A2 14     LDX#&14
8F3F      202         CA        DEX
8F40  >   240 062     F0 3E     BEQ 62   --> &8F80
8F42      032 226 190 20 E2 BE  JSR &BEE2
8F45      201 013     C9 0D     CMP#&0D
8F47  7   240 055     F0 37     BEQ 55   --> &8F80
8F49  @   201 064     C9 40     CMP#&40
8F4B      208 242     D0 F2     BNE -14   --> &8F3F
8F4D      032 226 190 20 E2 BE  JSR &BEE2
8F50      201 013     C9 0D     CMP#&0D
8F52  ,   208 044     D0 2C     BNE 44   --> &8F80
8F54      032 254 190 20 FE BE  JSR &BEFE
8F57      160 000     A0 00     LDY#&00
8F59 d    100 011     64 0B     STZ &0B
8F5B      169 007     A9 07     LDA#&07
8F5D      133 012     85 0C     STA &0C
8F5F      178 000     B2 00     LDA (&00)
8F61      240 032     F0 20     BEQ 32   --> &8F83
8F63      145 011     91 0B     STA (&0B),Y
8F65      200         C8        INY
8F66      240 024     F0 18     BEQ 24   --> &8F80
8F68      230 000     E6 00     INC &00
8F6A      208 002     D0 02     BNE 2   --> &8F6E
8F6C      230 001     E6 01     INC &01
8F6E      201 013     C9 0D     CMP#&0D
8F70      208 237     D0 ED     BNE -19   --> &8F5F
8F72      165 001     A5 01     LDA &01
8F74      197 007     C5 07     CMP &07
8F76      176 011     B0 0B     BCS 11   --> &8F83
8F78      032 235 186 20 EB BA  JSR &BAEB
8F7B      128 218     80 DA     BRA -38   --> &8F57
8F7D      032 166 155 20 A6 9B  JSR &9BA6
8F80      032 254 190 20 FE BE  JSR &BEFE
```

```
8F83      032 172 187 20 AC BB  JSR &BBAC
8F86      160 007     A0 07     LDY#&07
8F88      132 012     84 0C     STY &0C
8F8A d    100 011     64 0B     STZ &0B
8F8C      032 166 178 20 A6 B2  JSR &B2A6
8F8F  >   169 062     A9 3E     LDA#&3E
8F91      032 238 255 20 EE FF  JSR &FFEE
8F94  t   032 116 186 20 74 BA  JSR &BA74
8F97      162 255     A2 FF     LDX#&FF
8F99      154         9A        TXS
8F9A      032 166 178 20 A6 B2  JSR &B2A6
8F9D      032 235 186 20 EB BA  JSR &BAEB
8FA0      176 225     B0 E1     BCS -31   --> &8F83
8FA2  z   128 122     80 7A     BRA 122   --> &901E
8FA4      032 188 155 20 BC 9B  JSR &9BBC
8FA7      166 011     A6 0B     LDX &0B
8FA9      164 012     A4 0C     LDY &0C
8FAB      032 247 255 20 F7 FF  JSR &FFF7
8FAE      169 013     A9 0D     LDA#&0D
8FB0      164 010     A4 0A     LDY &0A
8FB2      136         88        DEY
8FB3      200         C8        INY
8FB4      209 011     D1 0B     CMP (&0B),Y
8FB6      208 251     D0 FB     BNE -5    --> &8FB3
8FB8      032 188 155 20 BC 9B  JSR &9BBC
8FBB      128 004     80 04     BRA 4     --> &8FC1
8FBD      201 013     C9 0D     CMP#&0D
8FBF      208 237     D0 ED     BNE -19   --> &8FAE
8FC1      165 012     A5 0C     LDA &0C
8FC3      201 007     C9 07     CMP#&07
8FC5      240 191     F0 BF     BEQ -65   --> &8F86
8FC7      160 001     A0 01     LDY#&01
8FC9      177 011     B1 0B     LDA (&0B),Y
8FCB 0    048 185     30 B9     BMI -71   --> &8F86
8FCD      166 032     A6 20     LDX &20
8FCF      240 010     F0 0A     BEQ 10    --> &8FDB
8FD1  +   133 043     85 2B     STA &2B
8FD3      200         C8        INY
8FD4      177 011     B1 0B     LDA (&0B),Y
8FD6  *   133 042     85 2A     STA &2A
8FD8  K   032 075 156 20 4B 9C  JSR &9C4B
8FDB      160 004     A0 04     LDY#&04
8FDD      132 010     84 0A     STY &0A
```

```
8FDF ,   128 044     80 2C     BRA 44   --> &900D
8FE1     169 003     A9 03     LDA#&03
8FE3 (   133 040     85 28     STA &28
8FE5 L    076 032 137 4C 20 89  JMP &8920
8FE8 L    076 147 190 4C 93 BE  JMP &BE93
8FEB     164 010     A4 0A     LDY &0A
8FED     136         88        DEY
8FEE     177 011     B1 0B     LDA (&0B),Y
8FF0 *   201 042     C9 2A     CMP#&2A
8FF2     240 176     F0 B0     BEQ -80   --> &8FA4
8FF4 [   201 091     C9 5B     CMP#&5B
8FF6     240 233     F0 E9     BEQ -23   --> &8FE1
8FF8     201 162     C9 A2     CMP#&A2
8FFA     240 236     F0 EC     BEQ -20   --> &8FE8
8FFC =   201 061     C9 3D     CMP#&3D
8FFE `   240 096     F0 60     BEQ 96   --> &9060
9000     198 010     C6 0A     DEC &0A
```

8000,,201 001,C9 01,CMP#&01
8002,',240 039,F0 27,BEQ 39   --> &802B
8004,`,096,60,RTS
8005,,234,EA,NOP
8006,`,096,60,RTS
8007,B,014 004 066,0E 04 42,ASL &4204
800A,AS,065 083,41 53,"EOR (&53,X)"
800C,IC,073 067,49 43,EOR#&43
800E,,000,00,BRK
800F,(,040,28,PLP
8010,C,067,43,xxx Invalid Code
8011,)1,041 049,29 31,AND#&31
8013,984,057 056 052,39 38 34,"AND &3438,Y"
8016,Ac,032 065 099,20 41 63,JSR &6341
8019,o,111,6F,xxx Invalid Code
801A,rn,114 110,72 6E,ADC (&6E)
801C,,010,0A,ASL A
801D,,013 000 000,0D 00 00,ORA &0000
8020,,128 000,80 00,BRA 0   --> &8022
8022,,000,00,BRK
8023,,000,00,BRK
8024,,003,03,xxx Invalid Code
8025,',039,27,xxx Invalid Code
8026,,001 010,01 0A,"ORA (&0A,X)"
8028,d,100 232,64 E8,STZ &E8
802A,,016,10,xxx Invalid Code
802B,%,037 017,25 11,AND &11
802D,,005 013,05 0D,ORA &0D
802F,,005 014,05 0E,ORA &0E
8031,,005 015,05 0F,ORA &0F
8033,,005 016,05 10,ORA &10
8035,,208 012,D0 0C,BNE 12   --> &8043
8037,A,169 065,A9 41,LDA#&41
8039,,133 013,85 0D,STA &0D
803B,I,073 019,49 13,EOR#&13
803D,,133 014,85 0E,STA &0E
803F,I,073 005,49 05,EOR#&05
8041,,133 015,85 0F,STA &0F
8043,,169 132,A9 84,LDA#&84
8045,,032 244 255,20 F4 FF,JSR &FFF4
8048,,134 006,86 06,STX &06
804A,,132 007,84 07,STY &07

804C,:,058,3A,DEC A
804D,,032 244 255,20 F4 FF,JSR &FFF4
8050,,132 024,84 18,STY &18
8052,d,100 031,64 1F,STZ &1F
8054,,156 002 004,9C 02 04,STZ &0402
8057,,156 003 004,9C 03 04,STZ &0403
805A,,162 255,A2 FF,LDX#&FF
805C,#,134 035,86 23,STX &23
805E,,162 010,A2 0A,LDX#&0A
8060,,142 000 004,8E 00 04,STX &0400
8063,,202,CA,DEX
8064,,142 001 004,8E 01 04,STX &0401
8067,x,169 120,A9 78,LDA#&78
8069,,141 002 002,8D 02 02,STA &0202
806C,,169 178,A9 B2,LDA#&B2
806E,,141 003 002,8D 03 02,STA &0203
8071,X,088,58,CLI
8072,L-,076 045 143,4C 2D 8F,JMP &8F2D
8075,9,132 057,84 39,STY &39
8077,,160 001,A0 01,LDY#&01
8079,7,177 055,B1 37,"LDA (&37),Y"
807B,,160 246,A0 F6,LDY#&F6
807D,,201 242,C9 F2,CMP#&F2
807F,,240 012,F0 0C,BEQ 12 --> &808D
8081,,160 248,A0 F8,LDY#&F8
8083,,128 008,80 08,BRA 8 --> &808D
8085,9,132 057,84 39,STY &39
8087,,160 001,A0 01,LDY#&01
8089,7,177 055,B1 37,"LDA (&37),Y"
808B,,010,0A,ASL A
808C,,168,A8,TAY
808D,,185 001 004,B9 01 04,"LDA &0401,Y"
8090,:,240 058,F0 3A,BEQ 58 --> &80CC
8092,+,133 043,85 2B,STA &2B
8094,,185 000 004,B9 00 04,"LDA &0400,Y"
8097,,128 011,80 0B,BRA 11 --> &80A4
8099,,160 001,A0 01,LDY#&01
809B,*,177 042,B1 2A,"LDA (&2A),Y"
809D,-,240 045,F0 2D,BEQ 45 --> &80CC
809F,,168,A8,TAY
80A0,*,178 042,B2 2A,LDA (&2A)
80A2,+,132 043,84 2B,STY &2B
80A4,*,133 042,85 2A,STA &2A

```
80A6,,160 002,A0 02,LDY#&02
80A8,*,177 042,B1 2A,"LDA (&2A),Y"
80AA,,208 010,D0 0A,BNE 10   --> &80B6
80AC,9,196 057,C4 39,CPY &39
80AE,,208 233,D0 E9,BNE -23   --> &8099
80B0,,128 017,80 11,BRA 17   --> &80C3
80B2,*,177 042,B1 2A,"LDA (&2A),Y"
80B4,,240 227,F0 E3,BEQ -29   --> &8099
80B6,7,209 055,D1 37,"CMP (&37),Y"
80B8,,208 223,D0 DF,BNE -33   --> &8099
80BA,,200,C8,INY
80BB,9,196 057,C4 39,CPY &39
80BD,,208 243,D0 F3,BNE -13   --> &80B2
80BF,*,177 042,B1 2A,"LDA (&2A),Y"
80C1,,208 214,D0 D6,BNE -42   --> &8099
80C3,,152,98,TYA
80C4,e*,101 042,65 2A,ADC &2A
80C6,*,133 042,85 2A,STA &2A
80C8,,144 002,90 02,BCC 2   --> &80CC
80CA,+,230 043,E6 2B,INC &2B
80CC,`,096,60,RTS
80CD,d=,100 061,64 3D,STZ &3D
80CF,,165 024,A5 18,LDA &18
80D1,>,133 062,85 3E,STA &3E
80D3,,160 001,A0 01,LDY#&01
80D5,=,177 061,B1 3D,"LDA (&3D),Y"
80D7,+,197 043,C5 2B,CMP &2B
80D9,,176 014,B0 0E,BCS 14   --> &80E9
80DB,,160 003,A0 03,LDY#&03
80DD,=,177 061,B1 3D,"LDA (&3D),Y"
80DF,e=,101 061,65 3D,ADC &3D
80E1,=,133 061,85 3D,STA &3D
80E3,,144 238,90 EE,BCC -18   --> &80D3
80E5,>,230 062,E6 3E,INC &3E
80E7,,128 234,80 EA,BRA -22   --> &80D3
80E9,,208 010,D0 0A,BNE 10   --> &80F5
80EB,,200,C8,INY
80EC,=,177 061,B1 3D,"LDA (&3D),Y"
80EE,*,197 042,C5 2A,CMP &2A
80F0,,144 233,90 E9,BCC -23   --> &80DB
80F2,,208 001,D0 01,BNE 1   --> &80F5
80F4,`,096,60,RTS
80F5,,160 002,A0 02,LDY#&02
```

80F7,,024,18,CLC
80F8,`,096,60,RTS
80F9,,032 190 150,20 BE 96,JSR &96BE
80FC,-,165 045,A5 2D,LDA &2D
80FE,H,072,48,PHA
80FF,,032 190 172,20 BE AC,JSR &ACBE
8102,,032 015 160,20 0F A0,JSR &A00F
8105,',134 039,86 27,STX &27
8107,,032 190 150,20 BE 96,JSR &96BE
810A,h,104,68,PLA
810B,8,133 056,85 38,STA &38
810D,E-,069 045,45 2D,EOR &2D
810F,7,133 055,85 37,STA &37
8111,,032 190 172,20 BE AC,JSR &ACBE
8114,9,162 057,A2 39,LDX#&39
8116,,032 008 189,20 08 BD,JSR &BD08
8119,d=,100 061,64 3D,STZ &3D
811B,d>,100 062,64 3E,STZ &3E
811D,d?,100 063,64 3F,STZ &3F
811F,d@,100 064,64 40,STZ &40
8121,-,165 045,A5 2D,LDA &2D
8123,*,005 042,05 2A,ORA &2A
8125,+,005 043,05 2B,ORA &2B
8127,",",005 044,05 2C,ORA &2C
8129,G,240 071,F0 47,BEQ 71   --> &8172
812B,,160 032,A0 20,LDY#&20
812D,,136,88,DEY
812E,A,240 065,F0 41,BEQ 65   --> &8171
8130,9,006 057,06 39,ASL &39
8132,&:,038 058,26 3A,ROL &3A
8134,&;,038 059,26 3B,ROL &3B
8136,&<,038 060,26 3C,ROL &3C
8138,,016 243,10 F3,BPL -13   --> &812D
813A,&9,038 057,26 39,ROL &39
813C,&:,038 058,26 3A,ROL &3A
813E,&;,038 059,26 3B,ROL &3B
8140,&<,038 060,26 3C,ROL &3C
8142,&=,038 061,26 3D,ROL &3D
8144,&>,038 062,26 3E,ROL &3E
8146,&?,038 063,26 3F,ROL &3F
8148,&@,038 064,26 40,ROL &40
814A,8,056,38,SEC
814B,=,165 061,A5 3D,LDA &3D

814D,*,229 042,E5 2A,SBC &2A
814F,H,072,48,PHA
8150,>,165 062,A5 3E,LDA &3E
8152,+,229 043,E5 2B,SBC &2B
8154,H,072,48,PHA
8155,?,165 063,A5 3F,LDA &3F
8157,",",229 044,E5 2C,SBC &2C
8159,,170,AA,TAX
815A,@,165 064,A5 40,LDA &40
815C,-,229 045,E5 2D,SBC &2D
815E,,144 012,90 0C,BCC 12 --> &816C
8160,@,133 064,85 40,STA &40
8162,?,134 063,86 3F,STX &3F
8164,h,104,68,PLA
8165,>,133 062,85 3E,STA &3E
8167,h,104,68,PLA
8168,=,133 061,85 3D,STA &3D
816A,,176 002,B0 02,BCS 2 --> &816E
816C,h,104,68,PLA
816D,h,104,68,PLA
816E,,136,88,DEY
816F,,208 201,D0 C9,BNE -55 --> &813A
8171,`,096,60,RTS
8172,,000,00,BRK
8173,,018,12,EQUB &12
8174,D,068,44,"""D"""
8175,i,105,69,"""i"""
8176,v,118,76,"""v"""
8177,i,105,69,"""i"""
8178,s,115,73,"""s"""
8179,i,105,69,"""i"""
817A,o,111,6F,"""o"""
817B,n,110,6E,"""n"""
817C,,032,20,""" """
817D,b,098,62,"""b"""
817E,y,121,79,"""y"""
817F,,032,20,""" """
8180,z,122,7A,"""z"""
8181,e,101,65,"""e"""
8182,r,114,72,"""r"""
8183,o,111,6F,"""o"""
8184,,000,00,EQUB &00
8185,d5,100 053,64 35,STZ &35

8187,d/,100 047,64 2F,STZ &2F
8189,-,165 045,A5 2D,LDA &2D
818B,.,133 046,85 2E,STA &2E
818D,,016 005,10 05,BPL 5 --> &8194
818F,,032 222 172,20 DE AC,JSR &ACDE
8192,-,165 045,A5 2D,LDA &2D
8194,&,208 038,D0 26,BNE 38 --> &81BC
8196,d4,100 052,64 34,STZ &34
8198,",",165 044,A5 2C,LDA &2C
819A,,208 020,D0 14,BNE 20 --> &81B0
819C,d3,100 051,64 33,STZ &33
819E,+,165 043,A5 2B,LDA &2B
81A0,,208 006,D0 06,BNE 6 --> &81A8
81A2,d2,100 050,64 32,STZ &32
81A4,*,165 042,A5 2A,LDA &2A
81A6,8,128 056,80 38,BRA 56 --> &81E0
81A8,*,164 042,A4 2A,LDY &2A
81AA,2,132 050,84 32,STY &32
81AC,,160 144,A0 90,LDY#&90
81AE,2,128 050,80 32,BRA 50 --> &81E2
81B0,+,164 043,A4 2B,LDY &2B
81B2,2,132 050,84 32,STY &32
81B4,*,164 042,A4 2A,LDY &2A
81B6,3,132 051,84 33,STY &33
81B8,,160 152,A0 98,LDY#&98
81BA,&,128 038,80 26,BRA 38 --> &81E2
81BC,",",164 044,A4 2C,LDY &2C
81BE,2,132 050,84 32,STY &32
81C0,+,164 043,A4 2B,LDY &2B
81C2,3,132 051,84 33,STY &33
81C4,*,164 042,A4 2A,LDY &2A
81C6,4,132 052,84 34,STY &34
81C8,,160 160,A0 A0,LDY#&A0
81CA,,128 022,80 16,BRA 22 --> &81E2
81CC,d.,100 046,64 2E,STZ &2E
81CE,d0,100 048,64 30,STZ &30
81D0,d/,100 047,64 2F,STZ &2F
81D2,d1,100 049,64 31,STZ &31
81D4,`,096,60,RTS
81D5,,032 180 166,20 B4 A6,JSR &A6B4
81D8,,168,A8,TAY
81D9,,016 005,10 05,BPL 5 --> &81E0
81DB,.,133 046,85 2E,STA &2E

81DD,I,073 255,49 FF,EOR#&FF
81DF,,026,1A,INC A
81E0,,160 136,A0 88,LDY#&88
81E2,,009 000,09 00,ORA#&00
81E4,0,048 012,30 0C,BMI 12 --> &81F2
81E6,,240 228,F0 E4,BEQ -28 --> &81CC
81E8,,136,88,DEY
81E9,4,006 052,06 34,ASL &34
81EB,&3,038 051,26 33,ROL &33
81ED,&2,038 050,26 32,ROL &32
81EF,*,042,2A,ROL A
81F0,,016 246,10 F6,BPL -10 --> &81E8
81F2,1,133 049,85 31,STA &31
81F4,0,132 048,84 30,STY &30
81F6,`,096,60,RTS
81F7,1,165 049,A5 31,LDA &31
81F9,0,048 217,30 D9,BMI -39 --> &81D4
81FB,-,208 045,D0 2D,BNE 45 --> &822A
81FD,2,005 050,05 32,ORA &32
81FF,3,005 051,05 33,ORA &33
8201,4,005 052,05 34,ORA &34
8203,5,005 053,05 35,ORA &35
8205,,240 197,F0 C5,BEQ -59 --> &81CC
8207,0,165 048,A5 30,LDA &30
8209,2,164 050,A4 32,LDY &32
820B,1,132 049,84 31,STY &31
820D,3,164 051,A4 33,LDY &33
820F,2,132 050,84 32,STY &32
8211,4,164 052,A4 34,LDY &34
8213,3,132 051,84 33,STY &33
8215,5,164 053,A4 35,LDY &35
8217,4,132 052,84 34,STY &34
8219,d5,100 053,64 35,STZ &35
821B,8,056,38,SEC
821C,,233 008,E9 08,SBC#&08
821E,,176 002,B0 02,BCS 2 --> &8222
8220,/,198 047,C6 2F,DEC &2F
8222,1,164 049,A4 31,LDY &31
8224,,240 227,F0 E3,BEQ -29 --> &8209
8226,0,048 023,30 17,BMI 23 --> &823F
8228,,128 002,80 02,BRA 2 --> &822C
822A,0,165 048,A5 30,LDA &30
822C,,024,18,CLC

822D,,233 000,E9 00,SBC#&00
822F,,176 002,B0 02,BCS 2   --> &8233
8231,/,198 047,C6 2F,DEC &2F
8233,5,006 053,06 35,ASL &35
8235,&4,038 052,26 34,ROL &34
8237,&3,038 051,26 33,ROL &33
8239,&2,038 050,26 32,ROL &32
823B,&1,038 049,26 31,ROL &31
823D,,016 238,10 EE,BPL -18   --> &822D
823F,0,133 048,85 30,STA &30
8241,`,096,60,RTS
8242,0,165 048,A5 30,LDA &30
8244,",",016 044,10 2C,BPL 44   --> &8272
8246,1,164 049,A4 31,LDY &31
8248,4,240 052,F0 34,BEQ 52   --> &827E
824A,F1,070 049,46 31,LSR &31
824C,f2,102 050,66 32,ROR &32
824E,f3,102 051,66 33,ROR &33
8250,f4,102 052,66 34,ROR &34
8252,,026,1A,INC A
8253,h,240 104,F0 68,BEQ 104   --> &82BD
8255,,201 160,C9 A0,CMP#&A0
8257,g,176 103,B0 67,BCS 103   --> &82C0
8259,,201 153,C9 99,CMP#&99
825B,,176 237,B0 ED,BCS -19   --> &824A
825D,i,105 008,69 08,ADC#&08
825F,3,164 051,A4 33,LDY &33
8261,4,132 052,84 34,STY &34
8263,2,164 050,A4 32,LDY &32
8265,3,132 051,84 33,STY &33
8267,1,164 049,A4 31,LDY &31
8269,2,132 050,84 32,STY &32
826B,d1,100 049,64 31,STZ &31
826D,,128 230,80 E6,BRA -26   --> &8255
826F,,032 011 164,20 0B A4,JSR &A40B
8272,L,076 180 166,4C B4 A6,JMP &A6B4
8275,0,165 048,A5 30,LDA &30
8277,,016 246,10 F6,BPL -10   --> &826F
8279,p,032 112 165,20 70 A5,JSR &A570
827C,1,164 049,A4 31,LDY &31
827E,D,240 068,F0 44,BEQ 68   --> &82C4
8280,F1,070 049,46 31,LSR &31
8282,f2,102 050,66 32,ROR &32

8284,f3,102 051,66 33,ROR &33
8286,f4,102 052,66 34,ROR &34
8288,f=,102 061,66 3D,ROR &3D
828A,f>,102 062,66 3E,ROR &3E
828C,f?,102 063,66 3F,ROR &3F
828E,f@,102 064,66 40,ROR &40
8290,,026,1A,INC A
8291,*,240 042,F0 2A,BEQ 42 --> &82BD
8293,,201 160,C9 A0,CMP#&A0
8295,),176 041,B0 29,BCS 41 --> &82C0
8297,,201 153,C9 99,CMP#&99
8299,,176 229,B0 E5,BCS -27 --> &8280
829B,i,105 008,69 08,ADC#&08
829D,?,164 063,A4 3F,LDY &3F
829F,@,132 064,84 40,STY &40
82A1,>,164 062,A4 3E,LDY &3E
82A3,?,132 063,84 3F,STY &3F
82A5,=,164 061,A4 3D,LDY &3D
82A7,>,132 062,84 3E,STY &3E
82A9,4,164 052,A4 34,LDY &34
82AB,=,132 061,84 3D,STY &3D
82AD,3,164 051,A4 33,LDY &33
82AF,4,132 052,84 34,STY &34
82B1,2,164 050,A4 32,LDY &32
82B3,3,132 051,84 33,STY &33
82B5,1,164 049,A4 31,LDY &31
82B7,2,132 050,84 32,STY &32
82B9,d1,100 049,64 31,STZ &31
82BB,,128 214,80 D6,BRA -42 --> &8293
82BD,L,076 197 166,4C C5 A6,JMP &A6C5
82C0,,208 251,D0 FB,BNE -5 --> &82BD
82C2,0,133 048,85 30,STA &30
82C4,.,165 046,A5 2E,LDA &2E
82C6,,016 023,10 17,BPL 23 --> &82DF
82C8,8,056,38,SEC
82C9,,160 000,A0 00,LDY#&00
82CB,,152,98,TYA
82CC,4,229 052,E5 34,SBC &34
82CE,4,133 052,85 34,STA &34
82D0,,152,98,TYA
82D1,3,229 051,E5 33,SBC &33
82D3,3,133 051,85 33,STA &33
82D5,,152,98,TYA

82D6,2,229 050,E5 32,SBC &32
82D8,2,133 050,85 32,STA &32
82DA,,152,98,TYA
82DB,1,229 049,E5 31,SBC &31
82DD,1,133 049,85 31,STA &31
82DF,`,096,60,RTS
82E0,0,165 048,A5 30,LDA &30
82E2,0,048 005,30 05,BMI 5 --> &82E9
82E4,dI,100 073,64 49,STZ &49
82E6,L,076 242 163,4C F2 A3,JMP &A3F2
82E9,u,032 117 130,20 75 82,JSR &8275
82EC,4,165 052,A5 34,LDA &34
82EE,I,133 073,85 49,STA &49
82F0,S,032 083 131,20 53 83,JSR &8353
82F3,,169 128,A9 80,LDA#&80
82F5,0,133 048,85 30,STA &30
82F7,1,166 049,A6 31,LDX &31
82F9,,016 015,10 0F,BPL 15 --> &830A
82FB,E.,069 046,45 2E,EOR &2E
82FD,.,133 046,85 2E,STA &2E
82FF,,016 004,10 04,BPL 4 --> &8305
8301,I,230 073,E6 49,INC &49
8303,,128 002,80 02,BRA 2 --> &8307
8305,I,198 073,C6 49,DEC &49
8307,,032 200 130,20 C8 82,JSR &82C8
830A,L,076 247 129,4C F7 81,JMP &81F7
830D,4,230 052,E6 34,INC &34
830F,,208 012,D0 0C,BNE 12 --> &831D
8311,3,230 051,E6 33,INC &33
8313,,208 008,D0 08,BNE 8 --> &831D
8315,2,230 050,E6 32,INC &32
8317,,208 004,D0 04,BNE 4 --> &831D
8319,1,230 049,E6 31,INC &31
831B,,240 160,F0 A0,BEQ -96 --> &82BD
831D,`,096,60,RTS
831E,,160 004,A0 04,LDY#&04
8320,f,102 017,66 11,ROR &11
8322,,165 016,A5 10,LDA &10
8324,,170,AA,TAX
8325,j,106,6A,ROR A
8326,,133 017,85 11,STA &11
8328,,165 015,A5 0F,LDA &0F
832A,,133 016,85 10,STA &10

832C,J,074,4A,LSR A
832D,E,069 014,45 0E,EOR &0E
832F,),041 015,29 0F,AND#&0F
8331,E,069 014,45 0E,EOR &0E
8333,j,106,6A,ROR A
8334,j,106,6A,ROR A
8335,j,106,6A,ROR A
8336,j,106,6A,ROR A
8337,E,069 017,45 11,EOR &11
8339,,134 017,86 11,STX &11
833B,,166 014,A6 0E,LDX &0E
833D,,134 015,86 0F,STX &0F
833F,,166 013,A6 0D,LDX &0D
8341,,134 014,86 0E,STX &0E
8343,,133 013,85 0D,STA &0D
8345,,136,88,DEY
8346,,208 216,D0 D8,BNE -40 --> &8320
8348,`,096,60,RTS
8349,;,165 059,A5 3B,LDA &3B
834B,.,133 046,85 2E,STA &2E
834D,d/,100 047,64 2F,STZ &2F
834F,<,165 060,A5 3C,LDA &3C
8351,0,133 048,85 30,STA &30
8353,=,165 061,A5 3D,LDA &3D
8355,1,133 049,85 31,STA &31
8357,>,165 062,A5 3E,LDA &3E
8359,2,133 050,85 32,STA &32
835B,?,165 063,A5 3F,LDA &3F
835D,3,133 051,85 33,STA &33
835F,@,165 064,A5 40,LDA &40
8361,4,133 052,85 34,STA &34
8363,A,165 065,A5 41,LDA &41
8365,5,133 053,85 35,STA &35
8367,`,096,60,RTS
8368,1,165 049,A5 31,LDA &31
836A,,240 221,F0 DD,BEQ -35 --> &8349
836C,8,056,38,SEC
836D,0,165 048,A5 30,LDA &30
836F,<,229 060,E5 3C,SBC &3C
8371,o,240 111,F0 6F,BEQ 111 --> &83E2
8373,4,144 052,90 34,BCC 52 --> &83A9
8375,%,201 037,C9 25,CMP#&25
8377,,176 238,B0 EE,BCS -18 --> &8367

8379,,168,A8,TAY
837A,)8,041 056,29 38,AND#&38
837C,,240 023,F0 17,BEQ 23   --> &8395
837E,8,056,38,SEC
837F,@,166 064,A6 40,LDX &40
8381,A,134 065,86 41,STX &41
8383,?,166 063,A6 3F,LDX &3F
8385,@,134 064,86 40,STX &40
8387,>,166 062,A6 3E,LDX &3E
8389,?,134 063,86 3F,STX &3F
838B,=,166 061,A6 3D,LDX &3D
838D,>,134 062,86 3E,STX &3E
838F,d=,100 061,64 3D,STZ &3D
8391,,233 008,E9 08,SBC#&08
8393,,208 234,D0 EA,BNE -22   --> &837F
8395,,152,98,TYA
8396,),041 007,29 07,AND#&07
8398,H,240 072,F0 48,BEQ 72   --> &83E2
839A,F=,070 061,46 3D,LSR &3D
839C,f>,102 062,66 3E,ROR &3E
839E,f?,102 063,66 3F,ROR &3F
83A0,f@,102 064,66 40,ROR &40
83A2,fA,102 065,66 41,ROR &41
83A4,:,058,3A,DEC A
83A5,,208 243,D0 F3,BNE -13   --> &839A
83A7,9,128 057,80 39,BRA 57   --> &83E2
83A9,I,073 255,49 FF,EOR#&FF
83AB,,026,1A,INC A
83AC,%,201 037,C9 25,CMP#&25
83AE,,176 153,B0 99,BCS -103   --> &8349
83B0,<,164 060,A4 3C,LDY &3C
83B2,0,132 048,84 30,STY &30
83B4,,168,A8,TAY
83B5,)8,041 056,29 38,AND#&38
83B7,,240 023,F0 17,BEQ 23   --> &83D0
83B9,8,056,38,SEC
83BA,4,166 052,A6 34,LDX &34
83BC,5,134 053,86 35,STX &35
83BE,3,166 051,A6 33,LDX &33
83C0,4,134 052,86 34,STX &34
83C2,2,166 050,A6 32,LDX &32
83C4,3,134 051,86 33,STX &33
83C6,1,166 049,A6 31,LDX &31

83C8,2,134 050,86 32,STX &32
83CA,d1,100 049,64 31,STZ &31
83CC,,233 008,E9 08,SBC#&08
83CE,,208 234,D0 EA,BNE -22 --> &83BA
83D0,,152,98,TYA
83D1,),041 007,29 07,AND#&07
83D3,,240 013,F0 0D,BEQ 13 --> &83E2
83D5,F1,070 049,46 31,LSR &31
83D7,f2,102 050,66 32,ROR &32
83D9,f3,102 051,66 33,ROR &33
83DB,f4,102 052,66 34,ROR &34
83DD,f5,102 053,66 35,ROR &35
83DF,:,058,3A,DEC A
83E0,,208 243,D0 F3,BNE -13 --> &83D5
83E2,.,165 046,A5 2E,LDA &2E
83E4,E;,069 059,45 3B,EOR &3B
83E6,0,048 004,30 04,BMI 4 --> &83EC
83E8,,024,18,CLC
83E9,LG,076 071 164,4C 47 A4,JMP &A447
83EC,1,165 049,A5 31,LDA &31
83EE,=,197 061,C5 3D,CMP &3D
83F0,,208 027,D0 1B,BNE 27 --> &840D
83F2,2,165 050,A5 32,LDA &32
83F4,>,197 062,C5 3E,CMP &3E
83F6,,208 021,D0 15,BNE 21 --> &840D
83F8,3,165 051,A5 33,LDA &33
83FA,?,197 063,C5 3F,CMP &3F
83FC,,208 015,D0 0F,BNE 15 --> &840D
83FE,4,165 052,A5 34,LDA &34
8400,@,197 064,C5 40,CMP &40
8402,,208 009,D0 09,BNE 9 --> &840D
8404,5,165 053,A5 35,LDA &35
8406,A,197 065,C5 41,CMP &41
8408,,208 003,D0 03,BNE 3 --> &840D
840A,L,076 180 166,4C B4 A6,JMP &A6B4
840D,&,176 038,B0 26,BCS 38 --> &8435
840F,;,165 059,A5 3B,LDA &3B
8411,.,133 046,85 2E,STA &2E
8413,8,056,38,SEC
8414,A,165 065,A5 41,LDA &41
8416,5,229 053,E5 35,SBC &35
8418,5,133 053,85 35,STA &35
841A,@,165 064,A5 40,LDA &40

841C,4,229 052,E5 34,SBC &34
841E,4,133 052,85 34,STA &34
8420,?,165 063,A5 3F,LDA &3F
8422,3,229 051,E5 33,SBC &33
8424,3,133 051,85 33,STA &33
8426,>,165 062,A5 3E,LDA &3E
8428,2,229 050,E5 32,SBC &32
842A,2,133 050,85 32,STA &32
842C,=,165 061,A5 3D,LDA &3D
842E,1,229 049,E5 31,SBC &31
8430,1,133 049,85 31,STA &31
8432,L,076 249 129,4C F9 81,JMP &81F9
8435,5,165 053,A5 35,LDA &35
8437,A,229 065,E5 41,SBC &41
8439,5,133 053,85 35,STA &35
843B,4,165 052,A5 34,LDA &34
843D,@,229 064,E5 40,SBC &40
843F,4,133 052,85 34,STA &34
8441,3,165 051,A5 33,LDA &33
8443,?,229 063,E5 3F,SBC &3F
8445,3,133 051,85 33,STA &33
8447,2,165 050,A5 32,LDA &32
8449,>,229 062,E5 3E,SBC &3E
844B,2,133 050,85 32,STA &32
844D,1,165 049,A5 31,LDA &31
844F,=,229 061,E5 3D,SBC &3D
8451,1,133 049,85 31,STA &31
8453,L,076 249 129,4C F9 81,JMP &81F9
8456,AN,065 078,41 4E,"EOR (&4E,X)"
8458,D,068,44,xxx Invalid Code
8459,,128 000,80 00,BRA 0   --> &845B
845B,AB,065 066,41 42,"EOR (&42,X)"
845D,S,083,53,xxx Invalid Code
845E,,148 000,94 00,"STY &00,X"
8460,AC,065 067,41 43,"EOR (&43,X)"
8462,S,083,53,xxx Invalid Code
8463,,149 000,95 00,"STA &00,X"
8465,AD,065 068,41 44,"EOR (&44,X)"
8467,VA,086 065,56 41,"LSR &41,X"
8469,L,076 150 000,4C 96 00,JMP &0096
846C,AS,065 083,41 53,"EOR (&53,X)"
846E,C,067,43,xxx Invalid Code
846F,,151,97,xxx Invalid Code

8470,,000,00,BRK
8471,AS,065 083,41 53,"EOR (&53,X)"
8473,N,078 152 000,4E 98 00,LSR &0098
8476,AT,065 084,41 54,"EOR (&54,X)"
8478,N,078 153 000,4E 99 00,LSR &0099
847B,AU,065 085,41 55,"EOR (&55,X)"
847D,T,084,54,xxx Invalid Code
847E,O,079,4F,xxx Invalid Code
847F,,198 016,C6 10,DEC &10
8481,B,066,42,xxx Invalid Code
8482,G,071,47,xxx Invalid Code
8483,ET,069 084,45 54,EOR &54
8485,,154,9A,TXS
8486,B,001 066,01 42,"ORA (&42,X)"
8488,PU,080 085,50 55,BVC 85 --> &84DF
848A,T,084,54,xxx Invalid Code
848B,,213 003,D5 03,"CMP &03,X"
848D,C,067,43,xxx Invalid Code
848E,O,079,4F,xxx Invalid Code
848F,LOU,076 079 085,4C 4F 55,JMP &554F
8492,R,082 251,52 FB,EOR (&FB)
8494,,002,02,xxx Invalid Code
8495,C,067,43,xxx Invalid Code
8496,AL,065 076,41 4C,"EOR (&4C,X)"
8498,L,076 214 002,4C D6 02,JMP &02D6
849B,C,067,43,xxx Invalid Code
849C,H,072,48,PHA
849D,AI,065 073,41 49,"EOR (&49,X)"
849F,N,078 215 002,4E D7 02,LSR &02D7
84A2,C,067,43,xxx Invalid Code
84A3,H,072,48,PHA
84A4,R$,082 036,52 24,EOR (&24)
84A6,C,189 000 067,BD 00 43,"LDA &4300,X"
84A9,LEA,076 069 065,4C 45 41,JMP &4145
84AC,R,082 216,52 D8,EOR (&D8)
84AE,C,001 067,01 43,"ORA (&43,X)"
84B0,LOS,076 079 083,4C 4F 53,JMP &534F
84B3,E,069 217,45 D9,EOR &D9
84B5,,003,03,xxx Invalid Code
84B6,C,067,43,xxx Invalid Code
84B7,LG,076 071 218,4C 47 DA,JMP &DA47
84BA,C,001 067,01 43,"ORA (&43,X)"
84BC,LS,076 083 219,4C 53 DB,JMP &DB53

84BF,C,001 067,01 43,"ORA (&43,X)"
84C1,O,079,4F,xxx Invalid Code
84C2,S,083,53,xxx Invalid Code
84C3,,155,9B,xxx Invalid Code
84C4,,000,00,BRK
84C5,C,067,43,xxx Invalid Code
84C6,O,079,4F,xxx Invalid Code
84C7,UN,085 078,55 4E,"EOR &4E,X"
84C9,T,084,54,xxx Invalid Code
84CA,C,156 001 067,9C 01 43,STZ &4301
84CD,O,079,4F,xxx Invalid Code
84CE,LOR,076 079 082,4C 4F 52,JMP &524F
84D1,,251,FB,xxx Invalid Code
84D2,,002,02,xxx Invalid Code
84D3,D,068,44,xxx Invalid Code
84D4,AT,065 084,41 54,"EOR (&54,X)"
84D6,A,065 220,41 DC,"EOR (&DC,X)"
84D8,DE,032 068 069,20 44 45,JSR &4544
84DB,G,071,47,xxx Invalid Code
84DC,D,157 000 068,9D 00 44,"STA &4400,X"
84DF,EF,069 070,45 46,EOR &46
84E1,D,221 000 068,DD 00 44,"CMP &4400,X"
84E4,EL,069 076,45 4C,EOR &4C
84E6,ET,069 084,45 54,EOR &54
84E8,E,069 199,45 C7,EOR &C7
84EA,D,016 068,10 44,BPL 68   --> &8530
84EC,IV,073 086,49 56,EOR#&56
84EE,,129 000,81 00,"STA (&00,X)"
84F0,D,068,44,xxx Invalid Code
84F1,IM,073 077,49 4D,EOR#&4D
84F3,D,222 002 068,DE 02 44,"DEC &4402,X"
84F6,RA,082 065,52 41,EOR (&41)
84F8,W,087,57,xxx Invalid Code
84F9,,223,DF,xxx Invalid Code
84FA,,002,02,xxx Invalid Code
84FB,EN,069 078,45 4E,EOR &4E
84FD,D,068,44,xxx Invalid Code
84FE,PR,080 082,50 52,BVC 82   --> &8552
8500,O,079,4F,xxx Invalid Code
8501,C,067,43,xxx Invalid Code
8502,,225 001,E1 01,"SBC (&01,X)"
8504,EN,069 078,45 4E,EOR &4E
8506,D,068,44,xxx Invalid Code

8507,,224 001,E0 01,CPX#&01
8509,EN,069 078,45 4E,EOR &4E
850B,VE,086 069,56 45,"LSR &45,X"
850D,LOP,076 079 080,4C 4F 50,JMP &504F
8510,E,069 226,45 E2,EOR &E2
8512,,002,02,xxx Invalid Code
8513,EL,069 076,45 4C,EOR &4C
8515,S,083,53,xxx Invalid Code
8516,E,069 139,45 8B,EOR &8B
8518,E,020 069,14 45,TRB &45
851A,VA,086 065,56 41,"LSR &41,X"
851C,L,076 160 000,4C A0 00,JMP &00A0
851F,ER,069 082,45 52,EOR &52
8521,L,076 158 001,4C 9E 01,JMP &019E
8524,ER,069 082,45 52,EOR &52
8526,RO,082 079,52 4F,EOR (&4F)
8528,R,082 133,52 85,EOR (&85)
852A,E,004 069,04 45,TSB &45
852C,O,079,4F,xxx Invalid Code
852D,F,070 197,46 C5,LSR &C5
852F,E,001 069,01 45,"ORA (&45,X)"
8531,O,079,4F,xxx Invalid Code
8532,R,082 130,52 82,EOR (&82)
8534,,000,00,BRK
8535,ER,069 082,45 52,EOR &52
8537,R,082 159,52 9F,EOR (&9F)
8539,E,001 069,01 45,"ORA (&45,X)"
853B,X,088,58,CLI
853C,P,080 161,50 A1,BVC -95   --> &84DF
853E,,000,00,BRK
853F,EX,069 088,45 58,EOR &58
8541,T,084,54,xxx Invalid Code
8542,,162 001,A2 01,LDX#&01
8544,ED,069 068,45 44,EOR &44
8546,IT,073 084,49 54,EOR#&54
8548,,206,CE,xxx Invalid Code
8549,F,016 070,10 46,BPL 70   --> &8591
854B,O,079,4F,xxx Invalid Code
854C,R,082 227,52 E3,EOR (&E3)
854E,,002,02,xxx Invalid Code
854F,FA,070 065,46 41,LSR &41
8551,LSE,076 083 069,4C 53 45,JMP &4553
8554,,163,A3,xxx Invalid Code

8555,F,001 070,01 46,"ORA (&46,X)"
8557,N,078 164 008,4E A4 08,LSR &08A4
855A,G,071,47,xxx Invalid Code
855B,O,079,4F,xxx Invalid Code
855C,T,084,54,xxx Invalid Code
855D,O,079,4F,xxx Invalid Code
855E,,229 018,E5 12,SBC &12
8560,G,071,47,xxx Invalid Code
8561,ET,069 084,45 54,EOR &54
8563,$,036 190,24 BE,BIT &BE
8565,,000,00,BRK
8566,G,071,47,xxx Invalid Code
8567,ET,069 084,45 54,EOR &54
8569,,165 000,A5 00,LDA &00
856B,G,071,47,xxx Invalid Code
856C,O,079,4F,xxx Invalid Code
856D,S,083,53,xxx Invalid Code
856E,UB,085 066,55 42,"EOR &42,X"
8570,,228 018,E4 12,CPX &12
8572,G,071,47,xxx Invalid Code
8573,C,067,43,xxx Invalid Code
8574,O,079,4F,xxx Invalid Code
8575,L,076 230 002,4C E6 02,JMP &02E6
8578,H,072,48,PHA
8579,IM,073 077,49 4D,EOR#&4D
857B,EM,069 077,45 4D,EOR &4D
857D,,147,93,xxx Invalid Code
857E,C,067,43,xxx Invalid Code
857F,IN,073 078,49 4E,EOR#&4E
8581,PU,080 085,50 55,BVC 85 --> &85D8
8583,T,084,54,xxx Invalid Code
8584,,232,E8,INX
8585,,002,02,xxx Invalid Code
8586,IF,073 070,49 46,EOR#&46
8588,,231,E7,xxx Invalid Code
8589,,002,02,xxx Invalid Code
858A,IN,073 078,49 4E,EOR#&4E
858C,K,075,4B,xxx Invalid Code
858D,EY,069 089,45 59,EOR &59
858F,$,036 191,24 BF,BIT &BF
8591,,000,00,BRK
8592,IN,073 078,49 4E,EOR#&4E
8594,K,075,4B,xxx Invalid Code

8595,EY,069 089,45 59,EOR &59
8597,,166 000,A6 00,LDX &00
8599,IN,073 078,49 4E,EOR#&4E
859B,T,084,54,xxx Invalid Code
859C,,168,A8,TAY
859D,,000,00,BRK
859E,IN,073 078,49 4E,EOR#&4E
85A0,S,083,53,xxx Invalid Code
85A1,T,084,54,xxx Invalid Code
85A2,R(,082 040,52 28,EOR (&28)
85A4,,167,A7,xxx Invalid Code
85A5,,000,00,BRK
85A6,LIS,076 073 083,4C 49 53,JMP &5349
85A9,T,084,54,xxx Invalid Code
85AA,,201 016,C9 10,CMP#&10
85AC,LIN,076 073 078,4C 49 4E,JMP &4E49
85AF,E,069 134,45 86,EOR &86
85B1,,000,00,BRK
85B2,LOA,076 079 065,4C 4F 41,JMP &414F
85B5,D,068,44,xxx Invalid Code
85B6,,200,C8,INY
85B7,,002,02,xxx Invalid Code
85B8,LOM,076 079 077,4C 4F 4D,JMP &4D4F
85BB,EM,069 077,45 4D,EOR &4D
85BD,C,146 067,92 43,STA (&43)
85BF,LOC,076 079 067,4C 4F 43,JMP &434F
85C2,AL,065 076,41 4C,"EOR (&4C,X)"
85C4,,234,EA,NOP
85C5,,002,02,xxx Invalid Code
85C6,LEF,076 069 070,4C 45 46,JMP &4645
85C9,T,084,54,xxx Invalid Code
85CA,$(,036 040,24 28,BIT &28
85CC,,192 000,C0 00,CPY#&00
85CE,LEN,076 069 078,4C 45 4E,JMP &4E45
85D1,,169 000,A9 00,LDA#&00
85D3,LET,076 069 084,4C 45 54,JMP &5445
85D6,,233 004,E9 04,SBC#&04
85D8,LOG,076 079 071,4C 4F 47,JMP &474F
85DB,,171,AB,xxx Invalid Code
85DC,,000,00,BRK
85DD,LN,076 078 170,4C 4E AA,JMP &AA4E
85E0,,000,00,BRK
85E1,MID,077 073 068,4D 49 44,EOR &4449

85E4,$(,036 040,24 28,BIT &28
85E6,,193 000,C1 00,"CMP (&00,X)"
85E8,MOD,077 079 068,4D 4F 44,EOR &444F
85EB,E,069 235,45 EB,EOR &EB
85ED,,002,02,xxx Invalid Code
85EE,MOD,077 079 068,4D 4F 44,EOR &444F
85F1,,131,83,xxx Invalid Code
85F2,,000,00,BRK
85F3,MOV,077 079 086,4D 4F 56,EOR &564F
85F6,E,069 236,45 EC,EOR &EC
85F8,,002,02,xxx Invalid Code
85F9,NEX,078 069 088,4E 45 58,LSR &5845
85FC,T,084,54,xxx Invalid Code
85FD,N,237 002 078,ED 02 4E,SBC &4E02
8600,EW,069 087,45 57,EOR &57
8602,,202,CA,DEX
8603,N,001 078,01 4E,"ORA (&4E,X)"
8605,O,079,4F,xxx Invalid Code
8606,T,084,54,xxx Invalid Code
8607,O,172 000 079,AC 00 4F,LDY &4F00
860A,LD,076 068 203,4C 44 CB,JMP &CB44
860D,O,001 079,01 4F,"ORA (&4F,X)"
860F,N,078 238 002,4E EE 02,LSR &02EE
8612,O,079,4F,xxx Invalid Code
8613,FF,070 070,46 46,LSR &46
8615,,135,87,xxx Invalid Code
8616,,000,00,BRK
8617,O,079,4F,xxx Invalid Code
8618,R,082 132,52 84,EOR (&84)
861A,,000,00,BRK
861B,O,079,4F,xxx Invalid Code
861C,PE,080 069,50 45,BVC 69 --> &8663
861E,NIN,078 073 078,4E 49 4E,LSR &4E49
8621,O,142 000 079,8E 00 4F,STX &4F00
8624,PE,080 069,50 45,BVC 69 --> &866B
8626,NOU,078 079 085,4E 4F 55,LSR &554F
8629,T,084,54,xxx Invalid Code
862A,O,174 000 079,AE 00 4F,LDX &4F00
862D,PE,080 069,50 45,BVC 69 --> &8674
862F,NUP,078 085 080,4E 55 50,LSR &5055
8632,O,173 000 079,AD 00 4F,LDA &4F00
8635,S,083,53,xxx Invalid Code
8636,C,067,43,xxx Invalid Code

8637,LI,076 073 255,4C 49 FF,JMP &FF49
863A,,002,02,xxx Invalid Code
863B,PR,080 082,50 52,BVC 82 --> &868F
863D,IN,073 078,49 4E,EOR#&4E
863F,T,084,54,xxx Invalid Code
8640,,241 002,F1 02,"SBC (&02),Y"
8642,PA,080 065,50 41,BVC 65 --> &8685
8644,G,071,47,xxx Invalid Code
8645,E,069 144,45 90,EOR &90
8647,C,067,43,xxx Invalid Code
8648,PT,080 084,50 54,BVC 84 --> &869E
864A,R,082 143,52 8F,EOR (&8F)
864C,C,067,43,xxx Invalid Code
864D,PI,080 073,50 49,BVC 73 --> &8698
864F,,175,AF,xxx Invalid Code
8650,P,001 080,01 50,"ORA (&50,X)"
8652,LOT,076 079 084,4C 4F 54,JMP &544F
8655,,240 002,F0 02,BEQ 2 --> &8659
8657,PO,080 079,50 4F,BVC 79 --> &86A8
8659,IN,073 078,49 4E,EOR#&4E
865B,T,084,54,xxx Invalid Code
865C,(,040,28,PLP
865D,,176 000,B0 00,BCS 0 --> &865F
865F,PR,080 082,50 52,BVC 82 --> &86B3
8661,O,079,4F,xxx Invalid Code
8662,C,067,43,xxx Invalid Code
8663,,242 010,F2 0A,SBC (&0A)
8665,PO,080 079,50 4F,BVC 79 --> &86B6
8667,S,083,53,xxx Invalid Code
8668,,177 001,B1 01,"LDA (&01),Y"
866A,RE,082 069,52 45,EOR (&45)
866C,T,084,54,xxx Invalid Code
866D,UR,085 082,55 52,"EOR &52,X"
866F,N,078 248 001,4E F8 01,LSR &01F8
8672,RE,082 069,52 45,EOR (&45)
8674,PE,080 069,50 45,BVC 69 --> &86BB
8676,AT,065 084,41 54,"EOR (&54,X)"
8678,,245 000,F5 00,"SBC &00,X"
867A,RE,082 069,52 45,EOR (&45)
867C,PO,080 079,50 4F,BVC 79 --> &86CD
867E,RT,082 084,52 54,EOR (&54)
8680,,246 001,F6 01,"INC &01,X"
8682,RE,082 069,52 45,EOR (&45)

8684,AD,065 068,41 44,"EOR (&44,X)"
8686,,243,F3,xxx Invalid Code
8687,,002,02,xxx Invalid Code
8688,RE,082 069,52 45,EOR (&45)
868A,M,077 244 032,4D F4 20,EOR &20F4
868D,RU,082 085,52 55,EOR (&55)
868F,N,078 249 001,4E F9 01,LSR &01F9
8692,RA,082 065,52 41,EOR (&41)
8694,D,068,44,xxx Invalid Code
8695,,178 000,B2 00,LDA (&00)
8697,RE,082 069,52 45,EOR (&45)
8699,S,083,53,xxx Invalid Code
869A,T,084,54,xxx Invalid Code
869B,O,079,4F,xxx Invalid Code
869C,RE,082 069,52 45,EOR (&45)
869E,,247,F7,xxx Invalid Code
869F,R,018 082,12 52,ORA (&52)
86A1,IG,073 071,49 47,EOR#&47
86A3,H,072,48,PHA
86A4,T,084,54,xxx Invalid Code
86A5,$(,036 040,24 28,BIT &28
86A7,,194,C2,xxx Invalid Code
86A8,,000,00,BRK
86A9,RN,082 078,52 4E,EOR (&4E)
86AB,D,068,44,xxx Invalid Code
86AC,,179,B3,xxx Invalid Code
86AD,R,001 082,01 52,"ORA (&52,X)"
86AF,EN,069 078,45 4E,EOR &4E
86B1,UM,085 077,55 4D,"EOR &4D,X"
86B3,B,066,42,xxx Invalid Code
86B4,ER,069 082,45 52,EOR &52
86B6,S,204 016 083,CC 10 53,CPY &5310
86B9,T,084,54,xxx Invalid Code
86BA,EP,069 080,45 50,EOR &50
86BC,,136,88,DEY
86BD,,000,00,BRK
86BE,S,083,53,xxx Invalid Code
86BF,AV,065 086,41 56,"EOR (&56,X)"
86C1,E,069 205,45 CD,EOR &CD
86C3,,002,02,xxx Invalid Code
86C4,S,083,53,xxx Invalid Code
86C5,G,071,47,xxx Invalid Code
86C6,N,078 180 000,4E B4 00,LSR &00B4

86C9,S,083,53,xxx Invalid Code
86CA,IN,073 078,49 4E,EOR#&4E
86CC,,181 000,B5 00,"LDA &00,X"
86CE,S,083,53,xxx Invalid Code
86CF,QR,081 082,51 52,"EOR (&52),Y"
86D1,,182 000,B6 00,"LDX &00,Y"
86D3,S,083,53,xxx Invalid Code
86D4,PC,080 067,50 43,BVC 67 --> &8719
86D6,,137 000,89 00,BIT#&00
86D8,S,083,53,xxx Invalid Code
86D9,T,084,54,xxx Invalid Code
86DA,R$,082 036,52 24,EOR (&24)
86DC,,195,C3,xxx Invalid Code
86DD,,000,00,BRK
86DE,S,083,53,xxx Invalid Code
86DF,T,084,54,xxx Invalid Code
86E0,RI,082 073,52 49,EOR (&49)
86E2,NG$,078 071 036,4E 47 24,LSR &2447
86E5,(,040,28,PLP
86E6,,196 000,C4 00,CPY &00
86E8,S,083,53,xxx Invalid Code
86E9,O,079,4F,xxx Invalid Code
86EA,UN,085 078,55 4E,"EOR &4E,X"
86EC,D,068,44,xxx Invalid Code
86ED,,212,D4,xxx Invalid Code
86EE,,002,02,xxx Invalid Code
86EF,S,083,53,xxx Invalid Code
86F0,T,084,54,xxx Invalid Code
86F1,O,079,4F,xxx Invalid Code
86F2,P,080 250,50 FA,BVC -6 --> &86EE
86F4,T,001 084,01 54,"ORA (&54,X)"
86F6,AN,065 078,41 4E,"EOR (&4E,X)"
86F8,,183,B7,xxx Invalid Code
86F9,,000,00,BRK
86FA,T,084,54,xxx Invalid Code
86FB,H,072,48,PHA
86FC,EN,069 078,45 4E,EOR &4E
86FE,T,140 020 084,8C 14 54,STY &5414
8701,O,079,4F,xxx Invalid Code
8702,,184,B8,CLV
8703,,000,00,BRK
8704,T,084,54,xxx Invalid Code
8705,AB,065 066,41 42,"EOR (&42,X)"

8707,(,040,28,PLP
8708,,138,8A,TXA
8709,,000,00,BRK
870A,T,084,54,xxx Invalid Code
870B,RA,082 065,52 41,EOR (&41)
870D,C,067,43,xxx Invalid Code
870E,E,069 252,45 FC,EOR &FC
8710,T,018 084,12 54,ORA (&54)
8712,IM,073 077,49 4D,EOR#&4D
8714,E,069 145,45 91,EOR &91
8716,C,067,43,xxx Invalid Code
8717,T,084,54,xxx Invalid Code
8718,RU,082 085,52 55,EOR (&55)
871A,E,069 185,45 B9,EOR &B9
871C,U,001 085,01 55,"ORA (&55,X)"
871E,NTI,078 084 073,4E 54 49,LSR &4954
8721,L,076 253 002,4C FD 02,JMP &02FD
8724,US,085 083,55 53,"EOR &53,X"
8726,R,082 186,52 BA,EOR (&BA)
8728,,000,00,BRK
8729,VD,086 068,56 44,"LSR &44,X"
872B,U,085 239,55 EF,"EOR &EF,X"
872D,,002,02,xxx Invalid Code
872E,VA,086 065,56 41,"LSR &41,X"
8730,L,076 187 000,4C BB 00,JMP &00BB
8733,VP,086 080,56 50,"LSR &50,X"
8735,O,079,4F,xxx Invalid Code
8736,S,083,53,xxx Invalid Code
8737,W,188 001 087,BC 01 57,"LDY &5701,X"
873A,ID,073 068,49 44,EOR#&44
873C,T,084,54,xxx Invalid Code
873D,H,072,48,PHA
873E,P,254 002 080,FE 02 50,"INC &5002,X"
8741,AG,065 071,41 47,"EOR (&47,X)"
8743,E,069 208,45 D0,EOR &D0
8745,,000,00,BRK
8746,PT,080 084,50 54,BVC 84   --> &879C
8748,R,082 207,52 CF,EOR (&CF)
874A,,000,00,BRK
874B,T,084,54,xxx Invalid Code
874C,IM,073 077,49 4D,EOR#&4D
874E,E,069 209,45 D1,EOR &D1
8750,,000,00,BRK

8751,LOM,076 079 077,4C 4F 4D,JMP &4D4F
8754,EM,069 077,45 4D,EOR &4D
8756,,210 000,D2 00,CMP (&00)
8758,H,072,48,PHA
8759,IM,073 077,49 4D,EOR#&4D
875B,EM,069 077,45 4D,EOR &4D
875D,,211,D3,xxx Invalid Code
875E,,000,00,BRK
875F,Mis,077 105 115,4D 69 73,EOR &7369
8762,s,115,73,xxx Invalid Code
8763,in,105 110,69 6E,ADC#&6E
8765,g,103,67,xxx Invalid Code
8766,,032 141 000,20 8D 00,JSR &008D
8769,,223,DF,xxx Invalid Code
876A,,170,AA,TAX
876B,,201 170,C9 AA,CMP#&AA
876D,,008,08,PHP
876E,D,174 068 174,AE 44 AE,LDX &AE44
8771,),041 174,29 AE,AND#&AE
8773,/,047,2F,xxx Invalid Code
8774,,174 183 172,AE B7 AC,LDX &ACB7
8777,,156 168 236,9C A8 EC,STZ &ECA8
877A,,173 179 171,AD B3 AB,LDA &ABB3
877D,,161 168,A1 A8,"LDA (&A8,X)"
877F,,195,C3,xxx Invalid Code
8780,,168,A8,TAY
8781,,215,D7,xxx Invalid Code
8782,,170,AA,TAX
8783,%,014 169 037,0E A9 25,ASL &25A9
8786,,174 216 169,AE D8 A9,LDX &A9D8
8789,5,053 174,35 AE,"AND &AE,X"
878B,;,059,3B,xxx Invalid Code
878C,,174 005 171,AE 05 AB,LDX &AB05
878F,,223,DF,xxx Invalid Code
8790,,169 197,A9 C5,LDA#&C5
8792,,170,AA,TAX
8793,,232,E8,INX
8794,,171,AB,xxx Invalid Code
8795,,023,17,xxx Invalid Code
8796,?,176 063,B0 3F,BCS 63 --> &87D7
8798,,174 194 171,AE C2 AB,LDX &ABC2
879B,6,054 172,36 AC,"ROL &AC,X"
879D,,138,8A,TXA

879E,,171,AB,xxx Invalid Code
879F,,017 174,11 AE,"ORA (&AE),Y"
87A1,F,070 167,46 A7,LSR &A7
87A3,,207,CF,xxx Invalid Code
87A4,,169 147,A9 93,LDA#&93
87A6,,170,AA,TAX
87A7,,231,E7,xxx Invalid Code
87A8,,170,AA,TAX
87A9,,227,E3,xxx Invalid Code
87AA,,170,AA,TAX
87AB,,255,FF,xxx Invalid Code
87AC,,170,AA,TAX
87AD,,014 172 163,0E AC A3,ASL &A3AC
87B0,,170,AA,TAX
87B1,,200,C8,INY
87B2,s,169 115,A9 73,LDA#&73
87B4,,170,AA,TAX
87B5,,245 171,F5 AB,"SBC &AB,X"
87B7,,013 169 181,0D A9 B5,ORA &B5A9
87BA,,167,A7,xxx Invalid Code
87BB,,155,9B,xxx Invalid Code
87BC,,165 249,A5 F9,LDA &F9
87BE,,173 219 171,AD DB AB,LDA &ABDB
87C1,,169 170,A9 AA,LDA#&AA
87C3,I,073 171,49 AB,EOR#&AB
87C5,/,188 170 047,BC AA 2F,"LDY &2FAA,X"
87C8,i,178 105,B2 69,LDA (&69)
87CA,,174 179 174,AE B3 AE,LDX &AEB3
87CD,s,115,73,xxx Invalid Code
87CE,,174 197 174,AE C5 AE,LDX &AEC5
87D1,t,116 174,74 AE,"STZ &AE,X"
87D3,G,028 175 071,1C AF 47,TRB &47AF
87D6,,175,AF,xxx Invalid Code
87D7,,207,CF,xxx Invalid Code
87D8,,171,AB,xxx Invalid Code
87D9,,137 148,89 94,BIT#&94
87DB,,023,17,xxx Invalid Code
87DC,,147,93,xxx Invalid Code
87DD,,032 143 221,20 8F DD,JSR &DD8F
87E0,,179,B3,xxx Invalid Code
87E1,},125 143 000,7D 8F 00,"ADC &008F,X"
87E4,,143,8F,xxx Invalid Code
87E5,,132 147,84 93,STY &93

87E7,U,085 190,55 BE,"EOR &BE,X"
87E9,,147,93,xxx Invalid Code
87EA,,179,B3,xxx Invalid Code
87EB,,151,97,xxx Invalid Code
87EC,4,190 052 150,BE 34 96,"LDX &9634,Y"
87EF,y,121 150 032,79 96 20,"ADC &2096,Y"
87F2,,150 015,96 0F,"STX &0F,Y"
87F4,,150 200,96 C8,"STX &C8,Y"
87F6,,178 189,B2 BD,LDA (&BD)
87F8,,190 190 146,BE BE 92,"LDX &92BE,Y"
87FB,,251,FB,xxx Invalid Code
87FC,>,142 062 150,8E 3E 96,STX &963E
87FF,,174 190 224,AE BE E0,LDX &E0BE
8802,,151,97,xxx Invalid Code
8803,,231,E7,xxx Invalid Code
8804,,151,97,xxx Invalid Code
8805,,174 143 174,AE 8F AE,LDX &AE8F
8808,,143,8F,xxx Invalid Code
8809,4,052 149,34 95,"BIT &95,X"
880B,,166 151,A6 97,LDX &97
880D,%,037 143,25 8F,AND &8F
880F,,154,9A,TXS
8810,,155,9B,xxx Invalid Code
8811,,236 178 024,EC B2 18,CPX &18B2
8814,,182 217,B6 D9,"LDX &D9,Y"
8816,,182 029,B6 1D,"LDX &1D,Y"
8818,,183,B7,xxx Invalid Code
8819,A,065 151,41 97,"EOR (&97,X)"
881B,,008,08,PHP
881C,,156 182 184,9C B6 B8,STZ &B8B6
881F,J,074,4A,LSR A
8820,,144 003,90 03,BCC 3 --> &8825
8822,,151,97,xxx Invalid Code
8823,_,095,5F,xxx Invalid Code
8824,,151,97,xxx Invalid Code
8825,,162 151,A2 97,LDX#&97
8827,,241 180,F1 B4,"SBC (&B4),Y"
8829,[,091,5B,xxx Invalid Code
882A,,183,B7,xxx Invalid Code
882B,,013 152 177,0D 98 B1,ORA &B198
882E,,151,97,xxx Invalid Code
882F,,141 145 228,8D 91 E4,STA &E491
8832,},150 125,96 7D,"STX &7D,Y"

8834,,185 174 143,B9 AE 8F,"LDA &8FAE,Y"
8837,X,088,58,CLI
8838,,186,BA,TSX
8839,,244,F4,xxx Invalid Code
883A,,151,97,xxx Invalid Code
883B,M,077 185 007,4D B9 07,EOR &07B9
883E,,183,B7,xxx Invalid Code
883F,,018 143,12 8F,ORA (&8F)
8841,,134 144,86 90,STX &90
8843,U,085 151,55 97,"EOR &97,X"
8845,F,070 150,46 96,LSR &96
8847,,023,17,xxx Invalid Code
8848,,186,BA,TSX
8849,,023,17,xxx Invalid Code
884A,,179,B3,xxx Invalid Code
884B,,135,87,xxx Invalid Code
884C,K,190 075 131,BE 4B 83,"LDX &834B,Y"
884F,,132 137,84 89,STY &89
8851,,150 184,96 B8,"STX &B8,Y"
8853,,185 216 217,B9 D8 D9,"LDA &D9D8,Y"
8856,,240 001,F0 01,BEQ 1   --> &8859
8858,,016 129,10 81,BPL -127   --> &87DB
885A,,144 137,90 89,BCC -119   --> &87E5
885C,,147,93,xxx Invalid Code
885D,,163,A3,xxx Invalid Code
885E,,164 169,A4 A9,LDY &A9
8860,8,056,38,SEC
8861,9x,057 120 001,39 78 01,"AND &0178,Y"
8864,,019,13,xxx Invalid Code
8865,!,033 161,21 A1,"AND (&A1,X)"
8867,,193 025,C1 19,"CMP (&19,X)"
8869,,024,18,CLC
886A,c,153 152 099,99 98 63,"STA &6398,Y"
886D,s,115,73,xxx Invalid Code
886E,,177 169,B1 A9,"LDA (&A9),Y"
8870,,197 012,C5 0C,CMP &0C
8872,,195,C3,xxx Invalid Code
8873,,211,D3,xxx Invalid Code
8874,A,065 196,41 C4,"EOR (&C4,X)"
8876,A,242 065,F2 41,SBC (&41)
8878,,131,83,xxx Invalid Code
8879,,176 129,B0 81,BCS -127   --> &87FC
887B,C,067,43,xxx Invalid Code

887C,lr,108 114 236,6C 72 EC,JMP (&EC72)
887F,,242 163,F2 A3,SBC (&A3)
8881,,195,C3,xxx Invalid Code
8882,,146 154,92 9A,STA (&9A)
8884,,024,18,CLC
8885,bB,025 098 066,19 62 42,"ORA &4262,Y"
8888,4,052 176,34 B0,"BIT &B0,X"
888A,r,114 152,72 98,ADC (&98)
888C,,153 129 152,99 81 98,"STA &9881,Y"
888F,5,153 020 053,99 14 35,"STA &3514,Y"
8892,,010,0A,ASL A
8893,,013 013 013,0D 0D 0D,ORA &0D0D
8896,,013 016 016,0D 10 10,ORA &1010
8899,%%,037 037,25 25,AND &25
889B,9AA,057 065 065,39 41 41,"AND &4141,Y"
889E,AA,065 065,41 41,"EOR (&41,X)"
88A0,J,074,4A,LSR A
88A1,J,074,4A,LSR A
88A2,LLL,076 076 076,4C 4C 4C,JMP &4C4C
88A5,PP,080 080,50 50,BVC 80   --> &88F7
88A7,RS,082 083,52 53,EOR (&53)
88A9,S,083,53,xxx Invalid Code
88AA,S,083,53,xxx Invalid Code
88AB,%,016 037,10 25,BPL 37   --> &88D2
88AD,AA,065 065,41 41,"EOR (&41,X)"
88AF,AA,065 065,41 41,"EOR (&41,X)"
88B1,,008,08,PHP
88B2,,008,08,PHP
88B3,,008,08,PHP
88B4,,009 009,09 09,ORA#&09
88B6,,010,0A,ASL A
88B7,,010,0A,ASL A
88B8,,010,0A,ASL A
88B9,,010,0A,ASL A
88BA,,005 021,05 15,ORA &15
88BC,>,062 004 013,3E 04 0D,"ROL &0D04,X"
88BF,0L,048 076,30 4C,BMI 76   --> &890D
88C1,2,006 050,06 32,ASL &32
88C3,II,073 073,49 49,EOR#&49
88C5,%,016 037,10 25,BPL 37   --> &88EC
88C7,N,013 078 014,0D 4E 0E,ORA &0E4E
88CA,RR,014 082 082,0E 52 52,ASL &5252
88CD,),009 041,09 29,ORA#&29

88CF,*,042,2A,ROL A
88D0,00,048 048,30 30,BMI 48   --> &8902
88D2,NNN,078 078 078,4E 4E 4E,LSR &4E4E
88D5,>,062 022 000,3E 16 00,"ROL &0016,X"
88D8,,024,18,CLC
88D9,,216,D8,CLD
88DA,X,088,58,CLI
88DB,,184,B8,CLV
88DC,,202,CA,DEX
88DD,,136,88,DEY
88DE,,232,E8,INX
88DF,,200,C8,INY
88E0,,234,EA,NOP
88E1,H,072,48,PHA
88E2,,008,08,PHP
88E3,h,104,68,PLA
88E4,(,040,28,PLP
88E5,@,064,40,RTI
88E6,`,096,60,RTS
88E7,8,056,38,SEC
88E8,,248,F8,SED
88E9,x,120,78,SEI
88EA,,170,AA,TAX
88EB,,168,A8,TAY
88EC,,186,BA,TSX
88ED,,138,8A,TXA
88EE,,154,9A,TXS
88EF,,152,98,TYA
88F0,:,058,3A,DEC A
88F1,,026,1A,INC A
88F2,Z,090,5A,PHY
88F3,,218,DA,PHX
88F4,z,122,7A,PLY
88F5,,250,FA,PLX
88F6,,144 176,90 B0,BCC -80   --> &88A8
88F8,0,240 048,F0 30,BEQ 48   --> &892A
88FA,,208 016,D0 10,BNE 16   --> &890C
88FC,Pp,080 112,50 70,BVC 112   --> &896E
88FE,!,128 033,80 21,BRA 33   --> &8921
8900,A,065 001,41 01,"EOR (&01,X)"
8902,a,097 193 161,61 C1 A1,"ADC (&A1C1,X)"
8905,,225 006,E1 06,"SBC (&06,X)"
8907,F&,070 038,46 26,LSR &26

```
8909,f,102 198,66 C6,ROR &C6
890B,,230 156,E6 9C,INC &9C
890D,,156 224 192,9C E0 C0,STZ &C0E0
8910,,000,00,BRK
8911,$,016 036,10 24,BPL 36   --> &8937
8913,L,076 032 162,4C 20 A2,JMP &A220
8916,,160 129,A0 81,LDY#&81
8918,,134 132,86 84,STX &84
891A,:,058,3A,DEC A
891B,(,133 040,85 28,STA &28
891D,L,076 011 144,4C 0B 90,JMP &900B
8920,,032 224 142,20 E0 8E,JSR &8EE0
8923,I],073 093,49 5D,EOR#&5D
8925,,240 243,F0 F3,BEQ -13   --> &891A
8927,,032 188 155,20 BC 9B,JSR &9BBC
892A,,198 010,C6 0A,DEC &0A
892C,,032 235 137,20 EB 89,JSR &89EB
892F,,198 010,C6 0A,DEC &0A
8931,(,165 040,A5 28,LDA &28
8933,J,074,4A,LSR A
8934,x,144 120,90 78,BCC 120   --> &89AE
8936,,165 030,A5 1E,LDA &1E
8938,i,105 004,69 04,ADC#&04
893A,?,133 063,85 3F,STA &3F
893C,8,165 056,A5 38,LDA &38
893E,l,032 108 189,20 6C BD,JSR &BD6C
8941,7,165 055,A5 37,LDA &37
8943,,032 143 189,20 8F BD,JSR &BD8F
8946,,162 252,A2 FC,LDX#&FC
8948,9,164 057,A4 39,LDY &39
894A,,016 002,10 02,BPL 2   --> &894E
894C,6,164 054,A4 36,LDY &36
894E,8,132 056,84 38,STY &38
8950,,240 025,F0 19,BEQ 25   --> &896B
8952,,160 000,A0 00,LDY#&00
8954,,232,E8,INX
8955,,208 010,D0 0A,BNE 10   --> &8961
8957,,032 146 186,20 92 BA,JSR &BA92
895A,?,166 063,A6 3F,LDX &3F
895C,,032 191 189,20 BF BD,JSR &BDBF
895F,,162 253,A2 FD,LDX#&FD
8961,:,177 058,B1 3A,"LDA (&3A),Y"
8963,,032 143 189,20 8F BD,JSR &BD8F
```

8966,,200,C8,INY
8967,8,198 056,C6 38,DEC &38
8969,,208 233,D0 E9,BNE -23 --> &8954
896B,,138,8A,TXA
896C,,168,A8,TAY
896D,,200,C8,INY
896E,,240 007,F0 07,BEQ 7 --> &8977
8970,,162 003,A2 03,LDX#&03
8972,,032 191 189,20 BF BD,JSR &BDBF
8975,,128 246,80 F6,BRA -10 --> &896D
8977,,162 010,A2 0A,LDX#&0A
8979,,178 011,B2 0B,LDA (&0B)
897B,.,201 046,C9 2E,CMP#&2E
897D,,208 015,D0 0F,BNE 15 --> &898E
897F,7,032 055 189,20 37 BD,JSR &BD37
8982,,202,CA,DEX
8983,,208 002,D0 02,BNE 2 --> &8987
8985,,162 001,A2 01,LDX#&01
8987,,200,C8,INY
8988,,177 011,B1 0B,"LDA (&0B),Y"
898A,N,196 078,C4 4E,CPY &4E
898C,,208 241,D0 F1,BNE -15 --> &897F
898E,,032 191 189,20 BF BD,JSR &BDBF
8991,,136,88,DEY
8992,,200,C8,INY
8993,,209 011,D1 0B,"CMP (&0B),Y"
8995,,240 251,F0 FB,BEQ -5 --> &8992
8997,,177 011,B1 0B,"LDA (&0B),Y"
8999,:,201 058,C9 3A,CMP#&3A
899B,,240 010,F0 0A,BEQ 10 --> &89A7
899D,,201 013,C9 0D,CMP#&0D
899F,,240 010,F0 0A,BEQ 10 --> &89AB
89A1,7,032 055 189,20 37 BD,JSR &BD37
89A4,,200,C8,INY
89A5,,128 240,80 F0,BRA -16 --> &8997
89A7,,196 010,C4 0A,CPY &0A
89A9,,144 246,90 F6,BCC -10 --> &89A1
89AB,,032 146 186,20 92 BA,JSR &BA92
89AE,,164 010,A4 0A,LDY &0A
89B0,,136,88,DEY
89B1,,200,C8,INY
89B2,,177 011,B1 0B,"LDA (&0B),Y"
89B4,:,201 058,C9 3A,CMP#&3A

89B6,,240 004,F0 04,BEQ 4   --> &89BC
89B8,,201 013,C9 0D,CMP#&0D
89BA,,208 245,D0 F5,BNE -11   --> &89B1
89BC,,032 168 155,20 A8 9B,JSR &9BA8
89BF,,178 011,B2 0B,LDA (&0B)
89C1,:,201 058,C9 3A,CMP#&3A
89C3,,240 012,F0 0C,BEQ 12   --> &89D1
89C5,,165 012,A5 0C,LDA &0C
89C7,,201 007,C9 07,CMP#&07
89C9,,208 003,D0 03,BNE 3   --> &89CE
89CB,L,076 134 143,4C 86 8F,JMP &8F86
89CE,,032 222 155,20 DE 9B,JSR &9BDE
89D1,L,076 032 137,4C 20 89,JMP &8920
89D4,,032 174 152,20 AE 98,JSR &98AE
89D7,\,240 092,F0 5C,BEQ 92   --> &8A35
89D9,Z,176 090,B0 5A,BCS 90   --> &8A35
89DB,C,032 067 188,20 43 BC,JSR &BC43
89DE,,032 132 173,20 84 AD,JSR &AD84
89E1,',133 039,85 27,STA &27
89E3,+,032 043 179,20 2B B3,JSR &B32B
89E6,u,032 117 146,20 75 92,JSR &9275
89E9,N,132 078,84 4E,STY &4E
89EB,,032 224 142,20 E0 8E,JSR &8EE0
89EE,,160 000,A0 00,LDY#&00
89F0,d=,100 061,64 3D,STZ &3D
89F2,:,201 058,C9 3A,CMP#&3A
89F4,h,240 104,F0 68,BEQ 104   --> &8A5E
89F6,,201 013,C9 0D,CMP#&0D
89F8,d,240 100,F0 64,BEQ 100   --> &8A5E
89FA,\,201 092,C9 5C,CMP#&5C
89FC,`,240 096,F0 60,BEQ 96   --> &8A5E
89FE,.,201 046,C9 2E,CMP#&2E
8A00,,240 210,F0 D2,BEQ -46   --> &89D4
8A02,,198 010,C6 0A,DEC &0A
8A04,,162 003,A2 03,LDX#&03
8A06,,164 010,A4 0A,LDY &0A
8A08,,230 010,E6 0A,INC &0A
8A0A,,177 011,B1 0B,"LDA (&0B),Y"
8A0C,0*,048 042,30 2A,BMI 42   --> &8A38
8A0E,,201 032,C9 20,CMP#&20
8A10,,240 016,F0 10,BEQ 16   --> &8A22
8A12,,160 005,A0 05,LDY#&05
8A14,,010,0A,ASL A

8A15,,010,0A,ASL A
8A16,,010,0A,ASL A
8A17,,010,0A,ASL A
8A18,&=,038 061,26 3D,ROL &3D
8A1A,&>,038 062,26 3E,ROL &3E
8A1C,,136,88,DEY
8A1D,,208 248,D0 F8,BNE -8 --> &8A17
8A1F,,202,CA,DEX
8A20,,208 228,D0 E4,BNE -28 --> &8A06
8A22,E,162 069,A2 45,LDX#&45
8A24,=,165 061,A5 3D,LDA &3D
8A26,L,221 076 136,DD 4C 88,"CMP &884C,X"
8A29,,208 007,D0 07,BNE 7 --> &8A32
8A2B,,188 145 136,BC 91 88,"LDY &8891,X"
8A2E,>,196 062,C4 3E,CPY &3E
8A30,!,240 033,F0 21,BEQ 33 --> &8A53
8A32,,202,CA,DEX
8A33,,208 241,D0 F1,BNE -15 --> &8A26
8A35,Li,076 105 155,4C 69 9B,JMP &9B69
8A38,),162 041,A2 29,LDX#&29
8A3A,,201 128,C9 80,CMP#&80
8A3C,,240 021,F0 15,BEQ 21 --> &8A53
8A3E,,232,E8,INX
8A3F,,201 130,C9 82,CMP#&82
8A41,,240 016,F0 10,BEQ 16 --> &8A53
8A43,,232,E8,INX
8A44,,201 132,C9 84,CMP#&84
8A46,,208 237,D0 ED,BNE -19 --> &8A35
8A48,,230 010,E6 0A,INC &0A
8A4A,,200,C8,INY
8A4B,,177 011,B1 0B,"LDA (&0B),Y"
8A4D,),041 223,29 DF,AND#&DF
8A4F,A,201 065,C9 41,CMP#&41
8A51,,208 226,D0 E2,BNE -30 --> &8A35
8A53,,189 214 136,BD D6 88,"LDA &88D6,X"
8A56,),133 041,85 29,STA &29
8A58,,160 001,A0 01,LDY#&01
8A5A,,224 032,E0 20,CPX#&20
8A5C,H,176 072,B0 48,BCS 72 --> &8AA6
8A5E,@,173 064 004,AD 40 04,LDA &0440
8A61,7,133 055,85 37,STA &37
8A63,9,132 057,84 39,STY &39
8A65,(,166 040,A6 28,LDX &28

8A67,,224 004,E0 04,CPX#&04
8A69,A,174 065 004,AE 41 04,LDX &0441
8A6C,8,134 056,86 38,STX &38
8A6E,,144 006,90 06,BCC 6   --> &8A76
8A70,<,173 060 004,AD 3C 04,LDA &043C
8A73,=,174 061 004,AE 3D 04,LDX &043D
8A76,:,133 058,85 3A,STA &3A
8A78,;,134 059,86 3B,STX &3B
8A7A,,152,98,TYA
8A7B,(,240 040,F0 28,BEQ 40   --> &8AA5
8A7D,,016 004,10 04,BPL 4   --> &8A83
8A7F,6,164 054,A4 36,LDY &36
8A81,"""",240 034,F0 22,BEQ 34   --> &8AA5
8A83,,136,88,DEY
8A84,),185 041 000,B9 29 00,"LDA &0029,Y"
8A87,$9,036 057,24 39,BIT &39
8A89,,016 003,10 03,BPL 3   --> &8A8E
8A8B,,185 000 006,B9 00 06,"LDA &0600,Y"
8A8E,:,145 058,91 3A,"STA (&3A),Y"
8A90,@,238 064 004,EE 40 04,INC &0440
8A93,,208 003,D0 03,BNE 3   --> &8A98
8A95,A,238 065 004,EE 41 04,INC &0441
8A98,,144 008,90 08,BCC 8   --> &8AA2
8A9A,<,238 060 004,EE 3C 04,INC &043C
8A9D,,208 003,D0 03,BNE 3   --> &8AA2
8A9F,=,238 061 004,EE 3D 04,INC &043D
8AA2,,152,98,TYA
8AA3,,208 222,D0 DE,BNE -34   --> &8A83
8AA5,`,096,60,RTS
8AA6,),224 041,E0 29,CPX#&29
8AA8,<,176 060,B0 3C,BCS 60   --> &8AE6
8AAA,o,032 111 146,20 6F 92,JSR &926F
8AAD,,024,18,CLC
8AAE,*,165 042,A5 2A,LDA &2A
8AB0,@,237 064 004,ED 40 04,SBC &0440
8AB3,,168,A8,TAY
8AB4,+,165 043,A5 2B,LDA &2B
8AB6,A,237 065 004,ED 41 04,SBC &0441
8AB9,,192 001,C0 01,CPY#&01
8ABB,,136,88,DEY
8ABC,,233 000,E9 00,SBC#&00
8ABE,,240 027,F0 1B,BEQ 27   --> &8ADB
8AC0,,026,1A,INC A

8AC1,,208 003,D0 03,BNE 3   --> &8AC6
8AC3,,152,98,TYA
8AC4,0,048 025,30 19,BMI 25   --> &8ADF
8AC6,(,165 040,A5 28,LDA &28
8AC8,),041 002,29 02,AND#&02
8ACA,,240 018,F0 12,BEQ 18   --> &8ADE
8ACC,,000,00,BRK
8ACD,,001,01,EQUB &01
8ACE,O,079,4F,"ORA (&4F,X)"
8ACF,ut,117 116,75 74,"ADC &74,X"
8AD1,of,032 111 102,20 6F 66,JSR &666F
8AD4,ra,032 114 097,20 72 61,JSR &6172
8AD7,nge,110 103 101,6E 67 65,ROR &6567
8ADA,,000,00,EQUB &00
8ADB,,152,98,TYA
8ADC,0,048 232,30 E8,BMI -24   --> &8AC6
8ADE,,168,A8,TAY
8ADF,*,132 042,84 2A,STY &2A
8AE1,,160 002,A0 02,LDY#&02
8AE3,L^,076 094 138,4C 5E 8A,JMP &8A5E
8AE6,0,224 048,E0 30,CPX#&30
8AE8,,176 022,B0 16,BCS 22   --> &8B00
8AEA,,032 223 140,20 DF 8C,JSR &8CDF
8AED,,208 024,D0 18,BNE 24   --> &8B07
8AEF,,032 204 140,20 CC 8C,JSR &8CCC
8AF2,o,032 111 146,20 6F 92,JSR &926F
8AF5,+,165 043,A5 2B,LDA &2B
8AF7,,240 232,F0 E8,BEQ -24   --> &8AE1
8AF9,,000,00,BRK
8AFA,,002,02,EQUB &02
8AFB,B,066,42,xxx Invalid Code
8AFC,yte,121 116 101,79 74 65,"ADC &6574,Y"
8AFF,,000,00,EQUB &00
8B00,A,224 065,E0 41,CPX#&41
8B02,c,208 099,D0 63,BNE 99   --> &8B67
8B04,,032 224 142,20 E0 8E,JSR &8EE0
8B07,(,201 040,C9 28,CMP#&28
8B09,9,208 057,D0 39,BNE 57   --> &8B44
8B0B,o,032 111 146,20 6F 92,JSR &926F
8B0E,,032 224 142,20 E0 8E,JSR &8EE0
8B11,),201 041,C9 29,CMP#&29
8B13,,208 023,D0 17,BNE 23   --> &8B2C
8B15,,032 201 140,20 C9 8C,JSR &8CC9

8B18,,032 229 140,20 E5 8C,JSR &8CE5
8B1B,,240 004,F0 04,BEQ 4 --> &8B21
8B1D,),230 041,E6 29,INC &29
8B1F,,128 212,80 D4,BRA -44 --> &8AF5
8B21,,032 224 142,20 E0 8E,JSR &8EE0
8B24,),041 223,29 DF,AND#&DF
8B26,Y,201 089,C9 59,CMP#&59
8B28,,240 203,F0 CB,BEQ -53 --> &8AF5
8B2A,,128 016,80 10,BRA 16 --> &8B3C
8B2C,",",201 044,C9 2C,CMP#&2C
8B2E,,208 012,D0 0C,BNE 12 --> &8B3C
8B30,,032 215 140,20 D7 8C,JSR &8CD7
8B33,,208 007,D0 07,BNE 7 --> &8B3C
8B35,,032 224 142,20 E0 8E,JSR &8EE0
8B38,),201 041,C9 29,CMP#&29
8B3A,,240 185,F0 B9,BEQ -71 --> &8AF5
8B3C,,000,00,BRK
8B3D,,003,03,EQUB &03
8B3E,In,073 110,49 6E,EOR#&6E
8B40,de,100 101,64 65,STZ &65
8B42,x,120,78,SEI
8B43,,000,00,EQUB &00
8B44,m,032 109 146,20 6D 92,JSR &926D
8B47,,032 229 140,20 E5 8C,JSR &8CE5
8B4A,,208 018,D0 12,BNE 18 --> &8B5E
8B4C,,032 201 140,20 C9 8C,JSR &8CC9
8B4F,,032 215 140,20 D7 8C,JSR &8CD7
8B52,,240 010,F0 0A,BEQ 10 --> &8B5E
8B54,Y,201 089,C9 59,CMP#&59
8B56,,208 228,D0 E4,BNE -28 --> &8B3C
8B58,,032 204 140,20 CC 8C,JSR &8CCC
8B5B,L,076 254 139,4C FE 8B,JMP &8BFE
8B5E,,032 207 140,20 CF 8C,JSR &8CCF
8B61,+,165 043,A5 2B,LDA &2B
8B63,,208 243,D0 F3,BNE -13 --> &8B58
8B65,,128 144,80 90,BRA -112 --> &8AF7
8B67,6,224 054,E0 36,CPX#&36
8B69,6,176 054,B0 36,BCS 54 --> &8BA1
8B6B,,032 224 142,20 E0 8E,JSR &8EE0
8B6E,),041 223,29 DF,AND#&DF
8B70,A,201 065,C9 41,CMP#&41
8B72,,240 018,F0 12,BEQ 18 --> &8B86
8B74,m,032 109 146,20 6D 92,JSR &926D

8B77,,032 229 140,20 E5 8C,JSR &8CE5
8B7A,,208 229,D0 E5,BNE -27 --> &8B61
8B7C,,032 201 140,20 C9 8C,JSR &8CC9
8B7F,,032 215 140,20 D7 8C,JSR &8CD7
8B82,,240 221,F0 DD,BEQ -35 --> &8B61
8B84,,128 182,80 B6,BRA -74 --> &8B3C
8B86,,200,C8,INY
8B87,,177 011,B1 0B,"LDA (&0B),Y"
8B89,,032 132 141,20 84 8D,JSR &8D84
8B8C,,176 230,B0 E6,BCS -26 --> &8B74
8B8E,,160 022,A0 16,LDY#&16
8B90,4,224 052,E0 34,CPX#&34
8B92,,144 006,90 06,BCC 6 --> &8B9A
8B94,,208 002,D0 02,BNE 2 --> &8B98
8B96,6,160 054,A0 36,LDY#&36
8B98,),132 041,84 29,STY &29
8B9A,,032 207 140,20 CF 8C,JSR &8CCF
8B9D,,160 001,A0 01,LDY#&01
8B9F,_,128 095,80 5F,BRA 95 --> &8C00
8BA1,8,224 056,E0 38,CPX#&38
8BA3,%,176 037,B0 25,BCS 37 --> &8BCA
8BA5,o,032 111 146,20 6F 92,JSR &926F
8BA8,,160 003,A0 03,LDY#&03
8BAA,,162 001,A2 01,LDX#&01
8BAC,+,165 043,A5 2B,LDA &2B
8BAE,,208 007,D0 07,BNE 7 --> &8BB7
8BB0,,162 015,A2 0F,LDX#&0F
8BB2,d,169 100,A9 64,LDA#&64
8BB4,),133 041,85 29,STA &29
8BB6,,136,88,DEY
8BB7,Z,090,5A,PHY
8BB8,,032 229 140,20 E5 8C,JSR &8CE5
8BBB,,208 010,D0 0A,BNE 10 --> &8BC7
8BBD,,032 215 140,20 D7 8C,JSR &8CD7
8BC0,,208 194,D0 C2,BNE -62 --> &8B84
8BC2,,138,8A,TXA
8BC3,e),101 041,65 29,ADC &29
8BC5,),133 041,85 29,STA &29
8BC7,z,122,7A,PLY
8BC8,6,128 054,80 36,BRA 54 --> &8C00
8BCA,<,224 060,E0 3C,CPX#&3C
8BCC,,176 028,B0 1C,BCS 28 --> &8BEA
8BCE,:,224 058,E0 3A,CPX#&3A

8BD0,,176 007,B0 07,BCS 7   --> &8BD9
8BD2,,032 223 140,20 DF 8C,JSR &8CDF
8BD5,,240 016,F0 10,BEQ 16   --> &8BE7
8BD7,,198 010,C6 0A,DEC &0A
8BD9,o,032 111 146,20 6F 92,JSR &926F
8BDC,,128 128,80 80,BRA 128   --> &8C5E
8BDE,,032 223 140,20 DF 8C,JSR &8CDF
8BE1,,208 145,D0 91,BNE -111   --> &8B74
8BE3,,160 137,A0 89,LDY#&89
8BE5,),132 041,84 29,STY &29
8BE7,L,076 242 138,4C F2 8A,JMP &8AF2
8BEA,,240 242,F0 F2,BEQ -14   --> &8BDE
8BEC,>,224 062,E0 3E,CPX#&3E
8BEE,,240 011,F0 0B,BEQ 11   --> &8BFB
8BF0,7,176 055,B0 37,BCS 55   --> &8C29
8BF2,,032 224 142,20 E0 8E,JSR &8EE0
8BF5,(,201 040,C9 28,CMP#&28
8BF7,,240 010,F0 0A,BEQ 10   --> &8C03
8BF9,,198 010,C6 0A,DEC &0A
8BFB,o,032 111 146,20 6F 92,JSR &926F
8BFE,,160 003,A0 03,LDY#&03
8C00,L^,076 094 138,4C 5E 8A,JMP &8A5E
8C03,,032 201 140,20 C9 8C,JSR &8CC9
8C06,,032 201 140,20 C9 8C,JSR &8CC9
8C09,o,032 111 146,20 6F 92,JSR &926F
8C0C,,032 224 142,20 E0 8E,JSR &8EE0
8C0F,),201 041,C9 29,CMP#&29
8C11,,240 235,F0 EB,BEQ -21   --> &8BFE
8C13,",",201 044,C9 2C,CMP#&2C
8C15,,208 015,D0 0F,BNE 15   --> &8C26
8C17,,032 201 140,20 C9 8C,JSR &8CC9
8C1A,,032 215 140,20 D7 8C,JSR &8CD7
8C1D,,208 007,D0 07,BNE 7   --> &8C26
8C1F,,032 224 142,20 E0 8E,JSR &8EE0
8C22,),201 041,C9 29,CMP#&29
8C24,,240 216,F0 D8,BEQ -40   --> &8BFE
8C26,L<,076 060 139,4C 3C 8B,JMP &8B3C
8C29,D,224 068,E0 44,CPX#&44
8C2B,M,176 077,B0 4D,BCS 77   --> &8C7A
8C2D,=,165 061,A5 3D,LDA &3D
8C2F,I,073 001,49 01,EOR#&01
8C31,),041 031,29 1F,AND#&1F
8C33,H,072,48,PHA

8C34,A,224 065,E0 41,CPX#&41
8C36,!,176 033,B0 21,BCS 33 --> &8C59
8C38,,032 223 140,20 DF 8C,JSR &8CDF
8C3B,,208 003,D0 03,BNE 3 --> &8C40
8C3D,h,104,68,PLA
8C3E,,128 167,80 A7,BRA -89 --> &8BE7
8C40,m,032 109 146,20 6D 92,JSR &926D
8C43,h,104,68,PLA
8C44,7,133 055,85 37,STA &37
8C46,,032 229 140,20 E5 8C,JSR &8CE5
8C49,,208 145,D0 91,BNE -111 --> &8BDC
8C4B,,032 224 142,20 E0 8E,JSR &8EE0
8C4E,),041 031,29 1F,AND#&1F
8C50,7,197 055,C5 37,CMP &37
8C52,,208 210,D0 D2,BNE -46 --> &8C26
8C54,,032 201 140,20 C9 8C,JSR &8CC9
8C57,,128 131,80 83,BRA -125 --> &8BDC
8C59,o,032 111 146,20 6F 92,JSR &926F
8C5C,h,104,68,PLA
8C5D,7,133 055,85 37,STA &37
8C5F,,032 229 140,20 E5 8C,JSR &8CE5
8C62,,208 019,D0 13,BNE 19 --> &8C77
8C64,,032 224 142,20 E0 8E,JSR &8EE0
8C67,),041 031,29 1F,AND#&1F
8C69,7,197 055,C5 37,CMP &37
8C6B,,208 185,D0 B9,BNE -71 --> &8C26
8C6D,,032 201 140,20 C9 8C,JSR &8CC9
8C70,+,165 043,A5 2B,LDA &2B
8C72,,240 003,F0 03,BEQ 3 --> &8C77
8C74,L,076 249 138,4C F9 8A,JMP &8AF9
8C77,La,076 097 139,4C 61 8B,JMP &8B61
8C7A,,208 011,D0 0B,BNE 11 --> &8C87
8C7C,o,032 111 146,20 6F 92,JSR &926F
8C7F,*,165 042,A5 2A,LDA &2A
8C81,(,133 040,85 28,STA &28
8C83,,160 000,A0 00,LDY#&00
8C85,*,128 042,80 2A,BRA 42 --> &8CB1
8C87,,162 001,A2 01,LDX#&01
8C89,,164 010,A4 0A,LDY &0A
8C8B,,230 010,E6 0A,INC &0A
8C8D,,177 011,B1 0B,"LDA (&0B),Y"
8C8F,),041 223,29 DF,AND#&DF
8C91,B,201 066,C9 42,CMP#&42

8C93,,240 018,F0 12,BEQ 18   --> &8CA7
8C95,,232,E8,INX
8C96,W,201 087,C9 57,CMP#&57
8C98,,240 013,F0 0D,BEQ 13   --> &8CA7
8C9A,,162 004,A2 04,LDX#&04
8C9C,D,201 068,C9 44,CMP#&44
8C9E,,240 007,F0 07,BEQ 7   --> &8CA7
8CA0,S,201 083,C9 53,CMP#&53
8CA2,,240 019,F0 13,BEQ 19   --> &8CB7
8CA4,Li,076 105 155,4C 69 9B,JMP &9B69
8CA7,,218,DA,PHX
8CA8,o,032 111 146,20 6F 92,JSR &926F
8CAB,),162 041,A2 29,LDX#&29
8CAD,,032 198 189,20 C6 BD,JSR &BDC6
8CB0,z,122,7A,PLY
8CB1,L^,076 094 138,4C 5E 8A,JMP &8A5E
8CB4,L,076 146 144,4C 92 90,JMP &9092
8CB7,(,165 040,A5 28,LDA &28
8CB9,H,072,48,PHA
8CBA,/,032 047 157,20 2F 9D,JSR &9D2F
8CBD,,208 245,D0 F5,BNE -11   --> &8CB4
8CBF,h,104,68,PLA
8CC0,(,133 040,85 28,STA &28
8CC2,u,032 117 146,20 75 92,JSR &9275
8CC5,,160 255,A0 FF,LDY#&FF
8CC7,,128 232,80 E8,BRA -24   --> &8CB1
8CC9,,032 204 140,20 CC 8C,JSR &8CCC
8CCC,,032 207 140,20 CF 8C,JSR &8CCF
8CCF,),165 041,A5 29,LDA &29
8CD1,,024,18,CLC
8CD2,i,105 004,69 04,ADC#&04
8CD4,),133 041,85 29,STA &29
8CD6,`,096,60,RTS
8CD7,,032 224 142,20 E0 8E,JSR &8EE0
8CDA,),041 223,29 DF,AND#&DF
8CDC,X,201 088,C9 58,CMP#&58
8CDE,`,096,60,RTS
8CDF,,032 224 142,20 E0 8E,JSR &8EE0
8CE2,#,201 035,C9 23,CMP#&23
8CE4,`,096,60,RTS
8CE5,,032 224 142,20 E0 8E,JSR &8EE0
8CE8,",",201 044,C9 2C,CMP#&2C
8CEA,`,096,60,RTS

8CEB,7,146 055,92 37,STA (&37)
8CED,,024,18,CLC
8CEE,,152,98,TYA
8CEF,e7,101 055,65 37,ADC &37
8CF1,9,133 057,85 39,STA &39
8CF3,,160 000,A0 00,LDY#&00
8CF5,,152,98,TYA
8CF6,e8,101 056,65 38,ADC &38
8CF8,:,133 058,85 3A,STA &3A
8CFA,,200,C8,INY
8CFB,9,177 057,B1 39,"LDA (&39),Y"
8CFD,7,145 055,91 37,"STA (&37),Y"
8CFF,,201 013,C9 0D,CMP#&0D
8D01,,208 247,D0 F7,BNE -9 --> &8CFA
8D03,`,096,60,RTS
8D04,),041 015,29 0F,AND#&0F
8D06,=,133 061,85 3D,STA &3D
8D08,,162 000,A2 00,LDX#&00
8D0A,,160 000,A0 00,LDY#&00
8D0C,,200,C8,INY
8D0D,7,177 055,B1 37,"LDA (&37),Y"
8D0F,,032 148 141,20 94 8D,JSR &8D94
8D12,.,144 046,90 2E,BCC 46 --> &8D42
8D14,),041 015,29 0F,AND#&0F
8D16,H,072,48,PHA
8D17,>,134 062,86 3E,STX &3E
8D19,=,165 061,A5 3D,LDA &3D
8D1B,,010,0A,ASL A
8D1C,&>,038 062,26 3E,ROL &3E
8D1E,0,048 031,30 1F,BMI 31 --> &8D3F
8D20,,010,0A,ASL A
8D21,&>,038 062,26 3E,ROL &3E
8D23,0,048 026,30 1A,BMI 26 --> &8D3F
8D25,e=,101 061,65 3D,ADC &3D
8D27,=,133 061,85 3D,STA &3D
8D29,,138,8A,TXA
8D2A,e>,101 062,65 3E,ADC &3E
8D2C,=,006 061,06 3D,ASL &3D
8D2E,*,042,2A,ROL A
8D2F,0,048 014,30 0E,BMI 14 --> &8D3F
8D31,,176 012,B0 0C,BCS 12 --> &8D3F
8D33,,170,AA,TAX
8D34,h,104,68,PLA

8D35,e=,101 061,65 3D,ADC &3D
8D37,=,133 061,85 3D,STA &3D
8D39,,144 209,90 D1,BCC -47 --> &8D0C
8D3B,,232,E8,INX
8D3C,,016 206,10 CE,BPL -50 --> &8D0C
8D3E,H,072,48,PHA
8D3F,h,104,68,PLA
8D40,8,056,38,SEC
8D41,`,096,60,RTS
8D42,,136,88,DEY
8D43,,169 141,A9 8D,LDA#&8D
8D45,,032 235 140,20 EB 8C,JSR &8CEB
8D48,7,165 055,A5 37,LDA &37
8D4A,9,133 057,85 39,STA &39
8D4C,8,165 056,A5 38,LDA &38
8D4E,:,133 058,85 3A,STA &3A
8D50,,032 162 141,20 A2 8D,JSR &8DA2
8D53,,032 162 141,20 A2 8D,JSR &8DA2
8D56,,032 162 141,20 A2 8D,JSR &8DA2
8D59,9,177 057,B1 39,"LDA (&39),Y"
8D5B,7,145 055,91 37,"STA (&37),Y"
8D5D,,136,88,DEY
8D5E,,208 249,D0 F9,BNE -7 --> &8D59
8D60,,160 003,A0 03,LDY#&03
8D62,,138,8A,TXA
8D63,@,009 064,09 40,ORA#&40
8D65,9,145 057,91 39,"STA (&39),Y"
8D67,,136,88,DEY
8D68,=,165 061,A5 3D,LDA &3D
8D6A,)?,041 063,29 3F,AND#&3F
8D6C,@,009 064,09 40,ORA#&40
8D6E,9,145 057,91 39,"STA (&39),Y"
8D70,,136,88,DEY
8D71,?,169 063,A9 3F,LDA#&3F
8D73,=,020 061,14 3D,TRB &3D
8D75,,138,8A,TXA
8D76,),041 192,29 C0,AND#&C0
8D78,J,074,4A,LSR A
8D79,J,074,4A,LSR A
8D7A,=,005 061,05 3D,ORA &3D
8D7C,J,074,4A,LSR A
8D7D,J,074,4A,LSR A
8D7E,IT,073 084,49 54,EOR#&54

```
8D80,9,145 057,91 39,"STA (&39),Y"
8D82,,024,18,CLC
8D83,`,096,60,RTS
8D84,{,201 123,C9 7B,CMP#&7B
8D86,,176 250,B0 FA,BCS -6   --> &8D82
8D88,_,201 095,C9 5F,CMP#&5F
8D8A,,176 014,B0 0E,BCS 14   --> &8D9A
8D8C,[,201 091,C9 5B,CMP#&5B
8D8E,,176 242,B0 F2,BCS -14   --> &8D82
8D90,A,201 065,C9 41,CMP#&41
8D92,,176 006,B0 06,BCS 6   --> &8D9A
8D94,:,201 058,C9 3A,CMP#&3A
8D96,,176 234,B0 EA,BCS -22   --> &8D82
8D98,0,201 048,C9 30,CMP#&30
8D9A,`,096,60,RTS
8D9B,.,201 046,C9 2E,CMP#&2E
8D9D,,208 245,D0 F5,BNE -11   --> &8D94
8D9F,`,096,60,RTS
8DA0,7,178 055,B2 37,LDA (&37)
8DA2,7,230 055,E6 37,INC &37
8DA4,9,208 057,D0 39,BNE 57   --> &8DDF
8DA6,8,230 056,E6 38,INC &38
8DA8,`,096,60,RTS
8DA9,,032 162 141,20 A2 8D,JSR &8DA2
8DAC,7,178 055,B2 37,LDA (&37)
8DAE,`,096,60,RTS
8DAF,,032 162 141,20 A2 8D,JSR &8DA2
8DB2,7,178 055,B2 37,LDA (&37)
8DB4,,201 013,C9 0D,CMP#&0D
8DB6,',240 039,F0 27,BEQ 39   --> &8DDF
8DB8,,201 032,C9 20,CMP#&20
8DBA,,240 243,F0 F3,BEQ -13   --> &8DAF
8DBC,&,201 038,C9 26,CMP#&26
8DBE,,208 016,D0 10,BNE 16   --> &8DD0
8DC0,,032 169 141,20 A9 8D,JSR &8DA9
8DC3,,032 148 141,20 94 8D,JSR &8D94
8DC6,,176 248,B0 F8,BCS -8   --> &8DC0
8DC8,A,201 065,C9 41,CMP#&41
8DCA,,144 230,90 E6,BCC -26   --> &8DB2
8DCC,G,201 071,C9 47,CMP#&47
8DCE,,144 240,90 F0,BCC -16   --> &8DC0
8DD0,"""",201 034,C9 22,CMP#&22
8DD2,,208 012,D0 0C,BNE 12   --> &8DE0
```

8DD4,,032 169 141,20 A9 8D,JSR &8DA9
8DD7,"""",201 034,C9 22,CMP#&22
8DD9,,240 212,F0 D4,BEQ -44 --> &8DAF
8DDB,,201 013,C9 0D,CMP#&0D
8DDD,,208 245,D0 F5,BNE -11 --> &8DD4
8DDF,`,096,60,RTS
8DE0,:,201 058,C9 3A,CMP#&3A
8DE2,,208 009,D0 09,BNE 9 --> &8DED
8DE4,,032 162 141,20 A2 8D,JSR &8DA2
8DE7,d;,100 059,64 3B,STZ &3B
8DE9,d<,100 060,64 3C,STZ &3C
8DEB,,128 197,80 C5,BRA -59 --> &8DB2
8DED,",",201 044,C9 2C,CMP#&2C
8DEF,,240 190,F0 BE,BEQ -66 --> &8DAF
8DF1,*,201 042,C9 2A,CMP#&2A
8DF3,,208 012,D0 0C,BNE 12 --> &8E01
8DF5,;,165 059,A5 3B,LDA &3B
8DF7,,240 230,F0 E6,BEQ -26 --> &8DDF
8DF9,,162 255,A2 FF,LDX#&FF
8DFB,;,134 059,86 3B,STX &3B
8DFD,d<,100 060,64 3C,STZ &3C
8DFF,,128 174,80 AE,BRA -82 --> &8DAF
8E01,.,201 046,C9 2E,CMP#&2E
8E03,,240 014,F0 0E,BEQ 14 --> &8E13
8E05,,032 148 141,20 94 8D,JSR &8D94
8E08,",",144 044,90 2C,BCC 44 --> &8E36
8E0A,<,166 060,A6 3C,LDX &3C
8E0C,,240 005,F0 05,BEQ 5 --> &8E13
8E0E,,032 004 141,20 04 8D,JSR &8D04
8E11,,144 156,90 9C,BCC -100 --> &8DAF
8E13,7,178 055,B2 37,LDA (&37)
8E15,,032 155 141,20 9B 8D,JSR &8D9B
8E18,,144 005,90 05,BCC 5 --> &8E1F
8E1A,,032 162 141,20 A2 8D,JSR &8DA2
8E1D,,128 244,80 F4,BRA -12 --> &8E13
8E1F,,162 255,A2 FF,LDX#&FF
8E21,;,134 059,86 3B,STX &3B
8E23,,128 196,80 C4,BRA -60 --> &8DE9
8E25,,032 132 141,20 84 8D,JSR &8D84
8E28,,144 207,90 CF,BCC -49 --> &8DF9
8E2A,7,178 055,B2 37,LDA (&37)
8E2C,,032 132 141,20 84 8D,JSR &8D84
8E2F,,144 238,90 EE,BCC -18 --> &8E1F

8E31,,032 162 141,20 A2 8D,JSR &8DA2
8E34,,128 244,80 F4,BRA -12 --> &8E2A
8E36,A,201 065,C9 41,CMP#&41
8E38,,144 191,90 BF,BCC -65 --> &8DF9
8E3A,X,201 088,C9 58,CMP#&58
8E3C,,176 231,B0 E7,BCS -25 --> &8E25
8E3E,V,162 086,A2 56,LDX#&56
8E40,9,134 057,86 39,STX &39
8E42,,162 132,A2 84,LDX#&84
8E44,:,134 058,86 3A,STX &3A
8E46,,160 000,A0 00,LDY#&00
8E48,9,210 057,D2 39,CMP (&39)
8E4A,,144 222,90 DE,BCC -34 --> &8E2A
8E4C,,208 015,D0 0F,BNE 15 --> &8E5D
8E4E,,200,C8,INY
8E4F,9,177 057,B1 39,"LDA (&39),Y"
8E51,01,048 049,30 31,BMI 49 --> &8E84
8E53,7,209 055,D1 37,"CMP (&37),Y"
8E55,,240 247,F0 F7,BEQ -9 --> &8E4E
8E57,7,177 055,B1 37,"LDA (&37),Y"
8E59,.,201 046,C9 2E,CMP#&2E
8E5B,,240 011,F0 0B,BEQ 11 --> &8E68
8E5D,,200,C8,INY
8E5E,9,177 057,B1 39,"LDA (&39),Y"
8E60,,016 251,10 FB,BPL -5 --> &8E5D
8E62,,201 254,C9 FE,CMP#&FE
8E64,,208 015,D0 0F,BNE 15 --> &8E75
8E66,,176 194,B0 C2,BCS -62 --> &8E2A
8E68,,200,C8,INY
8E69,9,177 057,B1 39,"LDA (&39),Y"
8E6B,0,048 023,30 17,BMI 23 --> &8E84
8E6D,9,230 057,E6 39,INC &39
8E6F,,208 248,D0 F8,BNE -8 --> &8E69
8E71,:,230 058,E6 3A,INC &3A
8E73,,128 244,80 F4,BRA -12 --> &8E69
8E75,8,056,38,SEC
8E76,,200,C8,INY
8E77,,152,98,TYA
8E78,e9,101 057,65 39,ADC &39
8E7A,9,133 057,85 39,STA &39
8E7C,,144 002,90 02,BCC 2 --> &8E80
8E7E,:,230 058,E6 3A,INC &3A
8E80,7,178 055,B2 37,LDA (&37)

```
8E82,,128 194,80 C2,BRA -62   --> &8E46
8E84,,170,AA,TAX
8E85,,200,C8,INY
8E86,9,177 057,B1 39,"LDA (&39),Y"
8E88,=,133 061,85 3D,STA &3D
8E8A,,136,88,DEY
8E8B,J,074,4A,LSR A
8E8C,,144 007,90 07,BCC 7   --> &8E95
8E8E,7,177 055,B1 37,"LDA (&37),Y"
8E90,,032 132 141,20 84 8D,JSR &8D84
8E93,,176 149,B0 95,BCS -107   --> &8E2A
8E95,,138,8A,TXA
8E96,$=,036 061,24 3D,BIT &3D
8E98,P,080 006,50 06,BVC 6   --> &8EA0
8E9A,;,166 059,A6 3B,LDX &3B
8E9C,,208 002,D0 02,BNE 2   --> &8EA0
8E9E,i@,105 064,69 40,ADC#&40
8EA0,,136,88,DEY
8EA1,,032 235 140,20 EB 8C,JSR &8CEB
8EA4,,162 255,A2 FF,LDX#&FF
8EA6,=,165 061,A5 3D,LDA &3D
8EA8,J,074,4A,LSR A
8EA9,J,074,4A,LSR A
8EAA,,144 004,90 04,BCC 4   --> &8EB0
8EAC,;,134 059,86 3B,STX &3B
8EAE,d<,100 060,64 3C,STZ &3C
8EB0,J,074,4A,LSR A
8EB1,,144 004,90 04,BCC 4   --> &8EB7
8EB3,d;,100 059,64 3B,STZ &3B
8EB5,d<,100 060,64 3C,STZ &3C
8EB7,J,074,4A,LSR A
8EB8,,144 016,90 10,BCC 16   --> &8ECA
8EBA,H,072,48,PHA
8EBB,,160 001,A0 01,LDY#&01
8EBD,7,177 055,B1 37,"LDA (&37),Y"
8EBF,,032 132 141,20 84 8D,JSR &8D84
8EC2,,144 005,90 05,BCC 5   --> &8EC9
8EC4,,032 162 141,20 A2 8D,JSR &8DA2
8EC7,,128 244,80 F4,BRA -12   --> &8EBD
8EC9,h,104,68,PLA
8ECA,J,074,4A,LSR A
8ECB,,144 002,90 02,BCC 2   --> &8ECF
8ECD,<,134 060,86 3C,STX &3C
```

8ECF,J,074,4A,LSR A
8ED0,,176 013,B0 0D,BCS 13 --> &8EDF
8ED2,L,076 175 141,4C AF 8D,JMP &8DAF
8ED5,,164 027,A4 1B,LDY &1B
8ED7,,230 027,E6 1B,INC &1B
8ED9,,177 025,B1 19,"LDA (&19),Y"
8EDB,,201 032,C9 20,CMP#&20
8EDD,,240 246,F0 F6,BEQ -10 --> &8ED5
8EDF,`,096,60,RTS
8EE0,,164 010,A4 0A,LDY &0A
8EE2,,230 010,E6 0A,INC &0A
8EE4,,177 011,B1 0B,"LDA (&0B),Y"
8EE6,,201 032,C9 20,CMP#&20
8EE8,,240 246,F0 F6,BEQ -10 --> &8EE0
8EEA,`,096,60,RTS
8EEB,,032 213 142,20 D5 8E,JSR &8ED5
8EEE,",",201 044,C9 2C,CMP#&2C
8EF0,`,096,60,RTS
8EF1,,032 235 142,20 EB 8E,JSR &8EEB
8EF4,,240 244,F0 F4,BEQ -12 --> &8EEA
8EF6,,000,00,BRK
8EF7,,005,05,EQUB &05
8EF8,,141,8D,xxx Invalid Code
8EF9,",",044,2C,xxx Invalid Code
8EFA,,000,00,EQUB &00
8EFB,,032 215 189,20 D7 BD,JSR &BDD7
8EFE,,128 021,80 15,BRA 21 --> &8F15
8F00,,032 166 155,20 A6 9B,JSR &9BA6
8F03,,165 024,A5 18,LDA &18
8F05,8,133 056,85 38,STA &38
8F07,d7,100 055,64 37,STZ &37
8F09,,169 000,A9 00,LDA#&00
8F0B,7,145 055,91 37,"STA (&37),Y"
8F0D,,032 229 189,20 E5 BD,JSR &BDE5
8F10,q,128 113,80 71,BRA 113 --> &8F83
8F12,,032 166 155,20 A6 9B,JSR &9BA6
8F15,,032 172 187,20 AC BB,JSR &BBAC
8F18,,165 024,A5 18,LDA &18
8F1A,,133 012,85 0C,STA &0C
8F1C,d,100 011,64 0B,STZ &0B
8F1E,w,128 119,80 77,BRA 119 --> &8F97
8F20,,032 215 189,20 D7 BD,JSR &BDD7
8F23,^,128 094,80 5E,BRA 94 --> &8F83

8F25,,032 166 155,20 A6 9B,JSR &9BA6
8F28,,032 229 189,20 E5 BD,JSR &BDE5
8F2B,Y,128 089,80 59,BRA 89 --> &8F86
8F2D,,169 242,A9 F2,LDA#&F2
8F2F,,032 024 174,20 18 AE,JSR &AE18
8F32,,032 226 190,20 E2 BE,JSR &BEE2
8F35,,170,AA,TAX
8F36,,032 226 190,20 E2 BE,JSR &BEE2
8F39,+,133 043,85 2B,STA &2B
8F3B,*,134 042,86 2A,STX &2A
8F3D,,162 020,A2 14,LDX#&14
8F3F,,202,CA,DEX
8F40,>,240 062,F0 3E,BEQ 62 --> &8F80
8F42,,032 226 190,20 E2 BE,JSR &BEE2
8F45,,201 013,C9 0D,CMP#&0D
8F47,7,240 055,F0 37,BEQ 55 --> &8F80
8F49,@,201 064,C9 40,CMP#&40
8F4B,,208 242,D0 F2,BNE -14 --> &8F3F
8F4D,,032 226 190,20 E2 BE,JSR &BEE2
8F50,,201 013,C9 0D,CMP#&0D
8F52,",",208 044,D0 2C,BNE 44 --> &8F80
8F54,,032 254 190,20 FE BE,JSR &BEFE
8F57,,160 000,A0 00,LDY#&00
8F59,d,100 011,64 0B,STZ &0B
8F5B,,169 007,A9 07,LDA#&07
8F5D,,133 012,85 0C,STA &0C
8F5F,,178 000,B2 00,LDA (&00)
8F61,,240 032,F0 20,BEQ 32 --> &8F83
8F63,,145 011,91 0B,"STA (&0B),Y"
8F65,,200,C8,INY
8F66,,240 024,F0 18,BEQ 24 --> &8F80
8F68,,230 000,E6 00,INC &00
8F6A,,208 002,D0 02,BNE 2 --> &8F6E
8F6C,,230 001,E6 01,INC &01
8F6E,,201 013,C9 0D,CMP#&0D
8F70,,208 237,D0 ED,BNE -19 --> &8F5F
8F72,,165 001,A5 01,LDA &01
8F74,,197 007,C5 07,CMP &07
8F76,,176 011,B0 0B,BCS 11 --> &8F83
8F78,,032 235 186,20 EB BA,JSR &BAEB
8F7B,,128 218,80 DA,BRA -38 --> &8F57
8F7D,,032 166 155,20 A6 9B,JSR &9BA6
8F80,,032 254 190,20 FE BE,JSR &BEFE

8F83,,032 172 187,20 AC BB,JSR &BBAC
8F86,,160 007,A0 07,LDY#&07
8F88,,132 012,84 0C,STY &0C
8F8A,d,100 011,64 0B,STZ &0B
8F8C,,032 166 178,20 A6 B2,JSR &B2A6
8F8F,>,169 062,A9 3E,LDA#&3E
8F91,,032 238 255,20 EE FF,JSR &FFEE
8F94,t,032 116 186,20 74 BA,JSR &BA74
8F97,,162 255,A2 FF,LDX#&FF
8F99,,154,9A,TXS
8F9A,,032 166 178,20 A6 B2,JSR &B2A6
8F9D,,032 235 186,20 EB BA,JSR &BAEB
8FA0,,176 225,B0 E1,BCS -31 --> &8F83
8FA2,z,128 122,80 7A,BRA 122 --> &901E
8FA4,,032 188 155,20 BC 9B,JSR &9BBC
8FA7,,166 011,A6 0B,LDX &0B
8FA9,,164 012,A4 0C,LDY &0C
8FAB,,032 247 255,20 F7 FF,JSR &FFF7
8FAE,,169 013,A9 0D,LDA#&0D
8FB0,,164 010,A4 0A,LDY &0A
8FB2,,136,88,DEY
8FB3,,200,C8,INY
8FB4,,209 011,D1 0B,"CMP (&0B),Y"
8FB6,,208 251,D0 FB,BNE -5 --> &8FB3
8FB8,,032 188 155,20 BC 9B,JSR &9BBC
8FBB,,128 004,80 04,BRA 4 --> &8FC1
8FBD,,201 013,C9 0D,CMP#&0D
8FBF,,208 237,D0 ED,BNE -19 --> &8FAE
8FC1,,165 012,A5 0C,LDA &0C
8FC3,,201 007,C9 07,CMP#&07
8FC5,,240 191,F0 BF,BEQ -65 --> &8F86
8FC7,,160 001,A0 01,LDY#&01
8FC9,,177 011,B1 0B,"LDA (&0B),Y"
8FCB,0,048 185,30 B9,BMI -71 --> &8F86
8FCD,,166 032,A6 20,LDX &20
8FCF,,240 010,F0 0A,BEQ 10 --> &8FDB
8FD1,+,133 043,85 2B,STA &2B
8FD3,,200,C8,INY
8FD4,,177 011,B1 0B,"LDA (&0B),Y"
8FD6,*,133 042,85 2A,STA &2A
8FD8,K,032 075 156,20 4B 9C,JSR &9C4B
8FDB,,160 004,A0 04,LDY#&04
8FDD,,132 010,84 0A,STY &0A

8FDF,",",128 044,80 2C,BRA 44   --> &900D
8FE1,,169 003,A9 03,LDA#&03
8FE3,(,133 040,85 28,STA &28
8FE5,L,076 032 137,4C 20 89,JMP &8920
8FE8,L,076 147 190,4C 93 BE,JMP &BE93
8FEB,,164 010,A4 0A,LDY &0A
8FED,,136,88,DEY
8FEE,,177 011,B1 0B,"LDA (&0B),Y"
8FF0,*,201 042,C9 2A,CMP#&2A
8FF2,,240 176,F0 B0,BEQ -80   --> &8FA4
8FF4,[,201 091,C9 5B,CMP#&5B
8FF6,,240 233,F0 E9,BEQ -23   --> &8FE1
8FF8,,201 162,C9 A2,CMP#&A2
8FFA,,240 236,F0 EC,BEQ -20   --> &8FE8
8FFC,=,201 061,C9 3D,CMP#&3D
8FFE,`,240 096,F0 60,BEQ 96   --> &9060
9000,,198 010,C6 0A,DEC &0A

BASIC IV ROM Routines

Disassembly 9000 to A000

9000		198 010	C6 0A	DEC &0A
9002		032 166 155	20 A6 9B	JSR &9BA6
9005		178 011	B2 0B	LDA (&0B)
9007	:	201 058	C9 3A	CMP#&3A
9009		208 178	D0 B2	BNE -78 --> &8FBD
900B		164 010	A4 0A	LDY &0A
900D		230 010	E6 0A	INC &0A
900F		177 011	B1 0B	LDA (&0B),Y
9011		201 032	C9 20	CMP#&20
9013		240 246	F0 F6	BEQ -10 --> &900B
9015		201 207	C9 CF	CMP#&CF
9017		144 012	90 0C	BCC 12 --> &9025
9019		010	0A	ASL A
901A		170	AA	TAX
901B	|M	124 077 135	7C 4D 87	JMP (&874D,X)

901E		032 224 142	20 E0 8E	JSR &8EE0
9021		201 198	C9 C6	CMP#&C6
9023		176 244	B0 F4	BCS -12 --> &9019
9025		166 011	A6 0B	LDX &0B
9027		134 025	86 19	STX &19
9029		166 012	A6 0C	LDX &0C
902B		134 026	86 1A	STX &1A
902D		132 027	84 1B	STY &1B
902F		032 009 153	20 09 99	JSR &9909
9032		208 027	D0 1B	BNE 27 --> &904F
9034		176 181	B0 B5	BCS -75 --> &8FEB
9036		134 027	86 1B	STX &1B
9038		032 134 155	20 86 9B	JSR &9B86
903B	T	032 084 152	20 54 98	JSR &9854
903E		162 005	A2 05	LDX#&05
9040	,	228 044	E4 2C	CPX &2C
9042		208 001	D0 01	BNE 1 --> &9045
9044		232	E8	INX
9045		032 131 152	20 83 98	JSR &9883
9048		198 010	C6 0A	DEC &0A
904A		032 174 152	20 AE 98	JSR &98AE
904D	4	240 052	F0 34	BEQ 52 --> &9083
904F	!	144 033	90 21	BCC 33 --> &9072
9051	&	032 038 188	20 26 BC	JSR &BC26
9054	R	032 082 155	20 52 9B	JSR &9B52
9057	'	165 039	A5 27	LDA &27
9059	7	208 055	D0 37	BNE 55 --> &9092
905B		032 171 144	20 AB 90	JSR &90AB
905E		128 165	80 A5	BRA -91 --> &9005
9060		186	BA	TSX
9061		224 252	E0 FC	CPX#&FC
9063	'	176 039	B0 27	BCS 39 --> &908C
9065		173 255 001	AD FF 01	LDA &01FF
9068		201 164	C9 A4	CMP#&A4
906A		208 032	D0 20	BNE 32 --> &908C

906C	/	032 047 157	20 2F 9D	JSR &9D2F
906F	L	076 145 155	4C 91 9B	JMP &9B91
9072	*	165 042	A5 2A	LDA &2A
9074	H	072	48	PHA
9075	+	165 043	A5 2B	LDA &2B
9077	H	072	48	PHA
9078	,	165 044	A5 2C	LDA &2C
907A	H	072	48	PHA
907B	R	032 082 155	20 52 9B	JSR &9B52
907E	+	032 043 179	20 2B B3	JSR &B32B
9081		128 130	80 82	BRA -126 --> &9005
9083	Li	076 105 155	4C 69 9B	JMP &9B69
9086		032 166 155	20 A6 9B	JSR &9BA6
9089		000	00	BRK
908A		000	00	EQUB &00
908B		250	FA	PLX
908C		000	00	BRK
908D		007	07	EQUB &07
908E	No	078 111 032	4E 6F 20	LSR &206F
9091		164	A4	xxx Invalid Code
9092		000	00	BRK
9093		006	06	EQUB &06
9094	T	084	54	xxx Invalid Code
9095	ype	121 112 101	79 70 65	ADC &6570,Y
9098	mi	032 109 105	20 6D 69	JSR &696D
909B	s	115	73	xxx Invalid Code
909C	mat	109 097 116	6D 61 74	ADC &7461
909F	c	099	63	xxx Invalid Code
90A0	h	104	68	PLA
90A1		000	00	BRK
90A2		000	00	EQUB &00
90A3	No	078 111 032	4E 6F 20	LSR &206F
90A6	ro	114 111	72 6F	ADC (&6F)
90A8	o	111	6F	xxx Invalid Code
90A9	m	109	6D	xxx Invalid Code

90AA		000	00	EQUB &00
90AB		032 230 188	20 E6 BC	JSR &BCE6
90AE	,	165 044	A5 2C	LDA &2C
90B0		201 128	C9 80	CMP#&80
90B2	x	240 120	F0 78	BEQ 120 --> &912C
90B4		160 002	A0 02	LDY#&02
90B6	*	177 042	B1 2A	LDA (&2A),Y
90B8	6	197 054	C5 36	CMP &36
90BA	R	176 082	B0 52	BCS 82 --> &910E
90BC		165 002	A5 02	LDA &02
90BE	,	133 044	85 2C	STA &2C
90C0		165 003	A5 03	LDA &03
90C2	-	133 045	85 2D	STA &2D
90C4	6	165 054	A5 36	LDA &36
90C6		201 008	C9 08	CMP#&08
90C8		144 006	90 06	BCC 6 --> &90D0
90CA	i	105 007	69 07	ADC#&07
90CC		144 002	90 02	BCC 2 --> &90D0
90CE		169 255	A9 FF	LDA#&FF
90D0		024	18	CLC
90D1	H	072	48	PHA
90D2		170	AA	TAX
90D3	*	177 042	B1 2A	LDA (&2A),Y
90D5	r*	114 042	72 2A	ADC (&2A)
90D7	E	069 002	45 02	EOR &02
90D9		208 015	D0 0F	BNE 15 --> &90EA
90DB		136	88	DEY
90DC	q*	113 042	71 2A	ADC (&2A),Y
90DE	E	069 003	45 03	EOR &03
90E0		208 008	D0 08	BNE 8 --> &90EA
90E2	-	133 045	85 2D	STA &2D
90E4		138	8A	TXA
90E5		200	C8	INY
90E6	8	056	38	SEC
90E7	*	241 042	F1 2A	SBC (&2A),Y

```
90E9        170          AA       TAX
90EA        138          8A       TXA
90EB        024          18       CLC
90EC    e   101 002      65 02    ADC &02
90EE        168          A8       TAY
90EF        165 003      A5 03    LDA &03
90F1    i   105 000      69 00    ADC#&00
90F3        170          AA       TAX
90F4        196 004      C4 04    CPY &04
90F6        229 005      E5 05    SBC &05
90F8        176 167      B0 A7    BCS -89 --> &90A1
90FA        132 002      84 02    STY &02
90FC        134 003      86 03    STX &03
90FE    h   104          68       PLA
90FF        160 002      A0 02    LDY#&02
9101    *   145 042      91 2A    STA (&2A),Y
9103        136          88       DEY
9104    -   165 045      A5 2D    LDA &2D
9106        240 006      F0 06    BEQ 6 --> &910E
9108    *   145 042      91 2A    STA (&2A),Y
910A    ,   165 044      A5 2C    LDA &2C
910C    *   146 042      92 2A    STA (&2A)
910E        160 003      A0 03    LDY#&03
9110    6   165 054      A5 36    LDA &36
9112    *   145 042      91 2A    STA (&2A),Y
9114        240 021      F0 15    BEQ 21 --> &912B
9116        160 001      A0 01    LDY#&01
9118    *   177 042      B1 2A    LDA (&2A),Y
911A    -   133 045      85 2D    STA &2D
911C    *   178 042      B2 2A    LDA (&2A)
911E    ,   133 044      85 2C    STA &2C
9120        136          88       DEY
9121        185 000 006  B9 00 06 LDA &0600,Y
9124    ,   145 044      91 2C    STA (&2C),Y
9126        200          C8       INY
```

9127	6	196 054	C4 36	CPY &36
9129		208 246	D0 F6	BNE -10 --> &9121
912B	`	096	60	RTS
912C	+	032 043 190	20 2B BE	JSR &BE2B
912F		192 000	C0 00	CPY#&00
9131		240 011	F0 0B	BEQ 11 --> &913E
9133		185 000 006	B9 00 06	LDA &0600,Y
9136	*	145 042	91 2A	STA (&2A),Y
9138		136	88	DEY
9139		208 248	D0 F8	BNE -8 --> &9133
913B		173 000 006	AD 00 06	LDA &0600
913E	*	146 042	92 2A	STA (&2A)
9140	`	096	60	RTS
9141	<	032 060 186	20 3C BA	JSR &BA3C
9144	Z	090	5A	PHY
9145		032 235 142	20 EB 8E	JSR &8EEB
9148	=	208 061	D0 3D	BNE 61 --> &9187
914A	;	032 059 157	20 3B 9D	JSR &9D3B
914D		032 017 165	20 11 A5	JSR &A511
9150	z	122	7A	PLY
9151	'	165 039	A5 27	LDA &27
9153		032 212 255	20 D4 FF	JSR &FFD4
9156		170	AA	TAX
9157		240 027	F0 1B	BEQ 27 --> &9174
9159	0	048 012	30 0C	BMI 12 --> &9167
915B		162 003	A2 03	LDX#&03
915D	*	181 042	B5 2A	LDA &2A,X
915F		032 212 255	20 D4 FF	JSR &FFD4
9162		202	CA	DEX
9163		016 248	10 F8	BPL -8 --> &915D
9165		128 221	80 DD	BRA -35 --> &9144
9167		162 004	A2 04	LDX#&04
9169	l	189 108 004	BD 6C 04	LDA &046C,X
916C		032 212 255	20 D4 FF	JSR &FFD4
916F		202	CA	DEX

9170		016 247	10 F7	BPL -9 --> &9169
9172		128 208	80 D0	BRA -48 --> &9144
9174	6	165 054	A5 36	LDA &36
9176		032 212 255	20 D4 FF	JSR &FFD4
9179		170	AA	TAX
917A		240 200	F0 C8	BEQ -56 --> &9144
917C		189 255 005	BD FF 05	LDA &05FF,X
917F		032 212 255	20 D4 FF	JSR &FFD4
9182		202	CA	DEX
9183		208 247	D0 F7	BNE -9 --> &917C
9185		128 189	80 BD	BRA -67 --> &9144
9187	h	104	68	PLA
9188		132 010	84 0A	STY &0A
918A	L	076 002 144	4C 02 90	JMP &9002
918D		032 223 140	20 DF 8C	JSR &8CDF
9190		240 175	F0 AF	BEQ -81 --> &9141
9192		198 010	C6 0A	DEC &0A
9194		128 021	80 15	BRA 21 --> &91AB
9196		173 000 004	AD 00 04	LDA &0400
9199		240 016	F0 10	BEQ 16 --> &91AB
919B		165 030	A5 1E	LDA &1E
919D		240 012	F0 0C	BEQ 12 --> &91AB
919F		237 000 004	ED 00 04	SBC &0400
91A2		176 249	B0 F9	BCS -7 --> &919D
91A4		168	A8	TAY
91A5		032 146 189	20 92 BD	JSR &BD92
91A8		200	C8	INY
91A9		208 250	D0 FA	BNE -6 --> &91A5
91AB		024	18	CLC
91AC		173 000 004	AD 00 04	LDA &0400
91AF		133 020	85 14	STA &14
91B1	f	102 021	66 15	ROR &15
91B3		032 224 142	20 E0 8E	JSR &8EE0
91B6	:	201 058	C9 3A	CMP#&3A
91B8		240 008	F0 08	BEQ 8 --> &91C2

91BA		201 013	C9 0D	CMP#&0D
91BC		240 004	F0 04	BEQ 4 --> &91C2
91BE		201 139	C9 8B	CMP#&8B
91C0		208 025	D0 19	BNE 25 --> &91DB
91C2		032 146 186	20 92 BA	JSR &BA92
91C5	L	076 000 144	4C 00 90	JMP &9000
91C8	d	100 020	64 14	STZ &14
91CA	d	100 021	64 15	STZ &15
91CC		032 224 142	20 E0 8E	JSR &8EE0
91CF	:	201 058	C9 3A	CMP#&3A
91D1		240 242	F0 F2	BEQ -14 --> &91C5
91D3		201 013	C9 0D	CMP#&0D
91D5		240 238	F0 EE	BEQ -18 --> &91C5
91D7		201 139	C9 8B	CMP#&8B
91D9		240 234	F0 EA	BEQ -22 --> &91C5
91DB	~	201 126	C9 7E	CMP#&7E
91DD		240 210	F0 D2	BEQ -46 --> &91B1
91DF	,	201 044	C9 2C	CMP#&2C
91E1		240 179	F0 B3	BEQ -77 --> &9196
91E3	;	201 059	C9 3B	CMP#&3B
91E5		240 225	F0 E1	BEQ -31 --> &91C8
91E7	z	032 122 146	20 7A 92	JSR &927A
91EA		144 199	90 C7	BCC -57 --> &91B3
91EC		165 020	A5 14	LDA &14
91EE	H	072	48	PHA
91EF		165 021	A5 15	LDA &15
91F1	H	072	48	PHA
91F2		198 027	C6 1B	DEC &1B
91F4	;	032 059 157	20 3B 9D	JSR &9D3B
91F7	h	104	68	PLA
91F8		133 021	85 15	STA &15
91FA	h	104	68	PLA
91FB		133 020	85 14	STA &14
91FD		165 027	A5 1B	LDA &1B
91FF		133 010	85 0A	STA &0A

9201		152	98	TYA
9202		240 019	F0 13	BEQ 19 --> &9217
9204		032 024 161	20 18 A1	JSR &A118
9207		165 020	A5 14	LDA &14
9209	8	056	38	SEC
920A	6	229 054	E5 36	SBC &36
920C		144 009	90 09	BCC 9 --> &9217
920E		240 007	F0 07	BEQ 7 --> &9217
9210		168	A8	TAY
9211		032 146 189	20 92 BD	JSR &BD92
9214		136	88	DEY
9215		208 250	D0 FA	BNE -6 --> &9211
9217	6	165 054	A5 36	LDA &36
9219		240 152	F0 98	BEQ -104 --> &91B3
921B		160 000	A0 00	LDY#&00
921D		185 000 006	B9 00 06	LDA &0600,Y
9220		032 152 189	20 98 BD	JSR &BD98
9223		200	C8	INY
9224	6	196 054	C4 36	CPY &36
9226		208 245	D0 F5	BNE -11 --> &921D
9228		128 137	80 89	BRA -119 --> &91B3
922A	L	076 246 142	4C F6 8E	JMP &8EF6
922D	*	165 042	A5 2A	LDA &2A
922F	H	072	48	PHA
9230		032 167 150	20 A7 96	JSR &96A7
9233		169 031	A9 1F	LDA#&1F
9235		032 238 255	20 EE FF	JSR &FFEE
9238	h	104	68	PLA
9239		032 238 255	20 EE FF	JSR &FFEE
923C	@	032 064 152	20 40 98	JSR &9840
923F	)	128 041	80 29	BRA 41 --> &926A
9241		032 175 150	20 AF 96	JSR &96AF
9244		032 235 142	20 EB 8E	JSR &8EEB
9247		240 228	F0 E4	BEQ -28 --> &922D
9249	)	201 041	C9 29	CMP#&29

```
924B       208 221        D0 DD       BNE -35 --> &922A
924D   *   165 042        A5 2A       LDA &2A
924F       229 030        E5 1E       SBC &1E
9251       240 023        F0 17       BEQ 23 --> &926A
9253       170            AA          TAX
9254       176 012        B0 0C       BCS 12 --> &9262
9256       032 146 186    20 92 BA    JSR &BA92
9259       128 003        80 03       BRA 3 --> &925E
925B       032 180 150    20 B4 96    JSR &96B4
925E   *   166 042        A6 2A       LDX &2A
9260       240 008        F0 08       BEQ 8 --> &926A
9262       032 191 189    20 BF BD    JSR &BDBF
9265       128 003        80 03       BRA 3 --> &926A
9267       032 146 186    20 92 BA    JSR &BA92
926A       024            18          CLC
926B       128 008        80 08       BRA 8 --> &9275
926D       198 010        C6 0A       DEC &0A
926F   /   032 047 157    20 2F 9D    JSR &9D2F
9272       032 191 150    20 BF 96    JSR &96BF
9275       164 027        A4 1B       LDY &1B
9277       132 010        84 0A       STY &0A
9279   `   096            60          RTS
927A       166 011        A6 0B       LDX &0B
927C       134 025        86 19       STX &19
927E       166 012        A6 0C       LDX &0C
9280       134 026        86 1A       STX &1A
9282       166 010        A6 0A       LDX &0A
9284       134 027        86 1B       STX &1B
9286   '   201 039        C9 27       CMP#&27
9288       240 221        F0 DD       BEQ -35 --> &9267
928A       201 138        C9 8A       CMP#&8A
928C       240 179        F0 B3       BEQ -77 --> &9241
928E       201 137        C9 89       CMP#&89
9290       240 201        F0 C9       BEQ -55 --> &925B
9292   8   056            38          SEC
```

9293	`	096	60	RTS
9294		000	00	BRK
9295		009	09	EQUB &09
9296		141	BD	xxx Invalid Code
9297	"	034	22	xxx Invalid Code
9298		000	00	EQUB &00
9299		032 224 142	20 E0 8E	JSR &8EE0
929C	z	032 122 146	20 7A 92	JSR &927A
929F		144 242	90 F2	BCC -14 --> &9293
92A1	"	201 034	C9 22	CMP#&22
92A3		208 237	D0 ED	BNE -19 --> &9292
92A5		200	C8	INY
92A6		177 025	B1 19	LDA (&19),Y
92A8		201 013	C9 0D	CMP#&0D
92AA		240 232	F0 E8	BEQ -24 --> &9294
92AC	"	201 034	C9 22	CMP#&22
92AE		208 009	D0 09	BNE 9 --> &92B9
92B0		200	C8	INY
92B1		132 027	84 1B	STY &1B
92B3		177 025	B1 19	LDA (&19),Y
92B5	"	201 034	C9 22	CMP#&22
92B7		208 177	D0 B1	BNE -79 --> &926A
92B9		032 152 189	20 98 BD	JSR &BD98
92BC		128 231	80 E7	BRA -25 --> &92A5
92BE	/	032 047 157	20 2F 9D	JSR &9D2F
92C1		032 188 150	20 BC 96	JSR &96BC
92C4	&	032 038 188	20 26 BC	JSR &BC26
92C7		156 000 006	9C 00 06	STZ &0600
92CA		160 000	A0 00	LDY#&00
92CC	Z	090	5A	PHY
92CD		032 235 142	20 EB 8E	JSR &8EEB
92D0		208 031	D0 1F	BNE 31 --> &92F1
92D2		164 027	A4 1B	LDY &1B
92D4		032 001 153	20 01 99	JSR &9901
92D7	(	240 040	F0 28	BEQ 40 --> &9301

92D9	z	122	7A	PLY
92DA		200	C8	INY
92DB	*	165 042	A5 2A	LDA &2A
92DD		153 000 006	99 00 06	STA &0600,Y
92E0		200	C8	INY
92E1	+	165 043	A5 2B	LDA &2B
92E3		153 000 006	99 00 06	STA &0600,Y
92E6		200	C8	INY
92E7	,	165 044	A5 2C	LDA &2C
92E9		153 000 006	99 00 06	STA &0600,Y
92EC		238 000 006	EE 00 06	INC &0600
92EF		128 219	80 DB	BRA -37 --> &92CC
92F1	z	122	7A	PLY
92F2		198 027	C6 1B	DEC &1B
92F4		032 150 155	20 96 9B	JSR &9B96
92F7		032 230 188	20 E6 BC	JSR &BCE6
92FA		032 004 147	20 04 93	JSR &9304
92FD		216	D8	CLD
92FE	L	076 005 144	4C 05 90	JMP &9005
9301	L	076 140 173	4C 8C AD	JMP &AD8C
9304		173 012 004	AD 0C 04	LDA &040C
9307	J	074	4A	LSR A
9308		173 004 004	AD 04 04	LDA &0404
930B	`	174 096 004	AE 60 04	LDX &0460
930E	d	172 100 004	AC 64 04	LDY &0464
9311	l*	108 042 000	6C 2A 00	JMP (&002A)
9314	Li	076 105 155	4C 69 9B	JMP &9B69
9317		032 030 155	20 1E 9B	JSR &9B1E
931A		144 248	90 F8	BCC -8 --> &9314
931C	&	032 038 188	20 26 BC	JSR &BC26
931F		032 229 140	20 E5 8C	JSR &8CE5
9322		208 240	D0 F0	BNE -16 --> &9314
9324		032 030 155	20 1E 9B	JSR &9B1E
9327		144 235	90 EB	BCC -21 --> &9314
9329		032 166 155	20 A6 9B	JSR &9BA6

932C	*	165 042	A5 2A	LDA &2A
932E	9	133 057	85 39	STA &39
9330	+	165 043	A5 2B	LDA &2B
9332	:	133 058	85 3A	STA &3A
9334		032 230 188	20 E6 BC	JSR &BCE6
9337		032 152 186	20 98 BA	JSR &BA98
933A		032 202 155	20 CA 9B	JSR &9BCA
933D		032 239 190	20 EF BE	JSR &BEEF
9340	9	165 057	A5 39	LDA &39
9342	*	197 042	C5 2A	CMP &2A
9344	:	165 058	A5 3A	LDA &3A
9346	+	229 043	E5 2B	SBC &2B
9348		176 237	B0 ED	BCS -19 --> &9337
934A	L	076 131 143	4C 83 8F	JMP &8F83
934D		169 010	A9 0A	LDA#&0A
934F		032 024 174	20 18 AE	JSR &AE18
9352		032 030 155	20 1E 9B	JSR &9B1E
9355	&	032 038 188	20 26 BC	JSR &BC26
9358		169 010	A9 0A	LDA#&0A
935A		032 024 174	20 18 AE	JSR &AE18
935D		032 229 140	20 E5 8C	JSR &8CE5
9360		208 014	D0 0E	BNE 14 --> &9370
9362		032 030 155	20 1E 9B	JSR &9B1E
9365	+	165 043	A5 2B	LDA &2B
9367	S	208 083	D0 53	BNE 83 --> &93BC
9369	*	165 042	A5 2A	LDA &2A
936B	O	240 079	F0 4F	BEQ 79 --> &93BC
936D	L	076 166 155	4C A6 9B	JMP &9BA6
9370	L	076 176 155	4C B0 9B	JMP &9BB0
9373		165 018	A5 12	LDA &12
9375	;	133 059	85 3B	STA &3B
9377		165 019	A5 13	LDA &13
9379	<	133 060	85 3C	STA &3C
937B		165 024	A5 18	LDA &18
937D	8	133 056	85 38	STA &38

```
937F        160 001       A0 01       LDY#&01
9381  7     132 055       84 37       STY &37
9383  `     096           60          RTS
9384  M     032 077 147   20 4D 93    JSR &934D
9387  9     162 057       A2 39       LDX#&39
9389        032 008 189   20 08 BD    JSR &BD08
938C        032 229 189   20 E5 BD    JSR &BDE5
938F  s     032 115 147   20 73 93    JSR &9373
9392  7     178 055       B2 37       LDA (&37)
9394  0.    048 046       30 2E       BMI 46 --> &93C4
9396  ;     146 059       92 3B       STA (&3B)
9398  7     177 055       B1 37       LDA (&37),Y
939A  ;     145 059       91 3B       STA (&3B),Y
939C  8     056           38          SEC
939D        152           98          TYA
939E  e;    101 059       65 3B       ADC &3B
93A0  ;     133 059       85 3B       STA &3B
93A2        144 002       90 02       BCC 2 --> &93A6
93A4  <     230 060       E6 3C       INC &3C
93A6        197 006       C5 06       CMP &06
93A8  <     165 060       A5 3C       LDA &3C
93AA        229 007       E5 07       SBC &07
93AC        176 005       B0 05       BCS 5 --> &93B3
93AE  z     032 122 148   20 7A 94    JSR &947A
93B1        128 223       80 DF       BRA -33 --> &9392
93B3        000           00          BRK
93B4        000           00          EQUB &00
93B5  s     204 032 115   CC 20 73    CPY &7320
93B8  pa    112 097       70 61       BVS 97 --> &941B
93BA  c     099           63          xxx Invalid Code
93BB  e     101           65          xxx Invalid Code
93BC        000           00          BRK
93BD        000           00          EQUB &00
93BE  S     083           53          xxx Invalid Code
93BF  il    105 108       69 6C       ADC#&6C
```

93C1	ly	108 121	6C 79	xxx Invalid Code
93C3		000	00	EQUB &00
93C4	{	032 123 147	20 7B 93	JSR &937B
93C7	7	178 055	B2 37	LDA (&37)
93C9	0	048 028	30 1C	BMI 28 --> &93E7
93CB	:	165 058	A5 3A	LDA &3A
93CD	7	146 055	92 37	STA (&37)
93CF	9	165 057	A5 39	LDA &39
93D1	7	145 055	91 37	STA (&37),Y
93D3		024	18	CLC
93D4	9	165 057	A5 39	LDA &39
93D6	e*	101 042	65 2A	ADC &2A
93D8	9	133 057	85 39	STA &39
93DA		169 000	A9 00	LDA#&00
93DC	e:	101 058	65 3A	ADC &3A
93DE	)	041 127	29 7F	AND#&7F
93E0	:	133 058	85 3A	STA &3A
93E2	z	032 122 148	20 7A 94	JSR &947A
93E5		128 224	80 E0	BRA -32 --> &93C7
93E7		165 024	A5 18	LDA &18
93E9		133 012	85 0C	STA &0C
93EB	d	100 011	64 0B	STZ &0B
93ED		160 001	A0 01	LDY#&01
93EF		177 011	B1 0B	LDA (&0B),Y
93F1	0g	048 103	30 67	BMI 103 --> &945A
93F3		160 004	A0 04	LDY#&04
93F5	d,	100 044	64 2C	STZ &2C
93F7		177 011	B1 0B	LDA (&0B),Y
93F9	,	166 044	A6 2C	LDX &2C
93FB		208 008	D0 08	BNE 8 --> &9405
93FD		201 141	C9 8D	CMP#&8D
93FF		240 026	F0 1A	BEQ 26 --> &941B
9401		201 244	C9 F4	CMP#&F4
9403		240 013	F0 0D	BEQ 13 --> &9412
9405		200	C8	INY

9406	"	201 034	C9 22	CMP#&22
9408		208 004	D0 04	BNE 4 --> &940E
940A	E,	069 044	45 2C	EOR &2C
940C	,	133 044	85 2C	STA &2C
940E		201 013	C9 0D	CMP#&0D
9410		208 229	D0 E5	BNE -27 --> &93F7
9412		160 003	A0 03	LDY#&03
9414		177 011	B1 0B	LDA (&0B),Y
9416		032 244 155	20 F4 9B	JSR &9BF4
9419		128 210	80 D2	BRA -46 --> &93ED
941B	*	032 042 155	20 2A 9B	JSR &9B2A
941E	s	032 115 147	20 73 93	JSR &9373
9421	7	178 055	B2 37	LDA (&37)
9423	07	048 055	30 37	BMI 55 --> &945C
9425	;	178 059	B2 3B	LDA (&3B)
9427	+	197 043	C5 2B	CMP &2B
9429		208 031	D0 1F	BNE 31 --> &944A
942B	;	177 059	B1 3B	LDA (&3B),Y
942D	*	197 042	C5 2A	CMP &2A
942F		208 025	D0 19	BNE 25 --> &944A
9431	7	177 055	B1 37	LDA (&37),Y
9433	=	133 061	85 3D	STA &3D
9435	7	178 055	B2 37	LDA (&37)
9437		170	AA	TAX
9438		164 010	A4 0A	LDY &0A
943A		136	88	DEY
943B		165 011	A5 0B	LDA &0B
943D	9	133 057	85 39	STA &39
943F		165 012	A5 0C	LDA &0C
9441	:	133 058	85 3A	STA &3A
9443	b	032 098 141	20 62 8D	JSR &8D62
9446		164 010	A4 0A	LDY &0A
9448		128 173	80 AD	BRA -83 --> &93F7
944A		024	18	CLC
944B	z	032 122 148	20 7A 94	JSR &947A

944E	;	165 059	A5 3B	LDA &3B
9450	i	105 002	69 02	ADC#&02
9452	;	133 059	85 3B	STA &3B
9454		144 203	90 CB	BCC -53 --> &9421
9456	<	230 060	E6 3C	INC &3C
9458		128 199	80 C7	BRA -57 --> &9421
945A	Z	128 090	80 5A	BRA 90 --> &94B6
945C		032 207 190	20 CF BE	JSR &BECF
945F	Fa	070 097	46 61	LSR &61
9461	il	105 108	69 6C	ADC#&6C
9463	ed	101 100	65 64	ADC &64
9465	at	032 097 116	20 61 74	JSR &7461
9468		032 177 011	20 B1 0B	JSR &0BB1
946B	+	133 043	85 2B	STA &2B
946D		200	C8	INY
946E		177 011	B1 0B	LDA (&0B),Y
9470	*	133 042	85 2A	STA &2A
9472		032 129 160	20 81 A0	JSR &A081
9475		032 146 186	20 92 BA	JSR &BA92
9478		128 204	80 CC	BRA -52 --> &9446
947A		200	C8	INY
947B	7	177 055	B1 37	LDA (&37),Y
947D		160 001	A0 01	LDY#&01
947F	e7	101 055	65 37	ADC &37
9481	7	133 055	85 37	STA &37
9483		144 003	90 03	BCC 3 --> &9488
9485	8	230 056	E6 38	INC &38
9487		024	18	CLC
9488	`	096	60	RTS
9489	M	032 077 147	20 4D 93	JSR &934D
948C	*	165 042	A5 2A	LDA &2A
948E	H	072	48	PHA
948F		032 230 188	20 E6 BC	JSR &BCE6
9492	&	032 038 188	20 26 BC	JSR &BC26
9495		032 133 160	20 85 A0	JSR &A085

9498	t	032 116 186	20 74 BA	JSR &BA74
949B		032 230 188	20 E6 BC	JSR &BCE6
949E		032 231 141	20 E7 8D	JSR &8DE7
94A1		160 000	A0 00	LDY#&00
94A3		032 008 187	20 08 BB	JSR &BB08
94A6		032 172 187	20 AC BB	JSR &BBAC
94A9	h	104	68	PLA
94AA	H	072	48	PHA
94AB		024	18	CLC
94AC	e*	101 042	65 2A	ADC &2A
94AE	*	133 042	85 2A	STA &2A
94B0		144 224	90 E0	BCC -32 --> &9492
94B2	+	230 043	E6 2B	INC &2B
94B4		016 220	10 DC	BPL -36 --> &9492
94B6	L	076 131 143	4C 83 8F	JMP &8F83
94B9	L	076 005 150	4C 05 96	JMP &9605
94BC		198 010	C6 0A	DEC &0A
94BE		032 174 152	20 AE 98	JSR &98AE
94C1	i	240 105	F0 69	BEQ 105 --> &952C
94C3	g	176 103	B0 67	BCS 103 --> &952C
94C5	C	032 067 188	20 43 BC	JSR &BC43
94C8		032 175 150	20 AF 96	JSR &96AF
94CB		032 239 190	20 EF BE	JSR &BEEF
94CE	-	165 045	A5 2D	LDA &2D
94D0	,	005 044	05 2C	ORA &2C
94D2	X	208 088	D0 58	BNE 88 --> &952C
94D4		024	18	CLC
94D5	*	165 042	A5 2A	LDA &2A
94D7	e	101 002	65 02	ADC &02
94D9		168	A8	TAY
94DA	+	165 043	A5 2B	LDA &2B
94DC	e	101 003	65 03	ADC &03
94DE		170	AA	TAX
94DF		196 004	C4 04	CPY &04
94E1		229 005	E5 05	SBC &05

94E3		176 212	B0 D4	BCS -44 --> &94B9
94E5		165 002	A5 02	LDA &02
94E7	*	133 042	85 2A	STA &2A
94E9		165 003	A5 03	LDA &03
94EB	+	133 043	85 2B	STA &2B
94ED		132 002	84 02	STY &02
94EF		134 003	86 03	STX &03
94F1	@	169 064	A9 40	LDA#&40
94F3	'	133 039	85 27	STA &27
94F5	+	032 043 179	20 2B B3	JSR &B32B
94F8	u	032 117 146	20 75 92	JSR &9275
94FB		032 229 140	20 E5 8C	JSR &8CE5
94FE	4	240 052	F0 34	BEQ 52 --> &9534
9500	L	076 000 144	4C 00 90	JMP &9000
9503	?	162 063	A2 3F	LDX#&3F
9505		032 008 189	20 08 BD	JSR &BD08
9508		162 000	A2 00	LDX#&00
950A		160 000	A0 00	LDY#&00
950C	F@	070 064	46 40	LSR &40
950E	f?	102 063	66 3F	ROR &3F
9510		144 011	90 0B	BCC 11 --> &951D
9512		024	18	CLC
9513		152	98	TYA
9514	e*	101 042	65 2A	ADC &2A
9516		168	A8	TAY
9517		138	8A	TXA
9518	e+	101 043	65 2B	ADC &2B
951A		170	AA	TAX
951B		176 015	B0 0F	BCS 15 --> &952C
951D	*	006 042	06 2A	ASL &2A
951F	&+	038 043	26 2B	ROL &2B
9521	?	165 063	A5 3F	LDA &3F
9523	@	005 064	05 40	ORA &40
9525		208 229	D0 E5	BNE -27 --> &950C
9527	*	132 042	84 2A	STY &2A

```
9529    +    134 043          86 2B          STX &2B
952B    `    096              60             RTS
952C         000              00             BRK
952D         010              0A             EQUB &0A
952E    B    066              42             xxx Invalid Code
952F    ad   097 100 032      61 64 20       ADC (&2064,X)
9532         222              DE             xxx Invalid Code
9533         000              00             EQUB &00
9534         032 224 142      20 E0 8E       JSR &8EE0
9537         152              98             TYA
9538         024              18             CLC
9539    e    101 011          65 0B          ADC &0B
953B         166 012          A6 0C          LDX &0C
953D         144 002          90 02          BCC 2 --> &9541
953F         232              E8             INX
9540         024              18             CLC
9541         233 000          E9 00          SBC#&00
9543    7    133 055          85 37          STA &37
9545         138              8A             TXA
9546         233 000          E9 00          SBC#&00
9548    8    133 056          85 38          STA &38
954A         162 005          A2 05          LDX#&05
954C    ?    134 063          86 3F          STX &3F
954E         166 010          A6 0A          LDX &0A
9550         032 246 154      20 F6 9A       JSR &9AF6
9553         192 001          C0 01          CPY#&01
9555         240 213          F0 D5          BEQ -43 --> &952C
9557    (    201 040          C9 28          CMP#&28
9559         240 021          F0 15          BEQ 21 --> &9570
955B    $    201 036          C9 24          CMP#&24
955D         240 004          F0 04          BEQ 4 --> &9563
955F    %    201 037          C9 25          CMP#&25
9561         208 010          D0 0A          BNE 10 --> &956D
9563    ?    198 063          C6 3F          DEC &3F
9565         200              C8             INY
```

9566		232	E8	INX
9567	7	177 055	B1 37	LDA (&37),Y
9569	(	201 040	C9 28	CMP#&28
956B		240 003	F0 03	BEQ 3 --> &9570
956D	L	076 188 148	4C BC 94	JMP &94BC
9570		200	C8	INY
9571		134 010	86 0A	STX &0A
9573		032 133 128	20 85 80	JSR &8085
9576		208 180	D0 B4	BNE -76 --> &952C
9578	T	032 084 152	20 54 98	JSR &9854
957B		162 001	A2 01	LDX#&01
957D		032 131 152	20 83 98	JSR &9883
9580	?	165 063	A5 3F	LDA &3F
9582	H	072	48	PHA
9583		169 001	A9 01	LDA#&01
9585	H	072	48	PHA
9586		032 024 174	20 18 AE	JSR &AE18
9589	&	032 038 188	20 26 BC	JSR &BC26
958C	o	032 111 146	20 6F 92	JSR &926F
958F	+	165 043	A5 2B	LDA &2B
9591	)	041 192	29 C0	AND#&C0
9593	,	005 044	05 2C	ORA &2C
9595	-	005 045	05 2D	ORA &2D
9597		208 147	D0 93	BNE -109 --> &952C
9599		032 239 190	20 EF BE	JSR &BEEF
959C	z	122	7A	PLY
959D	*	165 042	A5 2A	LDA &2A
959F		145 002	91 02	STA (&02),Y
95A1		200	C8	INY
95A2	+	165 043	A5 2B	LDA &2B
95A4		145 002	91 02	STA (&02),Y
95A6		200	C8	INY
95A7	Z	090	5A	PHY
95A8		032 003 149	20 03 95	JSR &9503
95AB		032 229 140	20 E5 8C	JSR &8CE5

95AE		240 217	F0 D9	BEQ -39 --> &9589
95B0	)	201 041	C9 29	CMP#&29
95B2		208 194	D0 C2	BNE -62 --> &9576
95B4		250	FA	PLX
95B5	h	104	68	PLA
95B6		218	DA	PHX
95B7	?	133 063	85 3F	STA &3F
95B9	d@	100 064	64 40	STZ &40
95BB		032 008 149	20 08 95	JSR &9508
95BE	h	104	68	PLA
95BF	H	072	48	PHA
95C0	e*	101 042	65 2A	ADC &2A
95C2	*	133 042	85 2A	STA &2A
95C4		144 002	90 02	BCC 2 --> &95C8
95C6	+	230 043	E6 2B	INC &2B
95C8		165 003	A5 03	LDA &03
95CA	8	133 056	85 38	STA &38
95CC		165 002	A5 02	LDA &02
95CE	7	133 055	85 37	STA &37
95D0		024	18	CLC
95D1	e*	101 042	65 2A	ADC &2A
95D3		168	A8	TAY
95D4	+	165 043	A5 2B	LDA &2B
95D6	e	101 003	65 03	ADC &03
95D8	+	176 043	B0 2B	BCS 43 --> &9605
95DA		170	AA	TAX
95DB		196 004	C4 04	CPY &04
95DD		229 005	E5 05	SBC &05
95DF	$	176 036	B0 24	BCS 36 --> &9605
95E1		132 002	84 02	STY &02
95E3		134 003	86 03	STX &03
95E5	h	104	68	PLA
95E6	7	146 055	92 37	STA (&37)
95E8	e7	101 055	65 37	ADC &37
95EA		168	A8	TAY

```
95EB        169 000        A9 00        LDA#&00
95ED  d7    100 055        64 37        STZ &37
95EF        144 002        90 02        BCC 2 --> &95F3
95F1  8     230 056        E6 38        INC &38
95F3  7     145 055        91 37        STA (&37),Y
95F5        200            C8           INY
95F6        208 002        D0 02        BNE 2 --> &95FA
95F8  8     230 056        E6 38        INC &38
95FA        196 002        C4 02        CPY &02
95FC        208 245        D0 F5        BNE -11 --> &95F3
95FE  8     228 056        E4 38        CPX &38
9600        208 241        D0 F1        BNE -15 --> &95F3
9602  L     076 251 148    4C FB 94     JMP &94FB
9605        000            00           BRK
9606        011            0B           EQUB &0B
9607  s     222 032 115    DE 20 73     DEC &7320,X
960A  pa    112 097        70 61        BVS 97 --> &966D
960C  c     099            63           xxx Invalid Code
960D  e     101            65           xxx Invalid Code
960E        000            00           EQUB &00
960F        032 185 150    20 B9 96     JSR &96B9
9612  *     165 042        A5 2A        LDA &2A
9614        133 006        85 06        STA &06
9616        133 004        85 04        STA &04
9618  +     165 043        A5 2B        LDA &2B
961A        133 007        85 07        STA &07
961C        133 005        85 05        STA &05
961E        128 027        80 1B        BRA 27 --> &963B
9620        032 185 150    20 B9 96     JSR &96B9
9623  *     165 042        A5 2A        LDA &2A
9625        133 000        85 00        STA &00
9627        133 002        85 02        STA &02
9629  +     165 043        A5 2B        LDA &2B
962B        133 001        85 01        STA &01
962D        133 003        85 03        STA &03
```

```
962F        032 187 187   20 BB BB    JSR &BBBB
9632        128 007       80 07       BRA 7 --> &963B
9634        032 185 150   20 B9 96    JSR &96B9
9637    +   165 043       A5 2B       LDA &2B
9639        133 024       85 18       STA &18
963B    L   076 005 144   4C 05 90    JMP &9005
963E        032 166 155   20 A6 9B    JSR &9BA6
9641        032 172 187   20 AC BB    JSR &BBAC
9644        128 245       80 F5       BRA -11 --> &963B
9646        032 030 155   20 1E 9B    JSR &9B1E
9649        176 011       B0 0B       BCS 11 --> &9656
964B        201 238       C9 EE       CMP#&EE
964D        240 024       F0 18       BEQ 24 --> &9667
964F        201 135       C9 87       CMP#&87
9651        240 029       F0 1D       BEQ 29 --> &9670
9653    o   032 111 146   20 6F 92    JSR &926F
9656        032 166 155   20 A6 9B    JSR &9BA6
9659    *   165 042       A5 2A       LDA &2A
965B    !   133 033       85 21       STA &21
965D    +   165 043       A5 2B       LDA &2B
965F    "   133 034       85 22       STA &22
9661        169 255       A9 FF       LDA#&FF
9663        133 032       85 20       STA &20
9665        128 212       80 D4       BRA -44 --> &963B
9667        230 010       E6 0A       INC &0A
9669        032 166 155   20 A6 9B    JSR &9BA6
966C        169 255       A9 FF       LDA#&FF
966E        208 239       D0 EF       BNE -17 --> &965F
9670        230 010       E6 0A       INC &0A
9672        032 166 155   20 A6 9B    JSR &9BA6
9675        169 000       A9 00       LDA#&00
9677        128 234       80 EA       BRA -22 --> &9663
9679        200           C8          INY
967A        177 011       B1 0B       LDA (&0B),Y
967C    $   201 036       C9 24       CMP#&24
```

967E		240 014	F0 0E	BEQ 14 --> &968E
9680		032 185 150	20 B9 96	JSR &96B9
9683	d.	100 046	64 2E	STZ &2E
9685	*	162 042	A2 2A	LDX#&2A
9687		160 000	A0 00	LDY#&00
9689		169 002	A9 02	LDA#&02
968B	L	076 018 179	4C 12 B3	JMP &B312
968E		230 010	E6 0A	INC &0A
9690	F	032 070 155	20 46 9B	JSR &9B46
9693	'	165 039	A5 27	LDA &27
9695	@	208 064	D0 40	BNE 64 --> &96D7
9697		169 015	A9 0F	LDA#&0F
9699	6	164 054	A4 36	LDY &36
969B		140 255 005	8C FF 05	STY &05FF
969E		162 255	A2 FF	LDX#&FF
96A0		160 005	A0 05	LDY#&05
96A2		128 231	80 E7	BRA -25 --> &968B
96A4	Q	032 081 188	20 51 BC	JSR &BC51
96A7		032 172 173	20 AC AD	JSR &ADAC
96AA		128 019	80 13	BRA 19 --> &96BF
96AC		032 241 142	20 F1 8E	JSR &8EF1
96AF	;	032 059 157	20 3B 9D	JSR &9D3B
96B2		128 011	80 0B	BRA 11 --> &96BF
96B4	6	032 054 173	20 36 AD	JSR &AD36
96B7		128 006	80 06	BRA 6 --> &96BF
96B9	F	032 070 155	20 46 9B	JSR &9B46
96BC	'	165 039	A5 27	LDA &27
96BE		168	A8	TAY
96BF		240 022	F0 16	BEQ 22 --> &96D7
96C1		016 019	10 13	BPL 19 --> &96D6
96C3	B	032 066 130	20 42 82	JSR &8242
96C6	1	165 049	A5 31	LDA &31
96C8	-	133 045	85 2D	STA &2D
96CA	2	165 050	A5 32	LDA &32
96CC	,	133 044	85 2C	STA &2C

96CE	3	165 051	A5 33	LDA &33
96D0	+	133 043	85 2B	STA &2B
96D2	4	165 052	A5 34	LDA &34
96D4	*	133 042	85 2A	STA &2A
96D6	`	096	60	RTS
96D7	L	076 146 144	4C 92 90	JMP &9092
96DA	6	032 054 173	20 36 AD	JSR &AD36
96DD		240 248	F0 F8	BEQ -8 --> &96D7
96DF	0	048 245	30 F5	BMI -11 --> &96D6
96E1	L	076 133 129	4C 85 81	JMP &8185
96E4		165 011	A5 0B	LDA &0B
96E6		133 025	85 19	STA &19
96E8		165 012	A5 0C	LDA &0C
96EA		133 026	85 1A	STA &1A
96EC		165 010	A5 0A	LDA &0A
96EE		133 027	85 1B	STA &1B
96F0		169 242	A9 F2	LDA#&F2
96F2		032 025 176	20 19 B0	JSR &B019
96F5		032 150 155	20 96 9B	JSR &9B96
96F8	L	076 005 144	4C 05 90	JMP &9005
96FB		160 003	A0 03	LDY#&03
96FD		169 000	A9 00	LDA#&00
96FF	*	145 042	91 2A	STA (&2A),Y
9701		240 028	F0 1C	BEQ 28 --> &971F
9703		186	BA	TSX
9704		224 252	E0 FC	CPX#&FC
9706	*	176 042	B0 2A	BCS 42 --> &9732
9708		032 174 152	20 AE 98	JSR &98AE
970B	"	240 034	F0 22	BEQ 34 --> &972F
970D		032 129 177	20 81 B1	JSR &B181
9710	,	164 044	A4 2C	LDY &2C
9712	0	048 231	30 E7	BMI -25 --> &96FB
9714	C	032 067 188	20 43 BC	JSR &BC43
9717		032 232 171	20 E8 AB	JSR &ABE8
971A	'	133 039	85 27	STA &27

971C	+	032 043 179	20 2B B3	JSR &B32B
971F		186	BA	TSX
9720		254 006 001	FE 06 01	INC &0106,X
9723		164 027	A4 1B	LDY &1B
9725		132 010	84 0A	STY &0A
9727		032 229 140	20 E5 8C	JSR &8CE5
972A		240 215	F0 D7	BEQ -41 --> &9703
972C	L	076 000 144	4C 00 90	JMP &9000
972F	L	076 002 144	4C 02 90	JMP &9002
9732		000	00	BRK
9733		012	0C	EQUB &0C
9734	No	078 111	4E 6F	xxx Invalid Code
9736	t	116 032	74 20	STZ &20,X
9738		234	EA	NOP
9739		000	00	BRK
973A		025	19	EQUB &19
973B	Ba	066 097	42 61	xxx Invalid Code
973D	d	100 032	64 20	STZ &20
973F		235	EB	xxx Invalid Code
9740		000	00	EQUB &00
9741	o	032 111 146	20 6F 92	JSR &926F
9744	*	165 042	A5 2A	LDA &2A
9746	H	072	48	PHA
9747		032 172 150	20 AC 96	JSR &96AC
974A		032 150 155	20 96 9B	JSR &9B96
974D		169 018	A9 12	LDA#&12
974F		032 238 255	20 EE FF	JSR &FFEE
9752	h	104	68	PLA
9753	F	128 070	80 46	BRA 70 --> &979B
9755	o	032 111 146	20 6F 92	JSR &926F
9758		032 166 155	20 A6 9B	JSR &9BA6
975B		169 017	A9 11	LDA#&11
975D	<	128 060	80 3C	BRA 60 --> &979B
975F	o	032 111 146	20 6F 92	JSR &926F
9762		032 166 155	20 A6 9B	JSR &9BA6

9765	K	032 075 190	20 4B BE	JSR &BE4B
9768		232	E8	INX
9769	,	208 044	D0 2C	BNE 44 --> &9797
976B		200	C8	INY
976C	)	208 041	D0 29	BNE 41 --> &9797
976E		165 004	A5 04	LDA &04
9770		197 006	C5 06	CMP &06
9772		208 197	D0 C5	BNE -59 --> &9739
9774		165 005	A5 05	LDA &05
9776		197 007	C5 07	CMP &07
9778		208 191	D0 BF	BNE -65 --> &9739
977A	*	166 042	A6 2A	LDX &2A
977C		169 133	A9 85	LDA#&85
977E		032 244 255	20 F4 FF	JSR &FFF4
9781		228 002	E4 02	CPX &02
9783		152	98	TYA
9784		229 003	E5 03	SBC &03
9786		144 177	90 B1	BCC -79 --> &9739
9788		228 018	E4 12	CPX &12
978A		152	98	TYA
978B		229 019	E5 13	SBC &13
978D		144 170	90 AA	BCC -86 --> &9739
978F		134 006	86 06	STX &06
9791		134 004	86 04	STX &04
9793		132 007	84 07	STY &07
9795		132 005	84 05	STY &05
9797	d	100 030	64 1E	STZ &1E
9799		169 022	A9 16	LDA#&16
979B		032 238 255	20 EE FF	JSR &FFEE
979E	*	165 042	A5 2A	LDA &2A
97A0	L	128 076	80 4C	BRA 76 --> &97EE
97A2		169 004	A9 04	LDA#&04
97A4		128 002	80 02	BRA 2 --> &97A8
97A6		169 005	A9 05	LDA#&05
97A8	H	072	48	PHA

97A9	/	032 047 157	20 2F 9D	JSR &9D2F
97AC		032 188 150	20 BC 96	JSR &96BC
97AF		128 009	80 09	BRA 9 --> &97BA
97B1	o	032 111 146	20 6F 92	JSR &926F
97B4	*	165 042	A5 2A	LDA &2A
97B6	H	072	48	PHA
97B7		032 172 150	20 AC 96	JSR &96AC
97BA	&	032 038 188	20 26 BC	JSR &BC26
97BD		032 172 150	20 AC 96	JSR &96AC
97C0		032 150 155	20 96 9B	JSR &9B96
97C3		169 025	A9 19	LDA#&19
97C5		032 238 255	20 EE FF	JSR &FFEE
97C8	h	104	68	PLA
97C9		032 238 255	20 EE FF	JSR &FFEE
97CC		032 006 189	20 06 BD	JSR &BD06
97CF	7	165 055	A5 37	LDA &37
97D1		032 238 255	20 EE FF	JSR &FFEE
97D4	8	165 056	A5 38	LDA &38
97D6		032 238 255	20 EE FF	JSR &FFEE
97D9	@	032 064 152	20 40 98	JSR &9840
97DC	+	165 043	A5 2B	LDA &2B
97DE		128 014	80 0E	BRA 14 --> &97EE
97E0		032 166 155	20 A6 9B	JSR &9BA6
97E3		169 016	A9 10	LDA#&10
97E5		128 007	80 07	BRA 7 --> &97EE
97E7		032 166 155	20 A6 9B	JSR &9BA6
97EA	d	100 030	64 1E	STZ &1E
97EC		169 012	A9 0C	LDA#&0C
97EE		032 238 255	20 EE FF	JSR &FFEE
97F1	L	076 005 144	4C 05 90	JMP &9005
97F4		032 166 155	20 A6 9B	JSR &9BA6
97F7		032 146 186	20 92 BA	JSR &BA92
97FA		160 001	A0 01	LDY#&01
97FC		177 253	B1 FD	LDA (&FD),Y
97FE		240 241	F0 F1	BEQ -15 --> &97F1

```
9800    7    032 055 189    20 37 BD    JSR &BD37
9803         200            C8          INY
9804         208 246        D0 F6       BNE -10 --> &97FC
9806         128 233        80 E9       BRA -23 --> &97F1
9808    +    165 043        A5 2B       LDA &2B
980A         032 238 255    20 EE FF    JSR &FFEE
980D         032 224 142    20 E0 8E    JSR &8EE0
9810    :    201 058        C9 3A       CMP#&3A
9812    )    240 041        F0 29       BEQ 41 --> &983D
9814         201 013        C9 0D       CMP#&0D
9816    %    240 037        F0 25       BEQ 37 --> &983D
9818         201 139        C9 8B       CMP#&8B
981A    !    240 033        F0 21       BEQ 33 --> &983D
981C         198 010        C6 0A       DEC &0A
981E    o    032 111 146    20 6F 92    JSR &926F
9821    @    032 064 152    20 40 98    JSR &9840
9824         032 229 140    20 E5 8C    JSR &8CE5
9827         240 228        F0 E4       BEQ -28 --> &980D
9829    ;    201 059        C9 3B       CMP#&3B
982B         240 219        F0 DB       BEQ -37 --> &9808
982D    |    201 124        C9 7C       CMP#&7C
982F         208 223        D0 DF       BNE -33 --> &9810
9831         169 000        A9 00       LDA#&00
9833         160 009        A0 09       LDY#&09
9835         032 238 255    20 EE FF    JSR &FFEE
9838         136            88          DEY
9839         208 250        D0 FA       BNE -6 --> &9835
983B         128 208        80 D0       BRA -48 --> &980D
983D    L    076 000 144    4C 00 90    JMP &9000
9840    *    165 042        A5 2A       LDA &2A
9842    l    108 014 002    6C 0E 02    JMP (&020E)
9845         160 001        A0 01       LDY#&01
9847    7    177 055        B1 37       LDA (&37),Y
9849         170            AA          TAX
984A         169 246        A9 F6       LDA#&F6
```

984C		224 242	E0 F2	CPX#&F2
984E		240 009	F0 09	BEQ 9 --> &9859
9850		169 248	A9 F8	LDA#&F8
9852		128 005	80 05	BRA 5 --> &9859
9854		160 001	A0 01	LDY#&01
9856	7	177 055	B1 37	LDA (&37),Y
9858		010	0A	ASL A
9859		162 004	A2 04	LDX#&04
985B	:	133 058	85 3A	STA &3A
985D	;	134 059	86 3B	STX &3B
985F	:	177 058	B1 3A	LDA (&3A),Y
9861		240 005	F0 05	BEQ 5 --> &9868
9863		170	AA	TAX
9864	:	178 058	B2 3A	LDA (&3A)
9866		128 243	80 F3	BRA -13 --> &985B
9868		165 003	A5 03	LDA &03
986A	:	145 058	91 3A	STA (&3A),Y
986C		165 002	A5 02	LDA &02
986E	:	146 058	92 3A	STA (&3A)
9870		169 000	A9 00	LDA#&00
9872		145 002	91 02	STA (&02),Y
9874		200	C8	INY
9875	9	196 057	C4 39	CPY &39
9877	1	240 049	F0 31	BEQ 49 --> &98AA
9879	7	177 055	B1 37	LDA (&37),Y
987B		145 002	91 02	STA (&02),Y
987D		200	C8	INY
987E	9	196 057	C4 39	CPY &39
9880		208 247	D0 F7	BNE -9 --> &9879
9882	`	096	60	RTS
9883		169 000	A9 00	LDA#&00
9885		145 002	91 02	STA (&02),Y
9887		200	C8	INY
9888		202	CA	DEX
9889		208 250	D0 FA	BNE -6 --> &9885

988B		024	18	CLC
988C		152	98	TYA
988D	e	101 002	65 02	ADC &02
988F		144 002	90 02	BCC 2 --> &9893
9891		230 003	E6 03	INC &03
9893		164 003	A4 03	LDY &03
9895		196 005	C4 05	CPY &05
9897		144 015	90 0F	BCC 15 --> &98A8
9899		208 004	D0 04	BNE 4 --> &989F
989B		197 004	C5 04	CMP &04
989D		144 009	90 09	BCC 9 --> &98A8
989F		169 000	A9 00	LDA#&00
98A1		160 001	A0 01	LDY#&01
98A3	:	145 058	91 3A	STA (&3A),Y
98A5	L	076 161 144	4C A1 90	JMP &90A1
98A8		133 002	85 02	STA &02
98AA	`	096	60	RTS
98AB		032 131 152	20 83 98	JSR &9883
98AE		032 245 152	20 F5 98	JSR &98F5
98B1		208 029	D0 1D	BNE 29 --> &98D0
98B3		176 027	B0 1B	BCS 27 --> &98D0
98B5	T	032 084 152	20 54 98	JSR &9854
98B8		162 005	A2 05	LDX#&05
98BA	,	228 044	E4 2C	CPX &2C
98BC		208 237	D0 ED	BNE -19 --> &98AB
98BE		232	E8	INX
98BF		128 234	80 EA	BRA -22 --> &98AB
98C1	!	201 033	C9 21	CMP#&21
98C3		240 012	F0 0C	BEQ 12 --> &98D1
98C5	$	201 036	C9 24	CMP#&24
98C7		240 019	F0 13	BEQ 19 --> &98DC
98C9	I?	073 063	49 3F	EOR#&3F
98CB		240 006	F0 06	BEQ 6 --> &98D3
98CD		169 000	A9 00	LDA#&00
98CF	8	056	38	SEC

98D0	`	096	60	RTS
98D1		169 004	A9 04	LDA#&04
98D3	H	072	48	PHA
98D4		230 027	E6 1B	INC &1B
98D6		032 180 150	20 B4 96	JSR &96B4
98D9	L	076 206 153	4C CE 99	JMP &99CE
98DC		230 027	E6 1B	INC &1B
98DE		032 180 150	20 B4 96	JSR &96B4
98E1	+	165 043	A5 2B	LDA &2B
98E3		240 006	F0 06	BEQ 6 --> &98EB
98E5		169 128	A9 80	LDA#&80
98E7	,	133 044	85 2C	STA &2C
98E9	8	056	38	SEC
98EA	`	096	60	RTS
98EB		000	00	BRK
98EC		008	08	EQUB &08
98ED	$	036 032	24 20	BIT &20
98EF	ra	114 097	72 61	ADC (&61)
98F1	nge	110 103 101	6E 67 65	ROR &6567
98F4		000	00	EQUB &00
98F5		165 011	A5 0B	LDA &0B
98F7		133 025	85 19	STA &19
98F9		165 012	A5 0C	LDA &0C
98FB		133 026	85 1A	STA &1A
98FD		164 010	A4 0A	LDY &0A
98FF		136	88	DEY
9900		200	C8	INY
9901		132 027	84 1B	STY &1B
9903		177 025	B1 19	LDA (&19),Y
9905		201 032	C9 20	CMP#&20
9907		240 247	F0 F7	BEQ -9 --> &9900
9909	@	201 064	C9 40	CMP#&40
990B		144 180	90 B4	BCC -76 --> &98C1
990D	[	201 091	C9 5B	CMP#&5B
990F		176 026	B0 1A	BCS 26 --> &992B

```
9911        010         0A          ASL A
9912        010         0A          ASL A
9913    *   133 042     85 2A       STA &2A
9915        200         C8          INY
9916        177 025     B1 19       LDA (&19),Y
9918    %   201 037     C9 25       CMP#&25
991A        208 015     D0 0F       BNE 15 --> &992B
991C        169 004     A9 04       LDA#&04
991E    +   133 043     85 2B       STA &2B
9920        162 004     A2 04       LDX#&04
9922    ,   134 044     86 2C       STX &2C
9924        200         C8          INY
9925        177 025     B1 19       LDA (&19),Y
9927    (   201 040     C9 28       CMP#&28
9929    m   208 109     D0 6D       BNE 109 --> &9998
992B        162 005     A2 05       LDX#&05
992D    ,   134 044     86 2C       STX &2C
992F        024         18          CLC
9930        164 026     A4 1A       LDY &1A
9932        165 027     A5 1B       LDA &1B
9934        170         AA          TAX
9935        208 008     D0 08       BNE 8 --> &993F
9937    :   058         3A          DEC A
9938    e   101 025     65 19       ADC &19
993A        176 009     B0 09       BCS 9 --> &9945
993C        136         88          DEY
993D        128 006     80 06       BRA 6 --> &9945
993F    :   058         3A          DEC A
9940    e   101 025     65 19       ADC &19
9942        144 001     90 01       BCC 1 --> &9945
9944        200         C8          INY
9945    7   133 055     85 37       STA &37
9947    8   132 056     84 38       STY &38
9949        160 001     A0 01       LDY#&01
994B    7   177 055     B1 37       LDA (&37),Y
```

994D	A	201 065	C9 41	CMP#&41
994F		176 026	B0 1A	BCS 26 --> &996B
9951	0	201 048	C9 30	CMP#&30
9953	"	144 034	90 22	BCC 34 --> &9977
9955	:	201 058	C9 3A	CMP#&3A
9957		176 030	B0 1E	BCS 30 --> &9977
9959		232	E8	INX
995A		200	C8	INY
995B	7	177 055	B1 37	LDA (&37),Y
995D	A	201 065	C9 41	CMP#&41
995F		176 010	B0 0A	BCS 10 --> &996B
9961	0	201 048	C9 30	CMP#&30
9963		144 018	90 12	BCC 18 --> &9977
9965	:	201 058	C9 3A	CMP#&3A
9967		144 240	90 F0	BCC -16 --> &9959
9969		128 012	80 0C	BRA 12 --> &9977
996B	[	201 091	C9 5B	CMP#&5B
996D		144 234	90 EA	BCC -22 --> &9959
996F	_	201 095	C9 5F	CMP#&5F
9971		144 004	90 04	BCC 4 --> &9977
9973	{	201 123	C9 7B	CMP#&7B
9975		144 226	90 E2	BCC -30 --> &9959
9977		192 001	C0 01	CPY#&01
9979	+	240 043	F0 2B	BEQ 43 --> &99A6
997B	$	201 036	C9 24	CMP#&24
997D	[	240 091	F0 5B	BEQ 91 --> &99DA
997F	%	201 037	C9 25	CMP#&25
9981		208 006	D0 06	BNE 6 --> &9989
9983	,	198 044	C6 2C	DEC &2C
9985		232	E8	INX
9986		200	C8	INY
9987	7	177 055	B1 37	LDA (&37),Y
9989	(	201 040	C9 28	CMP#&28
998B	H	240 072	F0 48	BEQ 72 --> &99D5
998D		032 133 128	20 85 80	JSR &8085

9990		240 024	F0 18	BEQ 24 --> &99AA
9992		134 027	86 1B	STX &1B
9994		164 027	A4 1B	LDY &1B
9996		177 025	B1 19	LDA (&19),Y
9998	!	201 033	C9 21	CMP#&21
999A		240 018	F0 12	BEQ 18 --> &99AE
999C	I?	073 063	49 3F	EOR#&3F
999E		240 016	F0 10	BEQ 16 --> &99B0
99A0		024	18	CLC
99A1		132 027	84 1B	STY &1B
99A3		169 255	A9 FF	LDA#&FF
99A5	`	096	60	RTS
99A6		169 000	A9 00	LDA#&00
99A8	8	056	38	SEC
99A9	`	096	60	RTS
99AA		169 000	A9 00	LDA#&00
99AC		024	18	CLC
99AD	`	096	60	RTS
99AE		169 004	A9 04	LDA#&04
99B0	H	072	48	PHA
99B1		200	C8	INY
99B2		132 027	84 1B	STY &1B
99B4		032 160 177	20 A0 B1	JSR &B1A0
99B7		032 191 150	20 BF 96	JSR &96BF
99BA	+	165 043	A5 2B	LDA &2B
99BC	H	072	48	PHA
99BD	*	165 042	A5 2A	LDA &2A
99BF	H	072	48	PHA
99C0		032 180 150	20 B4 96	JSR &96B4
99C3		024	18	CLC
99C4	h	104	68	PLA
99C5	e*	101 042	65 2A	ADC &2A
99C7	*	133 042	85 2A	STA &2A
99C9	h	104	68	PLA
99CA	e+	101 043	65 2B	ADC &2B

```
99CC  +   133 043      85 2B     STA &2B
99CE  h   104          68        PLA
99CF  ,   133 044      85 2C     STA &2C
99D1      024          18        CLC
99D2      169 255      A9 FF     LDA#&FF
99D4  `   096          60        RTS
99D5      032 254 153  20 FE 99  JSR &99FE
99D8      128 186      80 BA     BRA -70 --> &9994
99DA  ,   198 044      C6 2C     DEC &2C
99DC      232          E8        INX
99DD      200          C8        INY
99DE  7   177 055      B1 37     LDA (&37),Y
99E0  (   201 040      C9 28     CMP#&28
99E2      240 013      F0 0D     BEQ 13 --> &99F1
99E4      032 133 128  20 85 80  JSR &8085
99E7      240 193      F0 C1     BEQ -63 --> &99AA
99E9      134 027      86 1B     STX &1B
99EB      169 129      A9 81     LDA#&81
99ED  ,   133 044      85 2C     STA &2C
99EF  8   056          38        SEC
99F0  `   096          60        RTS
99F1      032 254 153  20 FE 99  JSR &99FE
99F4      128 245      80 F5     BRA -11 --> &99EB
99F6      000          00        BRK
99F7      014          0E        EQUB &0E
99F8  Ar  065 114      41 72     xxx Invalid Code
99FA  ra  114 097      72 61     ADC (&61)
99FC  y   121          79        xxx Invalid Code
99FD      000          00        EQUB &00
99FE      232          E8        INX
99FF      200          C8        INY
9A00      032 133 128  20 85 80  JSR &8085
9A03      240 241      F0 F1     BEQ -15 --> &99F6
9A05      134 027      86 1B     STX &1B
9A07  ,   165 044      A5 2C     LDA &2C
```

9A09	H	072	48	PHA
9A0A	*	165 042	A5 2A	LDA &2A
9A0C	H	072	48	PHA
9A0D	+	165 043	A5 2B	LDA &2B
9A0F	H	072	48	PHA
9A10	*	178 042	B2 2A	LDA (&2A)
9A12		201 004	C9 04	CMP#&04
9A14	o	144 111	90 6F	BCC 111 --> &9A85
9A16		032 232 171	20 E8 AB	JSR &ABE8
9A19		169 001	A9 01	LDA#&01
9A1B	-	133 045	85 2D	STA &2D
9A1D	&	032 038 188	20 26 BC	JSR &BC26
9A20		032 175 150	20 AF 96	JSR &96AF
9A23		230 027	E6 1B	INC &1B
9A25	,	224 044	E0 2C	CPX#&2C
9A27		208 205	D0 CD	BNE -51 --> &99F6
9A29	9	162 057	A2 39	LDX#&39
9A2B		032 008 189	20 08 BD	JSR &BD08
9A2E	<	164 060	A4 3C	LDY &3C
9A30	h	104	68	PLA
9A31	8	133 056	85 38	STA &38
9A33	h	104	68	PLA
9A34	7	133 055	85 37	STA &37
9A36	H	072	48	PHA
9A37	8	165 056	A5 38	LDA &38
9A39	H	072	48	PHA
9A3A		032 211 154	20 D3 9A	JSR &9AD3
9A3D	-	132 045	84 2D	STY &2D
9A3F	7	177 055	B1 37	LDA (&37),Y
9A41	?	133 063	85 3F	STA &3F
9A43		200	C8	INY
9A44	7	177 055	B1 37	LDA (&37),Y
9A46	@	133 064	85 40	STA &40
9A48	*	165 042	A5 2A	LDA &2A
9A4A	e9	101 057	65 39	ADC &39

9A4C	*	133 042	85 2A	STA &2A
9A4E	+	165 043	A5 2B	LDA &2B
9A50	e:	101 058	65 3A	ADC &3A
9A52	+	133 043	85 2B	STA &2B
9A54		032 008 149	20 08 95	JSR &9508
9A57	8	056	38	SEC
9A58	7	178 055	B2 37	LDA (&37)
9A5A	-	229 045	E5 2D	SBC &2D
9A5C		201 003	C9 03	CMP#&03
9A5E		176 189	B0 BD	BCS -67 --> &9A1D
9A60	&	032 038 188	20 26 BC	JSR &BC26
9A63		032 167 150	20 A7 96	JSR &96A7
9A66	h	104	68	PLA
9A67	8	133 056	85 38	STA &38
9A69	h	104	68	PLA
9A6A	7	133 055	85 37	STA &37
9A6C	9	162 057	A2 39	LDX#&39
9A6E		032 008 189	20 08 BD	JSR &BD08
9A71	<	164 060	A4 3C	LDY &3C
9A73		032 211 154	20 D3 9A	JSR &9AD3
9A76		024	18	CLC
9A77	9	165 057	A5 39	LDA &39
9A79	e*	101 042	65 2A	ADC &2A
9A7B	*	133 042	85 2A	STA &2A
9A7D	:	165 058	A5 3A	LDA &3A
9A7F	e+	101 043	65 2B	ADC &2B
9A81	+	133 043	85 2B	STA &2B
9A83		144 017	90 11	BCC 17 --> &9A96
9A85		032 172 173	20 AC AD	JSR &ADAC
9A88		032 191 150	20 BF 96	JSR &96BF
9A8B	h	104	68	PLA
9A8C	8	133 056	85 38	STA &38
9A8E	h	104	68	PLA
9A8F	7	133 055	85 37	STA &37
9A91		160 001	A0 01	LDY#&01

9A93		032 211 154	20 D3 9A	JSR &9AD3
9A96	h	104	68	PLA
9A97	,	133 044	85 2C	STA &2C
9A99		201 005	C9 05	CMP#&05
9A9B		208 023	D0 17	BNE 23 --> &9AB4
9A9D	+	166 043	A6 2B	LDX &2B
9A9F	*	165 042	A5 2A	LDA &2A
9AA1	*	006 042	06 2A	ASL &2A
9AA3	&+	038 043	26 2B	ROL &2B
9AA5	*	006 042	06 2A	ASL &2A
9AA7	&+	038 043	26 2B	ROL &2B
9AA9	e*	101 042	65 2A	ADC &2A
9AAB	*	133 042	85 2A	STA &2A
9AAD		138	8A	TXA
9AAE	e+	101 043	65 2B	ADC &2B
9AB0	+	133 043	85 2B	STA &2B
9AB2		128 008	80 08	BRA 8 --> &9ABC
9AB4	*	006 042	06 2A	ASL &2A
9AB6	&+	038 043	26 2B	ROL &2B
9AB8	*	006 042	06 2A	ASL &2A
9ABA	&+	038 043	26 2B	ROL &2B
9ABC		152	98	TYA
9ABD	e*	101 042	65 2A	ADC &2A
9ABF	*	133 042	85 2A	STA &2A
9AC1		144 003	90 03	BCC 3 --> &9AC6
9AC3	+	230 043	E6 2B	INC &2B
9AC5		024	18	CLC
9AC6	7	165 055	A5 37	LDA &37
9AC8	e*	101 042	65 2A	ADC &2A
9ACA	*	133 042	85 2A	STA &2A
9ACC	8	165 056	A5 38	LDA &38
9ACE	e+	101 043	65 2B	ADC &2B
9AD0	+	133 043	85 2B	STA &2B
9AD2	`	096	60	RTS
9AD3	+	165 043	A5 2B	LDA &2B

```
9AD5    )    041 192     29 C0     AND#&C0
9AD7    ,    005 044     05 2C     ORA &2C
9AD9    -    005 045     05 2D     ORA &2D
9ADB         208 013     D0 0D     BNE 13 --> &9AEA
9ADD    *    165 042     A5 2A     LDA &2A
9ADF    7    209 055     D1 37     CMP (&37),Y
9AE1         200         C8        INY
9AE2    +    165 043     A5 2B     LDA &2B
9AE4    7    241 055     F1 37     SBC (&37),Y
9AE6         176 002     B0 02     BCS 2 --> &9AEA
9AE8         200         C8        INY
9AE9    `    096         60        RTS
9AEA         000         00        BRK
9AEB         015         0F        EQUB &0F
9AEC    S    083         53        xxx Invalid Code
9AED    ub   117 098     75 62     ADC &62,X
9AEF    s    115         73        xxx Invalid Code
9AF0    c    099         63        xxx Invalid Code
9AF1    ri   114 105     72 69     ADC (&69)
9AF3    pt   112 116     70 74     BVS 116 --> &9B69
9AF5         000         00        EQUB &00
9AF6         160 001     A0 01     LDY#&01
9AF8    7    177 055     B1 37     LDA (&37),Y
9AFA    0    201 048     C9 30     CMP#&30
9AFC         144 024     90 18     BCC 24 --> &9B16
9AFE    @    201 064     C9 40     CMP#&40
9B00         176 012     B0 0C     BCS 12 --> &9B0E
9B02    :    201 058     C9 3A     CMP#&3A
9B04         176 016     B0 10     BCS 16 --> &9B16
9B06         192 001     C0 01     CPY#&01
9B08         240 012     F0 0C     BEQ 12 --> &9B16
9B0A         232         E8        INX
9B0B         200         C8        INY
9B0C         208 234     D0 EA     BNE -22 --> &9AF8
9B0E    _    201 095     C9 5F     CMP#&5F
```

9B10		176 005	B0 05	BCS 5 --> &9B17
9B12	[	201 091	C9 5B	CMP#&5B
9B14		144 244	90 F4	BCC -12 --> &9B0A
9B16	`	096	60	RTS
9B17	{	201 123	C9 7B	CMP#&7B
9B19		144 239	90 EF	BCC -17 --> &9B0A
9B1B	`	096	60	RTS
9B1C		230 010	E6 0A	INC &0A
9B1E		164 010	A4 0A	LDY &0A
9B20		177 011	B1 0B	LDA (&0B),Y
9B22		201 032	C9 20	CMP#&20
9B24		240 246	F0 F6	BEQ -10 --> &9B1C
9B26		201 141	C9 8D	CMP#&8D
9B28		208 026	D0 1A	BNE 26 --> &9B44
9B2A		200	C8	INY
9B2B		177 011	B1 0B	LDA (&0B),Y
9B2D		010	0A	ASL A
9B2E		010	0A	ASL A
9B2F		170	AA	TAX
9B30	)	041 192	29 C0	AND#&C0
9B32		200	C8	INY
9B33	Q	081 011	51 0B	EOR (&0B),Y
9B35	*	133 042	85 2A	STA &2A
9B37		138	8A	TXA
9B38		010	0A	ASL A
9B39		010	0A	ASL A
9B3A		200	C8	INY
9B3B	Q	081 011	51 0B	EOR (&0B),Y
9B3D	+	133 043	85 2B	STA &2B
9B3F		200	C8	INY
9B40		132 010	84 0A	STY &0A
9B42	8	056	38	SEC
9B43	`	096	60	RTS
9B44		024	18	CLC
9B45	`	096	60	RTS

```
9B46        165 011          A5 0B       LDA &0B
9B48        133 025          85 19       STA &19
9B4A        165 012          A5 0C       LDA &0C
9B4C        133 026          85 1A       STA &1A
9B4E        165 010          A5 0A       LDA &0A
9B50        133 027          85 1B       STA &1B
9B52        164 027          A4 1B       LDY &1B
9B54        230 027          E6 1B       INC &1B
9B56        177 025          B1 19       LDA (&19),Y
9B58        201 032          C9 20       CMP#&20
9B5A        240 246          F0 F6       BEQ -10 --> &9B52
9B5C   =    201 061          C9 3D       CMP#&3D
9B5E   .    240 046          F0 2E       BEQ 46 --> &9B8E
9B60        000              00          BRK
9B61        004              04          EQUB &04
9B62   M    077              4D          xxx Invalid Code
9B63   is   105 115          69 73       ADC#&73
9B65   ta   116 097          74 61       STZ &61,X
9B67   k    107              6B          xxx Invalid Code
9B68   e    101              65          xxx Invalid Code
9B69        000              00          BRK
9B6A        016              10          EQUB &10
9B6B   S    083              53          xxx Invalid Code
9B6C   ynt  121 110 116      79 6E 74    ADC &746E,Y
9B6F   ax   097 120 032      61 78 20    ADC (&2078,X)
9B72   er   101 114          65 72       ADC &72
9B74   ro   114 111          72 6F       ADC (&6F)
9B76   r    114              72          xxx Invalid Code
9B77        000              00          BRK
9B78        013              0D          EQUB &0D
9B79   No   078 111          4E 6F       xxx Invalid Code
9B7B        032 242          20 F2       xxx Invalid Code
9B7D        000              00          BRK
9B7E        017              11          EQUB &11
9B7F   E    069              45          xxx Invalid Code
```

9B80	s	115	73	xxx Invalid Code
9B81	c	099	63	xxx Invalid Code
9B82	ape	097 112 101	61 70 65	ADC (&6570,X)
9B85		000	00	EQUB &00
9B86		032 213 142	20 D5 8E	JSR &8ED5
9B89	=	201 061	C9 3D	CMP#&3D
9B8B		208 211	D0 D3	BNE -45 --> &9B60
9B8D	`	096	60	RTS
9B8E	;	032 059 157	20 3B 9D	JSR &9D3B
9B91		138	8A	TXA
9B92		164 027	A4 1B	LDY &1B
9B94		128 026	80 1A	BRA 26 --> &9BB0
9B96		164 027	A4 1B	LDY &1B
9B98		128 014	80 0E	BRA 14 --> &9BA8
9B9A		186	BA	TSX
9B9B		224 252	E0 FC	CPX#&FC
9B9D		176 216	B0 D8	BCS -40 --> &9B77
9B9F		173 255 001	AD FF 01	LDA &01FF
9BA2		201 242	C9 F2	CMP#&F2
9BA4		208 209	D0 D1	BNE -47 --> &9B77
9BA6		164 010	A4 0A	LDY &0A
9BA8		136	88	DEY
9BA9		200	C8	INY
9BAA		177 011	B1 0B	LDA (&0B),Y
9BAC		201 032	C9 20	CMP#&20
9BAE		240 249	F0 F9	BEQ -7 --> &9BA9
9BB0	:	201 058	C9 3A	CMP#&3A
9BB2		240 008	F0 08	BEQ 8 --> &9BBC
9BB4		201 013	C9 0D	CMP#&0D
9BB6		240 004	F0 04	BEQ 4 --> &9BBC
9BB8		201 139	C9 8B	CMP#&8B
9BBA		208 173	D0 AD	BNE -83 --> &9B69
9BBC		024	18	CLC
9BBD		152	98	TYA
9BBE	e	101 011	65 0B	ADC &0B

9BC0		133 011	85 0B	STA &0B
9BC2		144 002	90 02	BCC 2 --> &9BC6
9BC4		230 012	E6 0C	INC &0C
9BC6		160 001	A0 01	LDY#&01
9BC8		132 010	84 0A	STY &0A
9BCA	$	036 255	24 FF	BIT &FF
9BCC	0	048 175	30 AF	BMI -81 --> &9B7D
9BCE	`	096	60	RTS
9BCF		032 166 155	20 A6 9B	JSR &9BA6
9BD2		178 011	B2 0B	LDA (&0B)
9BD4	:	201 058	C9 3A	CMP#&3A
9BD6		240 246	F0 F6	BEQ -10 --> &9BCE
9BD8		165 012	A5 0C	LDA &0C
9BDA		201 007	C9 07	CMP#&07
9BDC	$	240 036	F0 24	BEQ 36 --> &9C02
9BDE		160 001	A0 01	LDY#&01
9BE0		177 011	B1 0B	LDA (&0B),Y
9BE2	0	048 030	30 1E	BMI 30 --> &9C02
9BE4		166 032	A6 20	LDX &20
9BE6		240 010	F0 0A	BEQ 10 --> &9BF2
9BE8	+	133 043	85 2B	STA &2B
9BEA		200	C8	INY
9BEB		177 011	B1 0B	LDA (&0B),Y
9BED	*	133 042	85 2A	STA &2A
9BEF	K	032 075 156	20 4B 9C	JSR &9C4B
9BF2		169 003	A9 03	LDA#&03
9BF4		024	18	CLC
9BF5	e	101 011	65 0B	ADC &0B
9BF7		133 011	85 0B	STA &0B
9BF9		144 002	90 02	BCC 2 --> &9BFD
9BFB		230 012	E6 0C	INC &0C
9BFD		160 001	A0 01	LDY#&01
9BFF		132 010	84 0A	STY &0A
9C01	`	096	60	RTS
9C02	L	076 134 143	4C 86 8F	JMP &8F86

9C05	L	076 146 144	4C 92 90	JMP &9092
9C08	/	032 047 157	20 2F 9D	JSR &9D2F
9C0B		240 248	F0 F8	BEQ -8 --> &9C05
9C0D		016 003	10 03	BPL 3 --> &9C12
9C0F		032 195 150	20 C3 96	JSR &96C3
9C12		164 027	A4 1B	LDY &1B
9C14		132 010	84 0A	STY &0A
9C16	*	165 042	A5 2A	LDA &2A
9C18	+	005 043	05 2B	ORA &2B
9C1A	,	005 044	05 2C	ORA &2C
9C1C	-	005 045	05 2D	ORA &2D
9C1E		240 023	F0 17	BEQ 23 --> &9C37
9C20		224 140	E0 8C	CPX#&8C
9C22		240 003	F0 03	BEQ 3 --> &9C27
9C24	L	076 011 144	4C 0B 90	JMP &900B
9C27		230 010	E6 0A	INC &0A
9C29		032 030 155	20 1E 9B	JSR &9B1E
9C2C		144 246	90 F6	BCC -10 --> &9C24
9C2E	6	032 054 184	20 36 B8	JSR &B836
9C31		032 198 155	20 C6 9B	JSR &9BC6
9C34	L#	076 035 183	4C 23 B7	JMP &B723
9C37		164 010	A4 0A	LDY &0A
9C39		177 011	B1 0B	LDA (&0B),Y
9C3B		201 013	C9 0D	CMP#&0D
9C3D		240 009	F0 09	BEQ 9 --> &9C48
9C3F		200	C8	INY
9C40		201 139	C9 8B	CMP#&8B
9C42		208 245	D0 F5	BNE -11 --> &9C39
9C44		132 010	84 0A	STY &0A
9C46		240 225	F0 E1	BEQ -31 --> &9C29
9C48	L	076 184 143	4C B8 8F	JMP &8FB8
9C4B	*	165 042	A5 2A	LDA &2A
9C4D	!	197 033	C5 21	CMP &21
9C4F	+	165 043	A5 2B	LDA &2B
9C51	"	229 034	E5 22	SBC &22

```
9C53        176 172        B0 AC       BCS -84 --> &9C01
9C55   [    169 091        A9 5B       LDA#&5B
9C57        032 152 189    20 98 BD    JSR &BD98
9C5A        032 129 160    20 81 A0    JSR &A081
9C5D   ]    169 093        A9 5D       LDA#&5D
9C5F        032 152 189    20 98 BD    JSR &BD98
9C62   L    076 146 189    4C 92 BD    JMP &BD92
9C65   h    104            68          PLA
9C66   *    133 042        85 2A       STA &2A
9C68   h    104            68          PLA
9C69   +    133 043        85 2B       STA &2B
9C6B   h    104            68          PLA
9C6C   ,    133 044        85 2C       STA &2C
9C6E   h    104            68          PLA
9C6F   -    133 045        85 2D       STA &2D
9C71        032 250 187    20 FA BB    JSR &BBFA
9C74        032 133 129    20 85 81    JSR &8185
9C77        032 011 164    20 0B A4    JSR &A40B
9C7A        032 232 187    20 E8 BB    JSR &BBE8
9C7D   A    032 065 165    20 41 A5    JSR &A541
9C80        128 016        80 10       BRA 16 --> &9C92
9C82        032 250 187    20 FA BB    JSR &BBFA
9C85   L    032 076 158    20 4C 9E    JSR &9E4C
9C88        168            A8          TAY
9C89        032 221 150    20 DD 96    JSR &96DD
9C8C        032 232 187    20 E8 BB    JSR &BBE8
9C8F        032 224 164    20 E0 A4    JSR &A4E0
9C92        160 000        A0 00       LDY#&00
9C94        169 127        A9 7F       LDA#&7F
9C96   ;    020 059        14 3B       TRB &3B
9C98   .    165 046        A5 2E       LDA &2E
9C9A   )    041 128        29 80       AND#&80
9C9C   ;    197 059        C5 3B       CMP &3B
9C9E        208 030        D0 1E       BNE 30 --> &9CBE
9CA0   <    165 060        A5 3C       LDA &3C
```

9CA2	0	197 048	C5 30	CMP &30
9CA4		208 025	D0 19	BNE 25 --> &9CBF
9CA6	=	165 061	A5 3D	LDA &3D
9CA8	1	197 049	C5 31	CMP &31
9CAA		208 019	D0 13	BNE 19 --> &9CBF
9CAC	>	165 062	A5 3E	LDA &3E
9CAE	2	197 050	C5 32	CMP &32
9CB0		208 013	D0 0D	BNE 13 --> &9CBF
9CB2	?	165 063	A5 3F	LDA &3F
9CB4	3	197 051	C5 33	CMP &33
9CB6		208 007	D0 07	BNE 7 --> &9CBF
9CB8	@	165 064	A5 40	LDA &40
9CBA	4	197 052	C5 34	CMP &34
9CBC		208 001	D0 01	BNE 1 --> &9CBF
9CBE	`	096	60	RTS
9CBF	j	106	6A	ROR A
9CC0	E;	069 059	45 3B	EOR &3B
9CC2	*	042	2A	ROL A
9CC3		169 001	A9 01	LDA#&01
9CC5	`	096	60	RTS
9CC6	L	076 146 144	4C 92 90	JMP &9092
9CC9		138	8A	TXA
9CCA	6	240 054	F0 36	BEQ 54 --> &9D02
9CCC	0	048 180	30 B4	BMI -76 --> &9C82
9CCE	-	165 045	A5 2D	LDA &2D
9CD0	H	072	48	PHA
9CD1	,	165 044	A5 2C	LDA &2C
9CD3	H	072	48	PHA
9CD4	+	165 043	A5 2B	LDA &2B
9CD6	H	072	48	PHA
9CD7	*	165 042	A5 2A	LDA &2A
9CD9	H	072	48	PHA
9CDA	L	032 076 158	20 4C 9E	JSR &9E4C
9CDD		168	A8	TAY
9CDE		240 230	F0 E6	BEQ -26 --> &9CC6

9CE0	0	048 131	30 83	BMI -125 --> &9C65
9CE2	-	165 045	A5 2D	LDA &2D
9CE4	I	073 128	49 80	EOR#&80
9CE6	-	133 045	85 2D	STA &2D
9CE8	8	056	38	SEC
9CE9	h	104	68	PLA
9CEA	*	229 042	E5 2A	SBC &2A
9CEC	*	133 042	85 2A	STA &2A
9CEE	h	104	68	PLA
9CEF	+	229 043	E5 2B	SBC &2B
9CF1	*	004 042	04 2A	TSB &2A
9CF3	h	104	68	PLA
9CF4	,	229 044	E5 2C	SBC &2C
9CF6	*	004 042	04 2A	TSB &2A
9CF8	h	104	68	PLA
9CF9		160 000	A0 00	LDY#&00
9CFB	I	073 128	49 80	EOR#&80
9CFD	-	229 045	E5 2D	SBC &2D
9CFF	*	005 042	05 2A	ORA &2A
9D01	`	096	60	RTS
9D02	Q	032 081 188	20 51 BC	JSR &BC51
9D05	L	032 076 158	20 4C 9E	JSR &9E4C
9D08		168	A8	TAY
9D09		208 187	D0 BB	BNE -69 --> &9CC6
9D0B		178 004	B2 04	LDA (&04)
9D0D	6	197 054	C5 36	CMP &36
9D0F		144 002	90 02	BCC 2 --> &9D13
9D11	6	165 054	A5 36	LDA &36
9D13	7	133 055	85 37	STA &37
9D15	7	196 055	C4 37	CPY &37
9D17		240 010	F0 0A	BEQ 10 --> &9D23
9D19		200	C8	INY
9D1A		177 004	B1 04	LDA (&04),Y
9D1C		217 255 005	D9 FF 05	CMP &05FF,Y
9D1F		240 244	F0 F4	BEQ -12 --> &9D15

9D21		128 004	80 04	BRA 4 --> &9D27
9D23		178 004	B2 04	LDA (&04)
9D25	6	197 054	C5 36	CMP &36
9D27		008	08	PHP
9D28		032 225 188	20 E1 BC	JSR &BCE1
9D2B		160 000	A0 00	LDY#&00
9D2D	(	040	28	PLP
9D2E	`	096	60	RTS
9D2F		165 011	A5 0B	LDA &0B
9D31		133 025	85 19	STA &19
9D33		165 012	A5 0C	LDA &0C
9D35		133 026	85 1A	STA &1A
9D37		165 010	A5 0A	LDA &0A
9D39		133 027	85 1B	STA &1B
9D3B		032 129 157	20 81 9D	JSR &9D81
9D3E		224 132	E0 84	CPX#&84
9D40		240 010	F0 0A	BEQ 10 --> &9D4C
9D42		224 130	E0 82	CPX#&82
9D44		240 032	F0 20	BEQ 32 --> &9D66
9D46		198 027	C6 1B	DEC &1B
9D48		168	A8	TAY
9D49	'	133 039	85 27	STA &27
9D4B	`	096	60	RTS
9D4C	{	032 123 157	20 7B 9D	JSR &9D7B
9D4F		032 190 150	20 BE 96	JSR &96BE
9D52		160 003	A0 03	LDY#&03
9D54		177 004	B1 04	LDA (&04),Y
9D56	*	025 042 000	19 2A 00	ORA &002A,Y
9D59	*	153 042 000	99 2A 00	STA &002A,Y
9D5C		136	88	DEY
9D5D		016 245	10 F5	BPL -11 --> &9D54
9D5F		032 250 188	20 FA BC	JSR &BCFA
9D62	@	169 064	A9 40	LDA#&40
9D64		128 216	80 D8	BRA -40 --> &9D3E
9D66	{	032 123 157	20 7B 9D	JSR &9D7B

9D69		032 190 150	20 BE 96	JSR &96BE
9D6C		160 003	A0 03	LDY#&03
9D6E		177 004	B1 04	LDA (&04),Y
9D70	Y*	089 042 000	59 2A 00	EOR &002A,Y
9D73	*	153 042 000	99 2A 00	STA &002A,Y
9D76		136	88	DEY
9D77		016 245	10 F5	BPL -11 --> &9D6E
9D79		128 228	80 E4	BRA -28 --> &9D5F
9D7B		032 190 150	20 BE 96	JSR &96BE
9D7E	&	032 038 188	20 26 BC	JSR &BC26
9D81		032 169 157	20 A9 9D	JSR &9DA9
9D84		224 128	E0 80	CPX#&80
9D86		240 001	F0 01	BEQ 1 --> &9D89
9D88	`	096	60	RTS
9D89		032 190 150	20 BE 96	JSR &96BE
9D8C	&	032 038 188	20 26 BC	JSR &BC26
9D8F		032 169 157	20 A9 9D	JSR &9DA9
9D92		032 190 150	20 BE 96	JSR &96BE
9D95		160 003	A0 03	LDY#&03
9D97		177 004	B1 04	LDA (&04),Y
9D99	9*	057 042 000	39 2A 00	AND &002A,Y
9D9C	*	153 042 000	99 2A 00	STA &002A,Y
9D9F		136	88	DEY
9DA0		016 245	10 F5	BPL -11 --> &9D97
9DA2		032 250 188	20 FA BC	JSR &BCFA
9DA5	@	169 064	A9 40	LDA#&40
9DA7		128 219	80 DB	BRA -37 --> &9D84
9DA9	L	032 076 158	20 4C 9E	JSR &9E4C
9DAC	?	224 063	E0 3F	CPX#&3F
9DAE		176 004	B0 04	BCS 4 --> &9DB4
9DB0	<	224 060	E0 3C	CPX#&3C
9DB2		176 001	B0 01	BCS 1 --> &9DB5
9DB4	`	096	60	RTS
9DB5		240 022	F0 16	BEQ 22 --> &9DCD
9DB7	>	224 062	E0 3E	CPX#&3E

```
9DB9    :   240 058         F0 3A       BEQ 58 --> &9DF5
9DBB        170             AA          TAX
9DBC        032 202 156     20 CA 9C    JSR &9CCA
9DBF        208 001         D0 01       BNE 1 --> &9DC2
9DC1        136             88          DEY
9DC2    *   132 042         84 2A       STY &2A
9DC4    +   132 043         84 2B       STY &2B
9DC6    ,   132 044         84 2C       STY &2C
9DC8    -   132 045         84 2D       STY &2D
9DCA    @   169 064         A9 40       LDA#&40
9DCC    `   096             60          RTS
9DCD        170             AA          TAX
9DCE        164 027         A4 1B       LDY &1B
9DD0        177 025         B1 19       LDA (&19),Y
9DD2    =   201 061         C9 3D       CMP#&3D
9DD4        240 011         F0 0B       BEQ 11 --> &9DE1
9DD6    >   201 062         C9 3E       CMP#&3E
9DD8        240 018         F0 12       BEQ 18 --> &9DEC
9DDA        032 201 156     20 C9 9C    JSR &9CC9
9DDD        144 226         90 E2       BCC -30 --> &9DC1
9DDF        128 225         80 E1       BRA -31 --> &9DC2
9DE1        230 027         E6 1B       INC &1B
9DE3        032 201 156     20 C9 9C    JSR &9CC9
9DE6        240 217         F0 D9       BEQ -39 --> &9DC1
9DE8        144 215         90 D7       BCC -41 --> &9DC1
9DEA        128 214         80 D6       BRA -42 --> &9DC2
9DEC        230 027         E6 1B       INC &1B
9DEE        032 201 156     20 C9 9C    JSR &9CC9
9DF1        208 206         D0 CE       BNE -50 --> &9DC1
9DF3        128 205         80 CD       BRA -51 --> &9DC2
9DF5        170             AA          TAX
9DF6        164 027         A4 1B       LDY &1B
9DF8        177 025         B1 19       LDA (&19),Y
9DFA    =   201 061         C9 3D       CMP#&3D
9DFC        240 009         F0 09       BEQ 9 --> &9E07
```

9DFE		032 201 156	20 C9 9C	JSR &9CC9
9E01		240 191	F0 BF	BEQ -65 --> &9DC2
9E03		176 188	B0 BC	BCS -68 --> &9DC1
9E05		128 187	80 BB	BRA -69 --> &9DC2
9E07		230 027	E6 1B	INC &1B
9E09		032 201 156	20 C9 9C	JSR &9CC9
9E0C		176 179	B0 B3	BCS -77 --> &9DC1
9E0E		128 178	80 B2	BRA -78 --> &9DC2
9E10		000	00	BRK
9E11		019	13	EQUB &13
9E12	S	083	53	xxx Invalid Code
9E13	tr	116 114	74 72	STZ &72,X
9E15	in	105 110	69 6E	ADC#&6E
9E17	g	103	67	xxx Invalid Code
9E18	to	032 116 111	20 74 6F	JSR &6F74
9E1B	o	111	6F	xxx Invalid Code
9E1C	lo	032 108 111	20 6C 6F	JSR &6F6C
9E1F	ng	110 103	6E 67	xxx Invalid Code
9E21		000	00	EQUB &00
9E22	Q	032 081 188	20 51 BC	JSR &BC51
9E25		032 018 160	20 12 A0	JSR &A012
9E28		168	A8	TAY
9E29	f	208 102	D0 66	BNE 102 --> &9E91
9E2B		024	18	CLC
9E2C		218	DA	PHX
9E2D		178 004	B2 04	LDA (&04)
9E2F	e6	101 054	65 36	ADC &36
9E31		176 221	B0 DD	BCS -35 --> &9E10
9E33		170	AA	TAX
9E34	H	072	48	PHA
9E35	6	164 054	A4 36	LDY &36
9E37		185 255 005	B9 FF 05	LDA &05FF,Y
9E3A		157 255 005	9D FF 05	STA &05FF,X
9E3D		202	CA	DEX
9E3E		136	88	DEY

9E3F		208 246	D0 F6	BNE -10 --> &9E37
9E41		032 210 188	20 D2 BC	JSR &BCD2
9E44	h	104	68	PLA
9E45	6	133 054	85 36	STA &36
9E47		250	FA	PLX
9E48		169 000	A9 00	LDA#&00
9E4A		128 003	80 03	BRA 3 --> &9E4F
9E4C		032 199 159	20 C7 9F	JSR &9FC7
9E4F	+	224 043	E0 2B	CPX#&2B
9E51		240 005	F0 05	BEQ 5 --> &9E58
9E53	-	224 045	E0 2D	CPX#&2D
9E55	f	240 102	F0 66	BEQ 102 --> &9EBD
9E57	`	096	60	RTS
9E58		168	A8	TAY
9E59		240 199	F0 C7	BEQ -57 --> &9E22
9E5B	07	048 055	30 37	BMI 55 --> &9E94
9E5D		032 196 159	20 C4 9F	JSR &9FC4
9E60		168	A8	TAY
9E61	.	240 046	F0 2E	BEQ 46 --> &9E91
9E63	0K	048 075	30 4B	BMI 75 --> &9EB0
9E65		024	18	CLC
9E66		178 004	B2 04	LDA (&04)
9E68	e*	101 042	65 2A	ADC &2A
9E6A	*	133 042	85 2A	STA &2A
9E6C		160 001	A0 01	LDY#&01
9E6E		177 004	B1 04	LDA (&04),Y
9E70	e+	101 043	65 2B	ADC &2B
9E72	+	133 043	85 2B	STA &2B
9E74		200	C8	INY
9E75		177 004	B1 04	LDA (&04),Y
9E77	e,	101 044	65 2C	ADC &2C
9E79	,	133 044	85 2C	STA &2C
9E7B		200	C8	INY
9E7C		177 004	B1 04	LDA (&04),Y
9E7E	e-	101 045	65 2D	ADC &2D

9E80	-	133 045	85 2D	STA &2D
9E82		024	18	CLC
9E83		165 004	A5 04	LDA &04
9E85	i	105 004	69 04	ADC#&04
9E87		133 004	85 04	STA &04
9E89	@	169 064	A9 40	LDA#&40
9E8B		144 194	90 C2	BCC -62 --> &9E4F
9E8D		230 005	E6 05	INC &05
9E8F		128 190	80 BE	BRA -66 --> &9E4F
9E91	L	076 146 144	4C 92 90	JMP &9092
9E94		032 250 187	20 FA BB	JSR &BBFA
9E97		032 199 159	20 C7 9F	JSR &9FC7
9E9A		168	A8	TAY
9E9B		240 244	F0 F4	BEQ -12 --> &9E91
9E9D	'	134 039	86 27	STX &27
9E9F	0	048 003	30 03	BMI 3 --> &9EA4
9EA1		032 133 129	20 85 81	JSR &8185
9EA4		032 232 187	20 E8 BB	JSR &BBE8
9EA7		032 141 166	20 8D A6	JSR &A68D
9EAA	'	166 039	A6 27	LDX &27
9EAC		169 255	A9 FF	LDA#&FF
9EAE		128 159	80 9F	BRA -97 --> &9E4F
9EB0	'	134 039	86 27	STX &27
9EB2		032 230 188	20 E6 BC	JSR &BCE6
9EB5		032 250 187	20 FA BB	JSR &BBFA
9EB8		032 133 129	20 85 81	JSR &8185
9EBB		128 231	80 E7	BRA -25 --> &9EA4
9EBD		168	A8	TAY
9EBE		240 209	F0 D1	BEQ -47 --> &9E91
9EC0	0%	048 037	30 25	BMI 37 --> &9EE7
9EC2		032 196 159	20 C4 9F	JSR &9FC4
9EC5		168	A8	TAY
9EC6		240 201	F0 C9	BEQ -55 --> &9E91
9EC8	05	048 053	30 35	BMI 53 --> &9EFF
9ECA	8	056	38	SEC

```
9ECB        178 004       B2 04       LDA (&04)
9ECD    *   229 042       E5 2A       SBC &2A
9ECF    *   133 042       85 2A       STA &2A
9ED1        160 001       A0 01       LDY#&01
9ED3        177 004       B1 04       LDA (&04),Y
9ED5    +   229 043       E5 2B       SBC &2B
9ED7    +   133 043       85 2B       STA &2B
9ED9        200           C8          INY
9EDA        177 004       B1 04       LDA (&04),Y
9EDC    ,   229 044       E5 2C       SBC &2C
9EDE    ,   133 044       85 2C       STA &2C
9EE0        200           C8          INY
9EE1        177 004       B1 04       LDA (&04),Y
9EE3    -   229 045       E5 2D       SBC &2D
9EE5        128 153       80 99       BRA -103 --> &9E80
9EE7        032 250 187   20 FA BB    JSR &BBFA
9EEA        032 199 159   20 C7 9F    JSR &9FC7
9EED        168           A8          TAY
9EEE        240 161       F0 A1       BEQ -95 --> &9E91
9EF0    '   134 039       86 27       STX &27
9EF2    0   048 003       30 03       BMI 3 --> &9EF7
9EF4        032 133 129   20 85 81    JSR &8185
9EF7        032 232 187   20 E8 BB    JSR &BBE8
9EFA        032 138 166   20 8A A6    JSR &A68A
9EFD        128 171       80 AB       BRA -85 --> &9EAA
9EFF    '   134 039       86 27       STX &27
9F01        032 230 188   20 E6 BC    JSR &BCE6
9F04        032 250 187   20 FA BB    JSR &BBFA
9F07        032 133 129   20 85 81    JSR &8185
9F0A        032 232 187   20 E8 BB    JSR &BBE8
9F0D        032 199 172   20 C7 AC    JSR &ACC7
9F10        128 152       80 98       BRA -104 --> &9EAA
9F12        032 133 129   20 85 81    JSR &8185
9F15        032 230 188   20 E6 BC    JSR &BCE6
9F18        032 250 187   20 FA BB    JSR &BBFA
```

9F1B		032 133 129	20 85 81	JSR &8185
9F1E		128 013	80 0D	BRA 13 --> &9F2D
9F20		032 133 129	20 85 81	JSR &8185
9F23		032 250 187	20 FA BB	JSR &BBFA
9F26		032 018 160	20 12 A0	JSR &A012
9F29		168	A8	TAY
9F2A		032 221 150	20 DD 96	JSR &96DD
9F2D		032 232 187	20 E8 BB	JSR &BBE8
9F30		032 166 166	20 A6 A6	JSR &A6A6
9F33		169 255	A9 FF	LDA#&FF
9F35	L	076 202 159	4C CA 9F	JMP &9FCA
9F38	L	076 146 144	4C 92 90	JMP &9092
9F3B		168	A8	TAY
9F3C		240 250	F0 FA	BEQ -6 --> &9F38
9F3E	0	048 227	30 E3	BMI -29 --> &9F23
9F40	-	164 045	A4 2D	LDY &2D
9F42	,	196 044	C4 2C	CPY &2C
9F44		208 218	D0 DA	BNE -38 --> &9F20
9F46	+	165 043	A5 2B	LDA &2B
9F48		010	0A	ASL A
9F49		152	98	TYA
9F4A	i	105 000	69 00	ADC#&00
9F4C		208 210	D0 D2	BNE -46 --> &9F20
9F4E		032 015 160	20 0F A0	JSR &A00F
9F51		168	A8	TAY
9F52		240 228	F0 E4	BEQ -28 --> &9F38
9F54	0	048 191	30 BF	BMI -65 --> &9F15
9F56	-	164 045	A4 2D	LDY &2D
9F58	,	196 044	C4 2C	CPY &2C
9F5A		208 182	D0 B6	BNE -74 --> &9F12
9F5C	+	165 043	A5 2B	LDA &2B
9F5E		010	0A	ASL A
9F5F		152	98	TYA
9F60	i	105 000	69 00	ADC#&00
9F62		208 174	D0 AE	BNE -82 --> &9F12

9F64	Z	090	5A	PHY
9F65		032 190 172	20 BE AC	JSR &ACBE
9F68	'	134 039	86 27	STX &27
9F6A	9	162 057	A2 39	LDX#&39
9F6C		032 198 189	20 C6 BD	JSR &BDC6
9F6F		032 230 188	20 E6 BC	JSR &BCE6
9F72	h	104	68	PLA
9F73	E-	069 045	45 2D	EOR &2D
9F75	7	133 055	85 37	STA &37
9F77		032 190 172	20 BE AC	JSR &ACBE
9F7A		160 000	A0 00	LDY#&00
9F7C		162 000	A2 00	LDX#&00
9F7E	d?	100 063	64 3F	STZ &3F
9F80	d@	100 064	64 40	STZ &40
9F82	F:	070 058	46 3A	LSR &3A
9F84	f9	102 057	66 39	ROR &39
9F86		144 021	90 15	BCC 21 --> &9F9D
9F88		024	18	CLC
9F89		152	98	TYA
9F8A	e*	101 042	65 2A	ADC &2A
9F8C		168	A8	TAY
9F8D		138	8A	TXA
9F8E	e+	101 043	65 2B	ADC &2B
9F90		170	AA	TAX
9F91	?	165 063	A5 3F	LDA &3F
9F93	e,	101 044	65 2C	ADC &2C
9F95	?	133 063	85 3F	STA &3F
9F97	@	165 064	A5 40	LDA &40
9F99	e-	101 045	65 2D	ADC &2D
9F9B	@	133 064	85 40	STA &40
9F9D	*	006 042	06 2A	ASL &2A
9F9F	&+	038 043	26 2B	ROL &2B
9FA1	&,	038 044	26 2C	ROL &2C
9FA3	&-	038 045	26 2D	ROL &2D
9FA5	9	165 057	A5 39	LDA &39

```
9FA7    :    005 058          05 3A       ORA &3A
9FA9         208 215          D0 D7       BNE -41 --> &9F82
9FAB    =    132 061          84 3D       STY &3D
9FAD    >    134 062          86 3E       STX &3E
9FAF    7    165 055          A5 37       LDA &37
9FB1         008              08          PHP
9FB2    =    162 061          A2 3D       LDX#&3D
9FB4         032 128 170      20 80 AA    JSR &AA80
9FB7    (    040              28          PLP
9FB8         016 003          10 03       BPL 3 --> &9FBD
9FBA         032 222 172      20 DE AC    JSR &ACDE
9FBD    '    166 039          A6 27       LDX &27
9FBF         128 009          80 09       BRA 9 --> &9FCA
9FC1    L;   076 059 159      4C 3B 9F    JMP &9F3B
9FC4    &    032 038 188      20 26 BC    JSR &BC26
9FC7         032 018 160      20 12 A0    JSR &A012
9FCA    *    224 042          E0 2A       CPX#&2A
9FCC         240 243          F0 F3       BEQ -13 --> &9FC1
9FCE    /    224 047          E0 2F       CPX#&2F
9FD0         240 009          F0 09       BEQ 9 --> &9FDB
9FD2         224 131          E0 83       CPX#&83
9FD4         240 031          F0 1F       BEQ 31 --> &9FF5
9FD6         224 129          E0 81       CPX#&81
9FD8    #    240 035          F0 23       BEQ 35 --> &9FFD
9FDA    `    096              60          RTS
9FDB         168              A8          TAY
9FDC         032 221 150      20 DD 96    JSR &96DD
9FDF         032 250 187      20 FA BB    JSR &BBFA
9FE2         032 018 160      20 12 A0    JSR &A012
9FE5    '    134 039          86 27       STX &27
9FE7         168              A8          TAY
9FE8         032 221 150      20 DD 96    JSR &96DD
9FEB         032 232 187      20 E8 BB    JSR &BBE8
9FEE         032 238 165      20 EE A5    JSR &A5EE
9FF1         169 255          A9 FF       LDA#&FF
```

9FF3		128 200	80 C8	BRA -56 --> &9FBD
9FF5		032 249 128	20 F9 80	JSR &80F9
9FF8	8	165 056	A5 38	LDA &38
9FFA		008	08	PHP
9FFB		128 181	80 B5	BRA -75 --> &9FB2
9FFD		032 249 128	20 F9 80	JSR &80F9
A000	&9	038 057	26 39	ROL &39

```
9000      198 010     C6 0A     DEC &0A
9002      032 166 155 20 A6 9B  JSR &9BA6
9005      178 011     B2 0B     LDA (&0B)
9007  :   201 058     C9 3A     CMP#&3A
9009      208 178     D0 B2     BNE -78   --> &8FBD
900B      164 010     A4 0A     LDY &0A
900D      230 010     E6 0A     INC &0A
900F      177 011     B1 0B     LDA (&0B),Y
9011      201 032     C9 20     CMP#&20
9013      240 246     F0 F6     BEQ -10   --> &900B
9015      201 207     C9 CF     CMP#&CF
9017      144 012     90 0C     BCC 12   --> &9025
9019      010         0A        ASL A
901A      170         AA        TAX
901B |M   124 077 135 7C 4D 87  JMP (&874D,X)
901E      032 224 142 20 E0 8E  JSR &8EE0
9021      201 198     C9 C6     CMP#&C6
9023      176 244     B0 F4     BCS -12   --> &9019
9025      166 011     A6 0B     LDX &0B
9027      134 025     86 19     STX &19
9029      166 012     A6 0C     LDX &0C
902B      134 026     86 1A     STX &1A
902D      132 027     84 1B     STY &1B
902F      032 009 153 20 09 99  JSR &9909
9032      208 027     D0 1B     BNE 27   --> &904F
9034      176 181     B0 B5     BCS -75   --> &8FEB
9036      134 027     86 1B     STX &1B
9038      032 134 155 20 86 9B  JSR &9B86
903B  T   032 084 152 20 54 98  JSR &9854
903E      162 005     A2 05     LDX#&05
9040  ,   228 044     E4 2C     CPX &2C
9042      208 001     D0 01     BNE 1   --> &9045
9044      232         E8        INX
9045      032 131 152 20 83 98  JSR &9883
9048      198 010     C6 0A     DEC &0A
904A      032 174 152 20 AE 98  JSR &98AE
904D  4   240 052     F0 34     BEQ 52   --> &9083
904F  !   144 033     90 21     BCC 33   --> &9072
9051  &   032 038 188 20 26 BC  JSR &BC26
9054  R   032 082 155 20 52 9B  JSR &9B52
9057  '   165 039     A5 27     LDA &27
9059  7   208 055     D0 37     BNE 55   --> &9092
```

```
905B      032 171 144 20 AB 90  JSR &90AB
905E      128 165     80 A5     BRA -91   --> &9005
9060      186         BA        TSX
9061      224 252     E0 FC     CPX#&FC
9063  '   176 039     B0 27     BCS 39   --> &908C
9065      173 255 001 AD FF 01  LDA &01FF
9068      201 164     C9 A4     CMP#&A4
906A      208 032     D0 20     BNE 32   --> &908C
906C  /   032 047 157 20 2F 9D  JSR &9D2F
906F L    076 145 155 4C 91 9B  JMP &9B91
9072  *   165 042     A5 2A     LDA &2A
9074 H    072         48        PHA
9075  +   165 043     A5 2B     LDA &2B
9077 H    072         48        PHA
9078  ,   165 044     A5 2C     LDA &2C
907A H    072         48        PHA
907B  R   032 082 155 20 52 9B  JSR &9B52
907E  +   032 043 179 20 2B B3  JSR &B32B
9081      128 130     80 82     BRA -126   --> &9005
9083 Li   076 105 155 4C 69 9B  JMP &9B69
9086      032 166 155 20 A6 9B  JSR &9BA6
9089      000         00        BRK
908A      000         00        EQUB &00
908B      250         FA        PLX
908C      000         00        BRK
908D      007         07        EQUB &07
908E No   078 111 032 4E 6F 20  LSR &206F
9091      164         A4        xxx Invalid Code
9092      000         00        BRK
9093      006         06        EQUB &06
9094 T    084         54        xxx Invalid Code
9095 ype  121 112 101 79 70 65  ADC &6570,Y
9098  mi  032 109 105 20 6D 69  JSR &696D
909B s    115         73        xxx Invalid Code
909C mat  109 097 116 6D 61 74  ADC &7461
909F c    099         63        xxx Invalid Code
90A0 h    104         68        PLA
90A1      000         00        BRK
90A2      000         00        EQUB &00
90A3 No   078 111 032 4E 6F 20  LSR &206F
90A6 ro   114 111     72 6F     ADC (&6F)
90A8 o    111         6F        xxx Invalid Code
90A9 m    109         6D        xxx Invalid Code
```

```
90AA      000         00        EQUB &00
90AB      032 230 188 20 E6 BC  JSR &BCE6
90AE  ,   165 044     A5 2C     LDA &2C
90B0      201 128     C9 80     CMP#&80
90B2  x   240 120     F0 78     BEQ 120   --> &912C
90B4      160 002     A0 02     LDY#&02
90B6  *   177 042     B1 2A     LDA (&2A),Y
90B8  6   197 054     C5 36     CMP &36
90BA  R   176 082     B0 52     BCS 82   --> &910E
90BC      165 002     A5 02     LDA &02
90BE  ,   133 044     85 2C     STA &2C
90C0      165 003     A5 03     LDA &03
90C2  -   133 045     85 2D     STA &2D
90C4  6   165 054     A5 36     LDA &36
90C6      201 008     C9 08     CMP#&08
90C8      144 006     90 06     BCC 6   --> &90D0
90CA i    105 007     69 07     ADC#&07
90CC      144 002     90 02     BCC 2   --> &90D0
90CE      169 255     A9 FF     LDA#&FF
90D0      024         18        CLC
90D1 H    072         48        PHA
90D2      170         AA        TAX
90D3  *   177 042     B1 2A     LDA (&2A),Y
90D5 r*   114 042     72 2A     ADC (&2A)
90D7 E    069 002     45 02     EOR &02
90D9      208 015     D0 0F     BNE 15   --> &90EA
90DB      136         88        DEY
90DC q*   113 042     71 2A     ADC (&2A),Y
90DE E    069 003     45 03     EOR &03
90E0      208 008     D0 08     BNE 8   --> &90EA
90E2  -   133 045     85 2D     STA &2D
90E4      138         8A        TXA
90E5      200         C8        INY
90E6 8    056         38        SEC
90E7  *   241 042     F1 2A     SBC (&2A),Y
90E9      170         AA        TAX
90EA      138         8A        TXA
90EB      024         18        CLC
90EC e    101 002     65 02     ADC &02
90EE      168         A8        TAY
90EF      165 003     A5 03     LDA &03
90F1 i    105 000     69 00     ADC#&00
90F3      170         AA        TAX
```

```
90F4      196 004     C4 04     CPY &04
90F6      229 005     E5 05     SBC &05
90F8      176 167     B0 A7     BCS -89   --> &90A1
90FA      132 002     84 02     STY &02
90FC      134 003     86 03     STX &03
90FE h    104         68        PLA
90FF      160 002     A0 02     LDY#&02
9101  *   145 042     91 2A     STA (&2A),Y
9103      136         88        DEY
9104  -   165 045     A5 2D     LDA &2D
9106      240 006     F0 06     BEQ 6   --> &910E
9108  *   145 042     91 2A     STA (&2A),Y
910A  ,   165 044     A5 2C     LDA &2C
910C  *   146 042     92 2A     STA (&2A)
910E      160 003     A0 03     LDY#&03
9110  6   165 054     A5 36     LDA &36
9112  *   145 042     91 2A     STA (&2A),Y
9114      240 021     F0 15     BEQ 21   --> &912B
9116      160 001     A0 01     LDY#&01
9118  *   177 042     B1 2A     LDA (&2A),Y
911A  -   133 045     85 2D     STA &2D
911C  *   178 042     B2 2A     LDA (&2A)
911E  ,   133 044     85 2C     STA &2C
9120      136         88        DEY
9121      185 000 006 B9 00 06  LDA &0600,Y
9124  ,   145 044     91 2C     STA (&2C),Y
9126      200         C8        INY
9127  6   196 054     C4 36     CPY &36
9129      208 246     D0 F6     BNE -10   --> &9121
912B `    096         60        RTS
912C  +   032 043 190 20 2B BE  JSR &BE2B
912F      192 000     C0 00     CPY#&00
9131      240 011     F0 0B     BEQ 11   --> &913E
9133      185 000 006 B9 00 06  LDA &0600,Y
9136  *   145 042     91 2A     STA (&2A),Y
9138      136         88        DEY
9139      208 248     D0 F8     BNE -8   --> &9133
913B      173 000 006 AD 00 06  LDA &0600
913E  *   146 042     92 2A     STA (&2A)
9140 `    096         60        RTS
9141  <   032 060 186 20 3C BA  JSR &BA3C
9144 Z    090         5A        PHY
9145      032 235 142 20 EB 8E  JSR &8EEB
```

```
9148  =   208 061     D0 3D     BNE 61   --> &9187
914A  ;   032 059 157 20 3B 9D  JSR &9D3B
914D      032 017 165 20 11 A5  JSR &A511
9150 z    122         7A        PLY
9151  '   165 039     A5 27     LDA &27
9153      032 212 255 20 D4 FF  JSR &FFD4
9156      170         AA        TAX
9157      240 027     F0 1B     BEQ 27   --> &9174
9159 0    048 012     30 0C     BMI 12   --> &9167
915B      162 003     A2 03     LDX#&03
915D  *   181 042     B5 2A     LDA &2A,X
915F      032 212 255 20 D4 FF  JSR &FFD4
9162      202         CA        DEX
9163      016 248     10 F8     BPL -8   --> &915D
9165      128 221     80 DD     BRA -35   --> &9144
9167      162 004     A2 04     LDX#&04
9169  l   189 108 004 BD 6C 04  LDA &046C,X
916C      032 212 255 20 D4 FF  JSR &FFD4
916F      202         CA        DEX
9170      016 247     10 F7     BPL -9   --> &9169
9172      128 208     80 D0     BRA -48   --> &9144
9174  6   165 054     A5 36     LDA &36
9176      032 212 255 20 D4 FF  JSR &FFD4
9179      170         AA        TAX
917A      240 200     F0 C8     BEQ -56   --> &9144
917C      189 255 005 BD FF 05  LDA &05FF,X
917F      032 212 255 20 D4 FF  JSR &FFD4
9182      202         CA        DEX
9183      208 247     D0 F7     BNE -9   --> &917C
9185      128 189     80 BD     BRA -67   --> &9144
9187 h    104         68        PLA
9188      132 010     84 0A     STY &0A
918A L    076 002 144 4C 02 90  JMP &9002
918D      032 223 140 20 DF 8C  JSR &8CDF
9190      240 175     F0 AF     BEQ -81   --> &9141
9192      198 010     C6 0A     DEC &0A
9194      128 021     80 15     BRA 21   --> &91AB
9196      173 000 004 AD 00 04  LDA &0400
9199      240 016     F0 10     BEQ 16   --> &91AB
919B      165 030     A5 1E     LDA &1E
919D      240 012     F0 0C     BEQ 12   --> &91AB
919F      237 000 004 ED 00 04  SBC &0400
91A2      176 249     B0 F9     BCS -7   --> &919D
```

```
91A4      168         A8        TAY
91A5      032 146 189 20 92 BD  JSR &BD92
91A8      200         C8        INY
91A9      208 250     D0 FA     BNE -6   --> &91A5
91AB      024         18        CLC
91AC      173 000 004 AD 00 04  LDA &0400
91AF      133 020     85 14     STA &14
91B1 f    102 021     66 15     ROR &15
91B3      032 224 142 20 E0 8E  JSR &8EE0
91B6  :   201 058     C9 3A     CMP#&3A
91B8      240 008     F0 08     BEQ 8   --> &91C2
91BA      201 013     C9 0D     CMP#&0D
91BC      240 004     F0 04     BEQ 4   --> &91C2
91BE      201 139     C9 8B     CMP#&8B
91C0      208 025     D0 19     BNE 25   --> &91DB
91C2      032 146 186 20 92 BA  JSR &BA92
91C5 L    076 000 144 4C 00 90  JMP &9000
91C8 d    100 020     64 14     STZ &14
91CA d    100 021     64 15     STZ &15
91CC      032 224 142 20 E0 8E  JSR &8EE0
91CF  :   201 058     C9 3A     CMP#&3A
91D1      240 242     F0 F2     BEQ -14   --> &91C5
91D3      201 013     C9 0D     CMP#&0D
91D5      240 238     F0 EE     BEQ -18   --> &91C5
91D7      201 139     C9 8B     CMP#&8B
91D9      240 234     F0 EA     BEQ -22   --> &91C5
91DB  ~   201 126     C9 7E     CMP#&7E
91DD      240 210     F0 D2     BEQ -46   --> &91B1
91DF  ,   201 044     C9 2C     CMP#&2C
91E1      240 179     F0 B3     BEQ -77   --> &9196
91E3  ;   201 059     C9 3B     CMP#&3B
91E5      240 225     F0 E1     BEQ -31   --> &91C8
91E7  z   032 122 146 20 7A 92  JSR &927A
91EA      144 199     90 C7     BCC -57   --> &91B3
91EC      165 020     A5 14     LDA &14
91EE H    072         48        PHA
91EF      165 021     A5 15     LDA &15
91F1 H    072         48        PHA
91F2      198 027     C6 1B     DEC &1B
91F4  ;   032 059 157 20 3B 9D  JSR &9D3B
91F7 h    104         68        PLA
91F8      133 021     85 15     STA &15
91FA h    104         68        PLA
```

```
91FB      133 020     85 14     STA &14
91FD      165 027     A5 1B     LDA &1B
91FF      133 010     85 0A     STA &0A
9201      152         98        TYA
9202      240 019     F0 13     BEQ 19   --> &9217
9204      032 024 161 20 18 A1  JSR &A118
9207      165 020     A5 14     LDA &14
9209 8    056         38        SEC
920A  6   229 054     E5 36     SBC &36
920C      144 009     90 09     BCC 9   --> &9217
920E      240 007     F0 07     BEQ 7   --> &9217
9210      168         A8        TAY
9211      032 146 189 20 92 BD  JSR &BD92
9214      136         88        DEY
9215      208 250     D0 FA     BNE -6   --> &9211
9217  6   165 054     A5 36     LDA &36
9219      240 152     F0 98     BEQ -104   --> &91B3
921B      160 000     A0 00     LDY#&00
921D      185 000 006 B9 00 06  LDA &0600,Y
9220      032 152 189 20 98 BD  JSR &BD98
9223      200         C8        INY
9224  6   196 054     C4 36     CPY &36
9226      208 245     D0 F5     BNE -11   --> &921D
9228      128 137     80 89     BRA -119   --> &91B3
922A L    076 246 142 4C F6 8E  JMP &8EF6
922D  *   165 042     A5 2A     LDA &2A
922F H    072         48        PHA
9230      032 167 150 20 A7 96  JSR &96A7
9233      169 031     A9 1F     LDA#&1F
9235      032 238 255 20 EE FF  JSR &FFEE
9238 h    104         68        PLA
9239      032 238 255 20 EE FF  JSR &FFEE
923C  @   032 064 152 20 40 98  JSR &9840
923F  )   128 041     80 29     BRA 41   --> &926A
9241      032 175 150 20 AF 96  JSR &96AF
9244      032 235 142 20 EB 8E  JSR &8EEB
9247      240 228     F0 E4     BEQ -28   --> &922D
9249  )   201 041     C9 29     CMP#&29
924B      208 221     D0 DD     BNE -35   --> &922A
924D  *   165 042     A5 2A     LDA &2A
924F      229 030     E5 1E     SBC &1E
9251      240 023     F0 17     BEQ 23   --> &926A
9253      170         AA        TAX
```

```
9254      176 012     B0 0C     BCS 12   --> &9262
9256      032 146 186 20 92 BA  JSR &BA92
9259      128 003     80 03     BRA 3   --> &925E
925B      032 180 150 20 B4 96  JSR &96B4
925E  *   166 042     A6 2A     LDX &2A
9260      240 008     F0 08     BEQ 8   --> &926A
9262      032 191 189 20 BF BD  JSR &BDBF
9265      128 003     80 03     BRA 3   --> &926A
9267      032 146 186 20 92 BA  JSR &BA92
926A      024         18        CLC
926B      128 008     80 08     BRA 8   --> &9275
926D      198 010     C6 0A     DEC &0A
926F  /   032 047 157 20 2F 9D  JSR &9D2F
9272      032 191 150 20 BF 96  JSR &96BF
9275      164 027     A4 1B     LDY &1B
9277      132 010     84 0A     STY &0A
9279 `    096         60        RTS
927A      166 011     A6 0B     LDX &0B
927C      134 025     86 19     STX &19
927E      166 012     A6 0C     LDX &0C
9280      134 026     86 1A     STX &1A
9282      166 010     A6 0A     LDX &0A
9284      134 027     86 1B     STX &1B
9286  '   201 039     C9 27     CMP#&27
9288      240 221     F0 DD     BEQ -35   --> &9267
928A      201 138     C9 8A     CMP#&8A
928C      240 179     F0 B3     BEQ -77   --> &9241
928E      201 137     C9 89     CMP#&89
9290      240 201     F0 C9     BEQ -55   --> &925B
9292 8    056         38        SEC
9293 `    096         60        RTS
9294      000         00        BRK
9295      009         09        EQUB &09
9296      141         BD        xxx Invalid Code
9297 "    034         22        xxx Invalid Code
9298      000         00        EQUB &00
9299      032 224 142 20 E0 8E  JSR &8EE0
929C  z   032 122 146 20 7A 92  JSR &927A
929F      144 242     90 F2     BCC -14   --> &9293
92A1  "   201 034     C9 22     CMP#&22
92A3      208 237     D0 ED     BNE -19   --> &9292
92A5      200         C8        INY
92A6      177 025     B1 19     LDA (&19),Y
```

```
92A8      201 013     C9 0D     CMP#&0D
92AA      240 232     F0 E8     BEQ -24   --> &9294
92AC  "   201 034     C9 22     CMP#&22
92AE      208 009     D0 09     BNE 9   --> &92B9
92B0      200         C8        INY
92B1      132 027     84 1B     STY &1B
92B3      177 025     B1 19     LDA (&19),Y
92B5  "   201 034     C9 22     CMP#&22
92B7      208 177     D0 B1     BNE -79   --> &926A
92B9      032 152 189 20 98 BD  JSR &BD98
92BC      128 231     80 E7     BRA -25   --> &92A5
92BE  /   032 047 157 20 2F 9D  JSR &9D2F
92C1      032 188 150 20 BC 96  JSR &96BC
92C4  &   032 038 188 20 26 BC  JSR &BC26
92C7      156 000 006 9C 00 06  STZ &0600
92CA      160 000     A0 00     LDY#&00
92CC Z    090         5A        PHY
92CD      032 235 142 20 EB 8E  JSR &8EEB
92D0      208 031     D0 1F     BNE 31   --> &92F1
92D2      164 027     A4 1B     LDY &1B
92D4      032 001 153 20 01 99  JSR &9901
92D7  (   240 040     F0 28     BEQ 40   --> &9301
92D9 z    122         7A        PLY
92DA      200         C8        INY
92DB  *   165 042     A5 2A     LDA &2A
92DD      153 000 006 99 00 06  STA &0600,Y
92E0      200         C8        INY
92E1  +   165 043     A5 2B     LDA &2B
92E3      153 000 006 99 00 06  STA &0600,Y
92E6      200         C8        INY
92E7  ,   165 044     A5 2C     LDA &2C
92E9      153 000 006 99 00 06  STA &0600,Y
92EC      238 000 006 EE 00 06  INC &0600
92EF      128 219     80 DB     BRA -37   --> &92CC
92F1 z    122         7A        PLY
92F2      198 027     C6 1B     DEC &1B
92F4      032 150 155 20 96 9B  JSR &9B96
92F7      032 230 188 20 E6 BC  JSR &BCE6
92FA      032 004 147 20 04 93  JSR &9304
92FD      216         D8        CLD
92FE L    076 005 144 4C 05 90  JMP &9005
9301 L    076 140 173 4C 8C AD  JMP &AD8C
9304      173 012 004 AD 0C 04  LDA &040C
```

```
9307 J    074         4A        LSR A
9308      173 004 004 AD 04 04  LDA &0404
930B `    174 096 004 AE 60 04  LDX &0460
930E d    172 100 004 AC 64 04  LDY &0464
9311 l*   108 042 000 6C 2A 00  JMP (&002A)
9314 Li   076 105 155 4C 69 9B  JMP &9B69
9317      032 030 155 20 1E 9B  JSR &9B1E
931A      144 248     90 F8     BCC -8   --> &9314
931C  &   032 038 188 20 26 BC  JSR &BC26
931F      032 229 140 20 E5 8C  JSR &8CE5
9322      208 240     D0 F0     BNE -16   --> &9314
9324      032 030 155 20 1E 9B  JSR &9B1E
9327      144 235     90 EB     BCC -21   --> &9314
9329      032 166 155 20 A6 9B  JSR &9BA6
932C  *   165 042     A5 2A     LDA &2A
932E  9   133 057     85 39     STA &39
9330  +   165 043     A5 2B     LDA &2B
9332  :   133 058     85 3A     STA &3A
9334      032 230 188 20 E6 BC  JSR &BCE6
9337      032 152 186 20 98 BA  JSR &BA98
933A      032 202 155 20 CA 9B  JSR &9BCA
933D      032 239 190 20 EF BE  JSR &BEEF
9340  9   165 057     A5 39     LDA &39
9342  *   197 042     C5 2A     CMP &2A
9344  :   165 058     A5 3A     LDA &3A
9346  +   229 043     E5 2B     SBC &2B
9348      176 237     B0 ED     BCS -19   --> &9337
934A L    076 131 143 4C 83 8F  JMP &8F83
934D      169 010     A9 0A     LDA#&0A
934F      032 024 174 20 18 AE  JSR &AE18
9352      032 030 155 20 1E 9B  JSR &9B1E
9355  &   032 038 188 20 26 BC  JSR &BC26
9358      169 010     A9 0A     LDA#&0A
935A      032 024 174 20 18 AE  JSR &AE18
935D      032 229 140 20 E5 8C  JSR &8CE5
9360      208 014     D0 0E     BNE 14   --> &9370
9362      032 030 155 20 1E 9B  JSR &9B1E
9365  +   165 043     A5 2B     LDA &2B
9367  S   208 083     D0 53     BNE 83   --> &93BC
9369  *   165 042     A5 2A     LDA &2A
936B  O   240 079     F0 4F     BEQ 79   --> &93BC
936D L    076 166 155 4C A6 9B  JMP &9BA6
9370 L    076 176 155 4C B0 9B  JMP &9BB0
```

```
9373      165 018     A5 12     LDA &12
9375  ;   133 059     85 3B     STA &3B
9377      165 019     A5 13     LDA &13
9379  <   133 060     85 3C     STA &3C
937B      165 024     A5 18     LDA &18
937D  8   133 056     85 38     STA &38
937F      160 001     A0 01     LDY#&01
9381  7   132 055     84 37     STY &37
9383 `    096         60        RTS
9384  M   032 077 147 20 4D 93  JSR &934D
9387  9   162 057     A2 39     LDX#&39
9389      032 008 189 20 08 BD  JSR &BD08
938C      032 229 189 20 E5 BD  JSR &BDE5
938F  s   032 115 147 20 73 93  JSR &9373
9392  7   178 055     B2 37     LDA (&37)
9394 0.   048 046     30 2E     BMI 46   --> &93C4
9396  ;   146 059     92 3B     STA (&3B)
9398  7   177 055     B1 37     LDA (&37),Y
939A  ;   145 059     91 3B     STA (&3B),Y
939C 8    056         38        SEC
939D      152         98        TYA
939E e;   101 059     65 3B     ADC &3B
93A0  ;   133 059     85 3B     STA &3B
93A2      144 002     90 02     BCC 2   --> &93A6
93A4  <   230 060     E6 3C     INC &3C
93A6      197 006     C5 06     CMP &06
93A8  <   165 060     A5 3C     LDA &3C
93AA      229 007     E5 07     SBC &07
93AC      176 005     B0 05     BCS 5   --> &93B3
93AE  z   032 122 148 20 7A 94  JSR &947A
93B1      128 223     80 DF     BRA -33   --> &9392
93B3      000         00        BRK
93B4      000         00        EQUB &00
93B5   s  204 032 115 CC 20 73  CPY &7320
93B8 pa   112 097     70 61     BVS 97   --> &941B
93BA c    099         63        xxx Invalid Code
93BB e    101         65        xxx Invalid Code
93BC      000         00        BRK
93BD      000         00        EQUB &00
93BE S    083         53        xxx Invalid Code
93BF il   105 108     69 6C     ADC#&6C
93C1 ly   108 121     6C 79     xxx Invalid Code
93C3      000         00        EQUB &00
```

```
93C4  {   032 123 147 20 7B 93  JSR &937B
93C7  7   178 055     B2 37     LDA (&37)
93C9 0    048 028     30 1C     BMI 28   --> &93E7
93CB  :   165 058     A5 3A     LDA &3A
93CD  7   146 055     92 37     STA (&37)
93CF  9   165 057     A5 39     LDA &39
93D1  7   145 055     91 37     STA (&37),Y
93D3      024         18        CLC
93D4  9   165 057     A5 39     LDA &39
93D6 e*   101 042     65 2A     ADC &2A
93D8  9   133 057     85 39     STA &39
93DA      169 000     A9 00     LDA#&00
93DC e:   101 058     65 3A     ADC &3A
93DE )    041 127     29 7F     AND#&7F
93E0  :   133 058     85 3A     STA &3A
93E2  z   032 122 148 20 7A 94  JSR &947A
93E5      128 224     80 E0     BRA -32   --> &93C7
93E7      165 024     A5 18     LDA &18
93E9      133 012     85 0C     STA &0C
93EB d    100 011     64 0B     STZ &0B
93ED      160 001     A0 01     LDY#&01
93EF      177 011     B1 0B     LDA (&0B),Y
93F1 0g   048 103     30 67     BMI 103   --> &945A
93F3      160 004     A0 04     LDY#&04
93F5 d,   100 044     64 2C     STZ &2C
93F7      177 011     B1 0B     LDA (&0B),Y
93F9  ,   166 044     A6 2C     LDX &2C
93FB      208 008     D0 08     BNE 8   --> &9405
93FD      201 141     C9 8D     CMP#&8D
93FF      240 026     F0 1A     BEQ 26   --> &941B
9401      201 244     C9 F4     CMP#&F4
9403      240 013     F0 0D     BEQ 13   --> &9412
9405      200         C8        INY
9406  "   201 034     C9 22     CMP#&22
9408      208 004     D0 04     BNE 4   --> &940E
940A E,   069 044     45 2C     EOR &2C
940C  ,   133 044     85 2C     STA &2C
940E      201 013     C9 0D     CMP#&0D
9410      208 229     D0 E5     BNE -27   --> &93F7
9412      160 003     A0 03     LDY#&03
9414      177 011     B1 0B     LDA (&0B),Y
9416      032 244 155 20 F4 9B  JSR &9BF4
9419      128 210     80 D2     BRA -46   --> &93ED
```

```
941B  *   032 042 155 20 2A 9B  JSR &9B2A
941E  s   032 115 147 20 73 93  JSR &9373
9421  7   178 055     B2 37     LDA (&37)
9423 07   048 055     30 37     BMI 55   --> &945C
9425  ;   178 059     B2 3B     LDA (&3B)
9427  +   197 043     C5 2B     CMP &2B
9429      208 031     D0 1F     BNE 31   --> &944A
942B  ;   177 059     B1 3B     LDA (&3B),Y
942D  *   197 042     C5 2A     CMP &2A
942F      208 025     D0 19     BNE 25   --> &944A
9431  7   177 055     B1 37     LDA (&37),Y
9433  =   133 061     85 3D     STA &3D
9435  7   178 055     B2 37     LDA (&37)
9437      170         AA        TAX
9438      164 010     A4 0A     LDY &0A
943A      136         88        DEY
943B      165 011     A5 0B     LDA &0B
943D  9   133 057     85 39     STA &39
943F      165 012     A5 0C     LDA &0C
9441  :   133 058     85 3A     STA &3A
9443  b   032 098 141 20 62 8D  JSR &8D62
9446      164 010     A4 0A     LDY &0A
9448      128 173     80 AD     BRA -83   --> &93F7
944A      024         18        CLC
944B  z   032 122 148 20 7A 94  JSR &947A
944E  ;   165 059     A5 3B     LDA &3B
9450 i    105 002     69 02     ADC#&02
9452  ;   133 059     85 3B     STA &3B
9454      144 203     90 CB     BCC -53   --> &9421
9456  <   230 060     E6 3C     INC &3C
9458      128 199     80 C7     BRA -57   --> &9421
945A  Z   128 090     80 5A     BRA 90   --> &94B6
945C      032 207 190 20 CF BE  JSR &BECF
945F Fa   070 097     46 61     LSR &61
9461 il   105 108     69 6C     ADC#&6C
9463 ed   101 100     65 64     ADC &64
9465  at  032 097 116 20 61 74  JSR &7461
9468      032 177 011 20 B1 0B  JSR &0BB1
946B  +   133 043     85 2B     STA &2B
946D      200         C8        INY
946E      177 011     B1 0B     LDA (&0B),Y
9470  *   133 042     85 2A     STA &2A
9472      032 129 160 20 81 A0  JSR &A081
```

```
9475      032 146 186 20 92 BA  JSR &BA92
9478      128 204     80 CC     BRA -52   --> &9446
947A      200         C8        INY
947B  7   177 055     B1 37     LDA (&37),Y
947D      160 001     A0 01     LDY#&01
947F e7   101 055     65 37     ADC &37
9481  7   133 055     85 37     STA &37
9483      144 003     90 03     BCC 3   --> &9488
9485  8   230 056     E6 38     INC &38
9487      024         18        CLC
9488 `    096         60        RTS
9489  M   032 077 147 20 4D 93  JSR &934D
948C  *   165 042     A5 2A     LDA &2A
948E H    072         48        PHA
948F      032 230 188 20 E6 BC  JSR &BCE6
9492  &   032 038 188 20 26 BC  JSR &BC26
9495      032 133 160 20 85 A0  JSR &A085
9498  t   032 116 186 20 74 BA  JSR &BA74
949B      032 230 188 20 E6 BC  JSR &BCE6
949E      032 231 141 20 E7 8D  JSR &8DE7
94A1      160 000     A0 00     LDY#&00
94A3      032 008 187 20 08 BB  JSR &BB08
94A6      032 172 187 20 AC BB  JSR &BBAC
94A9 h    104         68        PLA
94AA H    072         48        PHA
94AB      024         18        CLC
94AC e*   101 042     65 2A     ADC &2A
94AE  *   133 042     85 2A     STA &2A
94B0      144 224     90 E0     BCC -32   --> &9492
94B2  +   230 043     E6 2B     INC &2B
94B4      016 220     10 DC     BPL -36   --> &9492
94B6 L    076 131 143 4C 83 8F  JMP &8F83
94B9 L    076 005 150 4C 05 96  JMP &9605
94BC      198 010     C6 0A     DEC &0A
94BE      032 174 152 20 AE 98  JSR &98AE
94C1  i   240 105     F0 69     BEQ 105   --> &952C
94C3  g   176 103     B0 67     BCS 103   --> &952C
94C5  C   032 067 188 20 43 BC  JSR &BC43
94C8      032 175 150 20 AF 96  JSR &96AF
94CB      032 239 190 20 EF BE  JSR &BEEF
94CE  -   165 045     A5 2D     LDA &2D
94D0  ,   005 044     05 2C     ORA &2C
94D2  X   208 088     D0 58     BNE 88   --> &952C
```

```
94D4      024         18        CLC
94D5  *   165 042     A5 2A     LDA &2A
94D7 e    101 002     65 02     ADC &02
94D9      168         A8        TAY
94DA  +   165 043     A5 2B     LDA &2B
94DC e    101 003     65 03     ADC &03
94DE      170         AA        TAX
94DF      196 004     C4 04     CPY &04
94E1      229 005     E5 05     SBC &05
94E3      176 212     B0 D4     BCS -44   --> &94B9
94E5      165 002     A5 02     LDA &02
94E7  *   133 042     85 2A     STA &2A
94E9      165 003     A5 03     LDA &03
94EB  +   133 043     85 2B     STA &2B
94ED      132 002     84 02     STY &02
94EF      134 003     86 03     STX &03
94F1  @   169 064     A9 40     LDA#&40
94F3  '   133 039     85 27     STA &27
94F5  +   032 043 179 20 2B B3  JSR &B32B
94F8  u   032 117 146 20 75 92  JSR &9275
94FB      032 229 140 20 E5 8C  JSR &8CE5
94FE  4   240 052     F0 34     BEQ 52   --> &9534
9500 L    076 000 144 4C 00 90  JMP &9000
9503  ?   162 063     A2 3F     LDX#&3F
9505      032 008 189 20 08 BD  JSR &BD08
9508      162 000     A2 00     LDX#&00
950A      160 000     A0 00     LDY#&00
950C F@   070 064     46 40     LSR &40
950E f?   102 063     66 3F     ROR &3F
9510      144 011     90 0B     BCC 11   --> &951D
9512      024         18        CLC
9513      152         98        TYA
9514 e*   101 042     65 2A     ADC &2A
9516      168         A8        TAY
9517      138         8A        TXA
9518 e+   101 043     65 2B     ADC &2B
951A      170         AA        TAX
951B      176 015     B0 0F     BCS 15   --> &952C
951D  *   006 042     06 2A     ASL &2A
951F &+   038 043     26 2B     ROL &2B
9521  ?   165 063     A5 3F     LDA &3F
9523  @   005 064     05 40     ORA &40
9525      208 229     D0 E5     BNE -27   --> &950C
```

```
9527  *   132 042     84 2A     STY &2A
9529  +   134 043     86 2B     STX &2B
952B  `    096         60        RTS
952C       000         00        BRK
952D       010         0A        EQUB &0A
952E B     066         42        xxx Invalid Code
952F ad    097 100 032 61 64 20  ADC (&2064,X)
9532       222         DE        xxx Invalid Code
9533       000         00        EQUB &00
9534       032 224 142 20 E0 8E  JSR &8EE0
9537       152         98        TYA
9538       024         18        CLC
9539 e     101 011     65 0B     ADC &0B
953B       166 012     A6 0C     LDX &0C
953D       144 002     90 02     BCC 2   --> &9541
953F       232         E8        INX
9540       024         18        CLC
9541       233 000     E9 00     SBC#&00
9543  7    133 055     85 37     STA &37
9545       138         8A        TXA
9546       233 000     E9 00     SBC#&00
9548  8    133 056     85 38     STA &38
954A       162 005     A2 05     LDX#&05
954C  ?    134 063     86 3F     STX &3F
954E       166 010     A6 0A     LDX &0A
9550       032 246 154 20 F6 9A  JSR &9AF6
9553       192 001     C0 01     CPY#&01
9555       240 213     F0 D5     BEQ -43   --> &952C
9557  (    201 040     C9 28     CMP#&28
9559       240 021     F0 15     BEQ 21   --> &9570
955B  $    201 036     C9 24     CMP#&24
955D       240 004     F0 04     BEQ 4   --> &9563
955F  %    201 037     C9 25     CMP#&25
9561       208 010     D0 0A     BNE 10   --> &956D
9563  ?    198 063     C6 3F     DEC &3F
9565       200         C8        INY
9566       232         E8        INX
9567  7    177 055     B1 37     LDA (&37),Y
9569  (    201 040     C9 28     CMP#&28
956B       240 003     F0 03     BEQ 3   --> &9570
956D L     076 188 148 4C BC 94  JMP &94BC
9570       200         C8        INY
9571       134 010     86 0A     STX &0A
```

```
9573      032 133 128 20 85 80  JSR &8085
9576      208 180     D0 B4     BNE -76   --> &952C
9578  T   032 084 152 20 54 98  JSR &9854
957B      162 001     A2 01     LDX#&01
957D      032 131 152 20 83 98  JSR &9883
9580  ?   165 063     A5 3F     LDA &3F
9582 H    072         48        PHA
9583      169 001     A9 01     LDA#&01
9585 H    072         48        PHA
9586      032 024 174 20 18 AE  JSR &AE18
9589  &   032 038 188 20 26 BC  JSR &BC26
958C  o   032 111 146 20 6F 92  JSR &926F
958F  +   165 043     A5 2B     LDA &2B
9591 )    041 192     29 C0     AND#&C0
9593  ,   005 044     05 2C     ORA &2C
9595  -   005 045     05 2D     ORA &2D
9597      208 147     D0 93     BNE -109   --> &952C
9599      032 239 190 20 EF BE  JSR &BEEF
959C z    122         7A        PLY
959D  *   165 042     A5 2A     LDA &2A
959F      145 002     91 02     STA (&02),Y
95A1      200         C8        INY
95A2  +   165 043     A5 2B     LDA &2B
95A4      145 002     91 02     STA (&02),Y
95A6      200         C8        INY
95A7 Z    090         5A        PHY
95A8      032 003 149 20 03 95  JSR &9503
95AB      032 229 140 20 E5 8C  JSR &8CE5
95AE      240 217     F0 D9     BEQ -39   --> &9589
95B0  )   201 041     C9 29     CMP#&29
95B2      208 194     D0 C2     BNE -62   --> &9576
95B4      250         FA        PLX
95B5 h    104         68        PLA
95B6      218         DA        PHX
95B7  ?   133 063     85 3F     STA &3F
95B9 d@   100 064     64 40     STZ &40
95BB      032 008 149 20 08 95  JSR &9508
95BE h    104         68        PLA
95BF H    072         48        PHA
95C0 e*   101 042     65 2A     ADC &2A
95C2  *   133 042     85 2A     STA &2A
95C4      144 002     90 02     BCC 2   --> &95C8
95C6  +   230 043     E6 2B     INC &2B
```

```
95C8      165 003     A5 03     LDA &03
95CA  8   133 056     85 38     STA &38
95CC      165 002     A5 02     LDA &02
95CE  7   133 055     85 37     STA &37
95D0      024         18        CLC
95D1 e*   101 042     65 2A     ADC &2A
95D3      168         A8        TAY
95D4  +   165 043     A5 2B     LDA &2B
95D6 e    101 003     65 03     ADC &03
95D8  +   176 043     B0 2B     BCS 43   --> &9605
95DA      170         AA        TAX
95DB      196 004     C4 04     CPY &04
95DD      229 005     E5 05     SBC &05
95DF  $   176 036     B0 24     BCS 36   --> &9605
95E1      132 002     84 02     STY &02
95E3      134 003     86 03     STX &03
95E5 h    104         68        PLA
95E6  7   146 055     92 37     STA (&37)
95E8 e7   101 055     65 37     ADC &37
95EA      168         A8        TAY
95EB      169 000     A9 00     LDA#&00
95ED d7   100 055     64 37     STZ &37
95EF      144 002     90 02     BCC 2   --> &95F3
95F1  8   230 056     E6 38     INC &38
95F3  7   145 055     91 37     STA (&37),Y
95F5      200         C8        INY
95F6      208 002     D0 02     BNE 2   --> &95FA
95F8  8   230 056     E6 38     INC &38
95FA      196 002     C4 02     CPY &02
95FC      208 245     D0 F5     BNE -11   --> &95F3
95FE  8   228 056     E4 38     CPX &38
9600      208 241     D0 F1     BNE -15   --> &95F3
9602 L    076 251 148 4C FB 94  JMP &94FB
9605      000         00        BRK
9606      011         0B        EQUB &0B
9607   s  222 032 115 DE 20 73  DEC &7320,X
960A pa   112 097     70 61     BVS 97   --> &966D
960C c    099         63        xxx Invalid Code
960D e    101         65        xxx Invalid Code
960E      000         00        EQUB &00
960F      032 185 150 20 B9 96  JSR &96B9
9612  *   165 042     A5 2A     LDA &2A
9614      133 006     85 06     STA &06
```

```
9616      133 004     85 04     STA &04
9618  +   165 043     A5 2B     LDA &2B
961A      133 007     85 07     STA &07
961C      133 005     85 05     STA &05
961E      128 027     80 1B     BRA 27   --> &963B
9620      032 185 150 20 B9 96  JSR &96B9
9623  *   165 042     A5 2A     LDA &2A
9625      133 000     85 00     STA &00
9627      133 002     85 02     STA &02
9629  +   165 043     A5 2B     LDA &2B
962B      133 001     85 01     STA &01
962D      133 003     85 03     STA &03
962F      032 187 187 20 BB BB  JSR &BBBB
9632      128 007     80 07     BRA 7   --> &963B
9634      032 185 150 20 B9 96  JSR &96B9
9637  +   165 043     A5 2B     LDA &2B
9639      133 024     85 18     STA &18
963B L    076 005 144 4C 05 90  JMP &9005
963E      032 166 155 20 A6 9B  JSR &9BA6
9641      032 172 187 20 AC BB  JSR &BBAC
9644      128 245     80 F5     BRA -11   --> &963B
9646      032 030 155 20 1E 9B  JSR &9B1E
9649      176 011     B0 0B     BCS 11   --> &9656
964B      201 238     C9 EE     CMP#&EE
964D      240 024     F0 18     BEQ 24   --> &9667
964F      201 135     C9 87     CMP#&87
9651      240 029     F0 1D     BEQ 29   --> &9670
9653  o   032 111 146 20 6F 92  JSR &926F
9656      032 166 155 20 A6 9B  JSR &9BA6
9659  *   165 042     A5 2A     LDA &2A
965B  !   133 033     85 21     STA &21
965D  +   165 043     A5 2B     LDA &2B
965F  "   133 034     85 22     STA &22
9661      169 255     A9 FF     LDA#&FF
9663      133 032     85 20     STA &20
9665      128 212     80 D4     BRA -44   --> &963B
9667      230 010     E6 0A     INC &0A
9669      032 166 155 20 A6 9B  JSR &9BA6
966C      169 255     A9 FF     LDA#&FF
966E      208 239     D0 EF     BNE -17   --> &965F
9670      230 010     E6 0A     INC &0A
9672      032 166 155 20 A6 9B  JSR &9BA6
9675      169 000     A9 00     LDA#&00
```

```
9677      128 234     80 EA     BRA -22   --> &9663
9679      200         C8        INY
967A      177 011     B1 0B     LDA (&0B),Y
967C  $   201 036     C9 24     CMP#&24
967E      240 014     F0 0E     BEQ 14   --> &968E
9680      032 185 150 20 B9 96  JSR &96B9
9683 d.   100 046     64 2E     STZ &2E
9685  *   162 042     A2 2A     LDX#&2A
9687      160 000     A0 00     LDY#&00
9689      169 002     A9 02     LDA#&02
968B L    076 018 179 4C 12 B3  JMP &B312
968E      230 010     E6 0A     INC &0A
9690  F   032 070 155 20 46 9B  JSR &9B46
9693  '   165 039     A5 27     LDA &27
9695  @   208 064     D0 40     BNE 64   --> &96D7
9697      169 015     A9 0F     LDA#&0F
9699  6   164 054     A4 36     LDY &36
969B      140 255 005 8C FF 05  STY &05FF
969E      162 255     A2 FF     LDX#&FF
96A0      160 005     A0 05     LDY#&05
96A2      128 231     80 E7     BRA -25   --> &968B
96A4  Q   032 081 188 20 51 BC  JSR &BC51
96A7      032 172 173 20 AC AD  JSR &ADAC
96AA      128 019     80 13     BRA 19   --> &96BF
96AC      032 241 142 20 F1 8E  JSR &8EF1
96AF  ;   032 059 157 20 3B 9D  JSR &9D3B
96B2      128 011     80 0B     BRA 11   --> &96BF
96B4  6   032 054 173 20 36 AD  JSR &AD36
96B7      128 006     80 06     BRA 6   --> &96BF
96B9  F   032 070 155 20 46 9B  JSR &9B46
96BC  '   165 039     A5 27     LDA &27
96BE      168         A8        TAY
96BF      240 022     F0 16     BEQ 22   --> &96D7
96C1      016 019     10 13     BPL 19   --> &96D6
96C3  B   032 066 130 20 42 82  JSR &8242
96C6  1   165 049     A5 31     LDA &31
96C8  -   133 045     85 2D     STA &2D
96CA  2   165 050     A5 32     LDA &32
96CC  ,   133 044     85 2C     STA &2C
96CE  3   165 051     A5 33     LDA &33
96D0  +   133 043     85 2B     STA &2B
96D2  4   165 052     A5 34     LDA &34
96D4  *   133 042     85 2A     STA &2A
```

```
96D6 `    096        60       RTS
96D7 L    076 146 144 4C 92 90  JMP &9092
96DA  6   032 054 173 20 36 AD  JSR &AD36
96DD      240 248     F0 F8     BEQ -8   --> &96D7
96DF 0    048 245     30 F5     BMI -11   --> &96D6
96E1 L    076 133 129 4C 85 81  JMP &8185
96E4      165 011     A5 0B     LDA &0B
96E6      133 025     85 19     STA &19
96E8      165 012     A5 0C     LDA &0C
96EA      133 026     85 1A     STA &1A
96EC      165 010     A5 0A     LDA &0A
96EE      133 027     85 1B     STA &1B
96F0      169 242     A9 F2     LDA#&F2
96F2      032 025 176 20 19 B0  JSR &B019
96F5      032 150 155 20 96 9B  JSR &9B96
96F8 L    076 005 144 4C 05 90  JMP &9005
96FB      160 003     A0 03     LDY#&03
96FD      169 000     A9 00     LDA#&00
96FF  *   145 042     91 2A     STA (&2A),Y
9701      240 028     F0 1C     BEQ 28   --> &971F
9703      186        BA        TSX
9704      224 252     E0 FC     CPX#&FC
9706  *   176 042     B0 2A     BCS 42   --> &9732
9708      032 174 152 20 AE 98  JSR &98AE
970B  "   240 034     F0 22     BEQ 34   --> &972F
970D      032 129 177 20 81 B1  JSR &B181
9710  ,   164 044     A4 2C     LDY &2C
9712 0    048 231     30 E7     BMI -25   --> &96FB
9714  C   032 067 188 20 43 BC  JSR &BC43
9717      032 232 171 20 E8 AB  JSR &ABE8
971A  '   133 039     85 27     STA &27
971C  +   032 043 179 20 2B B3  JSR &B32B
971F      186        BA        TSX
9720      254 006 001 FE 06 01  INC &0106,X
9723      164 027     A4 1B     LDY &1B
9725      132 010     84 0A     STY &0A
9727      032 229 140 20 E5 8C  JSR &8CE5
972A      240 215     F0 D7     BEQ -41   --> &9703
972C L    076 000 144 4C 00 90  JMP &9000
972F L    076 002 144 4C 02 90  JMP &9002
9732      000        00       BRK
9733      012        0C        EQUB &0C
9734 No   078 111     4E 6F     xxx Invalid Code
```

```
9736 t    116 032     74 20     STZ &20,X
9738      234         EA        NOP
9739      000         00        BRK
973A      025         19        EQUB &19
973B Ba   066 097     42 61     xxx Invalid Code
973D d    100 032     64 20     STZ &20
973F      235         EB        xxx Invalid Code
9740      000         00        EQUB &00
9741  o   032 111 146 20 6F 92  JSR &926F
9744  *   165 042     A5 2A     LDA &2A
9746 H    072         48        PHA
9747      032 172 150 20 AC 96  JSR &96AC
974A      032 150 155 20 96 9B  JSR &9B96
974D      169 018     A9 12     LDA#&12
974F      032 238 255 20 EE FF  JSR &FFEE
9752 h    104         68        PLA
9753  F   128 070     80 46     BRA 70   --> &979B
9755  o   032 111 146 20 6F 92  JSR &926F
9758      032 166 155 20 A6 9B  JSR &9BA6
975B      169 017     A9 11     LDA#&11
975D  <   128 060     80 3C     BRA 60   --> &979B
975F  o   032 111 146 20 6F 92  JSR &926F
9762      032 166 155 20 A6 9B  JSR &9BA6
9765  K   032 075 190 20 4B BE  JSR &BE4B
9768      232         E8        INX
9769  ,   208 044     D0 2C     BNE 44   --> &9797
976B      200         C8        INY
976C  )   208 041     D0 29     BNE 41   --> &9797
976E      165 004     A5 04     LDA &04
9770      197 006     C5 06     CMP &06
9772      208 197     D0 C5     BNE -59   --> &9739
9774      165 005     A5 05     LDA &05
9776      197 007     C5 07     CMP &07
9778      208 191     D0 BF     BNE -65   --> &9739
977A  *   166 042     A6 2A     LDX &2A
977C      169 133     A9 85     LDA#&85
977E      032 244 255 20 F4 FF  JSR &FFF4
9781      228 002     E4 02     CPX &02
9783      152         98        TYA
9784      229 003     E5 03     SBC &03
9786      144 177     90 B1     BCC -79   --> &9739
9788      228 018     E4 12     CPX &12
978A      152         98        TYA
```

```
978B      229 019     E5 13     SBC &13
978D      144 170     90 AA     BCC -86   --> &9739
978F      134 006     86 06     STX &06
9791      134 004     86 04     STX &04
9793      132 007     84 07     STY &07
9795      132 005     84 05     STY &05
9797 d    100 030     64 1E     STZ &1E
9799      169 022     A9 16     LDA#&16
979B      032 238 255 20 EE FF  JSR &FFEE
979E  *   165 042     A5 2A     LDA &2A
97A0  L   128 076     80 4C     BRA 76   --> &97EE
97A2      169 004     A9 04     LDA#&04
97A4      128 002     80 02     BRA 2   --> &97A8
97A6      169 005     A9 05     LDA#&05
97A8 H    072         48        PHA
97A9  /   032 047 157 20 2F 9D  JSR &9D2F
97AC      032 188 150 20 BC 96  JSR &96BC
97AF      128 009     80 09     BRA 9   --> &97BA
97B1  o   032 111 146 20 6F 92  JSR &926F
97B4  *   165 042     A5 2A     LDA &2A
97B6 H    072         48        PHA
97B7      032 172 150 20 AC 96  JSR &96AC
97BA  &   032 038 188 20 26 BC  JSR &BC26
97BD      032 172 150 20 AC 96  JSR &96AC
97C0      032 150 155 20 96 9B  JSR &9B96
97C3      169 025     A9 19     LDA#&19
97C5      032 238 255 20 EE FF  JSR &FFEE
97C8 h    104         68        PLA
97C9      032 238 255 20 EE FF  JSR &FFEE
97CC      032 006 189 20 06 BD  JSR &BD06
97CF  7   165 055     A5 37     LDA &37
97D1      032 238 255 20 EE FF  JSR &FFEE
97D4  8   165 056     A5 38     LDA &38
97D6      032 238 255 20 EE FF  JSR &FFEE
97D9  @   032 064 152 20 40 98  JSR &9840
97DC  +   165 043     A5 2B     LDA &2B
97DE      128 014     80 0E     BRA 14   --> &97EE
97E0      032 166 155 20 A6 9B  JSR &9BA6
97E3      169 016     A9 10     LDA#&10
97E5      128 007     80 07     BRA 7   --> &97EE
97E7      032 166 155 20 A6 9B  JSR &9BA6
97EA d    100 030     64 1E     STZ &1E
97EC      169 012     A9 0C     LDA#&0C
```

```
97EE      032 238 255 20 EE FF  JSR &FFEE
97F1 L    076 005 144 4C 05 90  JMP &9005
97F4      032 166 155 20 A6 9B  JSR &9BA6
97F7      032 146 186 20 92 BA  JSR &BA92
97FA      160 001     A0 01     LDY#&01
97FC      177 253     B1 FD     LDA (&FD),Y
97FE      240 241     F0 F1     BEQ -15   --> &97F1
9800  7   032 055 189 20 37 BD  JSR &BD37
9803      200         C8        INY
9804      208 246     D0 F6     BNE -10   --> &97FC
9806      128 233     80 E9     BRA -23   --> &97F1
9808  +   165 043     A5 2B     LDA &2B
980A      032 238 255 20 EE FF  JSR &FFEE
980D      032 224 142 20 E0 8E  JSR &8EE0
9810  :   201 058     C9 3A     CMP#&3A
9812  )   240 041     F0 29     BEQ 41    --> &983D
9814      201 013     C9 0D     CMP#&0D
9816  %   240 037     F0 25     BEQ 37    --> &983D
9818      201 139     C9 8B     CMP#&8B
981A  !   240 033     F0 21     BEQ 33    --> &983D
981C      198 010     C6 0A     DEC &0A
981E  o   032 111 146 20 6F 92  JSR &926F
9821  @   032 064 152 20 40 98  JSR &9840
9824      032 229 140 20 E5 8C  JSR &8CE5
9827      240 228     F0 E4     BEQ -28   --> &980D
9829  ;   201 059     C9 3B     CMP#&3B
982B      240 219     F0 DB     BEQ -37   --> &9808
982D  |   201 124     C9 7C     CMP#&7C
982F      208 223     D0 DF     BNE -33   --> &9810
9831      169 000     A9 00     LDA#&00
9833      160 009     A0 09     LDY#&09
9835      032 238 255 20 EE FF  JSR &FFEE
9838      136         88        DEY
9839      208 250     D0 FA     BNE -6    --> &9835
983B      128 208     80 D0     BRA -48   --> &980D
983D L    076 000 144 4C 00 90  JMP &9000
9840  *   165 042     A5 2A     LDA &2A
9842 l    108 014 002 6C 0E 02  JMP (&020E)
9845      160 001     A0 01     LDY#&01
9847  7   177 055     B1 37     LDA (&37),Y
9849      170         AA        TAX
984A      169 246     A9 F6     LDA#&F6
984C      224 242     E0 F2     CPX#&F2
```

```
984E      240 009     F0 09     BEQ 9   --> &9859
9850      169 248     A9 F8     LDA#&F8
9852      128 005     80 05     BRA 5   --> &9859
9854      160 001     A0 01     LDY#&01
9856  7   177 055     B1 37     LDA (&37),Y
9858      010         0A        ASL A
9859      162 004     A2 04     LDX#&04
985B  :   133 058     85 3A     STA &3A
985D  ;   134 059     86 3B     STX &3B
985F  :   177 058     B1 3A     LDA (&3A),Y
9861      240 005     F0 05     BEQ 5   --> &9868
9863      170         AA        TAX
9864  :   178 058     B2 3A     LDA (&3A)
9866      128 243     80 F3     BRA -13   --> &985B
9868      165 003     A5 03     LDA &03
986A  :   145 058     91 3A     STA (&3A),Y
986C      165 002     A5 02     LDA &02
986E  :   146 058     92 3A     STA (&3A)
9870      169 000     A9 00     LDA#&00
9872      145 002     91 02     STA (&02),Y
9874      200         C8        INY
9875  9   196 057     C4 39     CPY &39
9877  1   240 049     F0 31     BEQ 49   --> &98AA
9879  7   177 055     B1 37     LDA (&37),Y
987B      145 002     91 02     STA (&02),Y
987D      200         C8        INY
987E  9   196 057     C4 39     CPY &39
9880      208 247     D0 F7     BNE -9   --> &9879
9882  `   096         60        RTS
9883      169 000     A9 00     LDA#&00
9885      145 002     91 02     STA (&02),Y
9887      200         C8        INY
9888      202         CA        DEX
9889      208 250     D0 FA     BNE -6   --> &9885
988B      024         18        CLC
988C      152         98        TYA
988D e    101 002     65 02     ADC &02
988F      144 002     90 02     BCC 2   --> &9893
9891      230 003     E6 03     INC &03
9893      164 003     A4 03     LDY &03
9895      196 005     C4 05     CPY &05
9897      144 015     90 0F     BCC 15   --> &98A8
9899      208 004     D0 04     BNE 4   --> &989F
```

```
989B      197 004     C5 04     CMP &04
989D      144 009     90 09     BCC 9   --> &98A8
989F      169 000     A9 00     LDA#&00
98A1      160 001     A0 01     LDY#&01
98A3  :   145 058     91 3A     STA (&3A),Y
98A5 L    076 161 144 4C A1 90  JMP &90A1
98A8      133 002     85 02     STA &02
98AA `    096         60        RTS
98AB      032 131 152 20 83 98  JSR &9883
98AE      032 245 152 20 F5 98  JSR &98F5
98B1      208 029     D0 1D     BNE 29   --> &98D0
98B3      176 027     B0 1B     BCS 27   --> &98D0
98B5  T   032 084 152 20 54 98  JSR &9854
98B8      162 005     A2 05     LDX#&05
98BA  ,   228 044     E4 2C     CPX &2C
98BC      208 237     D0 ED     BNE -19   --> &98AB
98BE      232         E8        INX
98BF      128 234     80 EA     BRA -22   --> &98AB
98C1  !   201 033     C9 21     CMP#&21
98C3      240 012     F0 0C     BEQ 12   --> &98D1
98C5  $   201 036     C9 24     CMP#&24
98C7      240 019     F0 13     BEQ 19   --> &98DC
98C9 I?   073 063     49 3F     EOR#&3F
98CB      240 006     F0 06     BEQ 6   --> &98D3
98CD      169 000     A9 00     LDA#&00
98CF 8    056         38        SEC
98D0 `    096         60        RTS
98D1      169 004     A9 04     LDA#&04
98D3 H    072         48        PHA
98D4      230 027     E6 1B     INC &1B
98D6      032 180 150 20 B4 96  JSR &96B4
98D9 L    076 206 153 4C CE 99  JMP &99CE
98DC      230 027     E6 1B     INC &1B
98DE      032 180 150 20 B4 96  JSR &96B4
98E1  +   165 043     A5 2B     LDA &2B
98E3      240 006     F0 06     BEQ 6   --> &98EB
98E5      169 128     A9 80     LDA#&80
98E7  ,   133 044     85 2C     STA &2C
98E9 8    056         38        SEC
98EA `    096         60        RTS
98EB      000         00        BRK
98EC      008         08        EQUB &08
98ED $    036 032     24 20     BIT &20
```

```
98EF ra   114 097     72 61     ADC (&61)
98F1 nge  110 103 101 6E 67 65  ROR &6567
98F4      000         00        EQUB &00
98F5      165 011     A5 0B     LDA &0B
98F7      133 025     85 19     STA &19
98F9      165 012     A5 0C     LDA &0C
98FB      133 026     85 1A     STA &1A
98FD      164 010     A4 0A     LDY &0A
98FF      136         88        DEY
9900      200         C8        INY
9901      132 027     84 1B     STY &1B
9903      177 025     B1 19     LDA (&19),Y
9905      201 032     C9 20     CMP#&20
9907      240 247     F0 F7     BEQ -9   --> &9900
9909  @   201 064     C9 40     CMP#&40
990B      144 180     90 B4     BCC -76   --> &98C1
990D  [   201 091     C9 5B     CMP#&5B
990F      176 026     B0 1A     BCS 26   --> &992B
9911      010         0A        ASL A
9912      010         0A        ASL A
9913  *   133 042     85 2A     STA &2A
9915      200         C8        INY
9916      177 025     B1 19     LDA (&19),Y
9918  %   201 037     C9 25     CMP#&25
991A      208 015     D0 0F     BNE 15   --> &992B
991C      169 004     A9 04     LDA#&04
991E  +   133 043     85 2B     STA &2B
9920      162 004     A2 04     LDX#&04
9922  ,   134 044     86 2C     STX &2C
9924      200         C8        INY
9925      177 025     B1 19     LDA (&19),Y
9927  (   201 040     C9 28     CMP#&28
9929  m   208 109     D0 6D     BNE 109   --> &9998
992B      162 005     A2 05     LDX#&05
992D  ,   134 044     86 2C     STX &2C
992F      024         18        CLC
9930      164 026     A4 1A     LDY &1A
9932      165 027     A5 1B     LDA &1B
9934      170         AA        TAX
9935      208 008     D0 08     BNE 8   --> &993F
9937 :    058         3A        DEC A
9938 e    101 025     65 19     ADC &19
993A      176 009     B0 09     BCS 9   --> &9945
```

```
993C      136         88        DEY
993D      128 006     80 06     BRA 6   --> &9945
993F :    058         3A        DEC A
9940 e    101 025     65 19     ADC &19
9942      144 001     90 01     BCC 1   --> &9945
9944      200         C8        INY
9945  7   133 055     85 37     STA &37
9947  8   132 056     84 38     STY &38
9949      160 001     A0 01     LDY#&01
994B  7   177 055     B1 37     LDA (&37),Y
994D  A   201 065     C9 41     CMP#&41
994F      176 026     B0 1A     BCS 26   --> &996B
9951  0   201 048     C9 30     CMP#&30
9953  "   144 034     90 22     BCC 34   --> &9977
9955  :   201 058     C9 3A     CMP#&3A
9957      176 030     B0 1E     BCS 30   --> &9977
9959      232         E8        INX
995A      200         C8        INY
995B  7   177 055     B1 37     LDA (&37),Y
995D  A   201 065     C9 41     CMP#&41
995F      176 010     B0 0A     BCS 10   --> &996B
9961  0   201 048     C9 30     CMP#&30
9963      144 018     90 12     BCC 18   --> &9977
9965  :   201 058     C9 3A     CMP#&3A
9967      144 240     90 F0     BCC -16   --> &9959
9969      128 012     80 0C     BRA 12   --> &9977
996B  [   201 091     C9 5B     CMP#&5B
996D      144 234     90 EA     BCC -22   --> &9959
996F  _   201 095     C9 5F     CMP#&5F
9971      144 004     90 04     BCC 4   --> &9977
9973  {   201 123     C9 7B     CMP#&7B
9975      144 226     90 E2     BCC -30   --> &9959
9977      192 001     C0 01     CPY#&01
9979  +   240 043     F0 2B     BEQ 43   --> &99A6
997B  $   201 036     C9 24     CMP#&24
997D  [   240 091     F0 5B     BEQ 91   --> &99DA
997F  %   201 037     C9 25     CMP#&25
9981      208 006     D0 06     BNE 6   --> &9989
9983  ,   198 044     C6 2C     DEC &2C
9985      232         E8        INX
9986      200         C8        INY
9987  7   177 055     B1 37     LDA (&37),Y
9989  (   201 040     C9 28     CMP#&28
```

```
998B  H   240 072     F0 48     BEQ 72   --> &99D5
998D      032 133 128 20 85 80  JSR &8085
9990      240 024     F0 18     BEQ 24   --> &99AA
9992      134 027     86 1B     STX &1B
9994      164 027     A4 1B     LDY &1B
9996      177 025     B1 19     LDA (&19),Y
9998  !   201 033     C9 21     CMP#&21
999A      240 018     F0 12     BEQ 18   --> &99AE
999C I?   073 063     49 3F     EOR#&3F
999E      240 016     F0 10     BEQ 16   --> &99B0
99A0      024         18        CLC
99A1      132 027     84 1B     STY &1B
99A3      169 255     A9 FF     LDA#&FF
99A5 `    096         60        RTS
99A6      169 000     A9 00     LDA#&00
99A8 8    056         38        SEC
99A9 `    096         60        RTS
99AA      169 000     A9 00     LDA#&00
99AC      024         18        CLC
99AD `    096         60        RTS
99AE      169 004     A9 04     LDA#&04
99B0 H    072         48        PHA
99B1      200         C8        INY
99B2      132 027     84 1B     STY &1B
99B4      032 160 177 20 A0 B1  JSR &B1A0
99B7      032 191 150 20 BF 96  JSR &96BF
99BA  +   165 043     A5 2B     LDA &2B
99BC H    072         48        PHA
99BD  *   165 042     A5 2A     LDA &2A
99BF H    072         48        PHA
99C0      032 180 150 20 B4 96  JSR &96B4
99C3      024         18        CLC
99C4 h    104         68        PLA
99C5 e*   101 042     65 2A     ADC &2A
99C7  *   133 042     85 2A     STA &2A
99C9 h    104         68        PLA
99CA e+   101 043     65 2B     ADC &2B
99CC  +   133 043     85 2B     STA &2B
99CE h    104         68        PLA
99CF  ,   133 044     85 2C     STA &2C
99D1      024         18        CLC
99D2      169 255     A9 FF     LDA#&FF
99D4 `    096         60        RTS
```

```
99D5      032 254 153 20 FE 99  JSR &99FE
99D8      128 186     80 BA     BRA -70   --> &9994
99DA  ,   198 044     C6 2C     DEC &2C
99DC      232         E8        INX
99DD      200         C8        INY
99DE  7   177 055     B1 37     LDA (&37),Y
99E0  (   201 040     C9 28     CMP#&28
99E2      240 013     F0 0D     BEQ 13   --> &99F1
99E4      032 133 128 20 85 80  JSR &8085
99E7      240 193     F0 C1     BEQ -63   --> &99AA
99E9      134 027     86 1B     STX &1B
99EB      169 129     A9 81     LDA#&81
99ED  ,   133 044     85 2C     STA &2C
99EF 8    056         38        SEC
99F0 `    096         60        RTS
99F1      032 254 153 20 FE 99  JSR &99FE
99F4      128 245     80 F5     BRA -11   --> &99EB
99F6      000         00        BRK
99F7      014         0E        EQUB &0E
99F8 Ar   065 114     41 72     xxx Invalid Code
99FA ra   114 097     72 61     ADC (&61)
99FC y    121         79        xxx Invalid Code
99FD      000         00        EQUB &00
99FE      232         E8        INX
99FF      200         C8        INY
9A00      032 133 128 20 85 80  JSR &8085
9A03      240 241     F0 F1     BEQ -15   --> &99F6
9A05      134 027     86 1B     STX &1B
9A07  ,   165 044     A5 2C     LDA &2C
9A09 H    072         48        PHA
9A0A  *   165 042     A5 2A     LDA &2A
9A0C H    072         48        PHA
9A0D  +   165 043     A5 2B     LDA &2B
9A0F H    072         48        PHA
9A10  *   178 042     B2 2A     LDA (&2A)
9A12      201 004     C9 04     CMP#&04
9A14  o   144 111     90 6F     BCC 111   --> &9A85
9A16      032 232 171 20 E8 AB  JSR &ABE8
9A19      169 001     A9 01     LDA#&01
9A1B  -   133 045     85 2D     STA &2D
9A1D  &   032 038 188 20 26 BC  JSR &BC26
9A20      032 175 150 20 AF 96  JSR &96AF
9A23      230 027     E6 1B     INC &1B
```

```
9A25  ,   224 044     E0 2C     CPX#&2C
9A27      208 205     D0 CD     BNE -51   --> &99F6
9A29  9   162 057     A2 39     LDX#&39
9A2B      032 008 189 20 08 BD  JSR &BD08
9A2E  <   164 060     A4 3C     LDY &3C
9A30 h    104         68        PLA
9A31  8   133 056     85 38     STA &38
9A33 h    104         68        PLA
9A34  7   133 055     85 37     STA &37
9A36 H    072         48        PHA
9A37  8   165 056     A5 38     LDA &38
9A39 H    072         48        PHA
9A3A      032 211 154 20 D3 9A  JSR &9AD3
9A3D  -   132 045     84 2D     STY &2D
9A3F  7   177 055     B1 37     LDA (&37),Y
9A41  ?   133 063     85 3F     STA &3F
9A43      200         C8        INY
9A44  7   177 055     B1 37     LDA (&37),Y
9A46  @   133 064     85 40     STA &40
9A48  *   165 042     A5 2A     LDA &2A
9A4A e9   101 057     65 39     ADC &39
9A4C  *   133 042     85 2A     STA &2A
9A4E  +   165 043     A5 2B     LDA &2B
9A50 e:   101 058     65 3A     ADC &3A
9A52  +   133 043     85 2B     STA &2B
9A54      032 008 149 20 08 95  JSR &9508
9A57 8    056         38        SEC
9A58  7   178 055     B2 37     LDA (&37)
9A5A  -   229 045     E5 2D     SBC &2D
9A5C      201 003     C9 03     CMP#&03
9A5E      176 189     B0 BD     BCS -67   --> &9A1D
9A60  &   032 038 188 20 26 BC  JSR &BC26
9A63      032 167 150 20 A7 96  JSR &96A7
9A66 h    104         68        PLA
9A67  8   133 056     85 38     STA &38
9A69 h    104         68        PLA
9A6A  7   133 055     85 37     STA &37
9A6C  9   162 057     A2 39     LDX#&39
9A6E      032 008 189 20 08 BD  JSR &BD08
9A71  <   164 060     A4 3C     LDY &3C
9A73      032 211 154 20 D3 9A  JSR &9AD3
9A76      024         18        CLC
9A77  9   165 057     A5 39     LDA &39
```

```
9A79 e*   101 042     65 2A     ADC &2A
9A7B  *   133 042     85 2A     STA &2A
9A7D  :   165 058     A5 3A     LDA &3A
9A7F e+   101 043     65 2B     ADC &2B
9A81  +   133 043     85 2B     STA &2B
9A83      144 017     90 11     BCC 17   --> &9A96
9A85      032 172 173 20 AC AD  JSR &ADAC
9A88      032 191 150 20 BF 96  JSR &96BF
9A8B h    104         68        PLA
9A8C  8   133 056     85 38     STA &38
9A8E h    104         68        PLA
9A8F  7   133 055     85 37     STA &37
9A91      160 001     A0 01     LDY#&01
9A93      032 211 154 20 D3 9A  JSR &9AD3
9A96 h    104         68        PLA
9A97  ,   133 044     85 2C     STA &2C
9A99      201 005     C9 05     CMP#&05
9A9B      208 023     D0 17     BNE 23   --> &9AB4
9A9D  +   166 043     A6 2B     LDX &2B
9A9F  *   165 042     A5 2A     LDA &2A
9AA1  *   006 042     06 2A     ASL &2A
9AA3 &+   038 043     26 2B     ROL &2B
9AA5  *   006 042     06 2A     ASL &2A
9AA7 &+   038 043     26 2B     ROL &2B
9AA9 e*   101 042     65 2A     ADC &2A
9AAB  *   133 042     85 2A     STA &2A
9AAD      138         8A        TXA
9AAE e+   101 043     65 2B     ADC &2B
9AB0  +   133 043     85 2B     STA &2B
9AB2      128 008     80 08     BRA 8   --> &9ABC
9AB4  *   006 042     06 2A     ASL &2A
9AB6 &+   038 043     26 2B     ROL &2B
9AB8  *   006 042     06 2A     ASL &2A
9ABA &+   038 043     26 2B     ROL &2B
9ABC      152         98        TYA
9ABD e*   101 042     65 2A     ADC &2A
9ABF  *   133 042     85 2A     STA &2A
9AC1      144 003     90 03     BCC 3   --> &9AC6
9AC3  +   230 043     E6 2B     INC &2B
9AC5      024         18        CLC
9AC6  7   165 055     A5 37     LDA &37
9AC8 e*   101 042     65 2A     ADC &2A
9ACA  *   133 042     85 2A     STA &2A
```

```
9ACC  8   165 056     A5 38     LDA &38
9ACE e+   101 043     65 2B     ADC &2B
9AD0  +   133 043     85 2B     STA &2B
9AD2 `    096         60        RTS
9AD3  +   165 043     A5 2B     LDA &2B
9AD5 )    041 192     29 C0     AND#&C0
9AD7  ,   005 044     05 2C     ORA &2C
9AD9  -   005 045     05 2D     ORA &2D
9ADB      208 013     D0 0D     BNE 13   --> &9AEA
9ADD  *   165 042     A5 2A     LDA &2A
9ADF  7   209 055     D1 37     CMP (&37),Y
9AE1      200         C8        INY
9AE2  +   165 043     A5 2B     LDA &2B
9AE4  7   241 055     F1 37     SBC (&37),Y
9AE6      176 002     B0 02     BCS 2   --> &9AEA
9AE8      200         C8        INY
9AE9 `    096         60        RTS
9AEA      000         00        BRK
9AEB      015         0F        EQUB &0F
9AEC S    083         53        xxx Invalid Code
9AED ub   117 098     75 62     ADC &62,X
9AEF s    115         73        xxx Invalid Code
9AF0 c    099         63        xxx Invalid Code
9AF1 ri   114 105     72 69     ADC (&69)
9AF3 pt   112 116     70 74     BVS 116   --> &9B69
9AF5      000         00        EQUB &00
9AF6      160 001     A0 01     LDY#&01
9AF8  7   177 055     B1 37     LDA (&37),Y
9AFA  0   201 048     C9 30     CMP#&30
9AFC      144 024     90 18     BCC 24   --> &9B16
9AFE  @   201 064     C9 40     CMP#&40
9B00      176 012     B0 0C     BCS 12   --> &9B0E
9B02  :   201 058     C9 3A     CMP#&3A
9B04      176 016     B0 10     BCS 16   --> &9B16
9B06      192 001     C0 01     CPY#&01
9B08      240 012     F0 0C     BEQ 12   --> &9B16
9B0A      232         E8        INX
9B0B      200         C8        INY
9B0C      208 234     D0 EA     BNE -22   --> &9AF8
9B0E  _   201 095     C9 5F     CMP#&5F
9B10      176 005     B0 05     BCS 5   --> &9B17
9B12  [   201 091     C9 5B     CMP#&5B
9B14      144 244     90 F4     BCC -12   --> &9B0A
```

```
9B16 `   096       60      RTS
9B17 {   201 123   C9 7B   CMP#&7B
9B19     144 239   90 EF   BCC -17  --> &9B0A
9B1B `   096       60      RTS
9B1C     230 010   E6 0A   INC &0A
9B1E     164 010   A4 0A   LDY &0A
9B20     177 011   B1 0B   LDA (&0B),Y
9B22     201 032   C9 20   CMP#&20
9B24     240 246   F0 F6   BEQ -10  --> &9B1C
9B26     201 141   C9 8D   CMP#&8D
9B28     208 026   D0 1A   BNE 26  --> &9B44
9B2A     200       C8      INY
9B2B     177 011   B1 0B   LDA (&0B),Y
9B2D     010       0A      ASL A
9B2E     010       0A      ASL A
9B2F     170       AA      TAX
9B30 )   041 192   29 C0   AND#&C0
9B32     200       C8      INY
9B33 Q   081 011   51 0B   EOR (&0B),Y
9B35 *   133 042   85 2A   STA &2A
9B37     138       8A      TXA
9B38     010       0A      ASL A
9B39     010       0A      ASL A
9B3A     200       C8      INY
9B3B Q   081 011   51 0B   EOR (&0B),Y
9B3D +   133 043   85 2B   STA &2B
9B3F     200       C8      INY
9B40     132 010   84 0A   STY &0A
9B42 8   056       38      SEC
9B43 `   096       60      RTS
9B44     024       18      CLC
9B45 `   096       60      RTS
9B46     165 011   A5 0B   LDA &0B
9B48     133 025   85 19   STA &19
9B4A     165 012   A5 0C   LDA &0C
9B4C     133 026   85 1A   STA &1A
9B4E     165 010   A5 0A   LDA &0A
9B50     133 027   85 1B   STA &1B
9B52     164 027   A4 1B   LDY &1B
9B54     230 027   E6 1B   INC &1B
9B56     177 025   B1 19   LDA (&19),Y
9B58     201 032   C9 20   CMP#&20
9B5A     240 246   F0 F6   BEQ -10  --> &9B52
```

```
9B5C  =   201 061     C9 3D     CMP#&3D
9B5E  .   240 046     F0 2E     BEQ 46   --> &9B8E
9B60      000         00        BRK
9B61      004         04        EQUB &04
9B62 M    077         4D        xxx Invalid Code
9B63 is   105 115     69 73     ADC#&73
9B65 ta   116 097     74 61     STZ &61,X
9B67 k    107         6B        xxx Invalid Code
9B68 e    101         65        xxx Invalid Code
9B69      000         00        BRK
9B6A      016         10        EQUB &10
9B6B S    083         53        xxx Invalid Code
9B6C ynt  121 110 116 79 6E 74  ADC &746E,Y
9B6F ax   097 120 032 61 78 20  ADC (&2078,X)
9B72 er   101 114     65 72     ADC &72
9B74 ro   114 111     72 6F     ADC (&6F)
9B76 r    114         72        xxx Invalid Code
9B77      000         00        BRK
9B78      013         0D        EQUB &0D
9B79 No   078 111     4E 6F     xxx Invalid Code
9B7B      032 242     20 F2     xxx Invalid Code
9B7D      000         00        BRK
9B7E      017         11        EQUB &11
9B7F E    069         45        xxx Invalid Code
9B80 s    115         73        xxx Invalid Code
9B81 c    099         63        xxx Invalid Code
9B82 ape  097 112 101 61 70 65  ADC (&6570,X)
9B85      000         00        EQUB &00
9B86      032 213 142 20 D5 8E  JSR &8ED5
9B89  =   201 061     C9 3D     CMP#&3D
9B8B      208 211     D0 D3     BNE -45   --> &9B60
9B8D `    096         60        RTS
9B8E  ;   032 059 157 20 3B 9D  JSR &9D3B
9B91      138         8A        TXA
9B92      164 027     A4 1B     LDY &1B
9B94      128 026     80 1A     BRA 26   --> &9BB0
9B96      164 027     A4 1B     LDY &1B
9B98      128 014     80 0E     BRA 14   --> &9BA8
9B9A      186         BA        TSX
9B9B      224 252     E0 FC     CPX#&FC
9B9D      176 216     B0 D8     BCS -40   --> &9B77
9B9F      173 255 001 AD FF 01  LDA &01FF
9BA2      201 242     C9 F2     CMP#&F2
```

```
9BA4      208 209     D0 D1     BNE -47   --> &9B77
9BA6      164 010     A4 0A     LDY &0A
9BA8      136         88        DEY
9BA9      200         C8        INY
9BAA      177 011     B1 0B     LDA (&0B),Y
9BAC      201 032     C9 20     CMP#&20
9BAE      240 249     F0 F9     BEQ -7   --> &9BA9
9BB0  :   201 058     C9 3A     CMP#&3A
9BB2      240 008     F0 08     BEQ 8   --> &9BBC
9BB4      201 013     C9 0D     CMP#&0D
9BB6      240 004     F0 04     BEQ 4   --> &9BBC
9BB8      201 139     C9 8B     CMP#&8B
9BBA      208 173     D0 AD     BNE -83   --> &9B69
9BBC      024         18        CLC
9BBD      152         98        TYA
9BBE e    101 011     65 0B     ADC &0B
9BC0      133 011     85 0B     STA &0B
9BC2      144 002     90 02     BCC 2   --> &9BC6
9BC4      230 012     E6 0C     INC &0C
9BC6      160 001     A0 01     LDY#&01
9BC8      132 010     84 0A     STY &0A
9BCA $    036 255     24 FF     BIT &FF
9BCC 0    048 175     30 AF     BMI -81   --> &9B7D
9BCE `    096         60        RTS
9BCF      032 166 155 20 A6 9B  JSR &9BA6
9BD2      178 011     B2 0B     LDA (&0B)
9BD4  :   201 058     C9 3A     CMP#&3A
9BD6      240 246     F0 F6     BEQ -10   --> &9BCE
9BD8      165 012     A5 0C     LDA &0C
9BDA      201 007     C9 07     CMP#&07
9BDC  $   240 036     F0 24     BEQ 36   --> &9C02
9BDE      160 001     A0 01     LDY#&01
9BE0      177 011     B1 0B     LDA (&0B),Y
9BE2 0    048 030     30 1E     BMI 30   --> &9C02
9BE4      166 032     A6 20     LDX &20
9BE6      240 010     F0 0A     BEQ 10   --> &9BF2
9BE8  +   133 043     85 2B     STA &2B
9BEA      200         C8        INY
9BEB      177 011     B1 0B     LDA (&0B),Y
9BED  *   133 042     85 2A     STA &2A
9BEF  K   032 075 156 20 4B 9C  JSR &9C4B
9BF2      169 003     A9 03     LDA#&03
9BF4      024         18        CLC
```

```
9BF5 e    101 011     65 0B     ADC &0B
9BF7      133 011     85 0B     STA &0B
9BF9      144 002     90 02     BCC 2   --> &9BFD
9BFB      230 012     E6 0C     INC &0C
9BFD      160 001     A0 01     LDY#&01
9BFF      132 010     84 0A     STY &0A
9C01 `    096         60        RTS
9C02 L    076 134 143 4C 86 8F  JMP &8F86
9C05 L    076 146 144 4C 92 90  JMP &9092
9C08  /   032 047 157 20 2F 9D  JSR &9D2F
9C0B      240 248     F0 F8     BEQ -8   --> &9C05
9C0D      016 003     10 03     BPL 3   --> &9C12
9C0F      032 195 150 20 C3 96  JSR &96C3
9C12      164 027     A4 1B     LDY &1B
9C14      132 010     84 0A     STY &0A
9C16  *   165 042     A5 2A     LDA &2A
9C18  +   005 043     05 2B     ORA &2B
9C1A  ,   005 044     05 2C     ORA &2C
9C1C  -   005 045     05 2D     ORA &2D
9C1E      240 023     F0 17     BEQ 23   --> &9C37
9C20      224 140     E0 8C     CPX#&8C
9C22      240 003     F0 03     BEQ 3   --> &9C27
9C24 L    076 011 144 4C 0B 90  JMP &900B
9C27      230 010     E6 0A     INC &0A
9C29      032 030 155 20 1E 9B  JSR &9B1E
9C2C      144 246     90 F6     BCC -10   --> &9C24
9C2E  6   032 054 184 20 36 B8  JSR &B836
9C31      032 198 155 20 C6 9B  JSR &9BC6
9C34 L#   076 035 183 4C 23 B7  JMP &B723
9C37      164 010     A4 0A     LDY &0A
9C39      177 011     B1 0B     LDA (&0B),Y
9C3B      201 013     C9 0D     CMP#&0D
9C3D      240 009     F0 09     BEQ 9   --> &9C48
9C3F      200         C8        INY
9C40      201 139     C9 8B     CMP#&8B
9C42      208 245     D0 F5     BNE -11   --> &9C39
9C44      132 010     84 0A     STY &0A
9C46      240 225     F0 E1     BEQ -31   --> &9C29
9C48 L    076 184 143 4C B8 8F  JMP &8FB8
9C4B  *   165 042     A5 2A     LDA &2A
9C4D  !   197 033     C5 21     CMP &21
9C4F  +   165 043     A5 2B     LDA &2B
9C51  "   229 034     E5 22     SBC &22
```

```
9C53      176 172     B0 AC     BCS -84   --> &9C01
9C55  [   169 091     A9 5B     LDA#&5B
9C57      032 152 189 20 98 BD  JSR &BD98
9C5A      032 129 160 20 81 A0  JSR &A081
9C5D  ]   169 093     A9 5D     LDA#&5D
9C5F      032 152 189 20 98 BD  JSR &BD98
9C62 L    076 146 189 4C 92 BD  JMP &BD92
9C65 h    104         68        PLA
9C66  *   133 042     85 2A     STA &2A
9C68 h    104         68        PLA
9C69  +   133 043     85 2B     STA &2B
9C6B h    104         68        PLA
9C6C  ,   133 044     85 2C     STA &2C
9C6E h    104         68        PLA
9C6F  -   133 045     85 2D     STA &2D
9C71      032 250 187 20 FA BB  JSR &BBFA
9C74      032 133 129 20 85 81  JSR &8185
9C77      032 011 164 20 0B A4  JSR &A40B
9C7A      032 232 187 20 E8 BB  JSR &BBE8
9C7D  A   032 065 165 20 41 A5  JSR &A541
9C80      128 016     80 10     BRA 16   --> &9C92
9C82      032 250 187 20 FA BB  JSR &BBFA
9C85  L   032 076 158 20 4C 9E  JSR &9E4C
9C88      168         A8        TAY
9C89      032 221 150 20 DD 96  JSR &96DD
9C8C      032 232 187 20 E8 BB  JSR &BBE8
9C8F      032 224 164 20 E0 A4  JSR &A4E0
9C92      160 000     A0 00     LDY#&00
9C94      169 127     A9 7F     LDA#&7F
9C96  ;   020 059     14 3B     TRB &3B
9C98  .   165 046     A5 2E     LDA &2E
9C9A )    041 128     29 80     AND#&80
9C9C  ;   197 059     C5 3B     CMP &3B
9C9E      208 030     D0 1E     BNE 30   --> &9CBE
9CA0  <   165 060     A5 3C     LDA &3C
9CA2  0   197 048     C5 30     CMP &30
9CA4      208 025     D0 19     BNE 25   --> &9CBF
9CA6  =   165 061     A5 3D     LDA &3D
9CA8  1   197 049     C5 31     CMP &31
9CAA      208 019     D0 13     BNE 19   --> &9CBF
9CAC  >   165 062     A5 3E     LDA &3E
9CAE  2   197 050     C5 32     CMP &32
9CB0      208 013     D0 0D     BNE 13   --> &9CBF
```

```
9CB2  ?   165 063     A5 3F     LDA &3F
9CB4  3   197 051     C5 33     CMP &33
9CB6      208 007     D0 07     BNE 7   --> &9CBF
9CB8  @   165 064     A5 40     LDA &40
9CBA  4   197 052     C5 34     CMP &34
9CBC      208 001     D0 01     BNE 1   --> &9CBF
9CBE `    096         60        RTS
9CBF j    106         6A        ROR A
9CC0 E;   069 059     45 3B     EOR &3B
9CC2 *    042         2A        ROL A
9CC3      169 001     A9 01     LDA#&01
9CC5 `    096         60        RTS
9CC6 L    076 146 144 4C 92 90  JMP &9092
9CC9      138         8A        TXA
9CCA  6   240 054     F0 36     BEQ 54   --> &9D02
9CCC 0    048 180     30 B4     BMI -76   --> &9C82
9CCE  -   165 045     A5 2D     LDA &2D
9CD0 H    072         48        PHA
9CD1  ,   165 044     A5 2C     LDA &2C
9CD3 H    072         48        PHA
9CD4  +   165 043     A5 2B     LDA &2B
9CD6 H    072         48        PHA
9CD7  *   165 042     A5 2A     LDA &2A
9CD9 H    072         48        PHA
9CDA  L   032 076 158 20 4C 9E  JSR &9E4C
9CDD      168         A8        TAY
9CDE      240 230     F0 E6     BEQ -26   --> &9CC6
9CE0 0    048 131     30 83     BMI -125   --> &9C65
9CE2  -   165 045     A5 2D     LDA &2D
9CE4 I    073 128     49 80     EOR#&80
9CE6  -   133 045     85 2D     STA &2D
9CE8 8    056         38        SEC
9CE9 h    104         68        PLA
9CEA  *   229 042     E5 2A     SBC &2A
9CEC  *   133 042     85 2A     STA &2A
9CEE h    104         68        PLA
9CEF  +   229 043     E5 2B     SBC &2B
9CF1  *   004 042     04 2A     TSB &2A
9CF3 h    104         68        PLA
9CF4  ,   229 044     E5 2C     SBC &2C
9CF6  *   004 042     04 2A     TSB &2A
9CF8 h    104         68        PLA
9CF9      160 000     A0 00     LDY#&00
```

```
9CFB I    073 128     49 80     EOR#&80
9CFD  -   229 045     E5 2D     SBC &2D
9CFF  *   005 042     05 2A     ORA &2A
9D01 `    096         60        RTS
9D02  Q   032 081 188 20 51 BC  JSR &BC51
9D05  L   032 076 158 20 4C 9E  JSR &9E4C
9D08      168         A8        TAY
9D09      208 187     D0 BB     BNE -69   --> &9CC6
9D0B      178 004     B2 04     LDA (&04)
9D0D  6   197 054     C5 36     CMP &36
9D0F      144 002     90 02     BCC 2   --> &9D13
9D11  6   165 054     A5 36     LDA &36
9D13  7   133 055     85 37     STA &37
9D15  7   196 055     C4 37     CPY &37
9D17      240 010     F0 0A     BEQ 10   --> &9D23
9D19      200         C8        INY
9D1A      177 004     B1 04     LDA (&04),Y
9D1C      217 255 005 D9 FF 05  CMP &05FF,Y
9D1F      240 244     F0 F4     BEQ -12   --> &9D15
9D21      128 004     80 04     BRA 4   --> &9D27
9D23      178 004     B2 04     LDA (&04)
9D25  6   197 054     C5 36     CMP &36
9D27      008         08        PHP
9D28      032 225 188 20 E1 BC  JSR &BCE1
9D2B      160 000     A0 00     LDY#&00
9D2D (    040         28        PLP
9D2E `    096         60        RTS
9D2F      165 011     A5 0B     LDA &0B
9D31      133 025     85 19     STA &19
9D33      165 012     A5 0C     LDA &0C
9D35      133 026     85 1A     STA &1A
9D37      165 010     A5 0A     LDA &0A
9D39      133 027     85 1B     STA &1B
9D3B      032 129 157 20 81 9D  JSR &9D81
9D3E      224 132     E0 84     CPX#&84
9D40      240 010     F0 0A     BEQ 10   --> &9D4C
9D42      224 130     E0 82     CPX#&82
9D44      240 032     F0 20     BEQ 32   --> &9D66
9D46      198 027     C6 1B     DEC &1B
9D48      168         A8        TAY
9D49  '   133 039     85 27     STA &27
9D4B `    096         60        RTS
9D4C  {   032 123 157 20 7B 9D  JSR &9D7B
```

```
9D4F      032 190 150 20 BE 96  JSR &96BE
9D52      160 003     A0 03     LDY#&03
9D54      177 004     B1 04     LDA (&04),Y
9D56  *   025 042 000 19 2A 00  ORA &002A,Y
9D59  *   153 042 000 99 2A 00  STA &002A,Y
9D5C      136         88        DEY
9D5D      016 245     10 F5     BPL -11   --> &9D54
9D5F      032 250 188 20 FA BC  JSR &BCFA
9D62  @   169 064     A9 40     LDA#&40
9D64      128 216     80 D8     BRA -40   --> &9D3E
9D66  {   032 123 157 20 7B 9D  JSR &9D7B
9D69      032 190 150 20 BE 96  JSR &96BE
9D6C      160 003     A0 03     LDY#&03
9D6E      177 004     B1 04     LDA (&04),Y
9D70 Y*   089 042 000 59 2A 00  EOR &002A,Y
9D73  *   153 042 000 99 2A 00  STA &002A,Y
9D76      136         88        DEY
9D77      016 245     10 F5     BPL -11   --> &9D6E
9D79      128 228     80 E4     BRA -28   --> &9D5F
9D7B      032 190 150 20 BE 96  JSR &96BE
9D7E  &   032 038 188 20 26 BC  JSR &BC26
9D81      032 169 157 20 A9 9D  JSR &9DA9
9D84      224 128     E0 80     CPX#&80
9D86      240 001     F0 01     BEQ 1   --> &9D89
9D88 `    096         60        RTS
9D89      032 190 150 20 BE 96  JSR &96BE
9D8C  &   032 038 188 20 26 BC  JSR &BC26
9D8F      032 169 157 20 A9 9D  JSR &9DA9
9D92      032 190 150 20 BE 96  JSR &96BE
9D95      160 003     A0 03     LDY#&03
9D97      177 004     B1 04     LDA (&04),Y
9D99 9*   057 042 000 39 2A 00  AND &002A,Y
9D9C  *   153 042 000 99 2A 00  STA &002A,Y
9D9F      136         88        DEY
9DA0      016 245     10 F5     BPL -11   --> &9D97
9DA2      032 250 188 20 FA BC  JSR &BCFA
9DA5  @   169 064     A9 40     LDA#&40
9DA7      128 219     80 DB     BRA -37   --> &9D84
9DA9  L   032 076 158 20 4C 9E  JSR &9E4C
9DAC  ?   224 063     E0 3F     CPX#&3F
9DAE      176 004     B0 04     BCS 4   --> &9DB4
9DB0  <   224 060     E0 3C     CPX#&3C
9DB2      176 001     B0 01     BCS 1   --> &9DB5
```

```
9DB4 `   096       60       RTS
9DB5     240 022    F0 16    BEQ 22   --> &9DCD
9DB7 >   224 062    E0 3E    CPX#&3E
9DB9 :   240 058    F0 3A    BEQ 58   --> &9DF5
9DBB     170        AA       TAX
9DBC     032 202 156 20 CA 9C  JSR &9CCA
9DBF     208 001    D0 01    BNE 1   --> &9DC2
9DC1     136        88       DEY
9DC2 *   132 042    84 2A    STY &2A
9DC4 +   132 043    84 2B    STY &2B
9DC6 ,   132 044    84 2C    STY &2C
9DC8 -   132 045    84 2D    STY &2D
9DCA @   169 064    A9 40    LDA#&40
9DCC `   096        60       RTS
9DCD     170        AA       TAX
9DCE     164 027    A4 1B    LDY &1B
9DD0     177 025    B1 19    LDA (&19),Y
9DD2 =   201 061    C9 3D    CMP#&3D
9DD4     240 011    F0 0B    BEQ 11   --> &9DE1
9DD6 >   201 062    C9 3E    CMP#&3E
9DD8     240 018    F0 12    BEQ 18   --> &9DEC
9DDA     032 201 156 20 C9 9C  JSR &9CC9
9DDD     144 226    90 E2    BCC -30   --> &9DC1
9DDF     128 225    80 E1    BRA -31   --> &9DC2
9DE1     230 027    E6 1B    INC &1B
9DE3     032 201 156 20 C9 9C  JSR &9CC9
9DE6     240 217    F0 D9    BEQ -39   --> &9DC1
9DE8     144 215    90 D7    BCC -41   --> &9DC1
9DEA     128 214    80 D6    BRA -42   --> &9DC2
9DEC     230 027    E6 1B    INC &1B
9DEE     032 201 156 20 C9 9C  JSR &9CC9
9DF1     208 206    D0 CE    BNE -50   --> &9DC1
9DF3     128 205    80 CD    BRA -51   --> &9DC2
9DF5     170        AA       TAX
9DF6     164 027    A4 1B    LDY &1B
9DF8     177 025    B1 19    LDA (&19),Y
9DFA =   201 061    C9 3D    CMP#&3D
9DFC     240 009    F0 09    BEQ 9   --> &9E07
9DFE     032 201 156 20 C9 9C  JSR &9CC9
9E01     240 191    F0 BF    BEQ -65   --> &9DC2
9E03     176 188    B0 BC    BCS -68   --> &9DC1
9E05     128 187    80 BB    BRA -69   --> &9DC2
9E07     230 027    E6 1B    INC &1B
```

```
9E09      032 201 156 20 C9 9C  JSR &9CC9
9E0C      176 179     B0 B3     BCS -77   --> &9DC1
9E0E      128 178     80 B2     BRA -78   --> &9DC2
9E10      000         00        BRK
9E11      019         13        EQUB &13
9E12 S    083         53        xxx Invalid Code
9E13 tr   116 114     74 72     STZ &72,X
9E15 in   105 110     69 6E     ADC#&6E
9E17 g    103         67        xxx Invalid Code
9E18  to  032 116 111 20 74 6F  JSR &6F74
9E1B o    111         6F        xxx Invalid Code
9E1C  lo  032 108 111 20 6C 6F  JSR &6F6C
9E1F ng   110 103     6E 67     xxx Invalid Code
9E21      000         00        EQUB &00
9E22  Q   032 081 188 20 51 BC  JSR &BC51
9E25      032 018 160 20 12 A0  JSR &A012
9E28      168         A8        TAY
9E29  f   208 102     D0 66     BNE 102   --> &9E91
9E2B      024         18        CLC
9E2C      218         DA        PHX
9E2D      178 004     B2 04     LDA (&04)
9E2F e6   101 054     65 36     ADC &36
9E31      176 221     B0 DD     BCS -35   --> &9E10
9E33      170         AA        TAX
9E34 H    072         48        PHA
9E35  6   164 054     A4 36     LDY &36
9E37      185 255 005 B9 FF 05  LDA &05FF,Y
9E3A      157 255 005 9D FF 05  STA &05FF,X
9E3D      202         CA        DEX
9E3E      136         88        DEY
9E3F      208 246     D0 F6     BNE -10   --> &9E37
9E41      032 210 188 20 D2 BC  JSR &BCD2
9E44 h    104         68        PLA
9E45  6   133 054     85 36     STA &36
9E47      250         FA        PLX
9E48      169 000     A9 00     LDA#&00
9E4A      128 003     80 03     BRA 3   --> &9E4F
9E4C      032 199 159 20 C7 9F  JSR &9FC7
9E4F  +   224 043     E0 2B     CPX#&2B
9E51      240 005     F0 05     BEQ 5   --> &9E58
9E53  -   224 045     E0 2D     CPX#&2D
9E55  f   240 102     F0 66     BEQ 102   --> &9EBD
9E57 `    096         60        RTS
```

```
9E58      168         A8        TAY
9E59      240 199     F0 C7     BEQ -57   --> &9E22
9E5B 07   048 055     30 37     BMI 55    --> &9E94
9E5D      032 196 159 20 C4 9F  JSR &9FC4
9E60      168         A8        TAY
9E61  .   240 046     F0 2E     BEQ 46    --> &9E91
9E63 0K   048 075     30 4B     BMI 75    --> &9EB0
9E65      024         18        CLC
9E66      178 004     B2 04     LDA (&04)
9E68 e*   101 042     65 2A     ADC &2A
9E6A  *   133 042     85 2A     STA &2A
9E6C      160 001     A0 01     LDY#&01
9E6E      177 004     B1 04     LDA (&04),Y
9E70 e+   101 043     65 2B     ADC &2B
9E72  +   133 043     85 2B     STA &2B
9E74      200         C8        INY
9E75      177 004     B1 04     LDA (&04),Y
9E77 e,   101 044     65 2C     ADC &2C
9E79  ,   133 044     85 2C     STA &2C
9E7B      200         C8        INY
9E7C      177 004     B1 04     LDA (&04),Y
9E7E e-   101 045     65 2D     ADC &2D
9E80  -   133 045     85 2D     STA &2D
9E82      024         18        CLC
9E83      165 004     A5 04     LDA &04
9E85 i    105 004     69 04     ADC#&04
9E87      133 004     85 04     STA &04
9E89  @   169 064     A9 40     LDA#&40
9E8B      144 194     90 C2     BCC -62   --> &9E4F
9E8D      230 005     E6 05     INC &05
9E8F      128 190     80 BE     BRA -66   --> &9E4F
9E91 L    076 146 144 4C 92 90  JMP &9092
9E94      032 250 187 20 FA BB  JSR &BBFA
9E97      032 199 159 20 C7 9F  JSR &9FC7
9E9A      168         A8        TAY
9E9B      240 244     F0 F4     BEQ -12   --> &9E91
9E9D  '   134 039     86 27     STX &27
9E9F 0    048 003     30 03     BMI 3     --> &9EA4
9EA1      032 133 129 20 85 81  JSR &8185
9EA4      032 232 187 20 E8 BB  JSR &BBE8
9EA7      032 141 166 20 8D A6  JSR &A68D
9EAA  '   166 039     A6 27     LDX &27
9EAC      169 255     A9 FF     LDA#&FF
```

```
9EAE      128 159     80 9F     BRA -97   --> &9E4F
9EB0  '   134 039     86 27     STX &27
9EB2      032 230 188 20 E6 BC  JSR &BCE6
9EB5      032 250 187 20 FA BB  JSR &BBFA
9EB8      032 133 129 20 85 81  JSR &8185
9EBB      128 231     80 E7     BRA -25   --> &9EA4
9EBD      168         A8        TAY
9EBE      240 209     F0 D1     BEQ -47   --> &9E91
9EC0 0%   048 037     30 25     BMI 37    --> &9EE7
9EC2      032 196 159 20 C4 9F  JSR &9FC4
9EC5      168         A8        TAY
9EC6      240 201     F0 C9     BEQ -55   --> &9E91
9EC8 05   048 053     30 35     BMI 53    --> &9EFF
9ECA 8    056         38        SEC
9ECB      178 004     B2 04     LDA (&04)
9ECD  *   229 042     E5 2A     SBC &2A
9ECF  *   133 042     85 2A     STA &2A
9ED1      160 001     A0 01     LDY#&01
9ED3      177 004     B1 04     LDA (&04),Y
9ED5  +   229 043     E5 2B     SBC &2B
9ED7  +   133 043     85 2B     STA &2B
9ED9      200         C8        INY
9EDA      177 004     B1 04     LDA (&04),Y
9EDC  ,   229 044     E5 2C     SBC &2C
9EDE  ,   133 044     85 2C     STA &2C
9EE0      200         C8        INY
9EE1      177 004     B1 04     LDA (&04),Y
9EE3  -   229 045     E5 2D     SBC &2D
9EE5      128 153     80 99     BRA -103  --> &9E80
9EE7      032 250 187 20 FA BB  JSR &BBFA
9EEA      032 199 159 20 C7 9F  JSR &9FC7
9EED      168         A8        TAY
9EEE      240 161     F0 A1     BEQ -95   --> &9E91
9EF0  '   134 039     86 27     STX &27
9EF2 0    048 003     30 03     BMI 3     --> &9EF7
9EF4      032 133 129 20 85 81  JSR &8185
9EF7      032 232 187 20 E8 BB  JSR &BBE8
9EFA      032 138 166 20 8A A6  JSR &A68A
9EFD      128 171     80 AB     BRA -85   --> &9EAA
9EFF  '   134 039     86 27     STX &27
9F01      032 230 188 20 E6 BC  JSR &BCE6
9F04      032 250 187 20 FA BB  JSR &BBFA
9F07      032 133 129 20 85 81  JSR &8185
```

```
9F0A      032 232 187 20 E8 BB  JSR &BBE8
9F0D      032 199 172 20 C7 AC  JSR &ACC7
9F10      128 152     80 98     BRA -104   --> &9EAA
9F12      032 133 129 20 85 81  JSR &8185
9F15      032 230 188 20 E6 BC  JSR &BCE6
9F18      032 250 187 20 FA BB  JSR &BBFA
9F1B      032 133 129 20 85 81  JSR &8185
9F1E      128 013     80 0D     BRA 13   --> &9F2D
9F20      032 133 129 20 85 81  JSR &8185
9F23      032 250 187 20 FA BB  JSR &BBFA
9F26      032 018 160 20 12 A0  JSR &A012
9F29      168         A8        TAY
9F2A      032 221 150 20 DD 96  JSR &96DD
9F2D      032 232 187 20 E8 BB  JSR &BBE8
9F30      032 166 166 20 A6 A6  JSR &A6A6
9F33      169 255     A9 FF     LDA#&FF
9F35 L    076 202 159 4C CA 9F  JMP &9FCA
9F38 L    076 146 144 4C 92 90  JMP &9092
9F3B      168         A8        TAY
9F3C      240 250     F0 FA     BEQ -6   --> &9F38
9F3E 0    048 227     30 E3     BMI -29   --> &9F23
9F40  -   164 045     A4 2D     LDY &2D
9F42  ,   196 044     C4 2C     CPY &2C
9F44      208 218     D0 DA     BNE -38   --> &9F20
9F46  +   165 043     A5 2B     LDA &2B
9F48      010         0A        ASL A
9F49      152         98        TYA
9F4A i    105 000     69 00     ADC#&00
9F4C      208 210     D0 D2     BNE -46   --> &9F20
9F4E      032 015 160 20 0F A0  JSR &A00F
9F51      168         A8        TAY
9F52      240 228     F0 E4     BEQ -28   --> &9F38
9F54 0    048 191     30 BF     BMI -65   --> &9F15
9F56  -   164 045     A4 2D     LDY &2D
9F58  ,   196 044     C4 2C     CPY &2C
9F5A      208 182     D0 B6     BNE -74   --> &9F12
9F5C  +   165 043     A5 2B     LDA &2B
9F5E      010         0A        ASL A
9F5F      152         98        TYA
9F60 i    105 000     69 00     ADC#&00
9F62      208 174     D0 AE     BNE -82   --> &9F12
9F64 Z    090         5A        PHY
9F65      032 190 172 20 BE AC  JSR &ACBE
```

```
9F68  '   134 039     86 27     STX &27
9F6A  9   162 057     A2 39     LDX#&39
9F6C      032 198 189 20 C6 BD  JSR &BDC6
9F6F      032 230 188 20 E6 BC  JSR &BCE6
9F72 h    104         68        PLA
9F73 E-   069 045     45 2D     EOR &2D
9F75  7   133 055     85 37     STA &37
9F77      032 190 172 20 BE AC  JSR &ACBE
9F7A      160 000     A0 00     LDY#&00
9F7C      162 000     A2 00     LDX#&00
9F7E d?   100 063     64 3F     STZ &3F
9F80 d@   100 064     64 40     STZ &40
9F82 F:   070 058     46 3A     LSR &3A
9F84 f9   102 057     66 39     ROR &39
9F86      144 021     90 15     BCC 21   --> &9F9D
9F88      024         18        CLC
9F89      152         98        TYA
9F8A e*   101 042     65 2A     ADC &2A
9F8C      168         A8        TAY
9F8D      138         8A        TXA
9F8E e+   101 043     65 2B     ADC &2B
9F90      170         AA        TAX
9F91  ?   165 063     A5 3F     LDA &3F
9F93 e,   101 044     65 2C     ADC &2C
9F95  ?   133 063     85 3F     STA &3F
9F97  @   165 064     A5 40     LDA &40
9F99 e-   101 045     65 2D     ADC &2D
9F9B  @   133 064     85 40     STA &40
9F9D  *   006 042     06 2A     ASL &2A
9F9F &+   038 043     26 2B     ROL &2B
9FA1 &,   038 044     26 2C     ROL &2C
9FA3 &-   038 045     26 2D     ROL &2D
9FA5  9   165 057     A5 39     LDA &39
9FA7  :   005 058     05 3A     ORA &3A
9FA9      208 215     D0 D7     BNE -41   --> &9F82
9FAB  =   132 061     84 3D     STY &3D
9FAD  >   134 062     86 3E     STX &3E
9FAF  7   165 055     A5 37     LDA &37
9FB1      008         08        PHP
9FB2  =   162 061     A2 3D     LDX#&3D
9FB4      032 128 170 20 80 AA  JSR &AA80
9FB7 (    040         28        PLP
9FB8      016 003     10 03     BPL 3   --> &9FBD
```

```
9FBA      032 222 172 20 DE AC  JSR &ACDE
9FBD  '   166 039     A6 27     LDX &27
9FBF      128 009     80 09     BRA 9   --> &9FCA
9FC1 L;   076 059 159 4C 3B 9F  JMP &9F3B
9FC4  &   032 038 188 20 26 BC  JSR &BC26
9FC7      032 018 160 20 12 A0  JSR &A012
9FCA  *   224 042     E0 2A     CPX#&2A
9FCC      240 243     F0 F3     BEQ -13   --> &9FC1
9FCE  /   224 047     E0 2F     CPX#&2F
9FD0      240 009     F0 09     BEQ 9   --> &9FDB
9FD2      224 131     E0 83     CPX#&83
9FD4      240 031     F0 1F     BEQ 31   --> &9FF5
9FD6      224 129     E0 81     CPX#&81
9FD8  #   240 035     F0 23     BEQ 35   --> &9FFD
9FDA `    096         60        RTS
9FDB      168         A8        TAY
9FDC      032 221 150 20 DD 96  JSR &96DD
9FDF      032 250 187 20 FA BB  JSR &BBFA
9FE2      032 018 160 20 12 A0  JSR &A012
9FE5  '   134 039     86 27     STX &27
9FE7      168         A8        TAY
9FE8      032 221 150 20 DD 96  JSR &96DD
9FEB      032 232 187 20 E8 BB  JSR &BBE8
9FEE      032 238 165 20 EE A5  JSR &A5EE
9FF1      169 255     A9 FF     LDA#&FF
9FF3      128 200     80 C8     BRA -56   --> &9FBD
9FF5      032 249 128 20 F9 80  JSR &80F9
9FF8  8   165 056     A5 38     LDA &38
9FFA      008         08        PHP
9FFB      128 181     80 B5     BRA -75   --> &9FB2
9FFD      032 249 128 20 F9 80  JSR &80F9
A000 &9   038 057     26 39     ROL &39
```

9000,,198 010,C6 0A,DEC &0A
9002,,032 166 155,20 A6 9B,JSR &9BA6
9005,,178 011,B2 0B,LDA (&0B)
9007,:,201 058,C9 3A,CMP#&3A
9009,,208 178,D0 B2,BNE -78 --> &8FBD
900B,,164 010,A4 0A,LDY &0A
900D,,230 010,E6 0A,INC &0A
900F,,177 011,B1 0B,"LDA (&0B),Y"
9011,,201 032,C9 20,CMP#&20
9013,,240 246,F0 F6,BEQ -10 --> &900B
9015,,201 207,C9 CF,CMP#&CF
9017,,144 012,90 0C,BCC 12 --> &9025
9019,,010,0A,ASL A
901A,,170,AA,TAX
901B,|M,124 077 135,7C 4D 87,"JMP (&874D,X)"
901E,,032 224 142,20 E0 8E,JSR &8EE0
9021,,201 198,C9 C6,CMP#&C6
9023,,176 244,B0 F4,BCS -12 --> &9019
9025,,166 011,A6 0B,LDX &0B
9027,,134 025,86 19,STX &19
9029,,166 012,A6 0C,LDX &0C
902B,,134 026,86 1A,STX &1A
902D,,132 027,84 1B,STY &1B
902F,,032 009 153,20 09 99,JSR &9909
9032,,208 027,D0 1B,BNE 27 --> &904F
9034,,176 181,B0 B5,BCS -75 --> &8FEB
9036,,134 027,86 1B,STX &1B
9038,,032 134 155,20 86 9B,JSR &9B86
903B,T,032 084 152,20 54 98,JSR &9854
903E,,162 005,A2 05,LDX#&05
9040,",",228 044,E4 2C,CPX &2C
9042,,208 001,D0 01,BNE 1 --> &9045
9044,,232,E8,INX
9045,,032 131 152,20 83 98,JSR &9883
9048,,198 010,C6 0A,DEC &0A
904A,,032 174 152,20 AE 98,JSR &98AE
904D,4,240 052,F0 34,BEQ 52 --> &9083
904F,!,144 033,90 21,BCC 33 --> &9072
9051,&,032 038 188,20 26 BC,JSR &BC26
9054,R,032 082 155,20 52 9B,JSR &9B52
9057,',165 039,A5 27,LDA &27
9059,7,208 055,D0 37,BNE 55 --> &9092

905B,,032 171 144,20 AB 90,JSR &90AB
905E,,128 165,80 A5,BRA -91 --> &9005
9060,,186,BA,TSX
9061,,224 252,E0 FC,CPX#&FC
9063,',176 039,B0 27,BCS 39 --> &908C
9065,,173 255 001,AD FF 01,LDA &01FF
9068,,201 164,C9 A4,CMP#&A4
906A,,208 032,D0 20,BNE 32 --> &908C
906C,/,032 047 157,20 2F 9D,JSR &9D2F
906F,L,076 145 155,4C 91 9B,JMP &9B91
9072,*,165 042,A5 2A,LDA &2A
9074,H,072,48,PHA
9075,+,165 043,A5 2B,LDA &2B
9077,H,072,48,PHA
9078,",",165 044,A5 2C,LDA &2C
907A,H,072,48,PHA
907B,R,032 082 155,20 52 9B,JSR &9B52
907E,+,032 043 179,20 2B B3,JSR &B32B
9081,,128 130,80 82,BRA -126 --> &9005
9083,Li,076 105 155,4C 69 9B,JMP &9B69
9086,,032 166 155,20 A6 9B,JSR &9BA6
9089,,000,00,BRK
908A,,000,00,EQUB &00
908B,,250,FA,PLX
908C,,000,00,BRK
908D,,007,07,EQUB &07
908E,No,078 111 032,4E 6F 20,LSR &206F
9091,,164,A4,xxx Invalid Code
9092,,000,00,BRK
9093,,006,06,EQUB &06
9094,T,084,54,xxx Invalid Code
9095,ype,121 112 101,79 70 65,"ADC &6570,Y"
9098,mi,032 109 105,20 6D 69,JSR &696D
909B,s,115,73,xxx Invalid Code
909C,mat,109 097 116,6D 61 74,ADC &7461
909F,c,099,63,xxx Invalid Code
90A0,h,104,68,PLA
90A1,,000,00,BRK
90A2,,000,00,EQUB &00
90A3,No,078 111 032,4E 6F 20,LSR &206F
90A6,ro,114 111,72 6F,ADC (&6F)
90A8,o,111,6F,xxx Invalid Code
90A9,m,109,6D,xxx Invalid Code

```
90AA,,000,00,EQUB &00
90AB,,032 230 188,20 E6 BC,JSR &BCE6
90AE,",",165 044,A5 2C,LDA &2C
90B0,,201 128,C9 80,CMP#&80
90B2,x,240 120,F0 78,BEQ 120   --> &912C
90B4,,160 002,A0 02,LDY#&02
90B6,*,177 042,B1 2A,"LDA (&2A),Y"
90B8,6,197 054,C5 36,CMP &36
90BA,R,176 082,B0 52,BCS 82   --> &910E
90BC,,165 002,A5 02,LDA &02
90BE,",",133 044,85 2C,STA &2C
90C0,,165 003,A5 03,LDA &03
90C2,-,133 045,85 2D,STA &2D
90C4,6,165 054,A5 36,LDA &36
90C6,,201 008,C9 08,CMP#&08
90C8,,144 006,90 06,BCC 6   --> &90D0
90CA,i,105 007,69 07,ADC#&07
90CC,,144 002,90 02,BCC 2   --> &90D0
90CE,,169 255,A9 FF,LDA#&FF
90D0,,024,18,CLC
90D1,H,072,48,PHA
90D2,,170,AA,TAX
90D3,*,177 042,B1 2A,"LDA (&2A),Y"
90D5,r*,114 042,72 2A,ADC (&2A)
90D7,E,069 002,45 02,EOR &02
90D9,,208 015,D0 0F,BNE 15   --> &90EA
90DB,,136,88,DEY
90DC,q*,113 042,71 2A,"ADC (&2A),Y"
90DE,E,069 003,45 03,"EOR &03"
90E0,,208 008,D0 08,BNE 8   --> &90EA
90E2,-,133 045,85 2D,STA &2D
90E4,,138,8A,TXA
90E5,,200,C8,INY
90E6,8,056,38,SEC
90E7,*,241 042,F1 2A,"SBC (&2A),Y"
90E9,,170,AA,TAX
90EA,,138,8A,TXA
90EB,,024,18,CLC
90EC,e,101 002,65 02,ADC &02
90EE,,168,A8,TAY
90EF,,165 003,A5 03,LDA &03
90F1,i,105 000,69 00,ADC#&00
90F3,,170,AA,TAX
```

90F4,,196 004,C4 04,CPY &04
90F6,,229 005,E5 05,SBC &05
90F8,,176 167,B0 A7,BCS -89   --> &90A1
90FA,,132 002,84 02,STY &02
90FC,,134 003,86 03,STX &03
90FE,h,104,68,PLA
90FF,,160 002,A0 02,LDY#&02
9101,*,145 042,91 2A,"STA (&2A),Y"
9103,,136,88,DEY
9104,-,165 045,A5 2D,LDA &2D
9106,,240 006,F0 06,BEQ 6   --> &910E
9108,*,145 042,91 2A,"STA (&2A),Y"
910A,",",165 044,A5 2C,LDA &2C
910C,*,146 042,92 2A,STA (&2A)
910E,,160 003,A0 03,LDY#&03
9110,6,165 054,A5 36,LDA &36
9112,*,145 042,91 2A,"STA (&2A),Y"
9114,,240 021,F0 15,BEQ 21   --> &912B
9116,,160 001,A0 01,LDY#&01
9118,*,177 042,B1 2A,"LDA (&2A),Y"
911A,-,133 045,85 2D,STA &2D
911C,*,178 042,B2 2A,LDA (&2A)
911E,",",133 044,85 2C,STA &2C
9120,,136,88,DEY
9121,,185 000 006,B9 00 06,"LDA &0600,Y"
9124,",",145 044,91 2C,"STA (&2C),Y"
9126,,200,C8,INY
9127,6,196 054,C4 36,CPY &36
9129,,208 246,D0 F6,BNE -10   --> &9121
912B,`,096,60,RTS
912C,+,032 043 190,20 2B BE,JSR &BE2B
912F,,192 000,C0 00,CPY#&00
9131,,240 011,F0 0B,BEQ 11   --> &913E
9133,,185 000 006,B9 00 06,"LDA &0600,Y"
9136,*,145 042,91 2A,"STA (&2A),Y"
9138,,136,88,DEY
9139,,208 248,D0 F8,BNE -8   --> &9133
913B,,173 000 006,AD 00 06,LDA &0600
913E,*,146 042,92 2A,STA (&2A)
9140,`,096,60,RTS
9141,<,032 060 186,20 3C BA,JSR &BA3C
9144,Z,090,5A,PHY
9145,,032 235 142,20 EB 8E,JSR &8EEB

9148,=,208 061,D0 3D,BNE 61   --> &9187
914A,;,032 059 157,20 3B 9D,JSR &9D3B
914D,,032 017 165,20 11 A5,JSR &A511
9150,z,122,7A,PLY
9151,',165 039,A5 27,LDA &27
9153,,032 212 255,20 D4 FF,JSR &FFD4
9156,,170,AA,TAX
9157,,240 027,F0 1B,BEQ 27   --> &9174
9159,0,048 012,30 0C,BMI 12   --> &9167
915B,,162 003,A2 03,LDX#&03
915D,*,181 042,B5 2A,"LDA &2A,X"
915F,,032 212 255,20 D4 FF,JSR &FFD4
9162,,202,CA,DEX
9163,,016 248,10 F8,BPL -8   --> &915D
9165,,128 221,80 DD,BRA -35   --> &9144
9167,,162 004,A2 04,LDX#&04
9169,l,189 108 004,BD 6C 04,"LDA &046C,X"
916C,,032 212 255,20 D4 FF,JSR &FFD4
916F,,202,CA,DEX
9170,,016 247,10 F7,BPL -9   --> &9169
9172,,128 208,80 D0,BRA -48   --> &9144
9174,6,165 054,A5 36,LDA &36
9176,,032 212 255,20 D4 FF,JSR &FFD4
9179,,170,AA,TAX
917A,,240 200,F0 C8,BEQ -56   --> &9144
917C,,189 255 005,BD FF 05,"LDA &05FF,X"
917F,,032 212 255,20 D4 FF,JSR &FFD4
9182,,202,CA,DEX
9183,,208 247,D0 F7,BNE -9   --> &917C
9185,,128 189,80 BD,BRA -67   --> &9144
9187,h,104,68,PLA
9188,,132 010,84 0A,STY &0A
918A,L,076 002 144,4C 02 90,JMP &9002
918D,,032 223 140,20 DF 8C,JSR &8CDF
9190,,240 175,F0 AF,BEQ -81   --> &9141
9192,,198 010,C6 0A,DEC &0A
9194,,128 021,80 15,BRA 21   --> &91AB
9196,,173 000 004,AD 00 04,LDA &0400
9199,,240 016,F0 10,BEQ 16   --> &91AB
919B,,165 030,A5 1E,LDA &1E
919D,,240 012,F0 0C,BEQ 12   --> &91AB
919F,,237 000 004,ED 00 04,SBC &0400
91A2,,176 249,B0 F9,BCS -7   --> &919D

91A4,,168,A8,TAY
91A5,,032 146 189,20 92 BD,JSR &BD92
91A8,,200,C8,INY
91A9,,208 250,D0 FA,BNE -6 --> &91A5
91AB,,024,18,CLC
91AC,,173 000 004,AD 00 04,LDA &0400
91AF,,133 020,85 14,STA &14
91B1,f,102 021,66 15,ROR &15
91B3,,032 224 142,20 E0 8E,JSR &8EE0
91B6,:,201 058,C9 3A,CMP#&3A
91B8,,240 008,F0 08,BEQ 8 --> &91C2
91BA,,201 013,C9 0D,CMP#&0D
91BC,,240 004,F0 04,BEQ 4 --> &91C2
91BE,,201 139,C9 8B,CMP#&8B
91C0,,208 025,D0 19,BNE 25 --> &91DB
91C2,,032 146 186,20 92 BA,JSR &BA92
91C5,L,076 000 144,4C 00 90,JMP &9000
91C8,d,100 020,64 14,STZ &14
91CA,d,100 021,64 15,STZ &15
91CC,,032 224 142,20 E0 8E,JSR &8EE0
91CF,:,201 058,C9 3A,CMP#&3A
91D1,,240 242,F0 F2,BEQ -14 --> &91C5
91D3,,201 013,C9 0D,CMP#&0D
91D5,,240 238,F0 EE,BEQ -18 --> &91C5
91D7,,201 139,C9 8B,CMP#&8B
91D9,,240 234,F0 EA,BEQ -22 --> &91C5
91DB,~,201 126,C9 7E,CMP#&7E
91DD,,240 210,F0 D2,BEQ -46 --> &91B1
91DF,",",201 044,C9 2C,CMP#&2C
91E1,,240 179,F0 B3,BEQ -77 --> &9196
91E3,;,201 059,C9 3B,CMP#&3B
91E5,,240 225,F0 E1,BEQ -31 --> &91C8
91E7,z,032 122 146,20 7A 92,JSR &927A
91EA,,144 199,90 C7,BCC -57 --> &91B3
91EC,,165 020,A5 14,LDA &14
91EE,H,072,48,PHA
91EF,,165 021,A5 15,LDA &15
91F1,H,072,48,PHA
91F2,,198 027,C6 1B,DEC &1B
91F4,;,032 059 157,20 3B 9D,JSR &9D3B
91F7,h,104,68,PLA
91F8,,133 021,85 15,STA &15
91FA,h,104,68,PLA

91FB,,133 020,85 14,STA &14
91FD,,165 027,A5 1B,LDA &1B
91FF,,133 010,85 0A,STA &0A
9201,,152,98,TYA
9202,,240 019,F0 13,BEQ 19   --> &9217
9204,,032 024 161,20 18 A1,JSR &A118
9207,,165 020,A5 14,LDA &14
9209,8,056,38,SEC
920A,6,229 054,E5 36,SBC &36
920C,,144 009,90 09,BCC 9   --> &9217
920E,,240 007,F0 07,BEQ 7   --> &9217
9210,,168,A8,TAY
9211,,032 146 189,20 92 BD,JSR &BD92
9214,,136,88,DEY
9215,,208 250,D0 FA,BNE -6   --> &9211
9217,6,165 054,A5 36,LDA &36
9219,,240 152,F0 98,BEQ -104   --> &91B3
921B,,160 000,A0 00,LDY#&00
921D,,185 000 006,B9 00 06,"LDA &0600,Y"
9220,,032 152 189,20 98 BD,JSR &BD98
9223,,200,C8,INY
9224,6,196 054,C4 36,CPY &36
9226,,208 245,D0 F5,BNE -11   --> &921D
9228,,128 137,80 89,BRA -119   --> &91B3
922A,L,076 246 142,4C F6 8E,JMP &8EF6
922D,*,165 042,A5 2A,LDA &2A
922F,H,072,48,PHA
9230,,032 167 150,20 A7 96,JSR &96A7
9233,,169 031,A9 1F,LDA#&1F
9235,,032 238 255,20 EE FF,JSR &FFEE
9238,h,104,68,PLA
9239,,032 238 255,20 EE FF,JSR &FFEE
923C,@,032 064 152,20 40 98,JSR &9840
923F,),128 041,80 29,BRA 41   --> &926A
9241,,032 175 150,20 AF 96,JSR &96AF
9244,,032 235 142,20 EB 8E,JSR &8EEB
9247,,240 228,F0 E4,BEQ -28   --> &922D
9249,),201 041,C9 29,CMP#&29
924B,,208 221,D0 DD,BNE -35   --> &922A
924D,*,165 042,A5 2A,LDA &2A
924F,,229 030,E5 1E,SBC &1E
9251,,240 023,F0 17,BEQ 23   --> &926A
9253,,170,AA,TAX

```
9254,,176 012,B0 0C,BCS 12   --> &9262
9256,,032 146 186,20 92 BA,JSR &BA92
9259,,128 003,80 03,BRA 3   --> &925E
925B,,032 180 150,20 B4 96,JSR &96B4
925E,*,166 042,A6 2A,LDX &2A
9260,,240 008,F0 08,BEQ 8   --> &926A
9262,,032 191 189,20 BF BD,JSR &BDBF
9265,,128 003,80 03,BRA 3   --> &926A
9267,,032 146 186,20 92 BA,JSR &BA92
926A,,024,18,CLC
926B,,128 008,80 08,BRA 8   --> &9275
926D,,198 010,C6 0A,DEC &0A
926F,/,032 047 157,20 2F 9D,JSR &9D2F
9272,,032 191 150,20 BF 96,JSR &96BF
9275,,164 027,A4 1B,LDY &1B
9277,,132 010,84 0A,STY &0A
9279,`,096,60,RTS
927A,,166 011,A6 0B,LDX &0B
927C,,134 025,86 19,STX &19
927E,,166 012,A6 0C,LDX &0C
9280,,134 026,86 1A,STX &1A
9282,,166 010,A6 0A,LDX &0A
9284,,134 027,86 1B,STX &1B
9286,',201 039,C9 27,CMP#&27
9288,,240 221,F0 DD,BEQ -35   --> &9267
928A,,201 138,C9 8A,CMP#&8A
928C,,240 179,F0 B3,BEQ -77   --> &9241
928E,,201 137,C9 89,CMP#&89
9290,,240 201,F0 C9,BEQ -55   --> &925B
9292,8,056,38,SEC
9293,`,096,60,RTS
9294,,000,00,BRK
9295,,009,09,EQUB &09
9296,,141,BD,xxx Invalid Code
9297,"""""",034,22,xxx Invalid Code
9298,,000,00,EQUB &00
9299,,032 224 142,20 E0 8E,JSR &8EE0
929C,z,032 122 146,20 7A 92,JSR &927A
929F,,144 242,90 F2,BCC -14   --> &9293
92A1,"""""",201 034,C9 22,CMP#&22
92A3,,208 237,D0 ED,BNE -19   --> &9292
92A5,,200,C8,INY
92A6,,177 025,B1 19,"LDA (&19),Y"
```

92A8,,201 013,C9 0D,CMP#&0D
92AA,,240 232,F0 E8,BEQ -24   --> &9294
92AC,"""",201 034,C9 22,CMP#&22
92AE,,208 009,D0 09,BNE 9   --> &92B9
92B0,,200,C8,INY
92B1,,132 027,84 1B,STY &1B
92B3,,177 025,B1 19,"LDA (&19),Y"
92B5,"""",201 034,C9 22,CMP#&22
92B7,,208 177,D0 B1,BNE -79   --> &926A
92B9,,032 152 189,20 98 BD,JSR &BD98
92BC,,128 231,80 E7,BRA -25   --> &92A5
92BE,/,032 047 157,20 2F 9D,JSR &9D2F
92C1,,032 188 150,20 BC 96,JSR &96BC
92C4,&,032 038 188,20 26 BC,JSR &BC26
92C7,,156 000 006,9C 00 06,STZ &0600
92CA,,160 000,A0 00,LDY#&00
92CC,Z,090,5A,PHY
92CD,,032 235 142,20 EB 8E,JSR &8EEB
92D0,,208 031,D0 1F,BNE 31   --> &92F1
92D2,,164 027,A4 1B,LDY &1B
92D4,,032 001 153,20 01 99,JSR &9901
92D7,(,240 040,F0 28,BEQ 40   --> &9301
92D9,z,122,7A,PLY
92DA,,200,C8,INY
92DB,*,165 042,A5 2A,LDA &2A
92DD,,153 000 006,99 00 06,"STA &0600,Y"
92E0,,200,C8,INY
92E1,+,165 043,A5 2B,LDA &2B
92E3,,153 000 006,99 00 06,"STA &0600,Y"
92E6,,200,C8,INY
92E7,",",165 044,A5 2C,LDA &2C
92E9,,153 000 006,99 00 06,"STA &0600,Y"
92EC,,238 000 006,EE 00 06,INC &0600
92EF,,128 219,80 DB,BRA -37   --> &92CC
92F1,z,122,7A,PLY
92F2,,198 027,C6 1B,DEC &1B
92F4,,032 150 155,20 96 9B,JSR &9B96
92F7,,032 230 188,20 E6 BC,JSR &BCE6
92FA,,032 004 147,20 04 93,JSR &9304
92FD,,216,D8,CLD
92FE,L,076 005 144,4C 05 90,JMP &9005
9301,L,076 140 173,4C 8C AD,JMP &AD8C
9304,,173 012 004,AD 0C 04,LDA &040C

9307,J,074,4A,LSR A
9308,,173 004 004,AD 04 04,LDA &0404
930B,`,174 096 004,AE 60 04,LDX &0460
930E,d,172 100 004,AC 64 04,LDY &0464
9311,l*,108 042 000,6C 2A 00,JMP (&002A)
9314,Li,076 105 155,4C 69 9B,JMP &9B69
9317,,032 030 155,20 1E 9B,JSR &9B1E
931A,,144 248,90 F8,BCC -8 --> &9314
931C,&,032 038 188,20 26 BC,JSR &BC26
931F,,032 229 140,20 E5 8C,JSR &8CE5
9322,,208 240,D0 F0,BNE -16 --> &9314
9324,,032 030 155,20 1E 9B,JSR &9B1E
9327,,144 235,90 EB,BCC -21 --> &9314
9329,,032 166 155,20 A6 9B,JSR &9BA6
932C,*,165 042,A5 2A,LDA &2A
932E,9,133 057,85 39,STA &39
9330,+,165 043,A5 2B,LDA &2B
9332,:,133 058,85 3A,STA &3A
9334,,032 230 188,20 E6 BC,JSR &BCE6
9337,,032 152 186,20 98 BA,JSR &BA98
933A,,032 202 155,20 CA 9B,JSR &9BCA
933D,,032 239 190,20 EF BE,JSR &BEEF
9340,9,165 057,A5 39,LDA &39
9342,*,197 042,C5 2A,CMP &2A
9344,:,165 058,A5 3A,LDA &3A
9346,+,229 043,E5 2B,SBC &2B
9348,,176 237,B0 ED,BCS -19 --> &9337
934A,L,076 131 143,4C 83 8F,JMP &8F83
934D,,169 010,A9 0A,LDA#&0A
934F,,032 024 174,20 18 AE,JSR &AE18
9352,,032 030 155,20 1E 9B,JSR &9B1E
9355,&,032 038 188,20 26 BC,JSR &BC26
9358,,169 010,A9 0A,LDA#&0A
935A,,032 024 174,20 18 AE,JSR &AE18
935D,,032 229 140,20 E5 8C,JSR &8CE5
9360,,208 014,D0 0E,BNE 14 --> &9370
9362,,032 030 155,20 1E 9B,JSR &9B1E
9365,+,165 043,A5 2B,LDA &2B
9367,S,208 083,D0 53,BNE 83 --> &93BC
9369,*,165 042,A5 2A,LDA &2A
936B,O,240 079,F0 4F,BEQ 79 --> &93BC
936D,L,076 166 155,4C A6 9B,JMP &9BA6
9370,L,076 176 155,4C B0 9B,JMP &9BB0

9373,,165 018,A5 12,LDA &12
9375,;,133 059,85 3B,STA &3B
9377,,165 019,A5 13,LDA &13
9379,<,133 060,85 3C,STA &3C
937B,,165 024,A5 18,LDA &18
937D,8,133 056,85 38,STA &38
937F,,160 001,A0 01,LDY#&01
9381,7,132 055,84 37,STY &37
9383,`,096,60,RTS
9384,M,032 077 147,20 4D 93,JSR &934D
9387,9,162 057,A2 39,LDX#&39
9389,,032 008 189,20 08 BD,JSR &BD08
938C,,032 229 189,20 E5 BD,JSR &BDE5
938F,s,032 115 147,20 73 93,JSR &9373
9392,7,178 055,B2 37,LDA (&37)
9394,0.,048 046,30 2E,BMI 46 --> &93C4
9396,;,146 059,92 3B,STA (&3B)
9398,7,177 055,B1 37,"LDA (&37),Y"
939A,;,145 059,91 3B,"STA (&3B),Y"
939C,8,056,38,SEC
939D,,152,98,TYA
939E,e;,101 059,65 3B,ADC &3B
93A0,;,133 059,85 3B,STA &3B
93A2,,144 002,90 02,BCC 2 --> &93A6
93A4,<,230 060,E6 3C,INC &3C
93A6,,197 006,C5 06,CMP &06
93A8,<,165 060,A5 3C,LDA &3C
93AA,,229 007,E5 07,SBC &07
93AC,,176 005,B0 05,BCS 5 --> &93B3
93AE,z,032 122 148,20 7A 94,JSR &947A
93B1,,128 223,80 DF,BRA -33 --> &9392
93B3,,000,00,BRK
93B4,,000,00,EQUB &00
93B5,s,204 032 115,CC 20 73,CPY &7320
93B8,pa,112 097,70 61,BVS 97 --> &941B
93BA,c,099,63,xxx Invalid Code
93BB,e,101,65,xxx Invalid Code
93BC,,000,00,BRK
93BD,,000,00,EQUB &00
93BE,S,083,53,xxx Invalid Code
93BF,il,105 108,69 6C,ADC#&6C
93C1,ly,108 121,6C 79,xxx Invalid Code
93C3,,000,00,EQUB &00

93C4,{,032 123 147,20 7B 93,JSR &937B
93C7,7,178 055,B2 37,LDA (&37)
93C9,0,048 028,30 1C,BMI 28   --> &93E7
93CB,:,165 058,A5 3A,LDA &3A
93CD,7,146 055,92 37,STA (&37)
93CF,9,165 057,A5 39,LDA &39
93D1,7,145 055,91 37,"STA (&37),Y"
93D3,,024,18,CLC
93D4,9,165 057,A5 39,LDA &39
93D6,e*,101 042,65 2A,ADC &2A
93D8,9,133 057,85 39,STA &39
93DA,,169 000,A9 00,LDA#&00
93DC,e:,101 058,65 3A,ADC &3A
93DE,),041 127,29 7F,AND#&7F
93E0,:,133 058,85 3A,STA &3A
93E2,z,032 122 148,20 7A 94,JSR &947A
93E5,,128 224,80 E0,BRA -32   --> &93C7
93E7,,165 024,A5 18,LDA &18
93E9,,133 012,85 0C,STA &0C
93EB,d,100 011,64 0B,STZ &0B
93ED,,160 001,A0 01,LDY#&01
93EF,,177 011,B1 0B,"LDA (&0B),Y"
93F1,0g,048 103,30 67,BMI 103   --> &945A
93F3,,160 004,A0 04,LDY#&04
93F5,"d,",100 044,64 2C,STZ &2C
93F7,,177 011,B1 0B,"LDA (&0B),Y"
93F9,",",166 044,A6 2C,LDX &2C
93FB,,208 008,D0 08,BNE 8   --> &9405
93FD,,201 141,C9 8D,CMP#&8D
93FF,,240 026,F0 1A,BEQ 26   --> &941B
9401,,201 244,C9 F4,CMP#&F4
9403,,240 013,F0 0D,BEQ 13   --> &9412
9405,,200,C8,INY
9406,"""",201 034,C9 22,CMP#&22
9408,,208 004,D0 04,BNE 4   --> &940E
940A,"E,",069 044,45 2C,EOR &2C
940C,",",133 044,85 2C,STA &2C
940E,,201 013,C9 0D,CMP#&0D
9410,,208 229,D0 E5,BNE -27   --> &93F7
9412,,160 003,A0 03,LDY#&03
9414,,177 011,B1 0B,"LDA (&0B),Y"
9416,,032 244 155,20 F4 9B,JSR &9BF4
9419,,128 210,80 D2,BRA -46   --> &93ED

941B,*,032 042 155,20 2A 9B,JSR &9B2A
941E,s,032 115 147,20 73 93,JSR &9373
9421,7,178 055,B2 37,LDA (&37)
9423,07,048 055,30 37,BMI 55   --> &945C
9425,;,178 059,B2 3B,LDA (&3B)
9427,+,197 043,C5 2B,CMP &2B
9429,,208 031,D0 1F,BNE 31   --> &944A
942B,;,177 059,B1 3B,"LDA (&3B),Y"
942D,*,197 042,C5 2A,CMP &2A
942F,,208 025,D0 19,BNE 25   --> &944A
9431,7,177 055,B1 37,"LDA (&37),Y"
9433,=,133 061,85 3D,STA &3D
9435,7,178 055,B2 37,LDA (&37)
9437,,170,AA,TAX
9438,,164 010,A4 0A,LDY &0A
943A,,136,88,DEY
943B,,165 011,A5 0B,LDA &0B
943D,9,133 057,85 39,STA &39
943F,,165 012,A5 0C,LDA &0C
9441,:,133 058,85 3A,STA &3A
9443,b,032 098 141,20 62 8D,JSR &8D62
9446,,164 010,A4 0A,LDY &0A
9448,,128 173,80 AD,BRA -83   --> &93F7
944A,,024,18,CLC
944B,z,032 122 148,20 7A 94,JSR &947A
944E,;,165 059,A5 3B,LDA &3B
9450,i,105 002,69 02,ADC#&02
9452,;,133 059,85 3B,STA &3B
9454,,144 203,90 CB,BCC -53   --> &9421
9456,<,230 060,E6 3C,INC &3C
9458,,128 199,80 C7,BRA -57   --> &9421
945A,Z,128 090,80 5A,BRA 90   --> &94B6
945C,,032 207 190,20 CF BE,JSR &BECF
945F,Fa,070 097,46 61,LSR &61
9461,il,105 108,69 6C,ADC#&6C
9463,ed,101 100,65 64,ADC &64
9465,at,032 097 116,20 61 74,JSR &7461
9468,,032 177 011,20 B1 0B,JSR &0BB1
946B,+,133 043,85 2B,STA &2B
946D,,200,C8,INY
946E,,177 011,B1 0B,"LDA (&0B),Y"
9470,*,133 042,85 2A,STA &2A
9472,,032 129 160,20 81 A0,JSR &A081

9475,,032 146 186,20 92 BA,JSR &BA92
9478,,128 204,80 CC,BRA -52 --> &9446
947A,,200,C8,INY
947B,7,177 055,B1 37,"LDA (&37),Y"
947D,,160 001,A0 01,LDY#&01
947F,e7,101 055,65 37,ADC &37
9481,7,133 055,85 37,STA &37
9483,,144 003,90 03,BCC 3 --> &9488
9485,8,230 056,E6 38,INC &38
9487,,024,18,CLC
9488,`,096,60,RTS
9489,M,032 077 147,20 4D 93,JSR &934D
948C,*,165 042,A5 2A,LDA &2A
948E,H,072,48,PHA
948F,,032 230 188,20 E6 BC,JSR &BCE6
9492,&,032 038 188,20 26 BC,JSR &BC26
9495,,032 133 160,20 85 A0,JSR &A085
9498,t,032 116 186,20 74 BA,JSR &BA74
949B,,032 230 188,20 E6 BC,JSR &BCE6
949E,,032 231 141,20 E7 8D,JSR &8DE7
94A1,,160 000,A0 00,LDY#&00
94A3,,032 008 187,20 08 BB,JSR &BB08
94A6,,032 172 187,20 AC BB,JSR &BBAC
94A9,h,104,68,PLA
94AA,H,072,48,PHA
94AB,,024,18,CLC
94AC,e*,101 042,65 2A,ADC &2A
94AE,*,133 042,85 2A,STA &2A
94B0,,144 224,90 E0,BCC -32 --> &9492
94B2,+,230 043,E6 2B,INC &2B
94B4,,016 220,10 DC,BPL -36 --> &9492
94B6,L,076 131 143,4C 83 8F,JMP &8F83
94B9,L,076 005 150,4C 05 96,JMP &9605
94BC,,198 010,C6 0A,DEC &0A
94BE,,032 174 152,20 AE 98,JSR &98AE
94C1,i,240 105,F0 69,BEQ 105 --> &952C
94C3,g,176 103,B0 67,BCS 103 --> &952C
94C5,C,032 067 188,20 43 BC,JSR &BC43
94C8,,032 175 150,20 AF 96,JSR &96AF
94CB,,032 239 190,20 EF BE,JSR &BEEF
94CE,-,165 045,A5 2D,LDA &2D
94D0,",",005 044,05 2C,ORA &2C
94D2,X,208 088,D0 58,BNE 88 --> &952C

94D4,,024,18,CLC
94D5,*,165 042,A5 2A,LDA &2A
94D7,e,101 002,65 02,ADC &02
94D9,,168,A8,TAY
94DA,+,165 043,A5 2B,LDA &2B
94DC,e,101 003,65 03,ADC &03
94DE,,170,AA,TAX
94DF,,196 004,C4 04,CPY &04
94E1,,229 005,E5 05,SBC &05
94E3,,176 212,B0 D4,BCS -44 --> &94B9
94E5,,165 002,A5 02,LDA &02
94E7,*,133 042,85 2A,STA &2A
94E9,,165 003,A5 03,LDA &03
94EB,+,133 043,85 2B,STA &2B
94ED,,132 002,84 02,STY &02
94EF,,134 003,86 03,STX &03
94F1,@,169 064,A9 40,LDA#&40
94F3,',133 039,85 27,STA &27
94F5,+,032 043 179,20 2B B3,JSR &B32B
94F8,u,032 117 146,20 75 92,JSR &9275
94FB,,032 229 140,20 E5 8C,JSR &8CE5
94FE,4,240 052,F0 34,BEQ 52 --> &9534
9500,L,076 000 144,4C 00 90,JMP &9000
9503,?,162 063,A2 3F,LDX#&3F
9505,,032 008 189,20 08 BD,JSR &BD08
9508,,162 000,A2 00,LDX#&00
950A,,160 000,A0 00,LDY#&00
950C,F@,070 064,46 40,LSR &40
950E,f?,102 063,66 3F,ROR &3F
9510,,144 011,90 0B,BCC 11 --> &951D
9512,,024,18,CLC
9513,,152,98,TYA
9514,e*,101 042,65 2A,ADC &2A
9516,,168,A8,TAY
9517,,138,8A,TXA
9518,e+,101 043,65 2B,ADC &2B
951A,,170,AA,TAX
951B,,176 015,B0 0F,BCS 15 --> &952C
951D,*,006 042,06 2A,ASL &2A
951F,&+,038 043,26 2B,ROL &2B
9521,?,165 063,A5 3F,LDA &3F
9523,@,005 064,05 40,ORA &40
9525,,208 229,D0 E5,BNE -27 --> &950C

9527,*,132 042,84 2A,STY &2A
9529,+,134 043,86 2B,STX &2B
952B,`,096,60,RTS
952C,,000,00,BRK
952D,,010,0A,EQUB &0A
952E,B,066,42,xxx Invalid Code
952F,ad,097 100 032,61 64 20,"ADC (&2064,X)"
9532,,222,DE,xxx Invalid Code
9533,,000,00,EQUB &00
9534,,032 224 142,20 E0 8E,JSR &8EE0
9537,,152,98,TYA
9538,,024,18,CLC
9539,e,101 011,65 0B,ADC &0B
953B,,166 012,A6 0C,LDX &0C
953D,,144 002,90 02,BCC 2   --> &9541
953F,,232,E8,INX
9540,,024,18,CLC
9541,,233 000,E9 00,SBC#&00
9543,7,133 055,85 37,STA &37
9545,,138,8A,TXA
9546,,233 000,E9 00,SBC#&00
9548,8,133 056,85 38,STA &38
954A,,162 005,A2 05,LDX#&05
954C,?,134 063,86 3F,STX &3F
954E,,166 010,A6 0A,LDX &0A
9550,,032 246 154,20 F6 9A,JSR &9AF6
9553,,192 001,C0 01,CPY#&01
9555,,240 213,F0 D5,BEQ -43   --> &952C
9557,(,201 040,C9 28,CMP#&28
9559,,240 021,F0 15,BEQ 21   --> &9570
955B,$,201 036,C9 24,CMP#&24
955D,,240 004,F0 04,BEQ 4   --> &9563
955F,%,201 037,C9 25,CMP#&25
9561,,208 010,D0 0A,BNE 10   --> &956D
9563,?,198 063,C6 3F,DEC &3F
9565,,200,C8,INY
9566,,232,E8,INX
9567,7,177 055,B1 37,"LDA (&37),Y"
9569,(,201 040,C9 28,CMP#&28
956B,,240 003,F0 03,BEQ 3   --> &9570
956D,L,076 188 148,4C BC 94,JMP &94BC
9570,,200,C8,INY
9571,,134 010,86 0A,STX &0A

9573,,032 133 128,20 85 80,JSR &8085
9576,,208 180,D0 B4,BNE -76 --> &952C
9578,T,032 084 152,20 54 98,JSR &9854
957B,,162 001,A2 01,LDX#&01
957D,,032 131 152,20 83 98,JSR &9883
9580,?,165 063,A5 3F,LDA &3F
9582,H,072,48,PHA
9583,,169 001,A9 01,LDA#&01
9585,H,072,48,PHA
9586,,032 024 174,20 18 AE,JSR &AE18
9589,&,032 038 188,20 26 BC,JSR &BC26
958C,o,032 111 146,20 6F 92,JSR &926F
958F,+,165 043,A5 2B,LDA &2B
9591,),041 192,29 C0,AND#&C0
9593,",",005 044,05 2C,ORA &2C
9595,-,005 045,05 2D,ORA &2D
9597,,208 147,D0 93,BNE -109 --> &952C
9599,,032 239 190,20 EF BE,JSR &BEEF
959C,z,122,7A,PLY
959D,*,165 042,A5 2A,LDA &2A
959F,,145 002,91 02,"STA (&02),Y"
95A1,,200,C8,INY
95A2,+,165 043,A5 2B,LDA &2B
95A4,,145 002,91 02,"STA (&02),Y"
95A6,,200,C8,INY
95A7,Z,090,5A,PHY
95A8,,032 003 149,20 03 95,JSR &9503
95AB,,032 229 140,20 E5 8C,JSR &8CE5
95AE,,240 217,F0 D9,BEQ -39 --> &9589
95B0,),201 041,C9 29,CMP#&29
95B2,,208 194,D0 C2,BNE -62 --> &9576
95B4,,250,FA,PLX
95B5,h,104,68,PLA
95B6,,218,DA,PHX
95B7,?,133 063,85 3F,STA &3F
95B9,d@,100 064,64 40,STZ &40
95BB,,032 008 149,20 08 95,JSR &9508
95BE,h,104,68,PLA
95BF,H,072,48,PHA
95C0,e*,101 042,65 2A,ADC &2A
95C2,*,133 042,85 2A,STA &2A
95C4,,144 002,90 02,BCC 2 --> &95C8
95C6,+,230 043,E6 2B,INC &2B

```
95C8,,165 003,A5 03,LDA &03
95CA,8,133 056,85 38,STA &38
95CC,,165 002,A5 02,LDA &02
95CE,7,133 055,85 37,STA &37
95D0,,024,18,CLC
95D1,e*,101 042,65 2A,ADC &2A
95D3,,168,A8,TAY
95D4,+,165 043,A5 2B,LDA &2B
95D6,e,101 003,65 03,ADC &03
95D8,+,176 043,B0 2B,BCS 43   --> &9605
95DA,,170,AA,TAX
95DB,,196 004,C4 04,CPY &04
95DD,,229 005,E5 05,SBC &05
95DF,$,176 036,B0 24,BCS 36   --> &9605
95E1,,132 002,84 02,STY &02
95E3,,134 003,86 03,STX &03
95E5,h,104,68,PLA
95E6,7,146 055,92 37,STA (&37)
95E8,e7,101 055,65 37,ADC &37
95EA,,168,A8,TAY
95EB,,169 000,A9 00,LDA#&00
95ED,d7,100 055,64 37,STZ &37
95EF,,144 002,90 02,BCC 2   --> &95F3
95F1,8,230 056,E6 38,INC &38
95F3,7,145 055,91 37,"STA (&37),Y"
95F5,,200,C8,INY
95F6,,208 002,D0 02,BNE 2   --> &95FA
95F8,8,230 056,E6 38,INC &38
95FA,,196 002,C4 02,CPY &02
95FC,,208 245,D0 F5,BNE -11   --> &95F3
95FE,8,228 056,E4 38,CPX &38
9600,,208 241,D0 F1,BNE -15   --> &95F3
9602,L,076 251 148,4C FB 94,JMP &94FB
9605,,000,00,BRK
9606,,011,0B,EQUB &0B
9607,s,222 032 115,DE 20 73,"DEC &7320,X"
960A,pa,112 097,70 61,BVS 97   --> &966D
960C,c,099,63,xxx Invalid Code
960D,e,101,65,xxx Invalid Code
960E,,000,00,EQUB &00
960F,,032 185 150,20 B9 96,JSR &96B9
9612,*,165 042,A5 2A,LDA &2A
9614,,133 006,85 06,STA &06
```

```
9616,,133 004,85 04,STA &04
9618,+,165 043,A5 2B,LDA &2B
961A,,133 007,85 07,STA &07
961C,,133 005,85 05,STA &05
961E,,128 027,80 1B,BRA 27   --> &963B
9620,,032 185 150,20 B9 96,JSR &96B9
9623,*,165 042,A5 2A,LDA &2A
9625,,133 000,85 00,STA &00
9627,,133 002,85 02,STA &02
9629,+,165 043,A5 2B,LDA &2B
962B,,133 001,85 01,STA &01
962D,,133 003,85 03,STA &03
962F,,032 187 187,20 BB BB,JSR &BBBB
9632,,128 007,80 07,BRA 7   --> &963B
9634,,032 185 150,20 B9 96,JSR &96B9
9637,+,165 043,A5 2B,LDA &2B
9639,,133 024,85 18,STA &18
963B,L,076 005 144,4C 05 90,JMP &9005
963E,,032 166 155,20 A6 9B,JSR &9BA6
9641,,032 172 187,20 AC BB,JSR &BBAC
9644,,128 245,80 F5,BRA -11   --> &963B
9646,,032 030 155,20 1E 9B,JSR &9B1E
9649,,176 011,B0 0B,BCS 11   --> &9656
964B,,201 238,C9 EE,CMP#&EE
964D,,240 024,F0 18,BEQ 24   --> &9667
964F,,201 135,C9 87,CMP#&87
9651,,240 029,F0 1D,BEQ 29   --> &9670
9653,o,032 111 146,20 6F 92,JSR &926F
9656,,032 166 155,20 A6 9B,JSR &9BA6
9659,*,165 042,A5 2A,LDA &2A
965B,!,133 033,85 21,STA &21
965D,+,165 043,A5 2B,LDA &2B
965F,"""",133 034,85 22,STA &22
9661,,169 255,A9 FF,LDA#&FF
9663,,133 032,85 20,STA &20
9665,,128 212,80 D4,BRA -44   --> &963B
9667,,230 010,E6 0A,INC &0A
9669,,032 166 155,20 A6 9B,JSR &9BA6
966C,,169 255,A9 FF,LDA#&FF
966E,,208 239,D0 EF,BNE -17   --> &965F
9670,,230 010,E6 0A,INC &0A
9672,,032 166 155,20 A6 9B,JSR &9BA6
9675,,169 000,A9 00,LDA#&00
```

9677,,128 234,80 EA,BRA -22   --> &9663
9679,,200,C8,INY
967A,,177 011,B1 0B,"LDA (&0B),Y"
967C,$,201 036,C9 24,CMP#&24
967E,,240 014,F0 0E,BEQ 14   --> &968E
9680,,032 185 150,20 B9 96,JSR &96B9
9683,d.,100 046,64 2E,STZ &2E
9685,*,162 042,A2 2A,LDX#&2A
9687,,160 000,A0 00,LDY#&00
9689,,169 002,A9 02,LDA#&02
968B,L,076 018 179,4C 12 B3,JMP &B312
968E,,230 010,E6 0A,INC &0A
9690,F,032 070 155,20 46 9B,JSR &9B46
9693,',165 039,A5 27,LDA &27
9695,@,208 064,D0 40,BNE 64   --> &96D7
9697,,169 015,A9 0F,LDA#&0F
9699,6,164 054,A4 36,LDY &36
969B,,140 255 005,8C FF 05,STY &05FF
969E,,162 255,A2 FF,LDX#&FF
96A0,,160 005,A0 05,LDY#&05
96A2,,128 231,80 E7,BRA -25   --> &968B
96A4,Q,032 081 188,20 51 BC,JSR &BC51
96A7,,032 172 173,20 AC AD,JSR &ADAC
96AA,,128 019,80 13,BRA 19   --> &96BF
96AC,,032 241 142,20 F1 8E,JSR &8EF1
96AF,;,032 059 157,20 3B 9D,JSR &9D3B
96B2,,128 011,80 0B,BRA 11   --> &96BF
96B4,6,032 054 173,20 36 AD,JSR &AD36
96B7,,128 006,80 06,BRA 6   --> &96BF
96B9,F,032 070 155,20 46 9B,JSR &9B46
96BC,',165 039,A5 27,LDA &27
96BE,,168,A8,TAY
96BF,,240 022,F0 16,BEQ 22   --> &96D7
96C1,,016 019,10 13,BPL 19   --> &96D6
96C3,B,032 066 130,20 42 82,JSR &8242
96C6,1,165 049,A5 31,LDA &31
96C8,-,133 045,85 2D,STA &2D
96CA,2,165 050,A5 32,LDA &32
96CC,",",133 044,85 2C,STA &2C
96CE,3,165 051,A5 33,LDA &33
96D0,+,133 043,85 2B,STA &2B
96D2,4,165 052,A5 34,LDA &34
96D4,*,133 042,85 2A,STA &2A

96D6,`,096,60,RTS
96D7,L,076 146 144,4C 92 90,JMP &9092
96DA,6,032 054 173,20 36 AD,JSR &AD36
96DD,,240 248,F0 F8,BEQ -8 --> &96D7
96DF,0,048 245,30 F5,BMI -11 --> &96D6
96E1,L,076 133 129,4C 85 81,JMP &8185
96E4,,165 011,A5 0B,LDA &0B
96E6,,133 025,85 19,STA &19
96E8,,165 012,A5 0C,LDA &0C
96EA,,133 026,85 1A,STA &1A
96EC,,165 010,A5 0A,LDA &0A
96EE,,133 027,85 1B,STA &1B
96F0,,169 242,A9 F2,LDA#&F2
96F2,,032 025 176,20 19 B0,JSR &B019
96F5,,032 150 155,20 96 9B,JSR &9B96
96F8,L,076 005 144,4C 05 90,JMP &9005
96FB,,160 003,A0 03,LDY#&03
96FD,,169 000,A9 00,LDA#&00
96FF,*,145 042,91 2A,"STA (&2A),Y"
9701,,240 028,F0 1C,BEQ 28 --> &971F
9703,,186,BA,TSX
9704,,224 252,E0 FC,CPX#&FC
9706,*,176 042,B0 2A,BCS 42 --> &9732
9708,,032 174 152,20 AE 98,JSR &98AE
970B,"""",240 034,F0 22,BEQ 34 --> &972F
970D,,032 129 177,20 81 B1,JSR &B181
9710,",",164 044,A4 2C,LDY &2C
9712,0,048 231,30 E7,BMI -25 --> &96FB
9714,C,032 067 188,20 43 BC,JSR &BC43
9717,,032 232 171,20 E8 AB,JSR &ABE8
971A,',133 039,85 27,STA &27
971C,+,032 043 179,20 2B B3,JSR &B32B
971F,,186,BA,TSX
9720,,254 006 001,FE 06 01,"INC &0106,X"
9723,,164 027,A4 1B,LDY &1B
9725,,132 010,84 0A,STY &0A
9727,,032 229 140,20 E5 8C,JSR &8CE5
972A,,240 215,F0 D7,BEQ -41 --> &9703
972C,L,076 000 144,4C 00 90,JMP &9000
972F,L,076 002 144,4C 02 90,JMP &9002
9732,,000,00,BRK
9733,,012,0C,EQUB &0C
9734,No,078 111,4E 6F,xxx Invalid Code

9736,t,116 032,74 20,"STZ &20,X"
9738,,234,EA,NOP
9739,,000,00,BRK
973A,,025,19,EQUB &19
973B,Ba,066 097,42 61,xxx Invalid Code
973D,d,100 032,64 20,STZ &20
973F,,235,EB,xxx Invalid Code
9740,,000,00,EQUB &00
9741,o,032 111 146,20 6F 92,JSR &926F
9744,*,165 042,A5 2A,LDA &2A
9746,H,072,48,PHA
9747,,032 172 150,20 AC 96,JSR &96AC
974A,,032 150 155,20 96 9B,JSR &9B96
974D,,169 018,A9 12,LDA#&12
974F,,032 238 255,20 EE FF,JSR &FFEE
9752,h,104,68,PLA
9753,F,128 070,80 46,BRA 70   --> &979B
9755,o,032 111 146,20 6F 92,JSR &926F
9758,,032 166 155,20 A6 9B,JSR &9BA6
975B,,169 017,A9 11,LDA#&11
975D,<,128 060,80 3C,BRA 60   --> &979B
975F,o,032 111 146,20 6F 92,JSR &926F
9762,,032 166 155,20 A6 9B,JSR &9BA6
9765,K,032 075 190,20 4B BE,JSR &BE4B
9768,,232,E8,INX
9769,",",208 044,D0 2C,BNE 44   --> &9797
976B,,200,C8,INY
976C,),208 041,D0 29,BNE 41   --> &9797
976E,,165 004,A5 04,LDA &04
9770,,197 006,C5 06,CMP &06
9772,,208 197,D0 C5,BNE -59   --> &9739
9774,,165 005,A5 05,LDA &05
9776,,197 007,C5 07,CMP &07
9778,,208 191,D0 BF,BNE -65   --> &9739
977A,*,166 042,A6 2A,LDX &2A
977C,,169 133,A9 85,LDA#&85
977E,,032 244 255,20 F4 FF,JSR &FFF4
9781,,228 002,E4 02,CPX &02
9783,,152,98,TYA
9784,,229 003,E5 03,SBC &03
9786,,144 177,90 B1,BCC -79   --> &9739
9788,,228 018,E4 12,CPX &12
978A,,152,98,TYA

978B,,229 019,E5 13,SBC &13
978D,,144 170,90 AA,BCC -86   --> &9739
978F,,134 006,86 06,STX &06
9791,,134 004,86 04,STX &04
9793,,132 007,84 07,STY &07
9795,,132 005,84 05,STY &05
9797,d,100 030,64 1E,STZ &1E
9799,,169 022,A9 16,LDA#&16
979B,,032 238 255,20 EE FF,JSR &FFEE
979E,*,165 042,A5 2A,LDA &2A
97A0,L,128 076,80 4C,BRA 76   --> &97EE
97A2,,169 004,A9 04,LDA#&04
97A4,,128 002,80 02,BRA 2   --> &97A8
97A6,,169 005,A9 05,LDA#&05
97A8,H,072,48,PHA
97A9,/,032 047 157,20 2F 9D,JSR &9D2F
97AC,,032 188 150,20 BC 96,JSR &96BC
97AF,,128 009,80 09,BRA 9   --> &97BA
97B1,o,032 111 146,20 6F 92,JSR &926F
97B4,*,165 042,A5 2A,LDA &2A
97B6,H,072,48,PHA
97B7,,032 172 150,20 AC 96,JSR &96AC
97BA,&,032 038 188,20 26 BC,JSR &BC26
97BD,,032 172 150,20 AC 96,JSR &96AC
97C0,,032 150 155,20 96 9B,JSR &9B96
97C3,,169 025,A9 19,LDA#&19
97C5,,032 238 255,20 EE FF,JSR &FFEE
97C8,h,104,68,PLA
97C9,,032 238 255,20 EE FF,JSR &FFEE
97CC,,032 006 189,20 06 BD,JSR &BD06
97CF,7,165 055,A5 37,LDA &37
97D1,,032 238 255,20 EE FF,JSR &FFEE
97D4,8,165 056,A5 38,LDA &38
97D6,,032 238 255,20 EE FF,JSR &FFEE
97D9,@,032 064 152,20 40 98,JSR &9840
97DC,+,165 043,A5 2B,LDA &2B
97DE,,128 014,80 0E,BRA 14   --> &97EE
97E0,,032 166 155,20 A6 9B,JSR &9BA6
97E3,,169 016,A9 10,LDA#&10
97E5,,128 007,80 07,BRA 7   --> &97EE
97E7,,032 166 155,20 A6 9B,JSR &9BA6
97EA,d,100 030,64 1E,STZ &1E
97EC,,169 012,A9 0C,LDA#&0C

97EE,,032 238 255,20 EE FF,JSR &FFEE
97F1,L,076 005 144,4C 05 90,JMP &9005
97F4,,032 166 155,20 A6 9B,JSR &9BA6
97F7,,032 146 186,20 92 BA,JSR &BA92
97FA,,160 001,A0 01,LDY#&01
97FC,,177 253,B1 FD,"LDA (&FD),Y"
97FE,,240 241,F0 F1,BEQ -15 --> &97F1
9800,7,032 055 189,20 37 BD,JSR &BD37
9803,,200,C8,INY
9804,,208 246,D0 F6,BNE -10 --> &97FC
9806,,128 233,80 E9,BRA -23 --> &97F1
9808,+,165 043,A5 2B,LDA &2B
980A,,032 238 255,20 EE FF,JSR &FFEE
980D,,032 224 142,20 E0 8E,JSR &8EE0
9810,:,201 058,C9 3A,CMP#&3A
9812,),240 041,F0 29,BEQ 41 --> &983D
9814,,201 013,C9 0D,CMP#&0D
9816,%,240 037,F0 25,BEQ 37 --> &983D
9818,,201 139,C9 8B,CMP#&8B
981A,!,240 033,F0 21,BEQ 33 --> &983D
981C,,198 010,C6 0A,DEC &0A
981E,o,032 111 146,20 6F 92,JSR &926F
9821,@,032 064 152,20 40 98,JSR &9840
9824,,032 229 140,20 E5 8C,JSR &8CE5
9827,,240 228,F0 E4,BEQ -28 --> &980D
9829,;,201 059,C9 3B,CMP#&3B
982B,,240 219,F0 DB,BEQ -37 --> &9808
982D,|,201 124,C9 7C,CMP#&7C
982F,,208 223,D0 DF,BNE -33 --> &9810
9831,,169 000,A9 00,LDA#&00
9833,,160 009,A0 09,LDY#&09
9835,,032 238 255,20 EE FF,JSR &FFEE
9838,,136,88,DEY
9839,,208 250,D0 FA,BNE -6 --> &9835
983B,,128 208,80 D0,BRA -48 --> &980D
983D,L,076 000 144,4C 00 90,JMP &9000
9840,*,165 042,A5 2A,LDA &2A
9842,l,108 014 002,6C 0E 02,JMP (&020E)
9845,,160 001,A0 01,LDY#&01
9847,7,177 055,B1 37,"LDA (&37),Y"
9849,,170,AA,TAX
984A,,169 246,A9 F6,LDA#&F6
984C,,224 242,E0 F2,CPX#&F2

984E,,240 009,F0 09,BEQ 9   --> &9859
9850,,169 248,A9 F8,LDA#&F8
9852,,128 005,80 05,BRA 5   --> &9859
9854,,160 001,A0 01,LDY#&01
9856,7,177 055,B1 37,"LDA (&37),Y"
9858,,010,0A,ASL A
9859,,162 004,A2 04,LDX#&04
985B,:,133 058,85 3A,STA &3A
985D,;,134 059,86 3B,STX &3B
985F,:,177 058,B1 3A,"LDA (&3A),Y"
9861,,240 005,F0 05,BEQ 5   --> &9868
9863,,170,AA,TAX
9864,:,178 058,B2 3A,LDA (&3A)
9866,,128 243,80 F3,BRA -13   --> &985B
9868,,165 003,A5 03,LDA &03
986A,:,145 058,91 3A,"STA (&3A),Y"
986C,,165 002,A5 02,LDA &02
986E,:,146 058,92 3A,STA (&3A)
9870,,169 000,A9 00,LDA#&00
9872,,145 002,91 02,"STA (&02),Y"
9874,,200,C8,INY
9875,9,196 057,C4 39,CPY &39
9877,1,240 049,F0 31,BEQ 49   --> &98AA
9879,7,177 055,B1 37,"LDA (&37),Y"
987B,,145 002,91 02,"STA (&02),Y"
987D,,200,C8,INY
987E,9,196 057,C4 39,CPY &39
9880,,208 247,D0 F7,BNE -9   --> &9879
9882,`,096,60,RTS
9883,,169 000,A9 00,LDA#&00
9885,,145 002,91 02,"STA (&02),Y"
9887,,200,C8,INY
9888,,202,CA,DEX
9889,,208 250,D0 FA,BNE -6   --> &9885
988B,,024,18,CLC
988C,,152,98,TYA
988D,e,101 002,65 02,ADC &02
988F,,144 002,90 02,BCC 2   --> &9893
9891,,230 003,E6 03,INC &03
9893,,164 003,A4 03,LDY &03
9895,,196 005,C4 05,CPY &05
9897,,144 015,90 0F,BCC 15   --> &98A8
9899,,208 004,D0 04,BNE 4   --> &989F

989B,,197 004,C5 04,CMP &04
989D,,144 009,90 09,BCC 9   --> &98A8
989F,,169 000,A9 00,LDA#&00
98A1,,160 001,A0 01,LDY#&01
98A3,:,145 058,91 3A,"STA (&3A),Y"
98A5,L,076 161 144,4C A1 90,JMP &90A1
98A8,,133 002,85 02,STA &02
98AA,`,096,60,RTS
98AB,,032 131 152,20 83 98,JSR &9883
98AE,,032 245 152,20 F5 98,JSR &98F5
98B1,,208 029,D0 1D,BNE 29   --> &98D0
98B3,,176 027,B0 1B,BCS 27   --> &98D0
98B5,T,032 084 152,20 54 98,JSR &9854
98B8,,162 005,A2 05,LDX#&05
98BA,",",228 044,E4 2C,CPX &2C
98BC,,208 237,D0 ED,BNE -19   --> &98AB
98BE,,232,E8,INX
98BF,,128 234,80 EA,BRA -22   --> &98AB
98C1,!,201 033,C9 21,CMP#&21
98C3,,240 012,F0 0C,BEQ 12   --> &98D1
98C5,$,201 036,C9 24,CMP#&24
98C7,,240 019,F0 13,BEQ 19   --> &98DC
98C9,I?,073 063,49 3F,EOR#&3F
98CB,,240 006,F0 06,BEQ 6   --> &98D3
98CD,,169 000,A9 00,LDA#&00
98CF,8,056,38,SEC
98D0,`,096,60,RTS
98D1,,169 004,A9 04,LDA#&04
98D3,H,072,48,PHA
98D4,,230 027,E6 1B,INC &1B
98D6,,032 180 150,20 B4 96,JSR &96B4
98D9,L,076 206 153,4C CE 99,JMP &99CE
98DC,,230 027,E6 1B,INC &1B
98DE,,032 180 150,20 B4 96,JSR &96B4
98E1,+,165 043,A5 2B,LDA &2B
98E3,,240 006,F0 06,BEQ 6   --> &98EB
98E5,,169 128,A9 80,LDA#&80
98E7,",",133 044,85 2C,STA &2C
98E9,8,056,38,SEC
98EA,`,096,60,RTS
98EB,,000,00,BRK
98EC,,008,08,EQUB &08
98ED,$,036 032,24 20,BIT &20

98EF,ra,114 097,72 61,ADC (&61)
98F1,nge,110 103 101,6E 67 65,ROR &6567
98F4,,000,00,EQUB &00
98F5,,165 011,A5 0B,LDA &0B
98F7,,133 025,85 19,STA &19
98F9,,165 012,A5 0C,LDA &0C
98FB,,133 026,85 1A,STA &1A
98FD,,164 010,A4 0A,LDY &0A
98FF,,136,88,DEY
9900,,200,C8,INY
9901,,132 027,84 1B,STY &1B
9903,,177 025,B1 19,"LDA (&19),Y"
9905,,201 032,C9 20,CMP#&20
9907,,240 247,F0 F7,BEQ -9 --> &9900
9909,@,201 064,C9 40,CMP#&40
990B,,144 180,90 B4,BCC -76 --> &98C1
990D,[,201 091,C9 5B,CMP#&5B
990F,,176 026,B0 1A,BCS 26 --> &992B
9911,,010,0A,ASL A
9912,,010,0A,ASL A
9913,*,133 042,85 2A,STA &2A
9915,,200,C8,INY
9916,,177 025,B1 19,"LDA (&19),Y"
9918,%,201 037,C9 25,CMP#&25
991A,,208 015,D0 0F,BNE 15 --> &992B
991C,,169 004,A9 04,LDA#&04
991E,+,133 043,85 2B,STA &2B
9920,,162 004,A2 04,LDX#&04
9922,",",134 044,86 2C,STX &2C
9924,,200,C8,INY
9925,,177 025,B1 19,"LDA (&19),Y"
9927,(,201 040,C9 28,CMP#&28
9929,m,208 109,D0 6D,BNE 109 --> &9998
992B,,162 005,A2 05,LDX#&05
992D,",",134 044,86 2C,STX &2C
992F,,024,18,CLC
9930,,164 026,A4 1A,LDY &1A
9932,,165 027,A5 1B,LDA &1B
9934,,170,AA,TAX
9935,,208 008,D0 08,BNE 8 --> &993F
9937,:,058,3A,DEC A
9938,e,101 025,65 19,ADC &19
993A,,176 009,B0 09,BCS 9 --> &9945

993C,,136,88,DEY
993D,,128 006,80 06,BRA 6 --> &9945
993F,:,058,3A,DEC A
9940,e,101 025,65 19,ADC &19
9942,,144 001,90 01,BCC 1 --> &9945
9944,,200,C8,INY
9945,7,133 055,85 37,STA &37
9947,8,132 056,84 38,STY &38
9949,,160 001,A0 01,LDY#&01
994B,7,177 055,B1 37,"LDA (&37),Y"
994D,A,201 065,C9 41,CMP#&41
994F,,176 026,B0 1A,BCS 26 --> &996B
9951,0,201 048,C9 30,CMP#&30
9953,"""",144 034,90 22,BCC 34 --> &9977
9955,:,201 058,C9 3A,CMP#&3A
9957,,176 030,B0 1E,BCS 30 --> &9977
9959,,232,E8,INX
995A,,200,C8,INY
995B,7,177 055,B1 37,"LDA (&37),Y"
995D,A,201 065,C9 41,CMP#&41
995F,,176 010,B0 0A,BCS 10 --> &996B
9961,0,201 048,C9 30,CMP#&30
9963,,144 018,90 12,BCC 18 --> &9977
9965,:,201 058,C9 3A,CMP#&3A
9967,,144 240,90 F0,BCC -16 --> &9959
9969,,128 012,80 0C,BRA 12 --> &9977
996B,[,201 091,C9 5B,CMP#&5B
996D,,144 234,90 EA,BCC -22 --> &9959
996F,_,201 095,C9 5F,CMP#&5F
9971,,144 004,90 04,BCC 4 --> &9977
9973,{,201 123,C9 7B,CMP#&7B
9975,,144 226,90 E2,BCC -30 --> &9959
9977,,192 001,C0 01,CPY#&01
9979,+,240 043,F0 2B,BEQ 43 --> &99A6
997B,$,201 036,C9 24,CMP#&24
997D,[,240 091,F0 5B,BEQ 91 --> &99DA
997F,%,201 037,C9 25,CMP#&25
9981,,208 006,D0 06,BNE 6 --> &9989
9983,",",198 044,C6 2C,DEC &2C
9985,,232,E8,INX
9986,,200,C8,INY
9987,7,177 055,B1 37,"LDA (&37),Y"
9989,(,201 040,C9 28,CMP#&28

998B,H,240 072,F0 48,BEQ 72   --> &99D5
998D,,032 133 128,20 85 80,JSR &8085
9990,,240 024,F0 18,BEQ 24   --> &99AA
9992,,134 027,86 1B,STX &1B
9994,,164 027,A4 1B,LDY &1B
9996,,177 025,B1 19,"LDA (&19),Y"
9998,!,201 033,C9 21,CMP#&21
999A,,240 018,F0 12,BEQ 18   --> &99AE
999C,I?,073 063,49 3F,EOR#&3F
999E,,240 016,F0 10,BEQ 16   --> &99B0
99A0,,024,18,CLC
99A1,,132 027,84 1B,STY &1B
99A3,,169 255,A9 FF,LDA#&FF
99A5,`,096,60,RTS
99A6,,169 000,A9 00,LDA#&00
99A8,8,056,38,SEC
99A9,`,096,60,RTS
99AA,,169 000,A9 00,LDA#&00
99AC,,024,18,CLC
99AD,`,096,60,RTS
99AE,,169 004,A9 04,LDA#&04
99B0,H,072,48,PHA
99B1,,200,C8,INY
99B2,,132 027,84 1B,STY &1B
99B4,,032 160 177,20 A0 B1,JSR &B1A0
99B7,,032 191 150,20 BF 96,JSR &96BF
99BA,+,165 043,A5 2B,LDA &2B
99BC,H,072,48,PHA
99BD,*,165 042,A5 2A,LDA &2A
99BF,H,072,48,PHA
99C0,,032 180 150,20 B4 96,JSR &96B4
99C3,,024,18,CLC
99C4,h,104,68,PLA
99C5,e*,101 042,65 2A,ADC &2A
99C7,*,133 042,85 2A,STA &2A
99C9,h,104,68,PLA
99CA,e+,101 043,65 2B,ADC &2B
99CC,+,133 043,85 2B,STA &2B
99CE,h,104,68,PLA
99CF,",",133 044,85 2C,STA &2C
99D1,,024,18,CLC
99D2,,169 255,A9 FF,LDA#&FF
99D4,`,096,60,RTS

99D5,,032 254 153,20 FE 99,JSR &99FE
99D8,,128 186,80 BA,BRA -70 --> &9994
99DA,",",198 044,C6 2C,DEC &2C
99DC,,232,E8,INX
99DD,,200,C8,INY
99DE,7,177 055,B1 37,"LDA (&37),Y"
99E0,(,201 040,C9 28,CMP#&28
99E2,,240 013,F0 0D,BEQ 13 --> &99F1
99E4,,032 133 128,20 85 80,JSR &8085
99E7,,240 193,F0 C1,BEQ -63 --> &99AA
99E9,,134 027,86 1B,STX &1B
99EB,,169 129,A9 81,LDA#&81
99ED,",",133 044,85 2C,STA &2C
99EF,8,056,38,SEC
99F0,`,096,60,RTS
99F1,,032 254 153,20 FE 99,JSR &99FE
99F4,,128 245,80 F5,BRA -11 --> &99EB
99F6,,000,00,BRK
99F7,,014,0E,EQUB &0E
99F8,Ar,065 114,41 72,xxx Invalid Code
99FA,ra,114 097,72 61,ADC (&61)
99FC,y,121,79,xxx Invalid Code
99FD,,000,00,EQUB &00
99FE,,232,E8,INX
99FF,,200,C8,INY
9A00,,032 133 128,20 85 80,JSR &8085
9A03,,240 241,F0 F1,BEQ -15 --> &99F6
9A05,,134 027,86 1B,STX &1B
9A07,",",165 044,A5 2C,LDA &2C
9A09,H,072,48,PHA
9A0A,*,165 042,A5 2A,LDA &2A
9A0C,H,072,48,PHA
9A0D,+,165 043,A5 2B,LDA &2B
9A0F,H,072,48,PHA
9A10,*,178 042,B2 2A,LDA (&2A)
9A12,,201 004,C9 04,CMP#&04
9A14,o,144 111,90 6F,BCC 111 --> &9A85
9A16,,032 232 171,20 E8 AB,JSR &ABE8
9A19,,169 001,A9 01,LDA#&01
9A1B,-,133 045,85 2D,STA &2D
9A1D,&,032 038 188,20 26 BC,JSR &BC26
9A20,,032 175 150,20 AF 96,JSR &96AF
9A23,,230 027,E6 1B,INC &1B

9A25,",",224 044,E0 2C,CPX#&2C
9A27,,208 205,D0 CD,BNE -51 --> &99F6
9A29,9,162 057,A2 39,LDX#&39
9A2B,,032 008 189,20 08 BD,JSR &BD08
9A2E,<,164 060,A4 3C,LDY &3C
9A30,h,104,68,PLA
9A31,8,133 056,85 38,STA &38
9A33,h,104,68,PLA
9A34,7,133 055,85 37,STA &37
9A36,H,072,48,PHA
9A37,8,165 056,A5 38,LDA &38
9A39,H,072,48,PHA
9A3A,,032 211 154,20 D3 9A,JSR &9AD3
9A3D,-,132 045,84 2D,STY &2D
9A3F,7,177 055,B1 37,"LDA (&37),Y"
9A41,?,133 063,85 3F,STA &3F
9A43,,200,C8,INY
9A44,7,177 055,B1 37,"LDA (&37),Y"
9A46,@,133 064,85 40,STA &40
9A48,*,165 042,A5 2A,LDA &2A
9A4A,e9,101 057,65 39,ADC &39
9A4C,*,133 042,85 2A,STA &2A
9A4E,+,165 043,A5 2B,LDA &2B
9A50,e:,101 058,65 3A,ADC &3A
9A52,+,133 043,85 2B,STA &2B
9A54,,032 008 149,20 08 95,JSR &9508
9A57,8,056,38,SEC
9A58,7,178 055,B2 37,LDA (&37)
9A5A,-,229 045,E5 2D,SBC &2D
9A5C,,201 003,C9 03,CMP#&03
9A5E,,176 189,B0 BD,BCS -67 --> &9A1D
9A60,&,032 038 188,20 26 BC,JSR &BC26
9A63,,032 167 150,20 A7 96,JSR &96A7
9A66,h,104,68,PLA
9A67,8,133 056,85 38,STA &38
9A69,h,104,68,PLA
9A6A,7,133 055,85 37,STA &37
9A6C,9,162 057,A2 39,LDX#&39
9A6E,,032 008 189,20 08 BD,JSR &BD08
9A71,<,164 060,A4 3C,LDY &3C
9A73,,032 211 154,20 D3 9A,JSR &9AD3
9A76,,024,18,CLC
9A77,9,165 057,A5 39,LDA &39

9A79,e*,101 042,65 2A,ADC &2A
9A7B,*,133 042,85 2A,STA &2A
9A7D,:,165 058,A5 3A,LDA &3A
9A7F,e+,101 043,65 2B,ADC &2B
9A81,+,133 043,85 2B,STA &2B
9A83,,144 017,90 11,BCC 17 --> &9A96
9A85,,032 172 173,20 AC AD,JSR &ADAC
9A88,,032 191 150,20 BF 96,JSR &96BF
9A8B,h,104,68,PLA
9A8C,8,133 056,85 38,STA &38
9A8E,h,104,68,PLA
9A8F,7,133 055,85 37,STA &37
9A91,,160 001,A0 01,LDY#&01
9A93,,032 211 154,20 D3 9A,JSR &9AD3
9A96,h,104,68,PLA
9A97,",",133 044,85 2C,STA &2C
9A99,,201 005,C9 05,CMP#&05
9A9B,,208 023,D0 17,BNE 23 --> &9AB4
9A9D,+,166 043,A6 2B,LDX &2B
9A9F,*,165 042,A5 2A,LDA &2A
9AA1,*,006 042,06 2A,ASL &2A
9AA3,&+,038 043,26 2B,ROL &2B
9AA5,*,006 042,06 2A,ASL &2A
9AA7,&+,038 043,26 2B,ROL &2B
9AA9,e*,101 042,65 2A,ADC &2A
9AAB,*,133 042,85 2A,STA &2A
9AAD,,138,8A,TXA
9AAE,e+,101 043,65 2B,ADC &2B
9AB0,+,133 043,85 2B,STA &2B
9AB2,,128 008,80 08,BRA 8 --> &9ABC
9AB4,*,006 042,06 2A,ASL &2A
9AB6,&+,038 043,26 2B,ROL &2B
9AB8,*,006 042,06 2A,ASL &2A
9ABA,&+,038 043,26 2B,ROL &2B
9ABC,,152,98,TYA
9ABD,e*,101 042,65 2A,ADC &2A
9ABF,*,133 042,85 2A,STA &2A
9AC1,,144 003,90 03,BCC 3 --> &9AC6
9AC3,+,230 043,E6 2B,INC &2B
9AC5,,024,18,CLC
9AC6,7,165 055,A5 37,LDA &37
9AC8,e*,101 042,65 2A,ADC &2A
9ACA,*,133 042,85 2A,STA &2A

9ACC,8,165 056,A5 38,LDA &38
9ACE,e+,101 043,65 2B,ADC &2B
9AD0,+,133 043,85 2B,STA &2B
9AD2,`,096,60,RTS
9AD3,+,165 043,A5 2B,LDA &2B
9AD5,),041 192,29 C0,AND#&C0
9AD7,",",005 044,05 2C,ORA &2C
9AD9,-,005 045,05 2D,ORA &2D
9ADB,,208 013,D0 0D,BNE 13 --> &9AEA
9ADD,*,165 042,A5 2A,LDA &2A
9ADF,7,209 055,D1 37,"CMP (&37),Y"
9AE1,,200,C8,INY
9AE2,+,165 043,A5 2B,LDA &2B
9AE4,7,241 055,F1 37,"SBC (&37),Y"
9AE6,,176 002,B0 02,BCS 2 --> &9AEA
9AE8,,200,C8,INY
9AE9,`,096,60,RTS
9AEA,,000,00,BRK
9AEB,,015,0F,EQUB &0F
9AEC,S,083,53,xxx Invalid Code
9AED,ub,117 098,75 62,"ADC &62,X"
9AEF,s,115,73,xxx Invalid Code
9AF0,c,099,63,xxx Invalid Code
9AF1,ri,114 105,72 69,ADC (&69)
9AF3,pt,112 116,70 74,BVS 116 --> &9B69
9AF5,,000,00,EQUB &00
9AF6,,160 001,A0 01,LDY#&01
9AF8,7,177 055,B1 37,"LDA (&37),Y"
9AFA,0,201 048,C9 30,CMP#&30
9AFC,,144 024,90 18,BCC 24 --> &9B16
9AFE,@,201 064,C9 40,CMP#&40
9B00,,176 012,B0 0C,BCS 12 --> &9B0E
9B02,:,201 058,C9 3A,CMP#&3A
9B04,,176 016,B0 10,BCS 16 --> &9B16
9B06,,192 001,C0 01,CPY#&01
9B08,,240 012,F0 0C,BEQ 12 --> &9B16
9B0A,,232,E8,INX
9B0B,,200,C8,INY
9B0C,,208 234,D0 EA,BNE -22 --> &9AF8
9B0E,_,201 095,C9 5F,CMP#&5F
9B10,,176 005,B0 05,BCS 5 --> &9B17
9B12,[,201 091,C9 5B,CMP#&5B
9B14,,144 244,90 F4,BCC -12 --> &9B0A

```
9B16,`,096,60,RTS
9B17,{,201 123,C9 7B,CMP#&7B
9B19,,144 239,90 EF,BCC -17   --> &9B0A
9B1B,`,096,60,RTS
9B1C,,230 010,E6 0A,INC &0A
9B1E,,164 010,A4 0A,LDY &0A
9B20,,177 011,B1 0B,"LDA (&0B),Y"
9B22,,201 032,C9 20,CMP#&20
9B24,,240 246,F0 F6,BEQ -10   --> &9B1C
9B26,,201 141,C9 8D,CMP#&8D
9B28,,208 026,D0 1A,BNE 26   --> &9B44
9B2A,,200,C8,INY
9B2B,,177 011,B1 0B,"LDA (&0B),Y"
9B2D,,010,0A,ASL A
9B2E,,010,0A,ASL A
9B2F,,170,AA,TAX
9B30,),041 192,29 C0,AND#&C0
9B32,,200,C8,INY
9B33,Q,081 011,51 0B,"EOR (&0B),Y"
9B35,*,133 042,85 2A,STA &2A
9B37,,138,8A,TXA
9B38,,010,0A,ASL A
9B39,,010,0A,ASL A
9B3A,,200,C8,INY
9B3B,Q,081 011,51 0B,"EOR (&0B),Y"
9B3D,+,133 043,85 2B,STA &2B
9B3F,,200,C8,INY
9B40,,132 010,84 0A,STY &0A
9B42,8,056,38,SEC
9B43,`,096,60,RTS
9B44,,024,18,CLC
9B45,`,096,60,RTS
9B46,,165 011,A5 0B,LDA &0B
9B48,,133 025,85 19,STA &19
9B4A,,165 012,A5 0C,LDA &0C
9B4C,,133 026,85 1A,STA &1A
9B4E,,165 010,A5 0A,LDA &0A
9B50,,133 027,85 1B,STA &1B
9B52,,164 027,A4 1B,LDY &1B
9B54,,230 027,E6 1B,INC &1B
9B56,,177 025,B1 19,"LDA (&19),Y"
9B58,,201 032,C9 20,CMP#&20
9B5A,,240 246,F0 F6,BEQ -10   --> &9B52
```

9B5C,=,201 061,C9 3D,CMP#&3D
9B5E,.,240 046,F0 2E,BEQ 46   --> &9B8E
9B60,,000,00,BRK
9B61,,004,04,EQUB &04
9B62,M,077,4D,xxx Invalid Code
9B63,is,105 115,69 73,ADC#&73
9B65,ta,116 097,74 61,"STZ &61,X"
9B67,k,107,6B,xxx Invalid Code
9B68,e,101,65,xxx Invalid Code
9B69,,000,00,BRK
9B6A,,016,10,EQUB &10
9B6B,S,083,53,xxx Invalid Code
9B6C,ynt,121 110 116,79 6E 74,"ADC &746E,Y"
9B6F,ax,097 120 032,61 78 20,"ADC (&2078,X)"
9B72,er,101 114,65 72,ADC &72
9B74,ro,114 111,72 6F,ADC (&6F)
9B76,r,114,72,xxx Invalid Code
9B77,,000,00,BRK
9B78,,013,0D,EQUB &0D
9B79,No,078 111,4E 6F,xxx Invalid Code
9B7B,,032 242,20 F2,xxx Invalid Code
9B7D,,000,00,BRK
9B7E,,017,11,EQUB &11
9B7F,E,069,45,xxx Invalid Code
9B80,s,115,73,xxx Invalid Code
9B81,c,099,63,xxx Invalid Code
9B82,ape,097 112 101,61 70 65,"ADC (&6570,X)"
9B85,,000,00,EQUB &00
9B86,,032 213 142,20 D5 8E,JSR &8ED5
9B89,=,201 061,C9 3D,CMP#&3D
9B8B,,208 211,D0 D3,BNE -45   --> &9B60
9B8D,`,096,60,RTS
9B8E,;,032 059 157,20 3B 9D,JSR &9D3B
9B91,,138,8A,TXA
9B92,,164 027,A4 1B,LDY &1B
9B94,,128 026,80 1A,BRA 26   --> &9BB0
9B96,,164 027,A4 1B,LDY &1B
9B98,,128 014,80 0E,BRA 14   --> &9BA8
9B9A,,186,BA,TSX
9B9B,,224 252,E0 FC,CPX#&FC
9B9D,,176 216,B0 D8,BCS -40   --> &9B77
9B9F,,173 255 001,AD FF 01,LDA &01FF
9BA2,,201 242,C9 F2,CMP#&F2

9BA4,,208 209,D0 D1,BNE -47   --> &9B77
9BA6,,164 010,A4 0A,LDY &0A
9BA8,,136,88,DEY
9BA9,,200,C8,INY
9BAA,,177 011,B1 0B,"LDA (&0B),Y"
9BAC,,201 032,C9 20,CMP#&20
9BAE,,240 249,F0 F9,BEQ -7   --> &9BA9
9BB0,:,201 058,C9 3A,CMP#&3A
9BB2,,240 008,F0 08,BEQ 8   --> &9BBC
9BB4,,201 013,C9 0D,CMP#&0D
9BB6,,240 004,F0 04,BEQ 4   --> &9BBC
9BB8,,201 139,C9 8B,CMP#&8B
9BBA,,208 173,D0 AD,BNE -83   --> &9B69
9BBC,,024,18,CLC
9BBD,,152,98,TYA
9BBE,e,101 011,65 0B,ADC &0B
9BC0,,133 011,85 0B,STA &0B
9BC2,,144 002,90 02,BCC 2   --> &9BC6
9BC4,,230 012,E6 0C,INC &0C
9BC6,,160 001,A0 01,LDY#&01
9BC8,,132 010,84 0A,STY &0A
9BCA,$,036 255,24 FF,BIT &FF
9BCC,0,048 175,30 AF,BMI -81   --> &9B7D
9BCE,`,096,60,RTS
9BCF,,032 166 155,20 A6 9B,JSR &9BA6
9BD2,,178 011,B2 0B,LDA (&0B)
9BD4,:,201 058,C9 3A,CMP#&3A
9BD6,,240 246,F0 F6,BEQ -10   --> &9BCE
9BD8,,165 012,A5 0C,LDA &0C
9BDA,,201 007,C9 07,CMP#&07
9BDC,$,240 036,F0 24,BEQ 36   --> &9C02
9BDE,,160 001,A0 01,LDY#&01
9BE0,,177 011,B1 0B,"LDA (&0B),Y"
9BE2,0,048 030,30 1E,BMI 30   --> &9C02
9BE4,,166 032,A6 20,LDX &20
9BE6,,240 010,F0 0A,BEQ 10   --> &9BF2
9BE8,+,133 043,85 2B,STA &2B
9BEA,,200,C8,INY
9BEB,,177 011,B1 0B,"LDA (&0B),Y"
9BED,*,133 042,85 2A,STA &2A
9BEF,K,032 075 156,20 4B 9C,JSR &9C4B
9BF2,,169 003,A9 03,LDA#&03
9BF4,,024,18,CLC

9BF5,e,101 011,65 0B,ADC &0B
9BF7,,133 011,85 0B,STA &0B
9BF9,,144 002,90 02,BCC 2   --> &9BFD
9BFB,,230 012,E6 0C,INC &0C
9BFD,,160 001,A0 01,LDY#&01
9BFF,,132 010,84 0A,STY &0A
9C01,`,096,60,RTS
9C02,L,076 134 143,4C 86 8F,JMP &8F86
9C05,L,076 146 144,4C 92 90,JMP &9092
9C08,/,032 047 157,20 2F 9D,JSR &9D2F
9C0B,,240 248,F0 F8,BEQ -8   --> &9C05
9C0D,,016 003,10 03,BPL 3   --> &9C12
9C0F,,032 195 150,20 C3 96,JSR &96C3
9C12,,164 027,A4 1B,LDY &1B
9C14,,132 010,84 0A,STY &0A
9C16,*,165 042,A5 2A,LDA &2A
9C18,+,005 043,05 2B,ORA &2B
9C1A,",",005 044,05 2C,ORA &2C
9C1C,-,005 045,05 2D,ORA &2D
9C1E,,240 023,F0 17,BEQ 23   --> &9C37
9C20,,224 140,E0 8C,CPX#&8C
9C22,,240 003,F0 03,BEQ 3   --> &9C27
9C24,L,076 011 144,4C 0B 90,JMP &900B
9C27,,230 010,E6 0A,INC &0A
9C29,,032 030 155,20 1E 9B,JSR &9B1E
9C2C,,144 246,90 F6,BCC -10   --> &9C24
9C2E,6,032 054 184,20 36 B8,JSR &B836
9C31,,032 198 155,20 C6 9B,JSR &9BC6
9C34,L#,076 035 183,4C 23 B7,JMP &B723
9C37,,164 010,A4 0A,LDY &0A
9C39,,177 011,B1 0B,"LDA (&0B),Y"
9C3B,,201 013,C9 0D,CMP#&0D
9C3D,,240 009,F0 09,BEQ 9   --> &9C48
9C3F,,200,C8,INY
9C40,,201 139,C9 8B,CMP#&8B
9C42,,208 245,D0 F5,BNE -11   --> &9C39
9C44,,132 010,84 0A,STY &0A
9C46,,240 225,F0 E1,BEQ -31   --> &9C29
9C48,L,076 184 143,4C B8 8F,JMP &8FB8
9C4B,*,165 042,A5 2A,LDA &2A
9C4D,!,197 033,C5 21,CMP &21
9C4F,+,165 043,A5 2B,LDA &2B
9C51,"""",229 034,E5 22,SBC &22

```
9C53,,176 172,B0 AC,BCS -84   --> &9C01
9C55,[,169 091,A9 5B,LDA#&5B
9C57,,032 152 189,20 98 BD,JSR &BD98
9C5A,,032 129 160,20 81 A0,JSR &A081
9C5D,],169 093,A9 5D,LDA#&5D
9C5F,,032 152 189,20 98 BD,JSR &BD98
9C62,L,076 146 189,4C 92 BD,JMP &BD92
9C65,h,104,68,PLA
9C66,*,133 042,85 2A,STA &2A
9C68,h,104,68,PLA
9C69,+,133 043,85 2B,STA &2B
9C6B,h,104,68,PLA
9C6C,",",133 044,85 2C,STA &2C
9C6E,h,104,68,PLA
9C6F,-,133 045,85 2D,STA &2D
9C71,,032 250 187,20 FA BB,JSR &BBFA
9C74,,032 133 129,20 85 81,JSR &8185
9C77,,032 011 164,20 0B A4,JSR &A40B
9C7A,,032 232 187,20 E8 BB,JSR &BBE8
9C7D,A,032 065 165,20 41 A5,JSR &A541
9C80,,128 016,80 10,BRA 16   --> &9C92
9C82,,032 250 187,20 FA BB,JSR &BBFA
9C85,L,032 076 158,20 4C 9E,JSR &9E4C
9C88,,168,A8,TAY
9C89,,032 221 150,20 DD 96,JSR &96DD
9C8C,,032 232 187,20 E8 BB,JSR &BBE8
9C8F,,032 224 164,20 E0 A4,JSR &A4E0
9C92,,160 000,A0 00,LDY#&00
9C94,,169 127,A9 7F,LDA#&7F
9C96,;,020 059,14 3B,TRB &3B
9C98,.,165 046,A5 2E,LDA &2E
9C9A,),041 128,29 80,AND#&80
9C9C,;,197 059,C5 3B,CMP &3B
9C9E,,208 030,D0 1E,BNE 30   --> &9CBE
9CA0,<,165 060,A5 3C,LDA &3C
9CA2,0,197 048,C5 30,CMP &30
9CA4,,208 025,D0 19,BNE 25   --> &9CBF
9CA6,=,165 061,A5 3D,LDA &3D
9CA8,1,197 049,C5 31,CMP &31
9CAA,,208 019,D0 13,BNE 19   --> &9CBF
9CAC,>,165 062,A5 3E,LDA &3E
9CAE,2,197 050,C5 32,CMP &32
9CB0,,208 013,D0 0D,BNE 13   --> &9CBF
```

```
9CB2,?,165 063,A5 3F,LDA &3F
9CB4,3,197 051,C5 33,CMP &33
9CB6,,208 007,D0 07,BNE 7   --> &9CBF
9CB8,@,165 064,A5 40,LDA &40
9CBA,4,197 052,C5 34,CMP &34
9CBC,,208 001,D0 01,BNE 1   --> &9CBF
9CBE,`,096,60,RTS
9CBF,j,106,6A,ROR A
9CC0,E;,069 059,45 3B,EOR &3B
9CC2,*,042,2A,ROL A
9CC3,,169 001,A9 01,LDA#&01
9CC5,`,096,60,RTS
9CC6,L,076 146 144,4C 92 90,JMP &9092
9CC9,,138,8A,TXA
9CCA,6,240 054,F0 36,BEQ 54   --> &9D02
9CCC,0,048 180,30 B4,BMI -76   --> &9C82
9CCE,-,165 045,A5 2D,LDA &2D
9CD0,H,072,48,PHA
9CD1,",",165 044,A5 2C,LDA &2C
9CD3,H,072,48,PHA
9CD4,+,165 043,A5 2B,LDA &2B
9CD6,H,072,48,PHA
9CD7,*,165 042,A5 2A,LDA &2A
9CD9,H,072,48,PHA
9CDA,L,032 076 158,20 4C 9E,JSR &9E4C
9CDD,,168,A8,TAY
9CDE,,240 230,F0 E6,BEQ -26   --> &9CC6
9CE0,0,048 131,30 83,BMI -125   --> &9C65
9CE2,-,165 045,A5 2D,LDA &2D
9CE4,I,073 128,49 80,EOR#&80
9CE6,-,133 045,85 2D,STA &2D
9CE8,8,056,38,SEC
9CE9,h,104,68,PLA
9CEA,*,229 042,E5 2A,SBC &2A
9CEC,*,133 042,85 2A,STA &2A
9CEE,h,104,68,PLA
9CEF,+,229 043,E5 2B,SBC &2B
9CF1,*,004 042,04 2A,TSB &2A
9CF3,h,104,68,PLA
9CF4,",",229 044,E5 2C,SBC &2C
9CF6,*,004 042,04 2A,TSB &2A
9CF8,h,104,68,PLA
9CF9,,160 000,A0 00,LDY#&00
```

9CFB,I,073 128,49 80,EOR#&80
9CFD,-,229 045,E5 2D,SBC &2D
9CFF,*,005 042,05 2A,ORA &2A
9D01,`,096,60,RTS
9D02,Q,032 081 188,20 51 BC,JSR &BC51
9D05,L,032 076 158,20 4C 9E,JSR &9E4C
9D08,,168,A8,TAY
9D09,,208 187,D0 BB,BNE -69   --> &9CC6
9D0B,,178 004,B2 04,LDA (&04)
9D0D,6,197 054,C5 36,CMP &36
9D0F,,144 002,90 02,BCC 2   --> &9D13
9D11,6,165 054,A5 36,LDA &36
9D13,7,133 055,85 37,STA &37
9D15,7,196 055,C4 37,CPY &37
9D17,,240 010,F0 0A,BEQ 10   --> &9D23
9D19,,200,C8,INY
9D1A,,177 004,B1 04,"LDA (&04),Y"
9D1C,,217 255 005,D9 FF 05,"CMP &05FF,Y"
9D1F,,240 244,F0 F4,BEQ -12   --> &9D15
9D21,,128 004,80 04,BRA 4   --> &9D27
9D23,,178 004,B2 04,LDA (&04)
9D25,6,197 054,C5 36,CMP &36
9D27,,008,08,PHP
9D28,,032 225 188,20 E1 BC,JSR &BCE1
9D2B,,160 000,A0 00,LDY#&00
9D2D,(,040,28,PLP
9D2E,`,096,60,RTS
9D2F,,165 011,A5 0B,LDA &0B
9D31,,133 025,85 19,STA &19
9D33,,165 012,A5 0C,LDA &0C
9D35,,133 026,85 1A,STA &1A
9D37,,165 010,A5 0A,LDA &0A
9D39,,133 027,85 1B,STA &1B
9D3B,,032 129 157,20 81 9D,JSR &9D81
9D3E,,224 132,E0 84,CPX#&84
9D40,,240 010,F0 0A,BEQ 10   --> &9D4C
9D42,,224 130,E0 82,CPX#&82
9D44,,240 032,F0 20,BEQ 32   --> &9D66
9D46,,198 027,C6 1B,DEC &1B
9D48,,168,A8,TAY
9D49,',133 039,85 27,STA &27
9D4B,`,096,60,RTS
9D4C,{,032 123 157,20 7B 9D,JSR &9D7B

9D4F,,032 190 150,20 BE 96,JSR &96BE
9D52,,160 003,A0 03,LDY#&03
9D54,,177 004,B1 04,"LDA (&04),Y"
9D56,*,025 042 000,19 2A 00,"ORA &002A,Y"
9D59,*,153 042 000,99 2A 00,"STA &002A,Y"
9D5C,,136,88,DEY
9D5D,,016 245,10 F5,BPL -11 --> &9D54
9D5F,,032 250 188,20 FA BC,JSR &BCFA
9D62,@,169 064,A9 40,LDA#&40
9D64,,128 216,80 D8,BRA -40 --> &9D3E
9D66,{,032 123 157,20 7B 9D,JSR &9D7B
9D69,,032 190 150,20 BE 96,JSR &96BE
9D6C,,160 003,A0 03,LDY#&03
9D6E,,177 004,B1 04,"LDA (&04),Y"
9D70,Y*,089 042 000,59 2A 00,"EOR &002A,Y"
9D73,*,153 042 000,99 2A 00,"STA &002A,Y"
9D76,,136,88,DEY
9D77,,016 245,10 F5,BPL -11 --> &9D6E
9D79,,128 228,80 E4,BRA -28 --> &9D5F
9D7B,,032 190 150,20 BE 96,JSR &96BE
9D7E,&,032 038 188,20 26 BC,JSR &BC26
9D81,,032 169 157,20 A9 9D,JSR &9DA9
9D84,,224 128,E0 80,CPX#&80
9D86,,240 001,F0 01,BEQ 1 --> &9D89
9D88,`,096,60,RTS
9D89,,032 190 150,20 BE 96,JSR &96BE
9D8C,&,032 038 188,20 26 BC,JSR &BC26
9D8F,,032 169 157,20 A9 9D,JSR &9DA9
9D92,,032 190 150,20 BE 96,JSR &96BE
9D95,,160 003,A0 03,LDY#&03
9D97,,177 004,B1 04,"LDA (&04),Y"
9D99,9*,057 042 000,39 2A 00,"AND &002A,Y"
9D9C,*,153 042 000,99 2A 00,"STA &002A,Y"
9D9F,,136,88,DEY
9DA0,,016 245,10 F5,BPL -11 --> &9D97
9DA2,,032 250 188,20 FA BC,JSR &BCFA
9DA5,@,169 064,A9 40,LDA#&40
9DA7,,128 219,80 DB,BRA -37 --> &9D84
9DA9,L,032 076 158,20 4C 9E,JSR &9E4C
9DAC,?,224 063,E0 3F,CPX#&3F
9DAE,,176 004,B0 04,BCS 4 --> &9DB4
9DB0,<,224 060,E0 3C,CPX#&3C
9DB2,,176 001,B0 01,BCS 1 --> &9DB5

```
9DB4,`,096,60,RTS
9DB5,,240 022,F0 16,BEQ 22   --> &9DCD
9DB7,>,224 062,E0 3E,CPX#&3E
9DB9,:,240 058,F0 3A,BEQ 58   --> &9DF5
9DBB,,170,AA,TAX
9DBC,,032 202 156,20 CA 9C,JSR &9CCA
9DBF,,208 001,D0 01,BNE 1   --> &9DC2
9DC1,,136,88,DEY
9DC2,*,132 042,84 2A,STY &2A
9DC4,+,132 043,84 2B,STY &2B
9DC6,",",132 044,84 2C,STY &2C
9DC8,-,132 045,84 2D,STY &2D
9DCA,@,169 064,A9 40,LDA#&40
9DCC,`,096,60,RTS
9DCD,,170,AA,TAX
9DCE,,164 027,A4 1B,LDY &1B
9DD0,,177 025,B1 19,"LDA (&19),Y"
9DD2,=,201 061,C9 3D,CMP#&3D
9DD4,,240 011,F0 0B,BEQ 11   --> &9DE1
9DD6,>,201 062,C9 3E,CMP#&3E
9DD8,,240 018,F0 12,BEQ 18   --> &9DEC
9DDA,,032 201 156,20 C9 9C,JSR &9CC9
9DDD,,144 226,90 E2,BCC -30   --> &9DC1
9DDF,,128 225,80 E1,BRA -31   --> &9DC2
9DE1,,230 027,E6 1B,INC &1B
9DE3,,032 201 156,20 C9 9C,JSR &9CC9
9DE6,,240 217,F0 D9,BEQ -39   --> &9DC1
9DE8,,144 215,90 D7,BCC -41   --> &9DC1
9DEA,,128 214,80 D6,BRA -42   --> &9DC2
9DEC,,230 027,E6 1B,INC &1B
9DEE,,032 201 156,20 C9 9C,JSR &9CC9
9DF1,,208 206,D0 CE,BNE -50   --> &9DC1
9DF3,,128 205,80 CD,BRA -51   --> &9DC2
9DF5,,170,AA,TAX
9DF6,,164 027,A4 1B,LDY &1B
9DF8,,177 025,B1 19,"LDA (&19),Y"
9DFA,=,201 061,C9 3D,CMP#&3D
9DFC,,240 009,F0 09,BEQ 9   --> &9E07
9DFE,,032 201 156,20 C9 9C,JSR &9CC9
9E01,,240 191,F0 BF,BEQ -65   --> &9DC2
9E03,,176 188,B0 BC,BCS -68   --> &9DC1
9E05,,128 187,80 BB,BRA -69   --> &9DC2
9E07,,230 027,E6 1B,INC &1B
```

9E09,,032 201 156,20 C9 9C,JSR &9CC9
9E0C,,176 179,B0 B3,BCS -77   --> &9DC1
9E0E,,128 178,80 B2,BRA -78   --> &9DC2
9E10,,000,00,BRK
9E11,,019,13,EQUB &13
9E12,S,083,53,xxx Invalid Code
9E13,tr,116 114,74 72,"STZ &72,X"
9E15,in,105 110,69 6E,ADC#&6E
9E17,g,103,67,xxx Invalid Code
9E18,to,032 116 111,20 74 6F,JSR &6F74
9E1B,o,111,6F,xxx Invalid Code
9E1C,lo,032 108 111,20 6C 6F,JSR &6F6C
9E1F,ng,110 103,6E 67,xxx Invalid Code
9E21,,000,00,EQUB &00
9E22,Q,032 081 188,20 51 BC,JSR &BC51
9E25,,032 018 160,20 12 A0,JSR &A012
9E28,,168,A8,TAY
9E29,f,208 102,D0 66,BNE 102   --> &9E91
9E2B,,024,18,CLC
9E2C,,218,DA,PHX
9E2D,,178 004,B2 04,LDA (&04)
9E2F,e6,101 054,65 36,ADC &36
9E31,,176 221,B0 DD,BCS -35   --> &9E10
9E33,,170,AA,TAX
9E34,H,072,48,PHA
9E35,6,164 054,A4 36,LDY &36
9E37,,185 255 005,B9 FF 05,"LDA &05FF,Y"
9E3A,,157 255 005,9D FF 05,"STA &05FF,X"
9E3D,,202,CA,DEX
9E3E,,136,88,DEY
9E3F,,208 246,D0 F6,BNE -10   --> &9E37
9E41,,032 210 188,20 D2 BC,JSR &BCD2
9E44,h,104,68,PLA
9E45,6,133 054,85 36,STA &36
9E47,,250,FA,PLX
9E48,,169 000,A9 00,LDA#&00
9E4A,,128 003,80 03,BRA 3   --> &9E4F
9E4C,,032 199 159,20 C7 9F,JSR &9FC7
9E4F,+,224 043,E0 2B,CPX#&2B
9E51,,240 005,F0 05,BEQ 5   --> &9E58
9E53,-,224 045,E0 2D,CPX#&2D
9E55,f,240 102,F0 66,BEQ 102   --> &9EBD
9E57,`,096,60,RTS

9E58,,168,A8,TAY
9E59,,240 199,F0 C7,BEQ -57 --> &9E22
9E5B,07,048 055,30 37,BMI 55 --> &9E94
9E5D,,032 196 159,20 C4 9F,JSR &9FC4
9E60,,168,A8,TAY
9E61,.,240 046,F0 2E,BEQ 46 --> &9E91
9E63,0K,048 075,30 4B,BMI 75 --> &9EB0
9E65,,024,18,CLC
9E66,,178 004,B2 04,LDA (&04)
9E68,e*,101 042,65 2A,ADC &2A
9E6A,*,133 042,85 2A,STA &2A
9E6C,,160 001,A0 01,LDY#&01
9E6E,,177 004,B1 04,"LDA (&04),Y"
9E70,e+,101 043,65 2B,ADC &2B
9E72,+,133 043,85 2B,STA &2B
9E74,,200,C8,INY
9E75,,177 004,B1 04,"LDA (&04),Y"
9E77,"e,",101 044,65 2C,ADC &2C
9E79,",",133 044,85 2C,STA &2C
9E7B,,200,C8,INY
9E7C,,177 004,B1 04,"LDA (&04),Y"
9E7E,e-,101 045,65 2D,ADC &2D
9E80,-,133 045,85 2D,STA &2D
9E82,,024,18,CLC
9E83,,165 004,A5 04,LDA &04
9E85,i,105 004,69 04,ADC#&04
9E87,,133 004,85 04,STA &04
9E89,@,169 064,A9 40,LDA#&40
9E8B,,144 194,90 C2,BCC -62 --> &9E4F
9E8D,,230 005,E6 05,INC &05
9E8F,,128 190,80 BE,BRA -66 --> &9E4F
9E91,L,076 146 144,4C 92 90,JMP &9092
9E94,,032 250 187,20 FA BB,JSR &BBFA
9E97,,032 199 159,20 C7 9F,JSR &9FC7
9E9A,,168,A8,TAY
9E9B,,240 244,F0 F4,BEQ -12 --> &9E91
9E9D,',134 039,86 27,STX &27
9E9F,0,048 003,30 03,BMI 3 --> &9EA4
9EA1,,032 133 129,20 85 81,JSR &8185
9EA4,,032 232 187,20 E8 BB,JSR &BBE8
9EA7,,032 141 166,20 8D A6,JSR &A68D
9EAA,',166 039,A6 27,LDX &27
9EAC,,169 255,A9 FF,LDA#&FF

9EAE,,128 159,80 9F,BRA -97   --> &9E4F
9EB0,',134 039,86 27,STX &27
9EB2,,032 230 188,20 E6 BC,JSR &BCE6
9EB5,,032 250 187,20 FA BB,JSR &BBFA
9EB8,,032 133 129,20 85 81,JSR &8185
9EBB,,128 231,80 E7,BRA -25   --> &9EA4
9EBD,,168,A8,TAY
9EBE,,240 209,F0 D1,BEQ -47   --> &9E91
9EC0,0%,048 037,30 25,BMI 37   --> &9EE7
9EC2,,032 196 159,20 C4 9F,JSR &9FC4
9EC5,,168,A8,TAY
9EC6,,240 201,F0 C9,BEQ -55   --> &9E91
9EC8,05,048 053,30 35,BMI 53   --> &9EFF
9ECA,8,056,38,SEC
9ECB,,178 004,B2 04,LDA (&04)
9ECD,*,229 042,E5 2A,SBC &2A
9ECF,*,133 042,85 2A,STA &2A
9ED1,,160 001,A0 01,LDY#&01
9ED3,,177 004,B1 04,"LDA (&04),Y"
9ED5,+,229 043,E5 2B,SBC &2B
9ED7,+,133 043,85 2B,STA &2B
9ED9,,200,C8,INY
9EDA,,177 004,B1 04,"LDA (&04),Y"
9EDC,",",229 044,E5 2C,SBC &2C
9EDE,",",133 044,85 2C,STA &2C
9EE0,,200,C8,INY
9EE1,,177 004,B1 04,"LDA (&04),Y"
9EE3,-,229 045,E5 2D,SBC &2D
9EE5,,128 153,80 99,BRA -103   --> &9E80
9EE7,,032 250 187,20 FA BB,JSR &BBFA
9EEA,,032 199 159,20 C7 9F,JSR &9FC7
9EED,,168,A8,TAY
9EEE,,240 161,F0 A1,BEQ -95   --> &9E91
9EF0,',134 039,86 27,STX &27
9EF2,0,048 003,30 03,BMI 3   --> &9EF7
9EF4,,032 133 129,20 85 81,JSR &8185
9EF7,,032 232 187,20 E8 BB,JSR &BBE8
9EFA,,032 138 166,20 8A A6,JSR &A68A
9EFD,,128 171,80 AB,BRA -85   --> &9EAA
9EFF,',134 039,86 27,STX &27
9F01,,032 230 188,20 E6 BC,JSR &BCE6
9F04,,032 250 187,20 FA BB,JSR &BBFA
9F07,,032 133 129,20 85 81,JSR &8185

9F0A,,032 232 187,20 E8 BB,JSR &BBE8
9F0D,,032 199 172,20 C7 AC,JSR &ACC7
9F10,,128 152,80 98,BRA -104 --> &9EAA
9F12,,032 133 129,20 85 81,JSR &8185
9F15,,032 230 188,20 E6 BC,JSR &BCE6
9F18,,032 250 187,20 FA BB,JSR &BBFA
9F1B,,032 133 129,20 85 81,JSR &8185
9F1E,,128 013,80 0D,BRA 13 --> &9F2D
9F20,,032 133 129,20 85 81,JSR &8185
9F23,,032 250 187,20 FA BB,JSR &BBFA
9F26,,032 018 160,20 12 A0,JSR &A012
9F29,,168,A8,TAY
9F2A,,032 221 150,20 DD 96,JSR &96DD
9F2D,,032 232 187,20 E8 BB,JSR &BBE8
9F30,,032 166 166,20 A6 A6,JSR &A6A6
9F33,,169 255,A9 FF,LDA#&FF
9F35,L,076 202 159,4C CA 9F,JMP &9FCA
9F38,L,076 146 144,4C 92 90,JMP &9092
9F3B,,168,A8,TAY
9F3C,,240 250,F0 FA,BEQ -6 --> &9F38
9F3E,0,048 227,30 E3,BMI -29 --> &9F23
9F40,-,164 045,A4 2D,LDY &2D
9F42,",",196 044,C4 2C,CPY &2C
9F44,,208 218,D0 DA,BNE -38 --> &9F20
9F46,+,165 043,A5 2B,LDA &2B
9F48,,010,0A,ASL A
9F49,,152,98,TYA
9F4A,i,105 000,69 00,ADC#&00
9F4C,,208 210,D0 D2,BNE -46 --> &9F20
9F4E,,032 015 160,20 0F A0,JSR &A00F
9F51,,168,A8,TAY
9F52,,240 228,F0 E4,BEQ -28 --> &9F38
9F54,0,048 191,30 BF,BMI -65 --> &9F15
9F56,-,164 045,A4 2D,LDY &2D
9F58,",",196 044,C4 2C,CPY &2C
9F5A,,208 182,D0 B6,BNE -74 --> &9F12
9F5C,+,165 043,A5 2B,LDA &2B
9F5E,,010,0A,ASL A
9F5F,,152,98,TYA
9F60,i,105 000,69 00,ADC#&00
9F62,,208 174,D0 AE,BNE -82 --> &9F12
9F64,Z,090,5A,PHY
9F65,,032 190 172,20 BE AC,JSR &ACBE

9F68,',134 039,86 27,STX &27
9F6A,9,162 057,A2 39,LDX#&39
9F6C,,032 198 189,20 C6 BD,JSR &BDC6
9F6F,,032 230 188,20 E6 BC,JSR &BCE6
9F72,h,104,68,PLA
9F73,E-,069 045,45 2D,EOR &2D
9F75,7,133 055,85 37,STA &37
9F77,,032 190 172,20 BE AC,JSR &ACBE
9F7A,,160 000,A0 00,LDY#&00
9F7C,,162 000,A2 00,LDX#&00
9F7E,d?,100 063,64 3F,STZ &3F
9F80,d@,100 064,64 40,STZ &40
9F82,F:,070 058,46 3A,LSR &3A
9F84,f9,102 057,66 39,ROR &39
9F86,,144 021,90 15,BCC 21 --> &9F9D
9F88,,024,18,CLC
9F89,,152,98,TYA
9F8A,e*,101 042,65 2A,ADC &2A
9F8C,,168,A8,TAY
9F8D,,138,8A,TXA
9F8E,e+,101 043,65 2B,ADC &2B
9F90,,170,AA,TAX
9F91,?,165 063,A5 3F,LDA &3F
9F93,"e,",101 044,65 2C,ADC &2C
9F95,?,133 063,85 3F,STA &3F
9F97,@,165 064,A5 40,LDA &40
9F99,e-,101 045,65 2D,ADC &2D
9F9B,@,133 064,85 40,STA &40
9F9D,*,006 042,06 2A,ASL &2A
9F9F,&+,038 043,26 2B,ROL &2B
9FA1,"&,",038 044,26 2C,ROL &2C
9FA3,&-,038 045,26 2D,ROL &2D
9FA5,9,165 057,A5 39,LDA &39
9FA7,:,005 058,05 3A,ORA &3A
9FA9,,208 215,D0 D7,BNE -41 --> &9F82
9FAB,=,132 061,84 3D,STY &3D
9FAD,>,134 062,86 3E,STX &3E
9FAF,7,165 055,A5 37,LDA &37
9FB1,,008,08,PHP
9FB2,=,162 061,A2 3D,LDX#&3D
9FB4,,032 128 170,20 80 AA,JSR &AA80
9FB7,(,040,28,PLP
9FB8,,016 003,10 03,BPL 3 --> &9FBD

9FBA,,032 222 172,20 DE AC,JSR &ACDE
9FBD,',166 039,A6 27,LDX &27
9FBF,,128 009,80 09,BRA 9   --> &9FCA
9FC1,L;,076 059 159,4C 3B 9F,JMP &9F3B
9FC4,&,032 038 188,20 26 BC,JSR &BC26
9FC7,,032 018 160,20 12 A0,JSR &A012
9FCA,*,224 042,E0 2A,CPX#&2A
9FCC,,240 243,F0 F3,BEQ -13   --> &9FC1
9FCE,/,224 047,E0 2F,CPX#&2F
9FD0,,240 009,F0 09,BEQ 9   --> &9FDB
9FD2,,224 131,E0 83,CPX#&83
9FD4,,240 031,F0 1F,BEQ 31   --> &9FF5
9FD6,,224 129,E0 81,CPX#&81
9FD8,#,240 035,F0 23,BEQ 35   --> &9FFD
9FDA,`,096,60,RTS
9FDB,,168,A8,TAY
9FDC,,032 221 150,20 DD 96,JSR &96DD
9FDF,,032 250 187,20 FA BB,JSR &BBFA
9FE2,,032 018 160,20 12 A0,JSR &A012
9FE5,',134 039,86 27,STX &27
9FE7,,168,A8,TAY
9FE8,,032 221 150,20 DD 96,JSR &96DD
9FEB,,032 232 187,20 E8 BB,JSR &BBE8
9FEE,,032 238 165,20 EE A5,JSR &A5EE
9FF1,,169 255,A9 FF,LDA#&FF
9FF3,,128 200,80 C8,BRA -56   --> &9FBD
9FF5,,032 249 128,20 F9 80,JSR &80F9
9FF8,8,165 056,A5 38,LDA &38
9FFA,,008,08,PHP
9FFB,,128 181,80 B5,BRA -75   --> &9FB2
9FFD,,032 249 128,20 F9 80,JSR &80F9
A000,&9,038 057,26 39,ROL &39

```
A000 &9   038 057     26 39     ROL &39
A002 &:   038 058     26 3A     ROL &3A
A004 &;   038 059     26 3B     ROL &3B
A006 &<   038 060     26 3C     ROL &3C
A008 $7   036 055     24 37     BIT &37
A00A      008         08        PHP
A00B  9   162 057     A2 39     LDX#&39
A00D      128 165     80 A5     BRA -91   --> &9FB4
A00F  &   032 038 188 20 26 BC  JSR &BC26
A012  6   032 054 173 20 36 AD  JSR &AD36
A015 H    072         48        PHA
A016      164 027     A4 1B     LDY &1B
A018      230 027     E6 1B     INC &1B
A01A      177 025     B1 19     LDA (&19),Y
A01C      201 032     C9 20     CMP#&20
A01E      240 246     F0 F6     BEQ -10   --> &A016
A020      170         AA        TAX
A021 h    104         68        PLA
A022  ^   224 094     E0 5E     CPX#&5E
A024      240 001     F0 01     BEQ 1   --> &A027
A026 `    096         60        RTS
A027      168         A8        TAY
A028      032 221 150 20 DD 96  JSR &96DD
A02B      032 250 187 20 FA BB  JSR &BBFA
A02E      032 218 150 20 DA 96  JSR &96DA
A031  0   165 048     A5 30     LDA &30
A033      201 135     C9 87     CMP#&87
A035  B   176 066     B0 42     BCS 66   --> &A079
A037      032 224 130 20 E0 82  JSR &82E0
A03A      208 013     D0 0D     BNE 13   --> &A049
A03C      032 232 187 20 E8 BB  JSR &BBE8
A03F  A   032 065 165 20 41 A5  JSR &A541
A042  I   165 073     A5 49     LDA &49
A044      032 190 165 20 BE A5  JSR &A5BE
A047  ,   128 044     80 2C     BRA 44   --> &A075
A049      032 013 165 20 0D A5  JSR &A50D
A04C      165 004     A5 04     LDA &04
A04E  J   133 074     85 4A     STA &4A
A050      165 005     A5 05     LDA &05
A052  K   133 075     85 4B     STA &4B
A054  A   032 065 165 20 41 A5  JSR &A541
A057  I   165 073     A5 49     LDA &49
```

```
A059      032 190 165 20 BE A5  JSR &A5BE
A05C  q   169 113     A9 71     LDA#&71
A05E      032 019 165 20 13 A5  JSR &A513
A061      032 232 187 20 E8 BB  JSR &BBE8
A064  A   032 065 165 20 41 A5  JSR &A541
A067  I   032 073 167 20 49 A7  JSR &A749
A06A      032 159 169 20 9F A9  JSR &A99F
A06D      032 226 169 20 E2 A9  JSR &A9E2
A070  q   169 113     A9 71     LDA#&71
A072      032 161 169 20 A1 A9  JSR &A9A1
A075      169 255     A9 FF     LDA#&FF
A077      128 156     80 9C     BRA -100   --> &A015
A079      032 013 165 20 0D A5  JSR &A50D
A07C      032 216 165 20 D8 A5  JSR &A5D8
A07F      128 219     80 DB     BRA -37   --> &A05C
A081      169 000     A9 00     LDA#&00
A083      128 002     80 02     BRA 2   --> &A087
A085      169 005     A9 05     LDA#&05
A087      133 020     85 14     STA &14
A089      162 004     A2 04     LDX#&04
A08B t?   116 063     74 3F     STZ &3F,X
A08D 8    056         38        SEC
A08E  *   165 042     A5 2A     LDA &2A
A090  &   253 038 128 FD 26 80  SBC &8026,X
A093      168         A8        TAY
A094  +   165 043     A5 2B     LDA &2B
A096  !   253 033 128 FD 21 80  SBC &8021,X
A099      144 008     90 08     BCC 8   --> &A0A3
A09B  +   133 043     85 2B     STA &2B
A09D  *   132 042     84 2A     STY &2A
A09F  ?   246 063     F6 3F     INC &3F,X
A0A1      128 235     80 EB     BRA -21   --> &A08E
A0A3      202         CA        DEX
A0A4      016 229     10 E5     BPL -27   --> &A08B
A0A6      162 005     A2 05     LDX#&05
A0A8      202         CA        DEX
A0A9      240 004     F0 04     BEQ 4   --> &A0AF
A0AB  ?   181 063     B5 3F     LDA &3F,X
A0AD      240 249     F0 F9     BEQ -7   --> &A0A8
A0AF  7   134 055     86 37     STX &37
A0B1      165 020     A5 14     LDA &14
A0B3      240 010     F0 0A     BEQ 10   --> &A0BF
A0B5  7   229 055     E5 37     SBC &37
```

```
A0B7      240 006     F0 06     BEQ 6   --> &A0BF
A0B9      170         AA        TAX
A0BA      032 191 189 20 BF BD  JSR &BDBF
A0BD  7   166 055     A6 37     LDX &37
A0BF  ?   181 063     B5 3F     LDA &3F,X
A0C1  0   009 048     09 30     ORA#&30
A0C3      032 148 189 20 94 BD  JSR &BD94
A0C6      202         CA        DEX
A0C7      016 246     10 F6     BPL -10   --> &A0BF
A0C9 `    096         60        RTS
A0CA      152         98        TYA
A0CB      016 003     10 03     BPL 3   --> &A0D0
A0CD      032 195 150 20 C3 96  JSR &96C3
A0D0      162 000     A2 00     LDX#&00
A0D2      160 000     A0 00     LDY#&00
A0D4  *   185 042 000 B9 2A 00  LDA &002A,Y
A0D7 H    072         48        PHA
A0D8 )    041 015     29 0F     AND#&0F
A0DA  ?   149 063     95 3F     STA &3F,X
A0DC h    104         68        PLA
A0DD J    074         4A        LSR A
A0DE J    074         4A        LSR A
A0DF J    074         4A        LSR A
A0E0 J    074         4A        LSR A
A0E1      232         E8        INX
A0E2  ?   149 063     95 3F     STA &3F,X
A0E4      232         E8        INX
A0E5      200         C8        INY
A0E6      192 004     C0 04     CPY#&04
A0E8      208 234     D0 EA     BNE -22   --> &A0D4
A0EA      202         CA        DEX
A0EB      240 004     F0 04     BEQ 4   --> &A0F1
A0ED  ?   181 063     B5 3F     LDA &3F,X
A0EF      240 249     F0 F9     BEQ -7   --> &A0EA
A0F1  ?   181 063     B5 3F     LDA &3F,X
A0F3      201 010     C9 0A     CMP#&0A
A0F5      144 002     90 02     BCC 2   --> &A0F9
A0F7 i    105 006     69 06     ADC#&06
A0F9 i0   105 048     69 30     ADC#&30
A0FB      032 208 162 20 D0 A2  JSR &A2D0
A0FE      202         CA        DEX
A0FF      016 240     10 F0     BPL -16   --> &A0F1
A101 `    096         60        RTS
```

```
A102      016 007     10 07     BPL 7   --> &A10B
A104  -   169 045     A9 2D     LDA#&2D
A106 d.   100 046     64 2E     STZ &2E
A108      032 208 162 20 D0 A2  JSR &A2D0
A10B  0   165 048     A5 30     LDA &30
A10D      201 129     C9 81     CMP#&81
A10F  K   176 075     B0 4B     BCS 75   --> &A15C
A111  6   032 054 164 20 36 A4  JSR &A436
A114  H   198 072     C6 48     DEC &48
A116      128 243     80 F3     BRA -13   --> &A10B
A118      174 002 004 AE 02 04  LDX &0402
A11B      224 003     E0 03     CPX#&03
A11D      144 002     90 02     BCC 2   --> &A121
A11F      162 000     A2 00     LDX#&00
A121  7   134 055     86 37     STX &37
A123      173 001 004 AD 01 04  LDA &0401
A126      240 006     F0 06     BEQ 6   --> &A12E
A128      201 010     C9 0A     CMP#&0A
A12A      176 006     B0 06     BCS 6   --> &A132
A12C      128 006     80 06     BRA 6   --> &A134
A12E      224 002     E0 02     CPX#&02
A130      240 002     F0 02     BEQ 2   --> &A134
A132      169 010     A9 0A     LDA#&0A
A134  8   133 056     85 38     STA &38
A136  M   133 077     85 4D     STA &4D
A138 d6   100 054     64 36     STZ &36
A13A dH   100 072     64 48     STZ &48
A13C $    036 021     24 15     BIT &15
A13E 0    048 138     30 8A     BMI -118   --> &A0CA
A140      152         98        TYA
A141 0    048 003     30 03     BMI 3   --> &A146
A143      032 133 129 20 85 81  JSR &8185
A146      032 242 163 20 F2 A3  JSR &A3F2
A149      208 183     D0 B7     BNE -73   --> &A102
A14B  7   165 055     A5 37     LDA &37
A14D      208 005     D0 05     BNE 5   --> &A154
A14F  0   169 048     A9 30     LDA#&30
A151 L    076 208 162 4C D0 A2  JMP &A2D0
A154 L    076 208 161 4C D0 A1  JMP &A1D0
A157      032 216 165 20 D8 A5  JSR &A5D8
A15A      128 015     80 0F     BRA 15   --> &A16B
A15C      201 132     C9 84     CMP#&84
A15E      144 015     90 0F     BCC 15   --> &A16F
```

```
A160      208 006     D0 06     BNE 6   --> &A168
A162  1   165 049     A5 31     LDA &31
A164      201 160     C9 A0     CMP#&A0
A166      144 007     90 07     BCC 7   --> &A16F
A168  x   032 120 164 20 78 A4  JSR &A478
A16B  H   230 072     E6 48     INC &48
A16D      128 156     80 9C     BRA -100   --> &A10B
A16F  5   165 053     A5 35     LDA &35
A171  '   133 039     85 27     STA &27
A173      032 017 165 20 11 A5  JSR &A511
A176  M   165 077     A5 4D     LDA &4D
A178  8   133 056     85 38     STA &38
A17A  7   166 055     A6 37     LDX &37
A17C      224 002     E0 02     CPX#&02
A17E      208 016     D0 10     BNE 16   --> &A190
A180 eH   101 072     65 48     ADC &48
A182 0P   048 080     30 50     BMI 80   --> &A1D4
A184  8   133 056     85 38     STA &38
A186      201 011     C9 0B     CMP#&0B
A188      144 006     90 06     BCC 6   --> &A190
A18A      169 010     A9 0A     LDA#&0A
A18C  8   133 056     85 38     STA &38
A18E d7   100 055     64 37     STZ &37
A190      032 184 166 20 B8 A6  JSR &A6B8
A193      169 160     A9 A0     LDA#&A0
A195  1   133 049     85 31     STA &31
A197      169 131     A9 83     LDA#&83
A199  0   133 048     85 30     STA &30
A19B  8   166 056     A6 38     LDX &38
A19D      240 006     F0 06     BEQ 6   --> &A1A5
A19F  x   032 120 164 20 78 A4  JSR &A478
A1A2      202         CA        DEX
A1A3      208 250     D0 FA     BNE -6   --> &A19F
A1A5      032 146 165 20 92 A5  JSR &A592
A1A8      032 224 164 20 E0 A4  JSR &A4E0
A1AB  '   165 039     A5 27     LDA &27
A1AD  A   133 065     85 41     STA &41
A1AF  h   032 104 131 20 68 83  JSR &8368
A1B2  0   165 048     A5 30     LDA &30
A1B4      201 132     C9 84     CMP#&84
A1B6      176 014     B0 0E     BCS 14   --> &A1C6
A1B8 f1   102 049     66 31     ROR &31
A1BA f2   102 050     66 32     ROR &32
```

```
A1BC f3   102 051     66 33     ROR &33
A1BE f4   102 052     66 34     ROR &34
A1C0 f5   102 053     66 35     ROR &35
A1C2  0   230 048     E6 30     INC &30
A1C4      208 236     D0 EC     BNE -20   --> &A1B2
A1C6  1   165 049     A5 31     LDA &31
A1C8      201 160     C9 A0     CMP#&A0
A1CA      176 139     B0 8B     BCS -117   --> &A157
A1CC  8   165 056     A5 38     LDA &38
A1CE      208 014     D0 0E     BNE 14   --> &A1DE
A1D0      201 001     C9 01     CMP#&01
A1D2  A   240 065     F0 41     BEQ 65   --> &A215
A1D4      032 180 166 20 B4 A6  JSR &A6B4
A1D7 dH   100 072     64 48     STZ &48
A1D9  M   165 077     A5 4D     LDA &4D
A1DB      026         1A        INC A
A1DC  8   133 056     85 38     STA &38
A1DE      169 001     A9 01     LDA#&01
A1E0  7   197 055     C5 37     CMP &37
A1E2  1   240 049     F0 31     BEQ 49   --> &A215
A1E4  H   164 072     A4 48     LDY &48
A1E6 0    048 010     30 0A     BMI 10   --> &A1F2
A1E8  8   196 056     C4 38     CPY &38
A1EA  )   176 041     B0 29     BCS 41   --> &A215
A1EC dH   100 072     64 48     STZ &48
A1EE      200         C8        INY
A1EF      152         98        TYA
A1F0  #   208 035     D0 23     BNE 35   --> &A215
A1F2  7   165 055     A5 37     LDA &37
A1F4      201 002     C9 02     CMP#&02
A1F6      240 006     F0 06     BEQ 6   --> &A1FE
A1F8      169 001     A9 01     LDA#&01
A1FA      192 255     C0 FF     CPY#&FF
A1FC      208 023     D0 17     BNE 23   --> &A215
A1FE  0   169 048     A9 30     LDA#&30
A200      032 208 162 20 D0 A2  JSR &A2D0
A203  .   169 046     A9 2E     LDA#&2E
A205      032 208 162 20 D0 A2  JSR &A2D0
A208  0   169 048     A9 30     LDA#&30
A20A  H   230 072     E6 48     INC &48
A20C      240 005     F0 05     BEQ 5   --> &A213
A20E      032 208 162 20 D0 A2  JSR &A2D0
A211      128 247     80 F7     BRA -9   --> &A20A
```

```
A213      169 128     A9 80     LDA#&80
A215  M   133 077     85 4D     STA &4D
A217  l   032 108 162 20 6C A2  JSR &A26C
A21A  M   198 077     C6 4D     DEC &4D
A21C      208 005     D0 05     BNE 5   --> &A223
A21E  .   169 046     A9 2E     LDA#&2E
A220      032 208 162 20 D0 A2  JSR &A2D0
A223  8   198 056     C6 38     DEC &38
A225      208 240     D0 F0     BNE -16   --> &A217
A227  7   164 055     A4 37     LDY &37
A229      136         88        DEY
A22A      240 024     F0 18     BEQ 24   --> &A244
A22C      136         88        DEY
A22D      240 017     F0 11     BEQ 17   --> &A240
A22F  6   164 054     A4 36     LDY &36
A231      136         88        DEY
A232      185 000 006 B9 00 06  LDA &0600,Y
A235  0   201 048     C9 30     CMP#&30
A237      240 248     F0 F8     BEQ -8   --> &A231
A239  .   201 046     C9 2E     CMP#&2E
A23B      240 001     F0 01     BEQ 1   --> &A23E
A23D      200         C8        INY
A23E  6   132 054     84 36     STY &36
A240  H   165 072     A5 48     LDA &48
A242  '   240 039     F0 27     BEQ 39   --> &A26B
A244  E   169 069     A9 45     LDA#&45
A246      032 208 162 20 D0 A2  JSR &A2D0
A249  H   165 072     A5 48     LDA &48
A24B      016 010     10 0A     BPL 10   --> &A257
A24D  -   169 045     A9 2D     LDA#&2D
A24F      032 208 162 20 D0 A2  JSR &A2D0
A252 8    056         38        SEC
A253      169 000     A9 00     LDA#&00
A255  H   229 072     E5 48     SBC &48
A257      032 188 162 20 BC A2  JSR &A2BC
A25A  7   165 055     A5 37     LDA &37
A25C      240 013     F0 0D     BEQ 13   --> &A26B
A25E      169 032     A9 20     LDA#&20
A260  H   164 072     A4 48     LDY &48
A262 0    048 003     30 03     BMI 3   --> &A267
A264      032 208 162 20 D0 A2  JSR &A2D0
A267      224 000     E0 00     CPX#&00
A269  e   240 101     F0 65     BEQ 101   --> &A2D0
```

```
A26B `   096        60       RTS
A26C  1  165 049    A5 31    LDA &31
A26E J   074        4A       LSR A
A26F J   074        4A       LSR A
A270 J   074        4A       LSR A
A271 J   074        4A       LSR A
A272     032 206 162 20 CE A2  JSR &A2CE
A275     169 240    A9 F0    LDA#&F0
A277  1  020 049    14 31    TRB &31
A279 H   072        48       PHA
A27A  4  166 052    A6 34    LDX &34
A27C  1  165 049    A5 31    LDA &31
A27E H   072        48       PHA
A27F  2  165 050    A5 32    LDA &32
A281 H   072        48       PHA
A282  3  165 051    A5 33    LDA &33
A284 H   072        48       PHA
A285  5  165 053    A5 35    LDA &35
A287     010        0A       ASL A
A288 &4  038 052    26 34    ROL &34
A28A &3  038 051    26 33    ROL &33
A28C &2  038 050    26 32    ROL &32
A28E &1  038 049    26 31    ROL &31
A290     010        0A       ASL A
A291 &4  038 052    26 34    ROL &34
A293 &3  038 051    26 33    ROL &33
A295 &2  038 050    26 32    ROL &32
A297 &1  038 049    26 31    ROL &31
A299 e5  101 053    65 35    ADC &35
A29B  5  133 053    85 35    STA &35
A29D     138        8A       TXA
A29E e4  101 052    65 34    ADC &34
A2A0  4  133 052    85 34    STA &34
A2A2 h   104        68       PLA
A2A3 e3  101 051    65 33    ADC &33
A2A5  3  133 051    85 33    STA &33
A2A7 h   104        68       PLA
A2A8 e2  101 050    65 32    ADC &32
A2AA  2  133 050    85 32    STA &32
A2AC h   104        68       PLA
A2AD e1  101 049    65 31    ADC &31
A2AF  5  006 053    06 35    ASL &35
A2B1 &4  038 052    26 34    ROL &34
```

```
A2B3 &3   038 051     26 33     ROL &33
A2B5 &2   038 050     26 32     ROL &32
A2B7 *    042         2A        ROL A
A2B8  1   133 049     85 31     STA &31
A2BA h    104         68        PLA
A2BB `    096         60        RTS
A2BC      162 255     A2 FF     LDX#&FF
A2BE 8    056         38        SEC
A2BF      232         E8        INX
A2C0      233 010     E9 0A     SBC#&0A
A2C2      176 251     B0 FB     BCS -5   --> &A2BF
A2C4 i    105 010     69 0A     ADC#&0A
A2C6 H    072         48        PHA
A2C7      138         8A        TXA
A2C8      240 003     F0 03     BEQ 3   --> &A2CD
A2CA      032 206 162 20 CE A2  JSR &A2CE
A2CD h    104         68        PLA
A2CE  0   009 048     09 30     ORA#&30
A2D0      218         DA        PHX
A2D1  6   166 054     A6 36     LDX &36
A2D3      157 000 006 9D 00 06  STA &0600,X
A2D6      250         FA        PLX
A2D7  6   230 054     E6 36     INC &36
A2D9 `    096         60        RTS
A2DA      032 004 164 20 04 A4  JSR &A404
A2DD      024         18        CLC
A2DE      169 255     A9 FF     LDA#&FF
A2E0 `    096         60        RTS
A2E1 d1   100 049     64 31     STZ &31
A2E3 d2   100 050     64 32     STZ &32
A2E5 d3   100 051     64 33     STZ &33
A2E7 d4   100 052     64 34     STZ &34
A2E9 d5   100 053     64 35     STZ &35
A2EB dG   100 071     64 47     STZ &47
A2ED dH   100 072     64 48     STZ &48
A2EF  .   201 046     C9 2E     CMP#&2E
A2F1  )   240 041     F0 29     BEQ 41   --> &A31C
A2F3  :   201 058     C9 3A     CMP#&3A
A2F5      176 227     B0 E3     BCS -29   --> &A2DA
A2F7  /   233 047     E9 2F     SBC#&2F
A2F9 0    048 223     30 DF     BMI -33   --> &A2DA
A2FB  5   133 053     85 35     STA &35
A2FD      200         C8        INY
```

```
A2FE      177 025     B1 19     LDA (&19),Y
A300  :   201 058     C9 3A     CMP#&3A
A302      176 032     B0 20     BCS 32   --> &A324
A304  /   233 047     E9 2F     SBC#&2F
A306      144 014     90 0E     BCC 14   --> &A316
A308  .   133 046     85 2E     STA &2E
A30A  5   165 053     A5 35     LDA &35
A30C      010         0A        ASL A
A30D      010         0A        ASL A
A30E e5   101 053     65 35     ADC &35
A310      010         0A        ASL A
A311 e.   101 046     65 2E     ADC &2E
A313  5   133 053     85 35     STA &35
A315      200         C8        INY
A316      177 025     B1 19     LDA (&19),Y
A318  .   201 046     C9 2E     CMP#&2E
A31A      208 008     D0 08     BNE 8   --> &A324
A31C  G   165 071     A5 47     LDA &47
A31E  D   208 068     D0 44     BNE 68   --> &A364
A320  G   230 071     E6 47     INC &47
A322      128 241     80 F1     BRA -15   --> &A315
A324  E   201 069     C9 45     CMP#&45
A326  5   240 053     F0 35     BEQ 53   --> &A35D
A328  :   201 058     C9 3A     CMP#&3A
A32A  8   176 056     B0 38     BCS 56   --> &A364
A32C  /   233 047     E9 2F     SBC#&2F
A32E  4   144 052     90 34     BCC 52   --> &A364
A330  1   166 049     A6 31     LDX &31
A332      224 024     E0 18     CPX#&18
A334      144 008     90 08     BCC 8   --> &A33E
A336  G   166 071     A6 47     LDX &47
A338      208 219     D0 DB     BNE -37   --> &A315
A33A  H   230 072     E6 48     INC &48
A33C      128 215     80 D7     BRA -41   --> &A315
A33E  G   166 071     A6 47     LDX &47
A340      240 002     F0 02     BEQ 2   --> &A344
A342  H   198 072     C6 48     DEC &48
A344  y   032 121 162 20 79 A2  JSR &A279
A347 e5   101 053     65 35     ADC &35
A349  5   133 053     85 35     STA &35
A34B      144 200     90 C8     BCC -56   --> &A315
A34D  4   230 052     E6 34     INC &34
A34F      208 196     D0 C4     BNE -60   --> &A315
```

```
A351  3   230 051     E6 33     INC &33
A353      208 192     D0 C0     BNE -64   --> &A315
A355  2   230 050     E6 32     INC &32
A357      208 188     D0 BC     BNE -68   --> &A315
A359  1   230 049     E6 31     INC &31
A35B      128 184     80 B8     BRA -72   --> &A315
A35D      032 186 163 20 BA A3  JSR &A3BA
A360 eH   101 072     65 48     ADC &48
A362  H   133 072     85 48     STA &48
A364      132 027     84 1B     STY &1B
A366  H   165 072     A5 48     LDA &48
A368  G   005 071     05 47     ORA &47
A36A  -   240 045     F0 2D     BEQ 45   --> &A399
A36C      032 242 163 20 F2 A3  JSR &A3F2
A36F  $   240 036     F0 24     BEQ 36   --> &A395
A371      169 168     A9 A8     LDA#&A8
A373  0   133 048     85 30     STA &30
A375 d/   100 047     64 2F     STZ &2F
A377 d.   100 046     64 2E     STZ &2E
A379      032 247 129 20 F7 81  JSR &81F7
A37C  H   165 072     A5 48     LDA &48
A37E 0    048 011     30 0B     BMI 11   --> &A38B
A380      240 016     F0 10     BEQ 16   --> &A392
A382  6   032 054 164 20 36 A4  JSR &A436
A385  H   198 072     C6 48     DEC &48
A387      208 249     D0 F9     BNE -7   --> &A382
A389      128 007     80 07     BRA 7   --> &A392
A38B  x   032 120 164 20 78 A4  JSR &A478
A38E  H   230 072     E6 48     INC &48
A390      208 249     D0 F9     BNE -7   --> &A38B
A392      032 149 166 20 95 A6  JSR &A695
A395 8    056         38        SEC
A396      169 255     A9 FF     LDA#&FF
A398 `    096         60        RTS
A399  2   165 050     A5 32     LDA &32
A39B  -   133 045     85 2D     STA &2D
A39D )    041 128     29 80     AND#&80
A39F  1   005 049     05 31     ORA &31
A3A1      208 206     D0 CE     BNE -50   --> &A371
A3A3  5   165 053     A5 35     LDA &35
A3A5  *   133 042     85 2A     STA &2A
A3A7  4   165 052     A5 34     LDA &34
A3A9  +   133 043     85 2B     STA &2B
```

```
A3AB  3   165 051     A5 33     LDA &33
A3AD  ,   133 044     85 2C     STA &2C
A3AF  @   169 064     A9 40     LDA#&40
A3B1 8    056         38        SEC
A3B2 `    096         60        RTS
A3B3      032 197 163 20 C5 A3  JSR &A3C5
A3B6 I    073 255     49 FF     EOR#&FF
A3B8 8    056         38        SEC
A3B9 `    096         60        RTS
A3BA      200         C8        INY
A3BB      177 025     B1 19     LDA (&19),Y
A3BD  -   201 045     C9 2D     CMP#&2D
A3BF      240 242     F0 F2     BEQ -14   --> &A3B3
A3C1  +   201 043     C9 2B     CMP#&2B
A3C3      208 003     D0 03     BNE 3   --> &A3C8
A3C5      200         C8        INY
A3C6      177 025     B1 19     LDA (&19),Y
A3C8  :   201 058     C9 3A     CMP#&3A
A3CA  "   176 034     B0 22     BCS 34   --> &A3EE
A3CC  /   233 047     E9 2F     SBC#&2F
A3CE      144 030     90 1E     BCC 30   --> &A3EE
A3D0  I   133 073     85 49     STA &49
A3D2      200         C8        INY
A3D3      177 025     B1 19     LDA (&19),Y
A3D5  :   201 058     C9 3A     CMP#&3A
A3D7      176 017     B0 11     BCS 17   --> &A3EA
A3D9  /   233 047     E9 2F     SBC#&2F
A3DB      144 013     90 0D     BCC 13   --> &A3EA
A3DD      200         C8        INY
A3DE  B   133 066     85 42     STA &42
A3E0  I   165 073     A5 49     LDA &49
A3E2      010         0A        ASL A
A3E3      010         0A        ASL A
A3E4 eI   101 073     65 49     ADC &49
A3E6      010         0A        ASL A
A3E7 eB   101 066     65 42     ADC &42
A3E9 `    096         60        RTS
A3EA  I   165 073     A5 49     LDA &49
A3EC      024         18        CLC
A3ED `    096         60        RTS
A3EE      169 000     A9 00     LDA#&00
A3F0      024         18        CLC
A3F1 `    096         60        RTS
```

```
A3F2  1   165 049     A5 31     LDA &31
A3F4  2   005 050     05 32     ORA &32
A3F6  3   005 051     05 33     ORA &33
A3F8  4   005 052     05 34     ORA &34
A3FA  5   005 053     05 35     ORA &35
A3FC      240 006     F0 06     BEQ 6   --> &A404
A3FE  .   165 046     A5 2E     LDA &2E
A400      208 008     D0 08     BNE 8   --> &A40A
A402      026         1A        INC A
A403 `    096         60        RTS
A404 d.   100 046     64 2E     STZ &2E
A406 d0   100 048     64 30     STZ &30
A408 d/   100 047     64 2F     STZ &2F
A40A `    096         60        RTS
A40B  .   165 046     A5 2E     LDA &2E
A40D  ;   133 059     85 3B     STA &3B
A40F  0   165 048     A5 30     LDA &30
A411  <   133 060     85 3C     STA &3C
A413  1   165 049     A5 31     LDA &31
A415  =   133 061     85 3D     STA &3D
A417  2   165 050     A5 32     LDA &32
A419  >   133 062     85 3E     STA &3E
A41B  3   165 051     A5 33     LDA &33
A41D  ?   133 063     85 3F     STA &3F
A41F  4   165 052     A5 34     LDA &34
A421  @   133 064     85 40     STA &40
A423  5   165 053     A5 35     LDA &35
A425  A   133 065     85 41     STA &41
A427 `    096         60        RTS
A428      032 011 164 20 0B A4  JSR &A40B
A42B F=   070 061     46 3D     LSR &3D
A42D f>   102 062     66 3E     ROR &3E
A42F f?   102 063     66 3F     ROR &3F
A431 f@   102 064     66 40     ROR &40
A433 fA   102 065     66 41     ROR &41
A435 `    096         60        RTS
A436      024         18        CLC
A437  0   165 048     A5 30     LDA &30
A439 i    105 003     69 03     ADC#&03
A43B  0   133 048     85 30     STA &30
A43D      144 002     90 02     BCC 2   --> &A441
A43F  /   230 047     E6 2F     INC &2F
A441  (   032 040 164 20 28 A4  JSR &A428
```

```
A444  +   032 043 164 20 2B A4  JSR &A42B
A447  5   165 053     A5 35     LDA &35
A449 eA   101 065     65 41     ADC &41
A44B  5   133 053     85 35     STA &35
A44D  4   165 052     A5 34     LDA &34
A44F e@   101 064     65 40     ADC &40
A451  4   133 052     85 34     STA &34
A453  3   165 051     A5 33     LDA &33
A455 e?   101 063     65 3F     ADC &3F
A457  3   133 051     85 33     STA &33
A459  2   165 050     A5 32     LDA &32
A45B e>   101 062     65 3E     ADC &3E
A45D  2   133 050     85 32     STA &32
A45F  1   165 049     A5 31     LDA &31
A461 e=   101 061     65 3D     ADC &3D
A463  1   133 049     85 31     STA &31
A465      144 016     90 10     BCC 16   --> &A477
A467 f1   102 049     66 31     ROR &31
A469 f2   102 050     66 32     ROR &32
A46B f3   102 051     66 33     ROR &33
A46D f4   102 052     66 34     ROR &34
A46F f5   102 053     66 35     ROR &35
A471  0   230 048     E6 30     INC &30
A473      208 002     D0 02     BNE 2   --> &A477
A475  /   230 047     E6 2F     INC &2F
A477 `    096         60        RTS
A478 8    056         38        SEC
A479  0   165 048     A5 30     LDA &30
A47B      233 004     E9 04     SBC#&04
A47D  0   133 048     85 30     STA &30
A47F      176 002     B0 02     BCS 2   --> &A483
A481  /   198 047     C6 2F     DEC &2F
A483  (   032 040 164 20 28 A4  JSR &A428
A486  G   032 071 164 20 47 A4  JSR &A447
A489  (   032 040 164 20 28 A4  JSR &A428
A48C  +   032 043 164 20 2B A4  JSR &A42B
A48F  +   032 043 164 20 2B A4  JSR &A42B
A492  +   032 043 164 20 2B A4  JSR &A42B
A495  G   032 071 164 20 47 A4  JSR &A447
A498 d=   100 061     64 3D     STZ &3D
A49A  1   165 049     A5 31     LDA &31
A49C  >   133 062     85 3E     STA &3E
A49E  2   165 050     A5 32     LDA &32
```

```
A4A0  ?   133 063     85 3F     STA &3F
A4A2  3   165 051     A5 33     LDA &33
A4A4  @   133 064     85 40     STA &40
A4A6  4   165 052     A5 34     LDA &34
A4A8  A   133 065     85 41     STA &41
A4AA  5   165 053     A5 35     LDA &35
A4AC *    042         2A        ROL A
A4AD  G   032 071 164 20 47 A4  JSR &A447
A4B0 d>   100 062     64 3E     STZ &3E
A4B2  1   165 049     A5 31     LDA &31
A4B4  ?   133 063     85 3F     STA &3F
A4B6  2   165 050     A5 32     LDA &32
A4B8  @   133 064     85 40     STA &40
A4BA  3   165 051     A5 33     LDA &33
A4BC  A   133 065     85 41     STA &41
A4BE  4   165 052     A5 34     LDA &34
A4C0 *    042         2A        ROL A
A4C1  G   032 071 164 20 47 A4  JSR &A447
A4C4  2   165 050     A5 32     LDA &32
A4C6 *    042         2A        ROL A
A4C7  1   165 049     A5 31     LDA &31
A4C9 e5   101 053     65 35     ADC &35
A4CB  5   133 053     85 35     STA &35
A4CD      144 016     90 10     BCC 16   --> &A4DF
A4CF  4   230 052     E6 34     INC &34
A4D1      208 012     D0 0C     BNE 12   --> &A4DF
A4D3  3   230 051     E6 33     INC &33
A4D5      208 008     D0 08     BNE 8   --> &A4DF
A4D7  2   230 050     E6 32     INC &32
A4D9      208 004     D0 04     BNE 4   --> &A4DF
A4DB  1   230 049     E6 31     INC &31
A4DD      240 136     F0 88     BEQ -120   --> &A467
A4DF `    096         60        RTS
A4E0 dA   100 065     64 41     STZ &41
A4E2      160 004     A0 04     LDY#&04
A4E4  J   177 074     B1 4A     LDA (&4A),Y
A4E6  @   133 064     85 40     STA &40
A4E8      136         88        DEY
A4E9  J   177 074     B1 4A     LDA (&4A),Y
A4EB  ?   133 063     85 3F     STA &3F
A4ED      136         88        DEY
A4EE  J   177 074     B1 4A     LDA (&4A),Y
A4F0  >   133 062     85 3E     STA &3E
```

```
A4F2      136         88        DEY
A4F3  J   177 074     B1 4A     LDA (&4A),Y
A4F5  ;   133 059     85 3B     STA &3B
A4F7      168         A8        TAY
A4F8  J   178 074     B2 4A     LDA (&4A)
A4FA  <   133 060     85 3C     STA &3C
A4FC      208 009     D0 09     BNE 9   --> &A507
A4FE      152         98        TYA
A4FF  >   005 062     05 3E     ORA &3E
A501  ?   005 063     05 3F     ORA &3F
A503  @   005 064     05 40     ORA &40
A505      240 003     F0 03     BEQ 3   --> &A50A
A507      152         98        TYA
A508      009 128     09 80     ORA#&80
A50A  =   133 061     85 3D     STA &3D
A50C  `   096         60        RTS
A50D  v   169 118     A9 76     LDA#&76
A50F      128 002     80 02     BRA 2   --> &A513
A511  l   169 108     A9 6C     LDA#&6C
A513  J   133 074     85 4A     STA &4A
A515      169 004     A9 04     LDA#&04
A517  K   133 075     85 4B     STA &4B
A519  0   165 048     A5 30     LDA &30
A51B  J   146 074     92 4A     STA (&4A)
A51D      160 001     A0 01     LDY#&01
A51F  .   165 046     A5 2E     LDA &2E
A521 E1   069 049     45 31     EOR &31
A523  )   041 128     29 80     AND#&80
A525 E1   069 049     45 31     EOR &31
A527  J   145 074     91 4A     STA (&4A),Y
A529  2   165 050     A5 32     LDA &32
A52B      200         C8        INY
A52C  J   145 074     91 4A     STA (&4A),Y
A52E  3   165 051     A5 33     LDA &33
A530      200         C8        INY
A531  J   145 074     91 4A     STA (&4A),Y
A533  4   165 052     A5 34     LDA &34
A535      200         C8        INY
A536  J   145 074     91 4A     STA (&4A),Y
A538  `   096         60        RTS
A539  l   169 108     A9 6C     LDA#&6C
A53B  J   133 074     85 4A     STA &4A
A53D      169 004     A9 04     LDA#&04
```

```
A53F  K   133 075     85 4B     STA &4B
A541 d5   100 053     64 35     STZ &35
A543 d/   100 047     64 2F     STZ &2F
A545      160 004     A0 04     LDY#&04
A547  J   177 074     B1 4A     LDA (&4A),Y
A549  4   133 052     85 34     STA &34
A54B      136         88        DEY
A54C  J   177 074     B1 4A     LDA (&4A),Y
A54E  3   133 051     85 33     STA &33
A550      136         88        DEY
A551  J   177 074     B1 4A     LDA (&4A),Y
A553  2   133 050     85 32     STA &32
A555      136         88        DEY
A556  J   177 074     B1 4A     LDA (&4A),Y
A558  .   133 046     85 2E     STA &2E
A55A      168         A8        TAY
A55B  J   178 074     B2 4A     LDA (&4A)
A55D  0   133 048     85 30     STA &30
A55F      208 009     D0 09     BNE 9   --> &A56A
A561      152         98        TYA
A562  2   005 050     05 32     ORA &32
A564  3   005 051     05 33     ORA &33
A566  4   005 052     05 34     ORA &34
A568      240 003     F0 03     BEQ 3   --> &A56D
A56A      152         98        TYA
A56B      009 128     09 80     ORA#&80
A56D  1   133 049     85 31     STA &31
A56F `    096         60        RTS
A570 d;   100 059     64 3B     STZ &3B
A572 d<   100 060     64 3C     STZ &3C
A574 d=   100 061     64 3D     STZ &3D
A576 d>   100 062     64 3E     STZ &3E
A578 d?   100 063     64 3F     STZ &3F
A57A d@   100 064     64 40     STZ &40
A57C dA   100 065     64 41     STZ &41
A57E `    096         60        RTS
A57F      024         18        CLC
A580  L   165 076     A5 4C     LDA &4C
A582 i    105 005     69 05     ADC#&05
A584  L   133 076     85 4C     STA &4C
A586  J   133 074     85 4A     STA &4A
A588 `    096         60        RTS
A589  .   169 046     A9 2E     LDA#&2E
```

```
A58B  J   133 074     85 4A     STA &4A
A58D      169 191     A9 BF     LDA#&BF
A58F  K   133 075     85 4B     STA &4B
A591 `    096         60        RTS
A592  l   169 108     A9 6C     LDA#&6C
A594  J   133 074     85 4A     STA &4A
A596      169 004     A9 04     LDA#&04
A598  K   133 075     85 4B     STA &4B
A59A `    096         60        RTS
A59B  :   032 058 169 20 3A A9  JSR &A93A
A59E  {   169 123     A9 7B     LDA#&7B
A5A0      032 019 165 20 13 A5  JSR &A513
A5A3      032 023 169 20 17 A9  JSR &A917
A5A6  v   169 118     A9 76     LDA#&76
A5A8      032 019 165 20 13 A5  JSR &A513
A5AB  {   169 123     A9 7B     LDA#&7B
A5AD  ;   032 059 165 20 3B A5  JSR &A53B
A5B0      032 021 169 20 15 A9  JSR &A915
A5B3  v   169 118     A9 76     LDA#&76
A5B5      032 148 165 20 94 A5  JSR &A594
A5B8      032 238 165 20 EE A5  JSR &A5EE
A5BB      169 255     A9 FF     LDA#&FF
A5BD `    096         60        RTS
A5BE      170         AA        TAX
A5BF      016 008     10 08     BPL 8   --> &A5C9
A5C1 :    058         3A        DEC A
A5C2 I    073 255     49 FF     EOR#&FF
A5C4 H    072         48        PHA
A5C5      032 233 165 20 E9 A5  JSR &A5E9
A5C8      250         FA        PLX
A5C9      240 013     F0 0D     BEQ 13   --> &A5D8
A5CB      032 017 165 20 11 A5  JSR &A511
A5CE      202         CA        DEX
A5CF      240 006     F0 06     BEQ 6   --> &A5D7
A5D1      032 166 166 20 A6 A6  JSR &A6A6
A5D4      202         CA        DEX
A5D5      208 250     D0 FA     BNE -6   --> &A5D1
A5D7 `    096         60        RTS
A5D8      169 128     A9 80     LDA#&80
A5DA  1   133 049     85 31     STA &31
A5DC      026         1A        INC A
A5DD  0   133 048     85 30     STA &30
A5DF L    076 184 166 4C B8 A6  JMP &A6B8
```

```
A5E2 Lr   076 114 129 4C 72 81  JMP &8172
A5E5      002         02        xxx Invalid Code
A5E6      008         08        PHP
A5E7      008         08        PHP
A5E8      008         08        PHP
A5E9      169 146     A9 92     LDA#&92
A5EB      032 139 165 20 8B A5  JSR &A58B
A5EE  1   165 049     A5 31     LDA &31
A5F0      240 240     F0 F0     BEQ -16   --> &A5E2
A5F2      032 224 164 20 E0 A4  JSR &A4E0
A5F5      208 003     D0 03     BNE 3   --> &A5FA
A5F7 L    076 180 166 4C B4 A6  JMP &A6B4
A5FA  ;   165 059     A5 3B     LDA &3B
A5FC E.   069 046     45 2E     EOR &2E
A5FE  .   133 046     85 2E     STA &2E
A600 8    056         38        SEC
A601  <   165 060     A5 3C     LDA &3C
A603 i    105 129     69 81     ADC#&81
A605 &/   038 047     26 2F     ROL &2F
A607  0   229 048     E5 30     SBC &30
A609      176 002     B0 02     BCS 2   --> &A60D
A60B  /   198 047     C6 2F     DEC &2F
A60D  0   133 048     85 30     STA &30
A60F      160 004     A0 04     LDY#&04
A611  <   132 060     84 3C     STY &3C
A613  =   165 061     A5 3D     LDA &3D
A615      162 008     A2 08     LDX#&08
A617      128 009     80 09     BRA 9   --> &A622
A619  C   150 067     96 43     STX &43,Y
A61B      190 229 165 BE E5 A5  LDX &A5E5,Y
A61E  <   132 060     84 3C     STY &3C
A620      176 022     B0 16     BCS 22   --> &A638
A622  1   197 049     C5 31     CMP &31
A624      208 016     D0 10     BNE 16   --> &A636
A626  >   164 062     A4 3E     LDY &3E
A628  2   196 050     C4 32     CPY &32
A62A      208 010     D0 0A     BNE 10   --> &A636
A62C  ?   164 063     A4 3F     LDY &3F
A62E  3   196 051     C4 33     CPY &33
A630      208 004     D0 04     BNE 4   --> &A636
A632  @   164 064     A4 40     LDY &40
A634  4   196 052     C4 34     CPY &34
A636      144 023     90 17     BCC 23   --> &A64F
```

```
A638      168         A8        TAY
A639  @   165 064     A5 40     LDA &40
A63B  4   229 052     E5 34     SBC &34
A63D  @   133 064     85 40     STA &40
A63F  ?   165 063     A5 3F     LDA &3F
A641  3   229 051     E5 33     SBC &33
A643  ?   133 063     85 3F     STA &3F
A645  >   165 062     A5 3E     LDA &3E
A647  2   229 050     E5 32     SBC &32
A649  >   133 062     85 3E     STA &3E
A64B      152         98        TYA
A64C  1   229 049     E5 31     SBC &31
A64E 8    056         38        SEC
A64F &;   038 059     26 3B     ROL &3B
A651  @   006 064     06 40     ASL &40
A653 &?   038 063     26 3F     ROL &3F
A655 &>   038 062     26 3E     ROL &3E
A657 *    042         2A        ROL A
A658      202         CA        DEX
A659      208 197     D0 C5     BNE -59   --> &A620
A65B  ;   166 059     A6 3B     LDX &3B
A65D  <   164 060     A4 3C     LDY &3C
A65F      136         88        DEY
A660      016 183     10 B7     BPL -73   --> &A619
A662  >   005 062     05 3E     ORA &3E
A664  ?   005 063     05 3F     ORA &3F
A666  @   005 064     05 40     ORA &40
A668      240 001     F0 01     BEQ 1   --> &A66B
A66A 8    056         38        SEC
A66B      138         8A        TXA
A66C j    106         6A        ROR A
A66D j    106         6A        ROR A
A66E j    106         6A        ROR A
A66F )    041 224     29 E0     AND#&E0
A671  5   133 053     85 35     STA &35
A673  C   165 067     A5 43     LDA &43
A675  4   133 052     85 34     STA &34
A677  D   165 068     A5 44     LDA &44
A679  3   133 051     85 33     STA &33
A67B  E   165 069     A5 45     LDA &45
A67D  2   133 050     85 32     STA &32
A67F  F   165 070     A5 46     LDA &46
A681  1   133 049     85 31     STA &31
```

```
A683 0    048 016     30 10     BMI 16   --> &A695
A685  *   032 042 130 20 2A 82  JSR &822A
A688      128 011     80 0B     BRA 11   --> &A695
A68A      032 202 172 20 CA AC  JSR &ACCA
A68D      032 224 164 20 E0 A4  JSR &A4E0
A690  2   240 050     F0 32     BEQ 50   --> &A6C4
A692  h   032 104 131 20 68 83  JSR &8368
A695  5   165 053     A5 35     LDA &35
A697      201 128     C9 80     CMP#&80
A699      144 019     90 13     BCC 19   --> &A6AE
A69B      240 014     F0 0E     BEQ 14   --> &A6AB
A69D  4   230 052     E6 34     INC &34
A69F      208 013     D0 0D     BNE 13   --> &A6AE
A6A1      032 211 164 20 D3 A4  JSR &A4D3
A6A4      128 008     80 08     BRA 8    --> &A6AE
A6A6      032 207 166 20 CF A6  JSR &A6CF
A6A9      128 234     80 EA     BRA -22  --> &A695
A6AB *    042         2A        ROL A
A6AC  4   004 052     04 34     TSB &34
A6AE  /   165 047     A5 2F     LDA &2F
A6B0      240 016     F0 10     BEQ 16   --> &A6C2
A6B2      016 017     10 11     BPL 17   --> &A6C5
A6B4 d0   100 048     64 30     STZ &30
A6B6 d1   100 049     64 31     STZ &31
A6B8 d.   100 046     64 2E     STZ &2E
A6BA d/   100 047     64 2F     STZ &2F
A6BC d2   100 050     64 32     STZ &32
A6BE d3   100 051     64 33     STZ &33
A6C0 d4   100 052     64 34     STZ &34
A6C2 d5   100 053     64 35     STZ &35
A6C4 `    096         60        RTS
A6C5      000         00        BRK
A6C6      020         14        EQUB &14
A6C7 T    084         54        xxx Invalid Code
A6C8 o    111         6F        xxx Invalid Code
A6C9 o    111         6F        xxx Invalid Code
A6CA  bi  032 098 105 20 62 69  JSR &6962
A6CD g    103         67        xxx Invalid Code
A6CE      000         00        EQUB &00
A6CF  1   165 049     A5 31     LDA &31
A6D1      240 241     F0 F1     BEQ -15  --> &A6C4
A6D3      032 224 164 20 E0 A4  JSR &A4E0
A6D6      240 220     F0 DC     BEQ -36  --> &A6B4
```

```
A6D8      024         18        CLC
A6D9  0   165 048     A5 30     LDA &30
A6DB e<   101 060     65 3C     ADC &3C
A6DD &/   038 047     26 2F     ROL &2F
A6DF      233 127     E9 7F     SBC#&7F
A6E1  0   133 048     85 30     STA &30
A6E3      176 002     B0 02     BCS 2   --> &A6E7
A6E5  /   198 047     C6 2F     DEC &2F
A6E7  .   165 046     A5 2E     LDA &2E
A6E9 E;   069 059     45 3B     EOR &3B
A6EB  .   133 046     85 2E     STA &2E
A6ED      218         DA        PHX
A6EE      162 248     A2 F8     LDX#&F8
A6F0      160 004     A0 04     LDY#&04
A6F2  9   181 057     B5 39     LDA &39,X
A6F4 t9   116 057     74 39     STZ &39,X
A6F6  A   153 065 000 99 41 00  STA &0041,Y
A6F9      232         E8        INX
A6FA      136         88        DEY
A6FB      208 245     D0 F5     BNE -11   --> &A6F2
A6FD d<   100 060     64 3C     STZ &3C
A6FF d;   100 059     64 3B     STZ &3B
A701 d:   100 058     64 3A     STZ &3A
A703  0   128 048     80 30     BRA 48   --> &A735
A705      218         DA        PHX
A706 F=   070 061     46 3D     LSR &3D
A708 f>   102 062     66 3E     ROR &3E
A70A f?   102 063     66 3F     ROR &3F
A70C f@   102 064     66 40     ROR &40
A70E fA   102 065     66 41     ROR &41
A710  F   022 070     16 46     ASL &46,X
A712      144 029     90 1D     BCC 29   --> &A731
A714      024         18        CLC
A715      152         98        TYA
A716 uB   117 066     75 42     ADC &42,X
A718      168         A8        TAY
A719  4   165 052     A5 34     LDA &34
A71B uA   117 065     75 41     ADC &41,X
A71D  4   133 052     85 34     STA &34
A71F  3   165 051     A5 33     LDA &33
A721 u@   117 064     75 40     ADC &40,X
A723  3   133 051     85 33     STA &33
A725  2   165 050     A5 32     LDA &32
```

```
A727 u?   117 063     75 3F     ADC &3F,X
A729  2   133 050     85 32     STA &32
A72B  1   165 049     A5 31     LDA &31
A72D u>   117 062     75 3E     ADC &3E,X
A72F  1   133 049     85 31     STA &31
A731      232         E8        INX
A732 0    048 220     30 DC     BMI -36   --> &A710
A734      250         FA        PLX
A735  F   181 070     B5 46     LDA &46,X
A737      208 204     D0 CC     BNE -52   --> &A705
A739      232         E8        INX
A73A 0    048 249     30 F9     BMI -7    --> &A735
A73C      250         FA        PLX
A73D  5   132 053     84 35     STY &35
A73F  1   165 049     A5 31     LDA &31
A741 0    048 129     30 81     BMI -127  --> &A6C4
A743 L    076 251 129 4C FB 81  JMP &81FB
A746      032 218 150 20 DA 96  JSR &96DA
A749      032 242 163 20 F2 A3  JSR &A3F2
A74C      240 002     F0 02     BEQ 2     --> &A750
A74E      016 022     10 16     BPL 22    --> &A766
A750      000         00        BRK
A751      022         16        EQUB &16
A752 L    076         4C        xxx Invalid Code
A753 o    111         6F        xxx Invalid Code
A754 g    103         67        xxx Invalid Code
A755  ra  032 114 097 20 72 61  JSR &6172
A758 nge  110 103 101 6E 67 65  ROR &6567
A75B      000         00        BRK
A75C      021         15        EQUB &15
A75D -    045         2D        xxx Invalid Code
A75E ve   118 101     76 65     ROR &65,X
A760  ro  032 114 111 20 72 6F  JSR &6F72
A763 o    111         6F        xxx Invalid Code
A764 t    116         74        xxx Invalid Code
A765      000         00        EQUB &00
A766  v   032 118 165 20 76 A5  JSR &A576
A769      160 128     A0 80     LDY#&80
A76B  ;   132 059     84 3B     STY &3B
A76D  =   132 061     84 3D     STY &3D
A76F      200         C8        INY
A770  <   132 060     84 3C     STY &3C
A772  0   166 048     A6 30     LDX &30
```

```
A774      240 006     F0 06     BEQ 6   --> &A77C
A776  1   165 049     A5 31     LDA &31
A778      201 181     C9 B5     CMP#&B5
A77A      144 002     90 02     BCC 2   --> &A77E
A77C      232         E8        INX
A77D      136         88        DEY
A77E      218         DA        PHX
A77F  0   132 048     84 30     STY &30
A781      032 146 166 20 92 A6  JSR &A692
A784  {   169 123     A9 7B     LDA#&7B
A786      032 019 165 20 13 A5  JSR &A513
A789  Q   162 081     A2 51     LDX#&51
A78B  o   169 111     A9 6F     LDA#&6F
A78D      160 002     A0 02     LDY#&02
A78F  a   032 097 168 20 61 A8  JSR &A861
A792  {   169 123     A9 7B     LDA#&7B
A794      032 161 169 20 A1 A9  JSR &A9A1
A797      032 166 166 20 A6 A6  JSR &A6A6
A79A      032 141 166 20 8D A6  JSR &A68D
A79D      032 017 165 20 11 A5  JSR &A511
A7A0 h    104         68        PLA
A7A1 8    056         38        SEC
A7A2      233 129     E9 81     SBC#&81
A7A4      032 213 129 20 D5 81  JSR &81D5
A7A7  L   169 076     A9 4C     LDA#&4C
A7A9      032 212 169 20 D4 A9  JSR &A9D4
A7AC      032 146 165 20 92 A5  JSR &A592
A7AF      032 141 166 20 8D A6  JSR &A68D
A7B2      169 255     A9 FF     LDA#&FF
A7B4 `    096         60        RTS
A7B5      032 218 150 20 DA 96  JSR &96DA
A7B8      032 242 163 20 F2 A3  JSR &A3F2
A7BB      240 245     F0 F5     BEQ -11   --> &A7B2
A7BD 0    048 156     30 9C     BMI -100  --> &A75B
A7BF  0   165 048     A5 30     LDA &30
A7C1 J    074         4A        LSR A
A7C2      008         08        PHP
A7C3 iA   105 065     69 41     ADC#&41
A7C5  0   133 048     85 30     STA &30
A7C7 (    040         28        PLP
A7C8      144 010     90 0A     BCC 10  --> &A7D4
A7CA F1   070 049     46 31     LSR &31
A7CC f2   102 050     66 32     ROR &32
```

```
A7CE f3   102 051     66 33     ROR &33
A7D0 f4   102 052     66 34     ROR &34
A7D2 f5   102 053     66 35     ROR &35
A7D4  p   032 112 165 20 70 A5  JSR &A570
A7D7 dC   100 067     64 43     STZ &43
A7D9 dD   100 068     64 44     STZ &44
A7DB dE   100 069     64 45     STZ &45
A7DD dF   100 070     64 46     STZ &46
A7DF  @   169 064     A9 40     LDA#&40
A7E1  =   133 061     85 3D     STA &3D
A7E3  B   133 066     85 42     STA &42
A7E5      162 251     A2 FB     LDX#&FB
A7E7      160 016     A0 10     LDY#&10
A7E9 8    056         38        SEC
A7EA  1   165 049     A5 31     LDA &31
A7EC  @   233 064     E9 40     SBC#&40
A7EE  1   133 049     85 31     STA &31
A7F0      152         98        TYA
A7F1 UB   085 066     55 42     EOR &42,X
A7F3  G   149 071     95 47     STA &47,X
A7F5  1   165 049     A5 31     LDA &31
A7F7  B   197 066     C5 42     CMP &42
A7F9      208 013     D0 0D     BNE 13   --> &A808
A7FB      218         DA        PHX
A7FC      162 252     A2 FC     LDX#&FC
A7FE  6   181 054     B5 36     LDA &36,X
A800  G   213 071     D5 47     CMP &47,X
A802      208 003     D0 03     BNE 3   --> &A807
A804      232         E8        INX
A805      208 247     D0 F7     BNE -9   --> &A7FE
A807      250         FA        PLX
A808  )   144 041     90 29     BCC 41   --> &A833
A80A  5   165 053     A5 35     LDA &35
A80C  F   229 070     E5 46     SBC &46
A80E  5   133 053     85 35     STA &35
A810  4   165 052     A5 34     LDA &34
A812  E   229 069     E5 45     SBC &45
A814  4   133 052     85 34     STA &34
A816  3   165 051     A5 33     LDA &33
A818  D   229 068     E5 44     SBC &44
A81A  3   133 051     85 33     STA &33
A81C  2   165 050     A5 32     LDA &32
A81E  C   229 067     E5 43     SBC &43
```

```
A820  2   133 050     85 32     STA &32
A822  1   165 049     A5 31     LDA &31
A824  B   229 066     E5 42     SBC &42
A826  1   133 049     85 31     STA &31
A828      152         98        TYA
A829      010         0A        ASL A
A82A      144 011     90 0B     BCC 11   --> &A837
A82C      026         1A        INC A
A82D UA   085 065     55 41     EOR &41,X
A82F  A   149 065     95 41     STA &41,X
A831  F   149 070     95 46     STA &46,X
A833  B   181 066     B5 42     LDA &42,X
A835      128 004     80 04     BRA 4   --> &A83B
A837 UB   085 066     55 42     EOR &42,X
A839  B   149 066     95 42     STA &42,X
A83B  G   149 071     95 47     STA &47,X
A83D  5   006 053     06 35     ASL &35
A83F &4   038 052     26 34     ROL &34
A841 &3   038 051     26 33     ROL &33
A843 &2   038 050     26 32     ROL &32
A845 &1   038 049     26 31     ROL &31
A847      152         98        TYA
A848 J    074         4A        LSR A
A849      168         A8        TAY
A84A      144 164     90 A4     BCC -92   --> &A7F0
A84C      160 128     A0 80     LDY#&80
A84E      232         E8        INX
A84F      208 159     D0 9F     BNE -97   --> &A7F0
A851  S   032 083 131 20 53 83  JSR &8353
A854  1   165 049     A5 31     LDA &31
A856 0    048 003     30 03     BMI 3   --> &A85B
A858      032 251 129 20 FB 81  JSR &81FB
A85B      032 149 166 20 95 A6  JSR &A695
A85E      169 255     A9 FF     LDA#&FF
A860 `    096         60        RTS
A861  G   132 071     84 47     STY &47
A863  L   134 076     86 4C     STX &4C
A865  0   166 048     A6 30     LDX &30
A867  @   224 064     E0 40     CPX#&40
A869  +   144 043     90 2B     BCC 43   --> &A896
A86B      032 233 165 20 E9 A5  JSR &A5E9
A86E      032 017 165 20 11 A5  JSR &A511
A871  L   165 076     A5 4C     LDA &4C
```

```
A873      032 139 165 20 8B A5  JSR &A58B
A876      032 141 166 20 8D A6  JSR &A68D
A879      032 134 168 20 86 A8  JSR &A886
A87C      032 146 165 20 92 A5  JSR &A592
A87F      032 141 166 20 8D A6  JSR &A68D
A882  G   198 071     C6 47     DEC &47
A884      208 243     D0 F3     BNE -13   --> &A879
A886      169 191     A9 BF     LDA#&BF
A888  K   133 075     85 4B     STA &4B
A88A      032 127 165 20 7F A5  JSR &A57F
A88D      032 238 165 20 EE A5  JSR &A5EE
A890      032 127 165 20 7F A5  JSR &A57F
A893 L    076 141 166 4C 8D A6  JMP &A68D
A896      032 139 165 20 8B A5  JSR &A58B
A899 LA   076 065 165 4C 41 A5  JMP &A541
A89C      032 161 168 20 A1 A8  JSR &A8A1
A89F  @   128 064     80 40     BRA 64   --> &A8E1
A8A1      032 218 150 20 DA 96  JSR &96DA
A8A4  .   165 046     A5 2E     LDA &2E
A8A6      016 007     10 07     BPL 7   --> &A8AF
A8A8 d.   100 046     64 2E     STZ &2E
A8AA      032 175 168 20 AF A8  JSR &A8AF
A8AD  #   128 035     80 23     BRA 35   --> &A8D2
A8AF      032 013 165 20 0D A5  JSR &A50D
A8B2  )   032 041 169 20 29 A9  JSR &A929
A8B5  1   165 049     A5 31     LDA &31
A8B7      240 005     F0 05     BEQ 5   --> &A8BE
A8B9      032 179 165 20 B3 A5  JSR &A5B3
A8BC      128 008     80 08     BRA 8   --> &A8C6
A8BE  .   169 046     A9 2E     LDA#&2E
A8C0 L    076 150 168 4C 96 A8  JMP &A896
A8C3      032 218 150 20 DA 96  JSR &96DA
A8C6      032 242 163 20 F2 A3  JSR &A3F2
A8C9  [   240 091     F0 5B     BEQ 91   --> &A926
A8CB      016 008     10 08     BPL 8   --> &A8D5
A8CD d.   100 046     64 2E     STZ &2E
A8CF      032 213 168 20 D5 A8  JSR &A8D5
A8D2  .   133 046     85 2E     STA &2E
A8D4 `    096         60        RTS
A8D5  0   165 048     A5 30     LDA &30
A8D7      201 129     C9 81     CMP#&81
A8D9      144 015     90 0F     BCC 15   --> &A8EA
A8DB      032 233 165 20 E9 A5  JSR &A5E9
```

```
A8DE      032 234 168 20 EA A8  JSR &A8EA
A8E1      032 137 165 20 89 A5  JSR &A589
A8E4      032 138 166 20 8A A6  JSR &A68A
A8E7      169 255     A9 FF     LDA#&FF
A8E9 `    096         60        RTS
A8EA  0   165 048     A5 30     LDA &30
A8EC  s   201 115     C9 73     CMP#&73
A8EE  6   144 054     90 36     BCC 54   --> &A926
A8F0      032 013 165 20 0D A5  JSR &A50D
A8F3  v   032 118 165 20 76 A5  JSR &A576
A8F6      169 128     A9 80     LDA#&80
A8F8  <   133 060     85 3C     STA &3C
A8FA  =   133 061     85 3D     STA &3D
A8FC  ;   133 059     85 3B     STA &3B
A8FE      032 146 166 20 92 A6  JSR &A692
A901      162 151     A2 97     LDX#&97
A903      169 201     A9 C9     LDA#&C9
A905      160 004     A0 04     LDY#&04
A907  a   032 097 168 20 61 A8  JSR &A861
A90A L    076 159 169 4C 9F A9  JMP &A99F
A90D      024         18        CLC
A90E      008         08        PHP
A90F  :   032 058 169 20 3A A9  JSR &A93A
A912 (    040         28        PLP
A913      144 002     90 02     BCC 2   --> &A917
A915  I   230 073     E6 49     INC &49
A917  I   165 073     A5 49     LDA &49
A919      137 002     89 02     BIT#&02
A91B      240 006     F0 06     BEQ 6   --> &A923
A91D  #   032 035 169 20 23 A9  JSR &A923
A920 L    076 202 172 4C CA AC  JMP &ACCA
A923 J    074         4A        LSR A
A924      176 003     B0 03     BCS 3   --> &A929
A926      169 255     A9 FF     LDA#&FF
A928 `    096         60        RTS
A929      032 017 165 20 11 A5  JSR &A511
A92C      032 166 166 20 A6 A6  JSR &A6A6
A92F      169 146     A9 92     LDA#&92
A931      032 139 165 20 8B A5  JSR &A58B
A934      032 138 166 20 8A A6  JSR &A68A
A937 L    076 184 167 4C B8 A7  JMP &A7B8
A93A      032 218 150 20 DA 96  JSR &96DA
A93D  0   165 048     A5 30     LDA &30
```

```
A93F      201 152     C9 98     CMP#&98
A941  j   176 106     B0 6A     BCS 106   --> &A9AD
A943      032 017 165 20 11 A5  JSR &A511
A946      032 137 165 20 89 A5  JSR &A589
A949      032 224 164 20 E0 A4  JSR &A4E0
A94C  .   165 046     A5 2E     LDA &2E
A94E  ;   133 059     85 3B     STA &3B
A950  <   198 060     C6 3C     DEC &3C
A952      032 146 166 20 92 A6  JSR &A692
A955  3   169 051     A9 33     LDA#&33
A957      032 212 169 20 D4 A9  JSR &A9D4
A95A      032 195 150 20 C3 96  JSR &96C3
A95D  I   133 073     85 49     STA &49
A95F  +   005 043     05 2B     ORA &2B
A961  ,   005 044     05 2C     ORA &2C
A963  (   240 040     F0 28     BEQ 40   --> &A98D
A965      032 137 129 20 89 81  JSR &8189
A968  q   169 113     A9 71     LDA#&71
A96A      032 019 165 20 13 A5  JSR &A513
A96D  $   169 036     A9 24     LDA#&24
A96F      032 212 169 20 D4 A9  JSR &A9D4
A972      032 146 165 20 92 A5  JSR &A592
A975      032 141 166 20 8D A6  JSR &A68D
A978      032 025 165 20 19 A5  JSR &A519
A97B  q   169 113     A9 71     LDA#&71
A97D  ;   032 059 165 20 3B A5  JSR &A53B
A980  )   169 041     A9 29     LDA#&29
A982      032 212 169 20 D4 A9  JSR &A9D4
A985      032 146 165 20 92 A5  JSR &A592
A988      032 141 166 20 8D A6  JSR &A68D
A98B      128 003     80 03     BRA 3   --> &A990
A98D  9   032 057 165 20 39 A5  JSR &A539
A990      032 013 165 20 0D A5  JSR &A50D
A993      032 166 166 20 A6 A6  JSR &A6A6
A996  t   162 116     A2 74     LDX#&74
A998      169 146     A9 92     LDA#&92
A99A      160 002     A0 02     LDY#&02
A99C  a   032 097 168 20 61 A8  JSR &A861
A99F  v   169 118     A9 76     LDA#&76
A9A1      160 004     A0 04     LDY#&04
A9A3  K   132 075     84 4B     STY &4B
A9A5  J   133 074     85 4A     STA &4A
A9A7      032 166 166 20 A6 A6  JSR &A6A6
```

```
A9AA      169 255     A9 FF     LDA#&FF
A9AC `    096         60        RTS
A9AD      000         00        BRK
A9AE      023         17        EQUB &17
A9AF Ac   065 099     41 63     EOR (&63,X)
A9B1 c    099         63        xxx Invalid Code
A9B2 ur   117 114     75 72     ADC &72,X
A9B4 acy  097 099 121 61 63 79  ADC (&7963,X)
A9B7  lo  032 108 111 20 6C 6F  JSR &6F6C
A9BA s    115         73        xxx Invalid Code
A9BB t    116         74        xxx Invalid Code
A9BC      000         00        BRK
A9BD      024         18        EQUB &18
A9BE Ex   069 120     45 78     EOR &78
A9C0 p    112 032     70 20     BVS 32   --> &A9E2
A9C2 ra   114 097     72 61     ADC (&61)
A9C4 nge  110 103 101 6E 67 65  ROR &6567
A9C7      000         00        EQUB &00
A9C8      032 218 150 20 DA 96  JSR &96DA
A9CB  8   169 056     A9 38     LDA#&38
A9CD      128 005     80 05     BRA 5   --> &A9D4
A9CF  F   032 070 167 20 46 A7  JSR &A746
A9D2  B   169 066     A9 42     LDA#&42
A9D4      160 191     A0 BF     LDY#&BF
A9D6      128 203     80 CB     BRA -53   --> &A9A3
A9D8      032 218 150 20 DA 96  JSR &96DA
A9DB  =   169 061     A9 3D     LDA#&3D
A9DD      128 245     80 F5     BRA -11   --> &A9D4
A9DF      032 218 150 20 DA 96  JSR &96DA
A9E2  0   165 048     A5 30     LDA &30
A9E4      201 135     C9 87     CMP#&87
A9E6      144 015     90 0F     BCC 15   --> &A9F7
A9E8      208 006     D0 06     BNE 6   --> &A9F0
A9EA  1   164 049     A4 31     LDY &31
A9EC      192 179     C0 B3     CPY#&B3
A9EE      144 007     90 07     BCC 7   --> &A9F7
A9F0  .   165 046     A5 2E     LDA &2E
A9F2      016 200     10 C8     BPL -56   --> &A9BC
A9F4 L    076 180 166 4C B4 A6  JMP &A6B4
A9F7      032 224 130 20 E0 82  JSR &82E0
A9FA      162 206     A2 CE     LDX#&CE
A9FC      169 246     A9 F6     LDA#&F6
A9FE      160 003     A0 03     LDY#&03
```

```
AA00  a   032 097 168 20 61 A8  JSR &A861
AA03      032 013 165 20 0D A5  JSR &A50D
AA06  G   169 071     A9 47     LDA#&47
AA08      032 150 168 20 96 A8  JSR &A896
AA0B  I   165 073     A5 49     LDA &49
AA0D      032 190 165 20 BE A5  JSR &A5BE
AA10      128 141     80 8D     BRA -115   --> &A99F
AA12      032 180 150 20 B4 96  JSR &96B4
AA15      169 129     A9 81     LDA#&81
AA17  *   166 042     A6 2A     LDX &2A
AA19  +   164 043     A4 2B     LDY &2B
AA1B L    076 244 255 4C F4 FF  JMP &FFF4
AA1E      032 030 131 20 1E 83  JSR &831E
AA21 d.   100 046     64 2E     STZ &2E
AA23 d/   100 047     64 2F     STZ &2F
AA25 d5   100 053     64 35     STZ &35
AA27      169 128     A9 80     LDA#&80
AA29  0   133 048     85 30     STA &30
AA2B      160 000     A0 00     LDY#&00
AA2D      162 003     A2 03     LDX#&03
AA2F Y    089 013 000 59 0D 00  EOR &000D,Y
AA32  1   149 049     95 31     STA &31,X
AA34      200         C8        INY
AA35      202         CA        DEX
AA36      016 247     10 F7     BPL -9   --> &AA2F
AA38 LT   076 084 168 4C 54 A8  JMP &A854
AA3B      230 027     E6 1B     INC &1B
AA3D      032 167 150 20 A7 96  JSR &96A7
AA40  -   165 045     A5 2D     LDA &2D
AA42 0%   048 037     30 25     BMI 37   --> &AA69
AA44  ,   005 044     05 2C     ORA &2C
AA46  +   005 043     05 2B     ORA &2B
AA48      208 008     D0 08     BNE 8   --> &AA52
AA4A  *   165 042     A5 2A     LDA &2A
AA4C      240 211     F0 D3     BEQ -45   --> &AA21
AA4E      201 001     C9 01     CMP#&01
AA50      240 204     F0 CC     BEQ -52   --> &AA1E
AA52      032 133 129 20 85 81  JSR &8185
AA55      032 250 187 20 FA BB  JSR &BBFA
AA58      032 030 170 20 1E AA  JSR &AA1E
AA5B      032 232 187 20 E8 BB  JSR &BBE8
AA5E      032 207 166 20 CF A6  JSR &A6CF
AA61      032 195 150 20 C3 96  JSR &96C3
```

```
AA64      032 239 190 20 EF BE  JSR &BEEF
AA67  '   128 039     80 27     BRA 39   --> &AA90
AA69      162 013     A2 0D     LDX#&0D
AA6B      032 198 189 20 C6 BD  JSR &BDC6
AA6E  @   169 064     A9 40     LDA#&40
AA70      133 017     85 11     STA &11
AA72  `   096         60        RTS
AA73      164 027     A4 1B     LDY &1B
AA75      177 025     B1 19     LDA (&19),Y
AA77  (   201 040     C9 28     CMP#&28
AA79      240 192     F0 C0     BEQ -64   --> &AA3B
AA7B      032 030 131 20 1E 83  JSR &831E
AA7E      162 013     A2 0D     LDX#&0D
AA80      181 000     B5 00     LDA &00,X
AA82  *   133 042     85 2A     STA &2A
AA84      181 001     B5 01     LDA &01,X
AA86  +   133 043     85 2B     STA &2B
AA88      181 002     B5 02     LDA &02,X
AA8A  ,   133 044     85 2C     STA &2C
AA8C      181 003     B5 03     LDA &03,X
AA8E  -   133 045     85 2D     STA &2D
AA90  @   169 064     A9 40     LDA#&40
AA92  `   096         60        RTS
AA93      032 180 150 20 B4 96  JSR &96B4
AA96      162 003     A2 03     LDX#&03
AA98  *   181 042     B5 2A     LDA &2A,X
AA9A I    073 255     49 FF     EOR#&FF
AA9C  *   149 042     95 2A     STA &2A,X
AA9E      202         CA        DEX
AA9F      016 247     10 F7     BPL -9   --> &AA98
AAA1      128 237     80 ED     BRA -19   --> &AA90
AAA3      032 188 170 20 BC AA  JSR &AABC
AAA6  *   134 042     86 2A     STX &2A
AAA8  `   096         60        RTS
AAA9      032 180 150 20 B4 96  JSR &96B4
AAAC      032 004 147 20 04 93  JSR &9304
AAAF  *   133 042     85 2A     STA &2A
AAB1  +   134 043     86 2B     STX &2B
AAB3  ,   132 044     84 2C     STY &2C
AAB5      008         08        PHP
AAB6 h    104         68        PLA
AAB7  -   133 045     85 2D     STA &2D
AAB9      216         D8        CLD
```

```
AABA      128 212     80 D4     BRA -44   --> &AA90
AABC      169 134     A9 86     LDA#&86
AABE      032 244 255 20 F4 FF  JSR &FFF4
AAC1      152         98        TYA
AAC2 L    076 024 174 4C 18 AE  JMP &AE18
AAC5      169 002     A9 02     LDA#&02
AAC7      128 002     80 02     BRA 2   --> &AACB
AAC9      169 000     A9 00     LDA#&00
AACB H    072         48        PHA
AACC  J   032 074 186 20 4A BA  JSR &BA4A
AACF  *   162 042     A2 2A     LDX#&2A
AAD1 h    104         68        PLA
AAD2      032 218 255 20 DA FF  JSR &FFDA
AAD5      128 185     80 B9     BRA -71   --> &AA90
AAD7  J   032 074 186 20 4A BA  JSR &BA4A
AADA      032 215 255 20 D7 FF  JSR &FFD7
AADD      128 227     80 E3     BRA -29   --> &AAC2
AADF  @   169 064     A9 40     LDA#&40
AAE1      128 006     80 06     BRA 6   --> &AAE9
AAE3      169 128     A9 80     LDA#&80
AAE5      128 002     80 02     BRA 2   --> &AAE9
AAE7      169 192     A9 C0     LDA#&C0
AAE9 H    072         48        PHA
AAEA  6   032 054 173 20 36 AD  JSR &AD36
AAED      208 013     D0 0D     BNE 13   --> &AAFC
AAEF  +   032 043 190 20 2B BE  JSR &BE2B
AAF2      162 000     A2 00     LDX#&00
AAF4      160 006     A0 06     LDY#&06
AAF6 h    104         68        PLA
AAF7      032 206 255 20 CE FF  JSR &FFCE
AAFA      128 198     80 C6     BRA -58   --> &AAC2
AAFC L    076 146 144 4C 92 90  JMP &9092
AAFF      032 190 168 20 BE A8  JSR &A8BE
AB02  0   230 048     E6 30     INC &30
AB04 `    096         60        RTS
AB05  6   032 054 173 20 36 AD  JSR &AD36
AB08  <   208 060     D0 3C     BNE 60   --> &AB46
AB0A  6   230 054     E6 36     INC &36
AB0C  6   164 054     A4 36     LDY &36
AB0E      169 013     A9 0D     LDA#&0D
AB10      153 255 005 99 FF 05  STA &05FF,Y
AB13  Q   032 081 188 20 51 BC  JSR &BC51
AB16      165 025     A5 19     LDA &19
```

```
AB18 H    072         48        PHA
AB19      165 026     A5 1A     LDA &1A
AB1B H    072         48        PHA
AB1C      165 027     A5 1B     LDA &1B
AB1E H    072         48        PHA
AB1F      164 004     A4 04     LDY &04
AB21      166 005     A6 05     LDX &05
AB23      200         C8        INY
AB24      132 025     84 19     STY &19
AB26  7   132 055     84 37     STY &37
AB28      208 001     D0 01     BNE 1   --> &AB2B
AB2A      232         E8        INX
AB2B      134 026     86 1A     STX &1A
AB2D  8   134 056     86 38     STX &38
AB2F      032 031 142 20 1F 8E  JSR &8E1F
AB32 d    100 027     64 1B     STZ &1B
AB34  ;   032 059 157 20 3B 9D  JSR &9D3B
AB37      032 225 188 20 E1 BC  JSR &BCE1
AB3A h    104         68        PLA
AB3B      133 027     85 1B     STA &1B
AB3D h    104         68        PLA
AB3E      133 026     85 1A     STA &1A
AB40 h    104         68        PLA
AB41      133 025     85 19     STA &19
AB43  '   165 039     A5 27     LDA &27
AB45 `    096         60        RTS
AB46 L    076 146 144 4C 92 90  JMP &9092
AB49  6   032 054 173 20 36 AD  JSR &AD36
AB4C      208 248     D0 F8     BNE -8   --> &AB46
AB4E  6   166 054     A6 36     LDX &36
AB50      158 000 006 9E 00 06  STZ &0600,X
AB53      165 025     A5 19     LDA &19
AB55 H    072         48        PHA
AB56      165 026     A5 1A     LDA &1A
AB58 H    072         48        PHA
AB59      165 027     A5 1B     LDA &1B
AB5B H    072         48        PHA
AB5C d    100 027     64 1B     STZ &1B
AB5E d    100 025     64 19     STZ &19
AB60      169 006     A9 06     LDA#&06
AB62      133 026     85 1A     STA &1A
AB64      032 213 142 20 D5 8E  JSR &8ED5
AB67  -   201 045     C9 2D     CMP#&2D
```

```
AB69      240 014     F0 0E     BEQ 14   --> &AB79
AB6B  +   201 043     C9 2B     CMP#&2B
AB6D      208 003     D0 03     BNE 3   --> &AB72
AB6F      032 213 142 20 D5 8E  JSR &8ED5
AB72      198 027     C6 1B     DEC &1B
AB74      032 225 162 20 E1 A2  JSR &A2E1
AB77      128 013     80 0D     BRA 13   --> &AB86
AB79      032 213 142 20 D5 8E  JSR &8ED5
AB7C      198 027     C6 1B     DEC &1B
AB7E      032 225 162 20 E1 A2  JSR &A2E1
AB81      144 003     90 03     BCC 3   --> &AB86
AB83      032 218 172 20 DA AC  JSR &ACDA
AB86  '   133 039     85 27     STA &27
AB88      128 176     80 B0     BRA -80   --> &AB3A
AB8A  6   032 054 173 20 36 AD  JSR &AD36
AB8D  =   240 061     F0 3D     BEQ 61   --> &ABCC
AB8F  !   016 033     10 21     BPL 33   --> &ABB2
AB91  .   165 046     A5 2E     LDA &2E
AB93      008         08        PHP
AB94  u   032 117 130 20 75 82  JSR &8275
AB97 (    040         28        PLP
AB98      016 019     10 13     BPL 19   --> &ABAD
AB9A  =   165 061     A5 3D     LDA &3D
AB9C  >   005 062     05 3E     ORA &3E
AB9E  ?   005 063     05 3F     ORA &3F
ABA0  @   005 064     05 40     ORA &40
ABA2      240 009     F0 09     BEQ 9   --> &ABAD
ABA4      032 200 130 20 C8 82  JSR &82C8
ABA7      032 013 131 20 0D 83  JSR &830D
ABAA      032 200 130 20 C8 82  JSR &82C8
ABAD      032 198 150 20 C6 96  JSR &96C6
ABB0  @   169 064     A9 40     LDA#&40
ABB2  `   096         60        RTS
ABB3  6   032 054 173 20 36 AD  JSR &AD36
ABB6      208 020     D0 14     BNE 20   --> &ABCC
ABB8  6   165 054     A5 36     LDA &36
ABBA      240 031     F0 1F     BEQ 31   --> &ABDB
ABBC      173 000 006 AD 00 06  LDA &0600
ABBF L    076 024 174 4C 18 AE  JMP &AE18
ABC2      032 018 170 20 12 AA  JSR &AA12
ABC5      152         98        TYA
ABC6      208 019     D0 13     BNE 19   --> &ABDB
ABC8      138         8A        TXA
```

```
ABC9 L    076 026 174 4C 1A AE  JMP &AE1A
ABCC L    076 146 144 4C 92 90  JMP &9092
ABCF  J   032 074 186 20 4A BA  JSR &BA4A
ABD2      170         AA        TAX
ABD3      169 127     A9 7F     LDA#&7F
ABD5      032 244 255 20 F4 FF  JSR &FFF4
ABD8      138         8A        TXA
ABD9      240 002     F0 02     BEQ 2   --> &ABDD
ABDB      162 255     A2 FF     LDX#&FF
ABDD  *   134 042     86 2A     STX &2A
ABDF  +   134 043     86 2B     STX &2B
ABE1  ,   134 044     86 2C     STX &2C
ABE3  -   134 045     86 2D     STX &2D
ABE5  @   169 064     A9 40     LDA#&40
ABE7  `   096         60        RTS
ABE8      162 000     A2 00     LDX#&00
ABEA      128 241     80 F1     BRA -15   --> &ABDD
ABEC      032 242 163 20 F2 A3  JSR &A3F2
ABEF      240 247     F0 F7     BEQ -9   --> &ABE8
ABF1      016 023     10 17     BPL 23   --> &AC0A
ABF3      128 230     80 E6     BRA -26   --> &ABDB
ABF5  6   032 054 173 20 36 AD  JSR &AD36
ABF8      240 210     F0 D2     BEQ -46   --> &ABCC
ABFA 0    048 240     30 F0     BMI -16   --> &ABEC
ABFC  -   165 045     A5 2D     LDA &2D
ABFE  ,   005 044     05 2C     ORA &2C
AC00  +   005 043     05 2B     ORA &2B
AC02  *   005 042     05 2A     ORA &2A
AC04      240 223     F0 DF     BEQ -33   --> &ABE5
AC06  -   165 045     A5 2D     LDA &2D
AC08 0    048 209     30 D1     BMI -47   --> &ABDB
AC0A      169 001     A9 01     LDA#&01
AC0C      128 177     80 B1     BRA -79   --> &ABBF
AC0E      032 175 150 20 AF 96  JSR &96AF
AC11  &   032 038 188 20 26 BC  JSR &BC26
AC14      032 241 142 20 F1 8E  JSR &8EF1
AC17      032 167 150 20 A7 96  JSR &96A7
AC1A  *   165 042     A5 2A     LDA &2A
AC1C H    072         48        PHA
AC1D  +   166 043     A6 2B     LDX &2B
AC1F      032 230 188 20 E6 BC  JSR &BCE6
AC22  -   134 045     86 2D     STX &2D
AC24 h    104         68        PLA
```

```
AC25  ,   133 044     85 2C     STA &2C
AC27      160 000     A0 00     LDY#&00
AC29  *   162 042     A2 2A     LDX#&2A
AC2B      169 009     A9 09     LDA#&09
AC2D      032 241 255 20 F1 FF  JSR &FFF1
AC30  .   165 046     A5 2E     LDA &2E
AC32 0    048 167     30 A7     BMI -89   --> &ABDB
AC34      128 214     80 D6     BRA -42   --> &AC0C
AC36  ;   032 059 157 20 3B 9D  JSR &9D3B
AC39      208 145     D0 91     BNE -111  --> &ABCC
AC3B  ,   224 044     E0 2C     CPX#&2C
AC3D      208 024     D0 18     BNE 24    --> &AC57
AC3F      230 027     E6 1B     INC &1B
AC41  Q   032 081 188 20 51 BC  JSR &BC51
AC44  ;   032 059 157 20 3B 9D  JSR &9D3B
AC47      208 131     D0 83     BNE -125  --> &ABCC
AC49      169 001     A9 01     LDA#&01
AC4B  *   133 042     85 2A     STA &2A
AC4D      230 027     E6 1B     INC &1B
AC4F  )   224 041     E0 29     CPX#&29
AC51      240 013     F0 0D     BEQ 13    --> &AC60
AC53  ,   224 044     E0 2C     CPX#&2C
AC55      240 003     F0 03     BEQ 3     --> &AC5A
AC57 L    076 246 142 4C F6 8E  JMP &8EF6
AC5A      032 164 150 20 A4 96  JSR &96A4
AC5D      032 210 188 20 D2 BC  JSR &BCD2
AC60  *   166 042     A6 2A     LDX &2A
AC62      208 002     D0 02     BNE 2     --> &AC66
AC64      162 001     A2 01     LDX#&01
AC66  *   134 042     86 2A     STX &2A
AC68      138         8A        TXA
AC69      202         CA        DEX
AC6A  -   134 045     86 2D     STX &2D
AC6C      024         18        CLC
AC6D e    101 004     65 04     ADC &04
AC6F  7   133 055     85 37     STA &37
AC71      169 000     A9 00     LDA#&00
AC73 e    101 005     65 05     ADC &05
AC75  8   133 056     85 38     STA &38
AC77      178 004     B2 04     LDA (&04)
AC79 8    056         38        SEC
AC7A  -   229 045     E5 2D     SBC &2D
AC7C  !   144 033     90 21     BCC 33    --> &AC9F
```

```
AC7E  6   229 054     E5 36     SBC &36
AC80      144 029     90 1D     BCC 29   --> &AC9F
AC82 i    105 000     69 00     ADC#&00
AC84  +   133 043     85 2B     STA &2B
AC86      032 225 188 20 E1 BC  JSR &BCE1
AC89      160 000     A0 00     LDY#&00
AC8B  6   166 054     A6 36     LDX &36
AC8D      240 011     F0 0B     BEQ 11   --> &AC9A
AC8F  7   177 055     B1 37     LDA (&37),Y
AC91      217 000 006 D9 00 06  CMP &0600,Y
AC94      208 016     D0 10     BNE 16   --> &ACA6
AC96      200         C8        INY
AC97      202         CA        DEX
AC98      208 245     D0 F5     BNE -11   --> &AC8F
AC9A  *   165 042     A5 2A     LDA &2A
AC9C L    076 024 174 4C 18 AE  JMP &AE18
AC9F      032 225 188 20 E1 BC  JSR &BCE1
ACA2      169 000     A9 00     LDA#&00
ACA4      128 246     80 F6     BRA -10   --> &AC9C
ACA6  *   230 042     E6 2A     INC &2A
ACA8  +   198 043     C6 2B     DEC &2B
ACAA      240 246     F0 F6     BEQ -10   --> &ACA2
ACAC  7   230 055     E6 37     INC &37
ACAE      208 217     D0 D9     BNE -39   --> &AC89
ACB0  8   230 056     E6 38     INC &38
ACB2      128 213     80 D5     BRA -43   --> &AC89
ACB4 L    076 146 144 4C 92 90  JMP &9092
ACB7  6   032 054 173 20 36 AD  JSR &AD36
ACBA      240 248     F0 F8     BEQ -8   --> &ACB4
ACBC 0    048 006     30 06     BMI 6   --> &ACC4
ACBE $-   036 045     24 2D     BIT &2D
ACC0 0    048 028     30 1C     BMI 28   --> &ACDE
ACC2  1   128 049     80 31     BRA 49   --> &ACF5
ACC4 d.   100 046     64 2E     STZ &2E
ACC6 `    096         60        RTS
ACC7      032 138 166 20 8A A6  JSR &A68A
ACCA  1   165 049     A5 31     LDA &31
ACCC      240 006     F0 06     BEQ 6   --> &ACD4
ACCE  .   165 046     A5 2E     LDA &2E
ACD0 I    073 128     49 80     EOR#&80
ACD2  .   133 046     85 2E     STA &2E
ACD4      169 255     A9 FF     LDA#&FF
ACD6 `    096         60        RTS
```

```
ACD7  L   032 076 173 20 4C AD  JSR &AD4C
ACDA      240 216     F0 D8     BEQ -40   --> &ACB4
ACDC 0    048 236     30 EC     BMI -20   --> &ACCA
ACDE 8    056         38        SEC
ACDF      169 000     A9 00     LDA#&00
ACE1      168         A8        TAY
ACE2  *   229 042     E5 2A     SBC &2A
ACE4  *   133 042     85 2A     STA &2A
ACE6      152         98        TYA
ACE7  +   229 043     E5 2B     SBC &2B
ACE9  +   133 043     85 2B     STA &2B
ACEB      152         98        TYA
ACEC  ,   229 044     E5 2C     SBC &2C
ACEE  ,   133 044     85 2C     STA &2C
ACF0      152         98        TYA
ACF1  -   229 045     E5 2D     SBC &2D
ACF3  -   133 045     85 2D     STA &2D
ACF5  @   169 064     A9 40     LDA#&40
ACF7  `   096         60        RTS
ACF8      032 213 142 20 D5 8E  JSR &8ED5
ACFB  "   201 034     C9 22     CMP#&22
ACFD      240 026     F0 1A     BEQ 26   --> &AD19
ACFF      162 000     A2 00     LDX#&00
AD01      177 025     B1 19     LDA (&19),Y
AD03      157 000 006 9D 00 06  STA &0600,X
AD06      200         C8        INY
AD07      232         E8        INX
AD08      201 013     C9 0D     CMP#&0D
AD0A      240 004     F0 04     BEQ 4   --> &AD10
AD0C  ,   201 044     C9 2C     CMP#&2C
AD0E      208 241     D0 F1     BNE -15   --> &AD01
AD10      136         88        DEY
AD11      202         CA        DEX
AD12  6   134 054     86 36     STX &36
AD14      132 027     84 1B     STY &1B
AD16      169 000     A9 00     LDA#&00
AD18  `   096         60        RTS
AD19      162 000     A2 00     LDX#&00
AD1B      200         C8        INY
AD1C      177 025     B1 19     LDA (&19),Y
AD1E      201 013     C9 0D     CMP#&0D
AD20      240 017     F0 11     BEQ 17   --> &AD33
AD22      157 000 006 9D 00 06  STA &0600,X
```

```
AD25      200        C8       INY
AD26      232        E8       INX
AD27  "   201 034    C9 22    CMP#&22
AD29      208 241    D0 F1    BNE -15   --> &AD1C
AD2B      177 025    B1 19    LDA (&19),Y
AD2D  "   201 034    C9 22    CMP#&22
AD2F      240 234    F0 EA    BEQ -22   --> &AD1B
AD31      208 222    D0 DE    BNE -34   --> &AD11
AD33 L    076 148 146 4C 94 92  JMP &9294
AD36      164 027    A4 1B    LDY &1B
AD38      230 027    E6 1B    INC &1B
AD3A      177 025    B1 19    LDA (&19),Y
AD3C      201 032    C9 20    CMP#&20
AD3E      240 246    F0 F6    BEQ -10   --> &AD36
AD40  -   201 045    C9 2D    CMP#&2D
AD42      240 147    F0 93    BEQ -109  --> &ACD7
AD44  "   201 034    C9 22    CMP#&22
AD46      240 209    F0 D1    BEQ -47   --> &AD19
AD48  +   201 043    C9 2B    CMP#&2B
AD4A      208 003    D0 03    BNE 3     --> &AD4F
AD4C      032 213 142 20 D5 8E  JSR &8ED5
AD4F      201 142    C9 8E    CMP#&8E
AD51      144 007    90 07    BCC 7     --> &AD5A
AD53      201 198    C9 C6    CMP#&C6
AD55  5   176 053    B0 35    BCS 53    --> &AD8C
AD57 L    076 025 144 4C 19 90  JMP &9019
AD5A  ?   201 063    C9 3F    CMP#&3F
AD5C      176 012    B0 0C    BCS 12    --> &AD6A
AD5E  .   201 046    C9 2E    CMP#&2E
AD60      176 018    B0 12    BCS 18    --> &AD74
AD62  &   201 038    C9 26    CMP#&26
AD64  Q   240 081    F0 51    BEQ 81    --> &ADB7
AD66  (   201 040    C9 28    CMP#&28
AD68  B   240 066    F0 42    BEQ 66    --> &ADAC
AD6A      198 027    C6 1B    DEC &1B
AD6C      032 009 153 20 09 99  JSR &9909
AD6F      240 009    F0 09    BEQ 9     --> &AD7A
AD71 L    076 160 177 4C A0 B1  JMP &B1A0
AD74      032 225 162 20 E1 A2  JSR &A2E1
AD77      144 019    90 13    BCC 19    --> &AD8C
AD79 `    096        60       RTS
AD7A  (   165 040    A5 28    LDA &28
AD7C )    041 002    29 02    AND#&02
```

```
AD7E      208 012     D0 0C     BNE 12   --> &AD8C
AD80      176 010     B0 0A     BCS 10   --> &AD8C
AD82      134 027     86 1B     STX &1B
AD84  @   173 064 004 AD 40 04  LDA &0440
AD87  A   172 065 004 AC 41 04  LDY &0441
AD8A  k   128 107     80 6B     BRA 107   --> &ADF7
AD8C      000         00        BRK
AD8D      026         1A        EQUB &1A
AD8E No   078 111 032 4E 6F 20  LSR &206F
AD91 s    115         73        xxx Invalid Code
AD92 uc   117 099     75 63     ADC &63,X
AD94 h    104         68        PLA
AD95  va  032 118 097 20 76 61  JSR &6176
AD98 ri   114 105     72 69     ADC (&69)
AD9A abl  097 098 108 61 62 6C  ADC (&6C62,X)
AD9D e    101         65        xxx Invalid Code
AD9E      000         00        BRK
AD9F      027         1B        EQUB &1B
ADA0  )   141 041 000 8D 29 00  STA &0029
ADA3  Ba  028 066 097 1C 42 61  TRB &6142
ADA6 d    100 032     64 20     STZ &20
ADA8 H    072         48        PHA
ADA9 ex   101 120     65 78     ADC &78
ADAB      000         00        EQUB &00
ADAC  ;   032 059 157 20 3B 9D  JSR &9D3B
ADAF      230 027     E6 1B     INC &1B
ADB1  )   224 041     E0 29     CPX#&29
ADB3      208 233     D0 E9     BNE -23   --> &AD9E
ADB5      168         A8        TAY
ADB6 `    096         60        RTS
ADB7      032 232 171 20 E8 AB  JSR &ABE8
ADBA      200         C8        INY
ADBB      177 025     B1 19     LDA (&19),Y
ADBD  0   201 048     C9 30     CMP#&30
ADBF  #   144 035     90 23     BCC 35   --> &ADE4
ADC1  :   201 058     C9 3A     CMP#&3A
ADC3      144 010     90 0A     BCC 10   --> &ADCF
ADC5  7   233 055     E9 37     SBC#&37
ADC7      201 010     C9 0A     CMP#&0A
ADC9      144 025     90 19     BCC 25   --> &ADE4
ADCB      201 016     C9 10     CMP#&10
ADCD      176 021     B0 15     BCS 21   --> &ADE4
ADCF      010         0A        ASL A
```

```
ADD0      010         0A        ASL A
ADD1      010         0A        ASL A
ADD2      010         0A        ASL A
ADD3      162 003     A2 03     LDX#&03
ADD5      010         0A        ASL A
ADD6 &*   038 042     26 2A     ROL &2A
ADD8 &+   038 043     26 2B     ROL &2B
ADDA &,   038 044     26 2C     ROL &2C
ADDC &-   038 045     26 2D     ROL &2D
ADDE      202         CA        DEX
ADDF      016 244     10 F4     BPL -12   --> &ADD5
ADE1      200         C8        INY
ADE2      208 215     D0 D7     BNE -41   --> &ADBB
ADE4      138         8A        TXA
ADE5      016 187     10 BB     BPL -69   --> &ADA2
ADE7      132 027     84 1B     STY &1B
ADE9  @   169 064     A9 40     LDA#&40
ADEB `    096         60        RTS
ADEC      032 180 150 20 B4 96  JSR &96B4
ADEF  *   166 042     A6 2A     LDX &2A
ADF1      169 128     A9 80     LDA#&80
ADF3      032 244 255 20 F4 FF  JSR &FFF4
ADF6      138         8A        TXA
ADF7  !   128 033     80 21     BRA 33   --> &AE1A
ADF9      200         C8        INY
ADFA      177 025     B1 19     LDA (&19),Y
ADFC  P   201 080     C9 50     CMP#&50
ADFE      208 140     D0 8C     BNE -116   --> &AD8C
AE00      230 027     E6 1B     INC &1B
AE02      165 018     A5 12     LDA &12
AE04      164 019     A4 13     LDY &13
AE06      128 018     80 12     BRA 18   --> &AE1A
AE08      164 024     A4 18     LDY &18
AE0A      169 000     A9 00     LDA#&00
AE0C      128 012     80 0C     BRA 12   --> &AE1A
AE0E L    076 146 144 4C 92 90  JMP &9092
AE11  6   032 054 173 20 36 AD  JSR &AD36
AE14      208 248     D0 F8     BNE -8   --> &AE0E
AE16  6   165 054     A5 36     LDA &36
AE18      160 000     A0 00     LDY#&00
AE1A  *   133 042     85 2A     STA &2A
AE1C  +   132 043     84 2B     STY &2B
AE1E d,   100 044     64 2C     STZ &2C
```

```
AE20 d-   100 045     64 2D     STZ &2D
AE22  @   169 064     A9 40     LDA#&40
AE24 `    096         60        RTS
AE25      165 030     A5 1E     LDA &1E
AE27      128 239     80 EF     BRA -17   --> &AE18
AE29      165 000     A5 00     LDA &00
AE2B      164 001     A4 01     LDY &01
AE2D      128 235     80 EB     BRA -21   --> &AE1A
AE2F      165 006     A5 06     LDA &06
AE31      164 007     A4 07     LDY &07
AE33      128 229     80 E5     BRA -27   --> &AE1A
AE35      164 009     A4 09     LDY &09
AE37      165 008     A5 08     LDA &08
AE39      128 223     80 DF     BRA -33   --> &AE1A
AE3B      178 253     B2 FD     LDA (&FD)
AE3D      128 217     80 D9     BRA -39   --> &AE18
AE3F      032 224 255 20 E0 FF  JSR &FFE0
AE42      128 212     80 D4     BRA -44   --> &AE18
AE44      200         C8        INY
AE45      177 025     B1 19     LDA (&19),Y
AE47  $   201 036     C9 24     CMP#&24
AE49      240 012     F0 0C     BEQ 12    --> &AE57
AE4B  *   162 042     A2 2A     LDX#&2A
AE4D      160 000     A0 00     LDY#&00
AE4F      169 001     A9 01     LDA#&01
AE51      032 241 255 20 F1 FF  JSR &FFF1
AE54  @   169 064     A9 40     LDA#&40
AE56 `    096         60        RTS
AE57      230 027     E6 1B     INC &1B
AE59      169 014     A9 0E     LDA#&0E
AE5B      162 000     A2 00     LDX#&00
AE5D      160 006     A0 06     LDY#&06
AE5F      156 000 006 9C 00 06  STZ &0600
AE62      032 241 255 20 F1 FF  JSR &FFF1
AE65      169 024     A9 18     LDA#&18
AE67  &   128 038     80 26     BRA 38    --> &AE8F
AE69      032 224 255 20 E0 FF  JSR &FFE0
AE6C      141 000 006 8D 00 06  STA &0600
AE6F      169 001     A9 01     LDA#&01
AE71      128 028     80 1C     BRA 28    --> &AE8F
AE73      024         18        CLC
AE74      008         08        PHP
AE75  ;   032 059 157 20 3B 9D  JSR &9D3B
```

```
AE78  E   208 069     D0 45     BNE 69   --> &AEBF
AE7A  ,   224 044     E0 2C     CPX#&2C
AE7C  D   208 068     D0 44     BNE 68   --> &AEC2
AE7E      230 027     E6 1B     INC &1B
AE80      032 164 150 20 A4 96  JSR &96A4
AE83      032 210 188 20 D2 BC  JSR &BCD2
AE86 (    040         28        PLP
AE87      176 011     B0 0B     BCS 11   --> &AE94
AE89  *   165 042     A5 2A     LDA &2A
AE8B  6   197 054     C5 36     CMP &36
AE8D      176 002     B0 02     BCS 2    --> &AE91
AE8F  6   133 054     85 36     STA &36
AE91      169 000     A9 00     LDA#&00
AE93 `    096         60        RTS
AE94  6   165 054     A5 36     LDA &36
AE96  *   229 042     E5 2A     SBC &2A
AE98      144 247     90 F7     BCC -9   --> &AE91
AE9A      240 247     F0 F7     BEQ -9   --> &AE93
AE9C      170         AA        TAX
AE9D  *   165 042     A5 2A     LDA &2A
AE9F  6   133 054     85 36     STA &36
AEA1      240 240     F0 F0     BEQ -16  --> &AE93
AEA3      160 000     A0 00     LDY#&00
AEA5      189 000 006 BD 00 06  LDA &0600,X
AEA8      153 000 006 99 00 06  STA &0600,Y
AEAB      232         E8        INX
AEAC      200         C8        INY
AEAD  *   198 042     C6 2A     DEC &2A
AEAF      208 244     D0 F4     BNE -12  --> &AEA5
AEB1      128 222     80 DE     BRA -34  --> &AE91
AEB3      032 018 170 20 12 AA  JSR &AA12
AEB6      138         8A        TXA
AEB7      192 000     C0 00     CPY#&00
AEB9      240 177     F0 B1     BEQ -79  --> &AE6C
AEBB      169 000     A9 00     LDA#&00
AEBD      128 208     80 D0     BRA -48  --> &AE8F
AEBF L    076 146 144 4C 92 90  JMP &9092
AEC2 L    076 246 142 4C F6 8E  JMP &8EF6
AEC5  ;   032 059 157 20 3B 9D  JSR &9D3B
AEC8      208 245     D0 F5     BNE -11  --> &AEBF
AECA  ,   224 044     E0 2C     CPX#&2C
AECC      208 244     D0 F4     BNE -12  --> &AEC2
AECE  Q   032 081 188 20 51 BC  JSR &BC51
```

```
AED1      230 027     E6 1B     INC &1B
AED3      032 175 150 20 AF 96  JSR &96AF
AED6  *   165 042     A5 2A     LDA &2A
AED8 H    072         48        PHA
AED9      169 255     A9 FF     LDA#&FF
AEDB  *   133 042     85 2A     STA &2A
AEDD      230 027     E6 1B     INC &1B
AEDF  )   224 041     E0 29     CPX#&29
AEE1      240 007     F0 07     BEQ 7   --> &AEEA
AEE3  ,   224 044     E0 2C     CPX#&2C
AEE5      208 219     D0 DB     BNE -37   --> &AEC2
AEE7      032 167 150 20 A7 96  JSR &96A7
AEEA      032 210 188 20 D2 BC  JSR &BCD2
AEED h    104         68        PLA
AEEE      168         A8        TAY
AEEF      024         18        CLC
AEF0      240 006     F0 06     BEQ 6   --> &AEF8
AEF2  6   229 054     E5 36     SBC &36
AEF4      176 197     B0 C5     BCS -59   --> &AEBB
AEF6      136         88        DEY
AEF7      152         98        TYA
AEF8  ,   133 044     85 2C     STA &2C
AEFA      170         AA        TAX
AEFB      160 000     A0 00     LDY#&00
AEFD  6   165 054     A5 36     LDA &36
AEFF 8    056         38        SEC
AF00  ,   229 044     E5 2C     SBC &2C
AF02  *   197 042     C5 2A     CMP &2A
AF04      176 002     B0 02     BCS 2   --> &AF08
AF06  *   133 042     85 2A     STA &2A
AF08  *   165 042     A5 2A     LDA &2A
AF0A      240 175     F0 AF     BEQ -81   --> &AEBB
AF0C      189 000 006 BD 00 06  LDA &0600,X
AF0F      153 000 006 99 00 06  STA &0600,Y
AF12      200         C8        INY
AF13      232         E8        INX
AF14  *   196 042     C4 2A     CPY &2A
AF16      208 244     D0 F4     BNE -12   --> &AF0C
AF18  6   132 054     84 36     STY &36
AF1A  ^   128 094     80 5E     BRA 94   --> &AF7A
AF1C      032 213 142 20 D5 8E  JSR &8ED5
AF1F      160 255     A0 FF     LDY#&FF
AF21  ~   201 126     C9 7E     CMP#&7E
```

```
AF23      240 004     F0 04     BEQ 4   --> &AF29
AF25      160 000     A0 00     LDY#&00
AF27      198 027     C6 1B     DEC &1B
AF29 Z    090         5A        PHY
AF2A  6   032 054 173 20 36 AD  JSR &AD36
AF2D      240 021     F0 15     BEQ 21   --> &AF44
AF2F      168         A8        TAY
AF30 h    104         68        PLA
AF31      133 021     85 15     STA &15
AF33      173 003 004 AD 03 04  LDA &0403
AF36      208 007     D0 07     BNE 7   --> &AF3F
AF38  7   133 055     85 37     STA &37
AF3A  2   032 050 161 20 32 A1  JSR &A132
AF3D  ;   128 059     80 3B     BRA 59   --> &AF7A
AF3F      032 024 161 20 18 A1  JSR &A118
AF42  6   128 054     80 36     BRA 54   --> &AF7A
AF44 L    076 146 144 4C 92 90  JMP &9092
AF47      032 175 150 20 AF 96  JSR &96AF
AF4A  &   032 038 188 20 26 BC  JSR &BC26
AF4D      032 241 142 20 F1 8E  JSR &8EF1
AF50      032 172 173 20 AC AD  JSR &ADAC
AF53      208 239     D0 EF     BNE -17   --> &AF44
AF55      032 230 188 20 E6 BC  JSR &BCE6
AF58  6   164 054     A4 36     LDY &36
AF5A      240 030     F0 1E     BEQ 30   --> &AF7A
AF5C  *   165 042     A5 2A     LDA &2A
AF5E      240 029     F0 1D     BEQ 29   --> &AF7D
AF60  *   198 042     C6 2A     DEC &2A
AF62      240 022     F0 16     BEQ 22   --> &AF7A
AF64      162 000     A2 00     LDX#&00
AF66      189 000 006 BD 00 06  LDA &0600,X
AF69      153 000 006 99 00 06  STA &0600,Y
AF6C      232         E8        INX
AF6D      200         C8        INY
AF6E      240 016     F0 10     BEQ 16   --> &AF80
AF70  6   228 054     E4 36     CPX &36
AF72      144 242     90 F2     BCC -14   --> &AF66
AF74  *   198 042     C6 2A     DEC &2A
AF76      208 236     D0 EC     BNE -20   --> &AF64
AF78  6   132 054     84 36     STY &36
AF7A      169 000     A9 00     LDA#&00
AF7C `    096         60        RTS
AF7D  6   133 054     85 36     STA &36
```

```
AF7F `   096       60       RTS
AF80 L   076 016 158 4C 10 9E  JMP &9E10
AF83 h   104       68       PLA
AF84     133 012   85 0C    STA &0C
AF86 h   104       68       PLA
AF87     133 011   85 0B    STA &0B
AF89     000       00       BRK
AF8A     029       1D       EQUB &1D
AF8B No  078 111   4E 6F    xxx Invalid Code
AF8D  su 032 115 117 20 73 75  JSR &7573
AF90 c   099       63       xxx Invalid Code
AF91 h   104       68       PLA
AF92   /  032 164 047 20 A4 2F  JSR &2FA4
AF95     242       F2       xxx Invalid Code
AF96     000       00       EQUB &00
AF97     165 024   A5 18    LDA &18
AF99     133 012   85 0C    STA &0C
AF9B d   100 011   64 0B    STZ &0B
AF9D     160 001   A0 01    LDY#&01
AF9F     177 011   B1 0B    LDA (&0B),Y
AFA1 0   048 224   30 E0    BMI -32   --> &AF83
AFA3     160 003   A0 03    LDY#&03
AFA5     200       C8       INY
AFA6     177 011   B1 0B    LDA (&0B),Y
AFA8     201 032   C9 20    CMP#&20
AFAA     240 249   F0 F9    BEQ -7   --> &AFA5
AFAC     201 221   C9 DD    CMP#&DD
AFAE     240 015   F0 0F    BEQ 15   --> &AFBF
AFB0     160 003   A0 03    LDY#&03
AFB2     177 011   B1 0B    LDA (&0B),Y
AFB4     024       18       CLC
AFB5 e   101 011   65 0B    ADC &0B
AFB7     133 011   85 0B    STA &0B
AFB9     144 226   90 E2    BCC -30   --> &AF9D
AFBB     230 012   E6 0C    INC &0C
AFBD     128 222   80 DE    BRA -34   --> &AF9D
AFBF     200       C8       INY
AFC0     132 010   84 0A    STY &0A
AFC2     032 224 142 20 E0 8E  JSR &8EE0
AFC5     152       98       TYA
AFC6     170       AA       TAX
AFC7     024       18       CLC
AFC8 e   101 011   65 0B    ADC &0B
```

```
AFCA     164 012    A4 0C    LDY &0C
AFCC     144 002    90 02    BCC 2   --> &AFD0
AFCE     200        C8       INY
AFCF     024        18       CLC
AFD0     233 000    E9 00    SBC#&00
AFD2  <  133 060    85 3C    STA &3C
AFD4     152        98       TYA
AFD5     233 000    E9 00    SBC#&00
AFD7  =  133 061    85 3D    STA &3D
AFD9     160 001    A0 01    LDY#&01
AFDB     232        E8       INX
AFDC  <  177 060    B1 3C    LDA (&3C),Y
AFDE  7  209 055    D1 37    CMP (&37),Y
AFE0     208 206    D0 CE    BNE -50   --> &AFB0
AFE2     200        C8       INY
AFE3  9  196 057    C4 39    CPY &39
AFE5     208 244    D0 F4    BNE -12   --> &AFDB
AFE7  <  177 060    B1 3C    LDA (&3C),Y
AFE9     032 132 141 20 84 8D  JSR &8D84
AFEC     176 194    B0 C2    BCS -62   --> &AFB0
AFEE     138        8A       TXA
AFEF     168        A8       TAY
AFF0     032 188 155 20 BC 9B  JSR &9BBC
AFF3  E  032 069 152 20 45 98  JSR &9845
AFF6     162 001    A2 01    LDX#&01
AFF8     032 131 152 20 83 98  JSR &9883
AFFB     165 011    A5 0B    LDA &0B
AFFD     146 002    92 02    STA (&02)
AFFF     160 001    A0 01    LDY#&01
B001     165 012    A5 0C    LDA &0C
```

A000,&9,038 057,26 39,ROL &39
A002,&:,038 058,26 3A,ROL &3A
A004,&;,038 059,26 3B,ROL &3B
A006,&<,038 060,26 3C,ROL &3C
A008,$7,036 055,24 37,BIT &37
A00A,,008,08,PHP
A00B,9,162 057,A2 39,LDX#&39
A00D,,128 165,80 A5,BRA -91 --> &9FB4
A00F,&,032 038 188,20 26 BC,JSR &BC26
A012,6,032 054 173,20 36 AD,JSR &AD36
A015,H,072,48,PHA
A016,,164 027,A4 1B,LDY &1B
A018,,230 027,E6 1B,INC &1B
A01A,,177 025,B1 19,"LDA (&19),Y"
A01C,,201 032,C9 20,CMP#&20
A01E,,240 246,F0 F6,BEQ -10 --> &A016
A020,,170,AA,TAX
A021,h,104,68,PLA
A022,^,224 094,E0 5E,CPX#&5E
A024,,240 001,F0 01,BEQ 1 --> &A027
A026,`,096,60,RTS
A027,,168,A8,TAY
A028,,032 221 150,20 DD 96,JSR &96DD
A02B,,032 250 187,20 FA BB,JSR &BBFA
A02E,,032 218 150,20 DA 96,JSR &96DA
A031,0,165 048,A5 30,LDA &30
A033,,201 135,C9 87,CMP#&87
A035,B,176 066,B0 42,BCS 66 --> &A079
A037,,032 224 130,20 E0 82,JSR &82E0
A03A,,208 013,D0 0D,BNE 13 --> &A049
A03C,,032 232 187,20 E8 BB,JSR &BBE8
A03F,A,032 065 165,20 41 A5,JSR &A541
A042,I,165 073,A5 49,LDA &49
A044,,032 190 165,20 BE A5,JSR &A5BE
A047,",",128 044,80 2C,BRA 44 --> &A075
A049,,032 013 165,20 0D A5,JSR &A50D
A04C,,165 004,A5 04,LDA &04
A04E,J,133 074,85 4A,STA &4A
A050,,165 005,A5 05,LDA &05
A052,K,133 075,85 4B,STA &4B
A054,A,032 065 165,20 41 A5,JSR &A541
A057,I,165 073,A5 49,LDA &49

A059,,032 190 165,20 BE A5,JSR &A5BE
A05C,q,169 113,A9 71,LDA#&71
A05E,,032 019 165,20 13 A5,JSR &A513
A061,,032 232 187,20 E8 BB,JSR &BBE8
A064,A,032 065 165,20 41 A5,JSR &A541
A067,I,032 073 167,20 49 A7,JSR &A749
A06A,,032 159 169,20 9F A9,JSR &A99F
A06D,,032 226 169,20 E2 A9,JSR &A9E2
A070,q,169 113,A9 71,LDA#&71
A072,,032 161 169,20 A1 A9,JSR &A9A1
A075,,169 255,A9 FF,LDA#&FF
A077,,128 156,80 9C,BRA -100 --> &A015
A079,,032 013 165,20 0D A5,JSR &A50D
A07C,,032 216 165,20 D8 A5,JSR &A5D8
A07F,,128 219,80 DB,BRA -37 --> &A05C
A081,,169 000,A9 00,LDA#&00
A083,,128 002,80 02,BRA 2 --> &A087
A085,,169 005,A9 05,LDA#&05
A087,,133 020,85 14,STA &14
A089,,162 004,A2 04,LDX#&04
A08B,t?,116 063,74 3F,"STZ &3F,X"
A08D,8,056,38,SEC
A08E,*,165 042,A5 2A,LDA &2A
A090,&,253 038 128,FD 26 80,"SBC &8026,X"
A093,,168,A8,TAY
A094,+,165 043,A5 2B,LDA &2B
A096,!,253 033 128,FD 21 80,"SBC &8021,X"
A099,,144 008,90 08,BCC 8 --> &A0A3
A09B,+,133 043,85 2B,STA &2B
A09D,*,132 042,84 2A,STY &2A
A09F,?,246 063,F6 3F,"INC &3F,X"
A0A1,,128 235,80 EB,BRA -21 --> &A08E
A0A3,,202,CA,DEX
A0A4,,016 229,10 E5,BPL -27 --> &A08B
A0A6,,162 005,A2 05,LDX#&05
A0A8,,202,CA,DEX
A0A9,,240 004,F0 04,BEQ 4 --> &A0AF
A0AB,?,181 063,B5 3F,"LDA &3F,X"
A0AD,,240 249,F0 F9,BEQ -7 --> &A0A8
A0AF,7,134 055,86 37,STX &37
A0B1,,165 020,A5 14,LDA &14
A0B3,,240 010,F0 0A,BEQ 10 --> &A0BF
A0B5,7,229 055,E5 37,SBC &37

A0B7,,240 006,F0 06,BEQ 6   --> &A0BF
A0B9,,170,AA,TAX
A0BA,,032 191 189,20 BF BD,JSR &BDBF
A0BD,7,166 055,A6 37,LDX &37
A0BF,?,181 063,B5 3F,"LDA &3F,X"
A0C1,0,009 048,09 30,ORA#&30
A0C3,,032 148 189,20 94 BD,JSR &BD94
A0C6,,202,CA,DEX
A0C7,,016 246,10 F6,BPL -10   --> &A0BF
A0C9,`,096,60,RTS
A0CA,,152,98,TYA
A0CB,,016 003,10 03,BPL 3   --> &A0D0
A0CD,,032 195 150,20 C3 96,JSR &96C3
A0D0,,162 000,A2 00,LDX#&00
A0D2,,160 000,A0 00,LDY#&00
A0D4,*,185 042 000,B9 2A 00,"LDA &002A,Y"
A0D7,H,072,48,PHA
A0D8,),041 015,29 0F,AND#&0F
A0DA,?,149 063,95 3F,"STA &3F,X"
A0DC,h,104,68,PLA
A0DD,J,074,4A,LSR A
A0DE,J,074,4A,LSR A
A0DF,J,074,4A,LSR A
A0E0,J,074,4A,LSR A
A0E1,,232,E8,INX
A0E2,?,149 063,95 3F,"STA &3F,X"
A0E4,,232,E8,INX
A0E5,,200,C8,INY
A0E6,,192 004,C0 04,CPY#&04
A0E8,,208 234,D0 EA,BNE -22   --> &A0D4
A0EA,,202,CA,DEX
A0EB,,240 004,F0 04,BEQ 4   --> &A0F1
A0ED,?,181 063,B5 3F,"LDA &3F,X"
A0EF,,240 249,F0 F9,BEQ -7   --> &A0EA
A0F1,?,181 063,B5 3F,"LDA &3F,X"
A0F3,,201 010,C9 0A,CMP#&0A
A0F5,,144 002,90 02,BCC 2   --> &A0F9
A0F7,i,105 006,69 06,ADC#&06
A0F9,i0,105 048,69 30,ADC#&30
A0FB,,032 208 162,20 D0 A2,JSR &A2D0
A0FE,,202,CA,DEX
A0FF,,016 240,10 F0,BPL -16   --> &A0F1
A101,`,096,60,RTS

A102,,016 007,10 07,BPL 7 --> &A10B
A104,-,169 045,A9 2D,LDA#&2D
A106,d.,100 046,64 2E,STZ &2E
A108,,032 208 162,20 D0 A2,JSR &A2D0
A10B,0,165 048,A5 30,LDA &30
A10D,,201 129,C9 81,CMP#&81
A10F,K,176 075,B0 4B,BCS 75 --> &A15C
A111,6,032 054 164,20 36 A4,JSR &A436
A114,H,198 072,C6 48,DEC &48
A116,,128 243,80 F3,BRA -13 --> &A10B
A118,,174 002 004,AE 02 04,LDX &0402
A11B,,224 003,E0 03,CPX#&03
A11D,,144 002,90 02,BCC 2 --> &A121
A11F,,162 000,A2 00,LDX#&00
A121,7,134 055,86 37,STX &37
A123,,173 001 004,AD 01 04,LDA &0401
A126,,240 006,F0 06,BEQ 6 --> &A12E
A128,,201 010,C9 0A,CMP#&0A
A12A,,176 006,B0 06,BCS 6 --> &A132
A12C,,128 006,80 06,BRA 6 --> &A134
A12E,,224 002,E0 02,CPX#&02
A130,,240 002,F0 02,BEQ 2 --> &A134
A132,,169 010,A9 0A,LDA#&0A
A134,8,133 056,85 38,STA &38
A136,M,133 077,85 4D,STA &4D
A138,d6,100 054,64 36,STZ &36
A13A,dH,100 072,64 48,STZ &48
A13C,$,036 021,24 15,BIT &15
A13E,0,048 138,30 8A,BMI -118 --> &A0CA
A140,,152,98,TYA
A141,0,048 003,30 03,BMI 3 --> &A146
A143,,032 133 129,20 85 81,JSR &8185
A146,,032 242 163,20 F2 A3,JSR &A3F2
A149,,208 183,D0 B7,BNE -73 --> &A102
A14B,7,165 055,A5 37,LDA &37
A14D,,208 005,D0 05,BNE 5 --> &A154
A14F,0,169 048,A9 30,LDA#&30
A151,L,076 208 162,4C D0 A2,JMP &A2D0
A154,L,076 208 161,4C D0 A1,JMP &A1D0
A157,,032 216 165,20 D8 A5,JSR &A5D8
A15A,,128 015,80 0F,BRA 15 --> &A16B
A15C,,201 132,C9 84,CMP#&84
A15E,,144 015,90 0F,BCC 15 --> &A16F

A160,,208 006,D0 06,BNE 6   --> &A168
A162,1,165 049,A5 31,LDA &31
A164,,201 160,C9 A0,CMP#&A0
A166,,144 007,90 07,BCC 7   --> &A16F
A168,x,032 120 164,20 78 A4,JSR &A478
A16B,H,230 072,E6 48,INC &48
A16D,,128 156,80 9C,BRA -100   --> &A10B
A16F,5,165 053,A5 35,LDA &35
A171,',133 039,85 27,STA &27
A173,,032 017 165,20 11 A5,JSR &A511
A176,M,165 077,A5 4D,LDA &4D
A178,8,133 056,85 38,STA &38
A17A,7,166 055,A6 37,LDX &37
A17C,,224 002,E0 02,CPX#&02
A17E,,208 016,D0 10,BNE 16   --> &A190
A180,eH,101 072,65 48,ADC &48
A182,0P,048 080,30 50,BMI 80   --> &A1D4
A184,8,133 056,85 38,STA &38
A186,,201 011,C9 0B,CMP#&0B
A188,,144 006,90 06,BCC 6   --> &A190
A18A,,169 010,A9 0A,LDA#&0A
A18C,8,133 056,85 38,STA &38
A18E,d7,100 055,64 37,STZ &37
A190,,032 184 166,20 B8 A6,JSR &A6B8
A193,,169 160,A9 A0,LDA#&A0
A195,1,133 049,85 31,STA &31
A197,,169 131,A9 83,LDA#&83
A199,0,133 048,85 30,STA &30
A19B,8,166 056,A6 38,LDX &38
A19D,,240 006,F0 06,BEQ 6   --> &A1A5
A19F,x,032 120 164,20 78 A4,JSR &A478
A1A2,,202,CA,DEX
A1A3,,208 250,D0 FA,BNE -6   --> &A19F
A1A5,,032 146 165,20 92 A5,JSR &A592
A1A8,,032 224 164,20 E0 A4,JSR &A4E0
A1AB,',165 039,A5 27,LDA &27
A1AD,A,133 065,85 41,STA &41
A1AF,h,032 104 131,20 68 83,JSR &8368
A1B2,0,165 048,A5 30,LDA &30
A1B4,,201 132,C9 84,CMP#&84
A1B6,,176 014,B0 0E,BCS 14   --> &A1C6
A1B8,f1,102 049,66 31,ROR &31
A1BA,f2,102 050,66 32,ROR &32

A1BC,f3,102 051,66 33,ROR &33
A1BE,f4,102 052,66 34,ROR &34
A1C0,f5,102 053,66 35,ROR &35
A1C2,0,230 048,E6 30,INC &30
A1C4,,208 236,D0 EC,BNE -20 --> &A1B2
A1C6,1,165 049,A5 31,LDA &31
A1C8,,201 160,C9 A0,CMP#&A0
A1CA,,176 139,B0 8B,BCS -117 --> &A157
A1CC,8,165 056,A5 38,LDA &38
A1CE,,208 014,D0 0E,BNE 14 --> &A1DE
A1D0,,201 001,C9 01,CMP#&01
A1D2,A,240 065,F0 41,BEQ 65 --> &A215
A1D4,,032 180 166,20 B4 A6,JSR &A6B4
A1D7,dH,100 072,64 48,STZ &48
A1D9,M,165 077,A5 4D,LDA &4D
A1DB,,026,1A,INC A
A1DC,8,133 056,85 38,STA &38
A1DE,,169 001,A9 01,LDA#&01
A1E0,7,197 055,C5 37,CMP &37
A1E2,1,240 049,F0 31,BEQ 49 --> &A215
A1E4,H,164 072,A4 48,LDY &48
A1E6,0,048 010,30 0A,BMI 10 --> &A1F2
A1E8,8,196 056,C4 38,CPY &38
A1EA,),176 041,B0 29,BCS 41 --> &A215
A1EC,dH,100 072,64 48,STZ &48
A1EE,,200,C8,INY
A1EF,,152,98,TYA
A1F0,#,208 035,D0 23,BNE 35 --> &A215
A1F2,7,165 055,A5 37,LDA &37
A1F4,,201 002,C9 02,CMP#&02
A1F6,,240 006,F0 06,BEQ 6 --> &A1FE
A1F8,,169 001,A9 01,LDA#&01
A1FA,,192 255,C0 FF,CPY#&FF
A1FC,,208 023,D0 17,BNE 23 --> &A215
A1FE,0,169 048,A9 30,LDA#&30
A200,,032 208 162,20 D0 A2,JSR &A2D0
A203,.,169 046,A9 2E,LDA#&2E
A205,,032 208 162,20 D0 A2,JSR &A2D0
A208,0,169 048,A9 30,LDA#&30
A20A,H,230 072,E6 48,INC &48
A20C,,240 005,F0 05,BEQ 5 --> &A213
A20E,,032 208 162,20 D0 A2,JSR &A2D0
A211,,128 247,80 F7,BRA -9 --> &A20A

A213,,169 128,A9 80,LDA#&80
A215,M,133 077,85 4D,STA &4D
A217,l,032 108 162,20 6C A2,JSR &A26C
A21A,M,198 077,C6 4D,DEC &4D
A21C,,208 005,D0 05,BNE 5 --> &A223
A21E,.,169 046,A9 2E,LDA#&2E
A220,,032 208 162,20 D0 A2,JSR &A2D0
A223,8,198 056,C6 38,DEC &38
A225,,208 240,D0 F0,BNE -16 --> &A217
A227,7,164 055,A4 37,LDY &37
A229,,136,88,DEY
A22A,,240 024,F0 18,BEQ 24 --> &A244
A22C,,136,88,DEY
A22D,,240 017,F0 11,BEQ 17 --> &A240
A22F,6,164 054,A4 36,LDY &36
A231,,136,88,DEY
A232,,185 000 006,B9 00 06,"LDA &0600,Y"
A235,0,201 048,C9 30,CMP#&30
A237,,240 248,F0 F8,BEQ -8 --> &A231
A239,.,201 046,C9 2E,CMP#&2E
A23B,,240 001,F0 01,BEQ 1 --> &A23E
A23D,,200,C8,INY
A23E,6,132 054,84 36,STY &36
A240,H,165 072,A5 48,LDA &48
A242,',240 039,F0 27,BEQ 39 --> &A26B
A244,E,169 069,A9 45,LDA#&45
A246,,032 208 162,20 D0 A2,JSR &A2D0
A249,H,165 072,A5 48,LDA &48
A24B,,016 010,10 0A,BPL 10 --> &A257
A24D,-,169 045,A9 2D,LDA#&2D
A24F,,032 208 162,20 D0 A2,JSR &A2D0
A252,8,056,38,SEC
A253,,169 000,A9 00,LDA#&00
A255,H,229 072,E5 48,SBC &48
A257,,032 188 162,20 BC A2,JSR &A2BC
A25A,7,165 055,A5 37,LDA &37
A25C,,240 013,F0 0D,BEQ 13 --> &A26B
A25E,,169 032,A9 20,LDA#&20
A260,H,164 072,A4 48,LDY &48
A262,0,048 003,30 03,BMI 3 --> &A267
A264,,032 208 162,20 D0 A2,JSR &A2D0
A267,,224 000,E0 00,CPX#&00
A269,e,240 101,F0 65,BEQ 101 --> &A2D0

A26B,`,096,60,RTS
A26C,1,165 049,A5 31,LDA &31
A26E,J,074,4A,LSR A
A26F,J,074,4A,LSR A
A270,J,074,4A,LSR A
A271,J,074,4A,LSR A
A272,,032 206 162,20 CE A2,JSR &A2CE
A275,,169 240,A9 F0,LDA#&F0
A277,1,020 049,14 31,TRB &31
A279,H,072,48,PHA
A27A,4,166 052,A6 34,LDX &34
A27C,1,165 049,A5 31,LDA &31
A27E,H,072,48,PHA
A27F,2,165 050,A5 32,LDA &32
A281,H,072,48,PHA
A282,3,165 051,A5 33,LDA &33
A284,H,072,48,PHA
A285,5,165 053,A5 35,LDA &35
A287,,010,0A,ASL A
A288,&4,038 052,26 34,ROL &34
A28A,&3,038 051,26 33,ROL &33
A28C,&2,038 050,26 32,ROL &32
A28E,&1,038 049,26 31,ROL &31
A290,,010,0A,ASL A
A291,&4,038 052,26 34,ROL &34
A293,&3,038 051,26 33,ROL &33
A295,&2,038 050,26 32,ROL &32
A297,&1,038 049,26 31,ROL &31
A299,e5,101 053,65 35,ADC &35
A29B,5,133 053,85 35,STA &35
A29D,,138,8A,TXA
A29E,e4,101 052,65 34,ADC &34
A2A0,4,133 052,85 34,STA &34
A2A2,h,104,68,PLA
A2A3,e3,101 051,65 33,ADC &33
A2A5,3,133 051,85 33,STA &33
A2A7,h,104,68,PLA
A2A8,e2,101 050,65 32,ADC &32
A2AA,2,133 050,85 32,STA &32
A2AC,h,104,68,PLA
A2AD,e1,101 049,65 31,ADC &31
A2AF,5,006 053,06 35,ASL &35
A2B1,&4,038 052,26 34,ROL &34

A2B3,&3,038 051,26 33,ROL &33
A2B5,&2,038 050,26 32,ROL &32
A2B7,*,042,2A,ROL A
A2B8,1,133 049,85 31,STA &31
A2BA,h,104,68,PLA
A2BB,`,096,60,RTS
A2BC,,162 255,A2 FF,LDX#&FF
A2BE,8,056,38,SEC
A2BF,,232,E8,INX
A2C0,,233 010,E9 0A,SBC#&0A
A2C2,,176 251,B0 FB,BCS -5 --> &A2BF
A2C4,i,105 010,69 0A,ADC#&0A
A2C6,H,072,48,PHA
A2C7,,138,8A,TXA
A2C8,,240 003,F0 03,BEQ 3 --> &A2CD
A2CA,,032 206 162,20 CE A2,JSR &A2CE
A2CD,h,104,68,PLA
A2CE,0,009 048,09 30,ORA#&30
A2D0,,218,DA,PHX
A2D1,6,166 054,A6 36,LDX &36
A2D3,,157 000 006,9D 00 06,"STA &0600,X"
A2D6,,250,FA,PLX
A2D7,6,230 054,E6 36,INC &36
A2D9,`,096,60,RTS
A2DA,,032 004 164,20 04 A4,JSR &A404
A2DD,,024,18,CLC
A2DE,,169 255,A9 FF,LDA#&FF
A2E0,`,096,60,RTS
A2E1,d1,100 049,64 31,STZ &31
A2E3,d2,100 050,64 32,STZ &32
A2E5,d3,100 051,64 33,STZ &33
A2E7,d4,100 052,64 34,STZ &34
A2E9,d5,100 053,64 35,STZ &35
A2EB,dG,100 071,64 47,STZ &47
A2ED,dH,100 072,64 48,STZ &48
A2EF,.,201 046,C9 2E,CMP#&2E
A2F1,),240 041,F0 29,BEQ 41 --> &A31C
A2F3,:,201 058,C9 3A,CMP#&3A
A2F5,,176 227,B0 E3,BCS -29 --> &A2DA
A2F7,/,233 047,E9 2F,SBC#&2F
A2F9,0,048 223,30 DF,BMI -33 --> &A2DA
A2FB,5,133 053,85 35,STA &35
A2FD,,200,C8,INY

A2FE,,177 025,B1 19,"LDA (&19),Y"
A300,:,201 058,C9 3A,CMP#&3A
A302,,176 032,B0 20,BCS 32 --> &A324
A304,/,233 047,E9 2F,SBC#&2F
A306,,144 014,90 0E,BCC 14 --> &A316
A308,.,133 046,85 2E,STA &2E
A30A,5,165 053,A5 35,LDA &35
A30C,,010,0A,ASL A
A30D,,010,0A,ASL A
A30E,e5,101 053,65 35,ADC &35
A310,,010,0A,ASL A
A311,e.,101 046,65 2E,ADC &2E
A313,5,133 053,85 35,STA &35
A315,,200,C8,INY
A316,,177 025,B1 19,"LDA (&19),Y"
A318,.,201 046,C9 2E,CMP#&2E
A31A,,208 008,D0 08,BNE 8 --> &A324
A31C,G,165 071,A5 47,LDA &47
A31E,D,208 068,D0 44,BNE 68 --> &A364
A320,G,230 071,E6 47,INC &47
A322,,128 241,80 F1,BRA -15 --> &A315
A324,E,201 069,C9 45,CMP#&45
A326,5,240 053,F0 35,BEQ 53 --> &A35D
A328,:,201 058,C9 3A,CMP#&3A
A32A,8,176 056,B0 38,BCS 56 --> &A364
A32C,/,233 047,E9 2F,SBC#&2F
A32E,4,144 052,90 34,BCC 52 --> &A364
A330,1,166 049,A6 31,LDX &31
A332,,224 024,E0 18,CPX#&18
A334,,144 008,90 08,BCC 8 --> &A33E
A336,G,166 071,A6 47,LDX &47
A338,,208 219,D0 DB,BNE -37 --> &A315
A33A,H,230 072,E6 48,INC &48
A33C,,128 215,80 D7,BRA -41 --> &A315
A33E,G,166 071,A6 47,LDX &47
A340,,240 002,F0 02,BEQ 2 --> &A344
A342,H,198 072,C6 48,DEC &48
A344,y,032 121 162,20 79 A2,JSR &A279
A347,e5,101 053,65 35,ADC &35
A349,5,133 053,85 35,STA &35
A34B,,144 200,90 C8,BCC -56 --> &A315
A34D,4,230 052,E6 34,INC &34
A34F,,208 196,D0 C4,BNE -60 --> &A315

A351,3,230 051,E6 33,INC &33
A353,,208 192,D0 C0,BNE -64 --> &A315
A355,2,230 050,E6 32,INC &32
A357,,208 188,D0 BC,BNE -68 --> &A315
A359,1,230 049,E6 31,INC &31
A35B,,128 184,80 B8,BRA -72 --> &A315
A35D,,032 186 163,20 BA A3,JSR &A3BA
A360,eH,101 072,65 48,ADC &48
A362,H,133 072,85 48,STA &48
A364,,132 027,84 1B,STY &1B
A366,H,165 072,A5 48,LDA &48
A368,G,005 071,05 47,ORA &47
A36A,-,240 045,F0 2D,BEQ 45 --> &A399
A36C,,032 242 163,20 F2 A3,JSR &A3F2
A36F,$,240 036,F0 24,BEQ 36 --> &A395
A371,,169 168,A9 A8,LDA#&A8
A373,0,133 048,85 30,STA &30
A375,d/,100 047,64 2F,STZ &2F
A377,d.,100 046,64 2E,STZ &2E
A379,,032 247 129,20 F7 81,JSR &81F7
A37C,H,165 072,A5 48,LDA &48
A37E,0,048 011,30 0B,BMI 11 --> &A38B
A380,,240 016,F0 10,BEQ 16 --> &A392
A382,6,032 054 164,20 36 A4,JSR &A436
A385,H,198 072,C6 48,DEC &48
A387,,208 249,D0 F9,BNE -7 --> &A382
A389,,128 007,80 07,BRA 7 --> &A392
A38B,x,032 120 164,20 78 A4,JSR &A478
A38E,H,230 072,E6 48,INC &48
A390,,208 249,D0 F9,BNE -7 --> &A38B
A392,,032 149 166,20 95 A6,JSR &A695
A395,8,056,38,SEC
A396,,169 255,A9 FF,LDA#&FF
A398,`,096,60,RTS
A399,2,165 050,A5 32,LDA &32
A39B,-,133 045,85 2D,STA &2D
A39D,),041 128,29 80,AND#&80
A39F,1,005 049,05 31,ORA &31
A3A1,,208 206,D0 CE,BNE -50 --> &A371
A3A3,5,165 053,A5 35,LDA &35
A3A5,*,133 042,85 2A,STA &2A
A3A7,4,165 052,A5 34,LDA &34
A3A9,+,133 043,85 2B,STA &2B

A3AB,3,165 051,A5 33,LDA &33
A3AD,",",133 044,85 2C,STA &2C
A3AF,@,169 064,A9 40,LDA#&40
A3B1,8,056,38,SEC
A3B2,`,096,60,RTS
A3B3,,032 197 163,20 C5 A3,JSR &A3C5
A3B6,I,073 255,49 FF,EOR#&FF
A3B8,8,056,38,SEC
A3B9,`,096,60,RTS
A3BA,,200,C8,INY
A3BB,,177 025,B1 19,"LDA (&19),Y"
A3BD,-,201 045,C9 2D,CMP#&2D
A3BF,,240 242,F0 F2,BEQ -14 --> &A3B3
A3C1,+,201 043,C9 2B,CMP#&2B
A3C3,,208 003,D0 03,BNE 3 --> &A3C8
A3C5,,200,C8,INY
A3C6,,177 025,B1 19,"LDA (&19),Y"
A3C8,:,201 058,C9 3A,CMP#&3A
A3CA,"""",176 034,B0 22,BCS 34 --> &A3EE
A3CC,/,233 047,E9 2F,SBC#&2F
A3CE,,144 030,90 1E,BCC 30 --> &A3EE
A3D0,I,133 073,85 49,STA &49
A3D2,,200,C8,INY
A3D3,,177 025,B1 19,"LDA (&19),Y"
A3D5,:,201 058,C9 3A,CMP#&3A
A3D7,,176 017,B0 11,BCS 17 --> &A3EA
A3D9,/,233 047,E9 2F,SBC#&2F
A3DB,,144 013,90 0D,BCC 13 --> &A3EA
A3DD,,200,C8,INY
A3DE,B,133 066,85 42,STA &42
A3E0,I,165 073,A5 49,LDA &49
A3E2,,010,0A,ASL A
A3E3,,010,0A,ASL A
A3E4,eI,101 073,65 49,ADC &49
A3E6,,010,0A,ASL A
A3E7,eB,101 066,65 42,ADC &42
A3E9,`,096,60,RTS
A3EA,I,165 073,A5 49,LDA &49
A3EC,,024,18,CLC
A3ED,`,096,60,RTS
A3EE,,169 000,A9 00,LDA#&00
A3F0,,024,18,CLC
A3F1,`,096,60,RTS

```
A3F2,1,165 049,A5 31,LDA &31
A3F4,2,005 050,05 32,ORA &32
A3F6,3,005 051,05 33,ORA &33
A3F8,4,005 052,05 34,ORA &34
A3FA,5,005 053,05 35,ORA &35
A3FC,,240 006,F0 06,BEQ 6   --> &A404
A3FE,.,165 046,A5 2E,LDA &2E
A400,,208 008,D0 08,BNE 8   --> &A40A
A402,,026,1A,INC A
A403,`,096,60,RTS
A404,d.,100 046,64 2E,STZ &2E
A406,d0,100 048,64 30,STZ &30
A408,d/,100 047,64 2F,STZ &2F
A40A,`,096,60,RTS
A40B,.,165 046,A5 2E,LDA &2E
A40D,;,133 059,85 3B,STA &3B
A40F,0,165 048,A5 30,LDA &30
A411,<,133 060,85 3C,STA &3C
A413,1,165 049,A5 31,LDA &31
A415,=,133 061,85 3D,STA &3D
A417,2,165 050,A5 32,LDA &32
A419,>,133 062,85 3E,STA &3E
A41B,3,165 051,A5 33,LDA &33
A41D,?,133 063,85 3F,STA &3F
A41F,4,165 052,A5 34,LDA &34
A421,@,133 064,85 40,STA &40
A423,5,165 053,A5 35,LDA &35
A425,A,133 065,85 41,STA &41
A427,`,096,60,RTS
A428,,032 011 164,20 0B A4,JSR &A40B
A42B,F=,070 061,46 3D,LSR &3D
A42D,f>,102 062,66 3E,ROR &3E
A42F,f?,102 063,66 3F,ROR &3F
A431,f@,102 064,66 40,ROR &40
A433,fA,102 065,66 41,ROR &41
A435,`,096,60,RTS
A436,,024,18,CLC
A437,0,165 048,A5 30,LDA &30
A439,i,105 003,69 03,ADC#&03
A43B,0,133 048,85 30,STA &30
A43D,,144 002,90 02,BCC 2   --> &A441
A43F,/,230 047,E6 2F,INC &2F
A441,(,032 040 164,20 28 A4,JSR &A428
```

```
A444,+,032 043 164,20 2B A4,JSR &A42B
A447,5,165 053,A5 35,LDA &35
A449,eA,101 065,65 41,ADC &41
A44B,5,133 053,85 35,STA &35
A44D,4,165 052,A5 34,LDA &34
A44F,e@,101 064,65 40,ADC &40
A451,4,133 052,85 34,STA &34
A453,3,165 051,A5 33,LDA &33
A455,e?,101 063,65 3F,ADC &3F
A457,3,133 051,85 33,STA &33
A459,2,165 050,A5 32,LDA &32
A45B,e>,101 062,65 3E,ADC &3E
A45D,2,133 050,85 32,STA &32
A45F,1,165 049,A5 31,LDA &31
A461,e=,101 061,65 3D,ADC &3D
A463,1,133 049,85 31,STA &31
A465,,144 016,90 10,BCC 16   --> &A477
A467,f1,102 049,66 31,ROR &31
A469,f2,102 050,66 32,ROR &32
A46B,f3,102 051,66 33,ROR &33
A46D,f4,102 052,66 34,ROR &34
A46F,f5,102 053,66 35,ROR &35
A471,0,230 048,E6 30,INC &30
A473,,208 002,D0 02,BNE 2   --> &A477
A475,/,230 047,E6 2F,INC &2F
A477,`,096,60,RTS
A478,8,056,38,SEC
A479,0,165 048,A5 30,LDA &30
A47B,,233 004,E9 04,SBC#&04
A47D,0,133 048,85 30,STA &30
A47F,,176 002,B0 02,BCS 2   --> &A483
A481,/,198 047,C6 2F,DEC &2F
A483,(,032 040 164,20 28 A4,JSR &A428
A486,G,032 071 164,20 47 A4,JSR &A447
A489,(,032 040 164,20 28 A4,JSR &A428
A48C,+,032 043 164,20 2B A4,JSR &A42B
A48F,+,032 043 164,20 2B A4,JSR &A42B
A492,+,032 043 164,20 2B A4,JSR &A42B
A495,G,032 071 164,20 47 A4,JSR &A447
A498,d=,100 061,64 3D,STZ &3D
A49A,1,165 049,A5 31,LDA &31
A49C,>,133 062,85 3E,STA &3E
A49E,2,165 050,A5 32,LDA &32
```

```
A4A0,?,133 063,85 3F,STA &3F
A4A2,3,165 051,A5 33,LDA &33
A4A4,@,133 064,85 40,STA &40
A4A6,4,165 052,A5 34,LDA &34
A4A8,A,133 065,85 41,STA &41
A4AA,5,165 053,A5 35,LDA &35
A4AC,*,042,2A,ROL A
A4AD,G,032 071 164,20 47 A4,JSR &A447
A4B0,d>,100 062,64 3E,STZ &3E
A4B2,1,165 049,A5 31,LDA &31
A4B4,?,133 063,85 3F,STA &3F
A4B6,2,165 050,A5 32,LDA &32
A4B8,@,133 064,85 40,STA &40
A4BA,3,165 051,A5 33,LDA &33
A4BC,A,133 065,85 41,STA &41
A4BE,4,165 052,A5 34,LDA &34
A4C0,*,042,2A,ROL A
A4C1,G,032 071 164,20 47 A4,JSR &A447
A4C4,2,165 050,A5 32,LDA &32
A4C6,*,042,2A,ROL A
A4C7,1,165 049,A5 31,LDA &31
A4C9,e5,101 053,65 35,ADC &35
A4CB,5,133 053,85 35,STA &35
A4CD,,144 016,90 10,BCC 16   --> &A4DF
A4CF,4,230 052,E6 34,INC &34
A4D1,,208 012,D0 0C,BNE 12   --> &A4DF
A4D3,3,230 051,E6 33,INC &33
A4D5,,208 008,D0 08,BNE 8   --> &A4DF
A4D7,2,230 050,E6 32,INC &32
A4D9,,208 004,D0 04,BNE 4   --> &A4DF
A4DB,1,230 049,E6 31,INC &31
A4DD,,240 136,F0 88,BEQ -120   --> &A467
A4DF,`,096,60,RTS
A4E0,dA,100 065,64 41,STZ &41
A4E2,,160 004,A0 04,LDY#&04
A4E4,J,177 074,B1 4A,"LDA (&4A),Y"
A4E6,@,133 064,85 40,STA &40
A4E8,,136,88,DEY
A4E9,J,177 074,B1 4A,"LDA (&4A),Y"
A4EB,?,133 063,85 3F,STA &3F
A4ED,,136,88,DEY
A4EE,J,177 074,B1 4A,"LDA (&4A),Y"
A4F0,>,133 062,85 3E,STA &3E
```

```
A4F2,,136,88,DEY
A4F3,J,177 074,B1 4A,"LDA (&4A),Y"
A4F5,;,133 059,85 3B,STA &3B
A4F7,,168,A8,TAY
A4F8,J,178 074,B2 4A,LDA (&4A)
A4FA,<,133 060,85 3C,STA &3C
A4FC,,208 009,D0 09,BNE 9   --> &A507
A4FE,,152,98,TYA
A4FF,>,005 062,05 3E,ORA &3E
A501,?,005 063,05 3F,ORA &3F
A503,@,005 064,05 40,ORA &40
A505,,240 003,F0 03,BEQ 3   --> &A50A
A507,,152,98,TYA
A508,,009 128,09 80,ORA#&80
A50A,=,133 061,85 3D,STA &3D
A50C,`,096,60,RTS
A50D,v,169 118,A9 76,LDA#&76
A50F,,128 002,80 02,BRA 2   --> &A513
A511,l,169 108,A9 6C,LDA#&6C
A513,J,133 074,85 4A,STA &4A
A515,,169 004,A9 04,LDA#&04
A517,K,133 075,85 4B,STA &4B
A519,0,165 048,A5 30,LDA &30
A51B,J,146 074,92 4A,STA (&4A)
A51D,,160 001,A0 01,LDY#&01
A51F,.,165 046,A5 2E,LDA &2E
A521,E1,069 049,45 31,EOR &31
A523,),041 128,29 80,AND#&80
A525,E1,069 049,45 31,EOR &31
A527,J,145 074,91 4A,"STA (&4A),Y"
A529,2,165 050,A5 32,LDA &32
A52B,,200,C8,INY
A52C,J,145 074,91 4A,"STA (&4A),Y"
A52E,3,165 051,A5 33,LDA &33
A530,,200,C8,INY
A531,J,145 074,91 4A,"STA (&4A),Y"
A533,4,165 052,A5 34,LDA &34
A535,,200,C8,INY
A536,J,145 074,91 4A,"STA (&4A),Y"
A538,`,096,60,RTS
A539,l,169 108,A9 6C,LDA#&6C
A53B,J,133 074,85 4A,STA &4A
A53D,,169 004,A9 04,LDA#&04
```

```
A53F,K,133 075,85 4B,STA &4B
A541,d5,100 053,64 35,STZ &35
A543,d/,100 047,64 2F,STZ &2F
A545,,160 004,A0 04,LDY#&04
A547,J,177 074,B1 4A,"LDA (&4A),Y"
A549,4,133 052,85 34,STA &34
A54B,,136,88,DEY
A54C,J,177 074,B1 4A,"LDA (&4A),Y"
A54E,3,133 051,85 33,STA &33
A550,,136,88,DEY
A551,J,177 074,B1 4A,"LDA (&4A),Y"
A553,2,133 050,85 32,STA &32
A555,,136,88,DEY
A556,J,177 074,B1 4A,"LDA (&4A),Y"
A558,.,133 046,85 2E,STA &2E
A55A,,168,A8,TAY
A55B,J,178 074,B2 4A,LDA (&4A)
A55D,0,133 048,85 30,STA &30
A55F,,208 009,D0 09,BNE 9   --> &A56A
A561,,152,98,TYA
A562,2,005 050,05 32,ORA &32
A564,3,005 051,05 33,ORA &33
A566,4,005 052,05 34,ORA &34
A568,,240 003,F0 03,BEQ 3   --> &A56D
A56A,,152,98,TYA
A56B,,009 128,09 80,ORA#&80
A56D,1,133 049,85 31,STA &31
A56F,`,096,60,RTS
A570,d;,100 059,64 3B,STZ &3B
A572,d<,100 060,64 3C,STZ &3C
A574,d=,100 061,64 3D,STZ &3D
A576,d>,100 062,64 3E,STZ &3E
A578,d?,100 063,64 3F,STZ &3F
A57A,d@,100 064,64 40,STZ &40
A57C,dA,100 065,64 41,STZ &41
A57E,`,096,60,RTS
A57F,,024,18,CLC
A580,L,165 076,A5 4C,LDA &4C
A582,i,105 005,69 05,ADC#&05
A584,L,133 076,85 4C,STA &4C
A586,J,133 074,85 4A,STA &4A
A588,`,096,60,RTS
A589,.,169 046,A9 2E,LDA#&2E
```

```
A58B,J,133 074,85 4A,STA &4A
A58D,,169 191,A9 BF,LDA#&BF
A58F,K,133 075,85 4B,STA &4B
A591,`,096,60,RTS
A592,l,169 108,A9 6C,LDA#&6C
A594,J,133 074,85 4A,STA &4A
A596,,169 004,A9 04,LDA#&04
A598,K,133 075,85 4B,STA &4B
A59A,`,096,60,RTS
A59B,:,032 058 169,20 3A A9,JSR &A93A
A59E,{,169 123,A9 7B,LDA#&7B
A5A0,,032 019 165,20 13 A5,JSR &A513
A5A3,,032 023 169,20 17 A9,JSR &A917
A5A6,v,169 118,A9 76,LDA#&76
A5A8,,032 019 165,20 13 A5,JSR &A513
A5AB,{,169 123,A9 7B,LDA#&7B
A5AD,;,032 059 165,20 3B A5,JSR &A53B
A5B0,,032 021 169,20 15 A9,JSR &A915
A5B3,v,169 118,A9 76,LDA#&76
A5B5,,032 148 165,20 94 A5,JSR &A594
A5B8,,032 238 165,20 EE A5,JSR &A5EE
A5BB,,169 255,A9 FF,LDA#&FF
A5BD,`,096,60,RTS
A5BE,,170,AA,TAX
A5BF,,016 008,10 08,BPL 8   --> &A5C9
A5C1,:,058,3A,DEC A
A5C2,I,073 255,49 FF,EOR#&FF
A5C4,H,072,48,PHA
A5C5,,032 233 165,20 E9 A5,JSR &A5E9
A5C8,,250,FA,PLX
A5C9,,240 013,F0 0D,BEQ 13   --> &A5D8
A5CB,,032 017 165,20 11 A5,JSR &A511
A5CE,,202,CA,DEX
A5CF,,240 006,F0 06,BEQ 6   --> &A5D7
A5D1,,032 166 166,20 A6 A6,JSR &A6A6
A5D4,,202,CA,DEX
A5D5,,208 250,D0 FA,BNE -6   --> &A5D1
A5D7,`,096,60,RTS
A5D8,,169 128,A9 80,LDA#&80
A5DA,1,133 049,85 31,STA &31
A5DC,,026,1A,INC A
A5DD,0,133 048,85 30,STA &30
A5DF,L,076 184 166,4C B8 A6,JMP &A6B8
```

A5E2,Lr,076 114 129,4C 72 81,JMP &8172
A5E5,,002,02,xxx Invalid Code
A5E6,,008,08,PHP
A5E7,,008,08,PHP
A5E8,,008,08,PHP
A5E9,,169 146,A9 92,LDA#&92
A5EB,,032 139 165,20 8B A5,JSR &A58B
A5EE,1,165 049,A5 31,LDA &31
A5F0,,240 240,F0 F0,BEQ -16 --> &A5E2
A5F2,,032 224 164,20 E0 A4,JSR &A4E0
A5F5,,208 003,D0 03,BNE 3 --> &A5FA
A5F7,L,076 180 166,4C B4 A6,JMP &A6B4
A5FA,;,165 059,A5 3B,LDA &3B
A5FC,E.,069 046,45 2E,EOR &2E
A5FE,.,133 046,85 2E,STA &2E
A600,8,056,38,SEC
A601,<,165 060,A5 3C,LDA &3C
A603,i,105 129,69 81,ADC#&81
A605,&/,038 047,26 2F,ROL &2F
A607,0,229 048,E5 30,SBC &30
A609,,176 002,B0 02,BCS 2 --> &A60D
A60B,/,198 047,C6 2F,DEC &2F
A60D,0,133 048,85 30,STA &30
A60F,,160 004,A0 04,LDY#&04
A611,<,132 060,84 3C,STY &3C
A613,=,165 061,A5 3D,LDA &3D
A615,,162 008,A2 08,LDX#&08
A617,,128 009,80 09,BRA 9 --> &A622
A619,C,150 067,96 43,"STX &43,Y"
A61B,,190 229 165,BE E5 A5,"LDX &A5E5,Y"
A61E,<,132 060,84 3C,STY &3C
A620,,176 022,B0 16,BCS 22 --> &A638
A622,1,197 049,C5 31,CMP &31
A624,,208 016,D0 10,BNE 16 --> &A636
A626,>,164 062,A4 3E,LDY &3E
A628,2,196 050,C4 32,CPY &32
A62A,,208 010,D0 0A,BNE 10 --> &A636
A62C,?,164 063,A4 3F,LDY &3F
A62E,3,196 051,C4 33,CPY &33
A630,,208 004,D0 04,BNE 4 --> &A636
A632,@,164 064,A4 40,LDY &40
A634,4,196 052,C4 34,CPY &34
A636,,144 023,90 17,BCC 23 --> &A64F

A638,,168,A8,TAY
A639,@,165 064,A5 40,LDA &40
A63B,4,229 052,E5 34,SBC &34
A63D,@,133 064,85 40,STA &40
A63F,?,165 063,A5 3F,LDA &3F
A641,3,229 051,E5 33,SBC &33
A643,?,133 063,85 3F,STA &3F
A645,>,165 062,A5 3E,LDA &3E
A647,2,229 050,E5 32,SBC &32
A649,>,133 062,85 3E,STA &3E
A64B,,152,98,TYA
A64C,1,229 049,E5 31,SBC &31
A64E,8,056,38,SEC
A64F,&;,038 059,26 3B,ROL &3B
A651,@,006 064,06 40,ASL &40
A653,&?,038 063,26 3F,ROL &3F
A655,&>,038 062,26 3E,ROL &3E
A657,*,042,2A,ROL A
A658,,202,CA,DEX
A659,,208 197,D0 C5,BNE -59 --> &A620
A65B,;,166 059,A6 3B,LDX &3B
A65D,<,164 060,A4 3C,LDY &3C
A65F,,136,88,DEY
A660,,016 183,10 B7,BPL -73 --> &A619
A662,>,005 062,05 3E,ORA &3E
A664,?,005 063,05 3F,ORA &3F
A666,@,005 064,05 40,ORA &40
A668,,240 001,F0 01,BEQ 1 --> &A66B
A66A,8,056,38,SEC
A66B,,138,8A,TXA
A66C,j,106,6A,ROR A
A66D,j,106,6A,ROR A
A66E,j,106,6A,ROR A
A66F,),041 224,29 E0,AND#&E0
A671,5,133 053,85 35,STA &35
A673,C,165 067,A5 43,LDA &43
A675,4,133 052,85 34,STA &34
A677,D,165 068,A5 44,LDA &44
A679,3,133 051,85 33,STA &33
A67B,E,165 069,A5 45,LDA &45
A67D,2,133 050,85 32,STA &32
A67F,F,165 070,A5 46,LDA &46
A681,1,133 049,85 31,STA &31

A683,0,048 016,30 10,BMI 16   --> &A695
A685,*,032 042 130,20 2A 82,JSR &822A
A688,,128 011,80 0B,BRA 11   --> &A695
A68A,,032 202 172,20 CA AC,JSR &ACCA
A68D,,032 224 164,20 E0 A4,JSR &A4E0
A690,2,240 050,F0 32,BEQ 50   --> &A6C4
A692,h,032 104 131,20 68 83,JSR &8368
A695,5,165 053,A5 35,LDA &35
A697,,201 128,C9 80,CMP#&80
A699,,144 019,90 13,BCC 19   --> &A6AE
A69B,,240 014,F0 0E,BEQ 14   --> &A6AB
A69D,4,230 052,E6 34,INC &34
A69F,,208 013,D0 0D,BNE 13   --> &A6AE
A6A1,,032 211 164,20 D3 A4,JSR &A4D3
A6A4,,128 008,80 08,BRA 8   --> &A6AE
A6A6,,032 207 166,20 CF A6,JSR &A6CF
A6A9,,128 234,80 EA,BRA -22   --> &A695
A6AB,*,042,2A,ROL A
A6AC,4,004 052,04 34,TSB &34
A6AE,/,165 047,A5 2F,LDA &2F
A6B0,,240 016,F0 10,BEQ 16   --> &A6C2
A6B2,,016 017,10 11,BPL 17   --> &A6C5
A6B4,d0,100 048,64 30,STZ &30
A6B6,d1,100 049,64 31,STZ &31
A6B8,d.,100 046,64 2E,STZ &2E
A6BA,d/,100 047,64 2F,STZ &2F
A6BC,d2,100 050,64 32,STZ &32
A6BE,d3,100 051,64 33,STZ &33
A6C0,d4,100 052,64 34,STZ &34
A6C2,d5,100 053,64 35,STZ &35
A6C4,`,096,60,RTS
A6C5,,000,00,BRK
A6C6,,020,14,EQUB &14
A6C7,T,084,54,xxx Invalid Code
A6C8,o,111,6F,xxx Invalid Code
A6C9,o,111,6F,xxx Invalid Code
A6CA,bi,032 098 105,20 62 69,JSR &6962
A6CD,g,103,67,xxx Invalid Code
A6CE,,000,00,EQUB &00
A6CF,1,165 049,A5 31,LDA &31
A6D1,,240 241,F0 F1,BEQ -15   --> &A6C4
A6D3,,032 224 164,20 E0 A4,JSR &A4E0
A6D6,,240 220,F0 DC,BEQ -36   --> &A6B4

A6D8,,024,18,CLC
A6D9,0,165 048,A5 30,LDA &30
A6DB,e<,101 060,65 3C,ADC &3C
A6DD,&/,038 047,26 2F,ROL &2F
A6DF,,233 127,E9 7F,SBC#&7F
A6E1,0,133 048,85 30,STA &30
A6E3,,176 002,B0 02,BCS 2 --> &A6E7
A6E5,/,198 047,C6 2F,DEC &2F
A6E7,.,165 046,A5 2E,LDA &2E
A6E9,E;,069 059,45 3B,EOR &3B
A6EB,.,133 046,85 2E,STA &2E
A6ED,,218,DA,PHX
A6EE,,162 248,A2 F8,LDX#&F8
A6F0,,160 004,A0 04,LDY#&04
A6F2,9,181 057,B5 39,"LDA &39,X"
A6F4,t9,116 057,74 39,"STZ &39,X"
A6F6,A,153 065 000,99 41 00,"STA &0041,Y"
A6F9,,232,E8,INX
A6FA,,136,88,DEY
A6FB,,208 245,D0 F5,BNE -11 --> &A6F2
A6FD,d<,100 060,64 3C,STZ &3C
A6FF,d;,100 059,64 3B,STZ &3B
A701,d:,100 058,64 3A,STZ &3A
A703,0,128 048,80 30,BRA 48 --> &A735
A705,,218,DA,PHX
A706,F=,070 061,46 3D,LSR &3D
A708,f>,102 062,66 3E,ROR &3E
A70A,f?,102 063,66 3F,ROR &3F
A70C,f@,102 064,66 40,ROR &40
A70E,fA,102 065,66 41,ROR &41
A710,F,022 070,16 46,"ASL &46,X"
A712,,144 029,90 1D,BCC 29 --> &A731
A714,,024,18,CLC
A715,,152,98,TYA
A716,uB,117 066,75 42,"ADC &42,X"
A718,,168,A8,TAY
A719,4,165 052,A5 34,LDA &34
A71B,uA,117 065,75 41,"ADC &41,X"
A71D,4,133 052,85 34,STA &34
A71F,3,165 051,A5 33,LDA &33
A721,u@,117 064,75 40,"ADC &40,X"
A723,3,133 051,85 33,STA &33
A725,2,165 050,A5 32,LDA &32

```
A727,u?,117 063,75 3F,"ADC &3F,X"
A729,2,133 050,85 32,STA &32
A72B,1,165 049,A5 31,LDA &31
A72D,u>,117 062,75 3E,"ADC &3E,X"
A72F,1,133 049,85 31,STA &31
A731,,232,E8,INX
A732,0,048 220,30 DC,BMI -36   --> &A710
A734,,250,FA,PLX
A735,F,181 070,B5 46,"LDA &46,X"
A737,,208 204,D0 CC,BNE -52   --> &A705
A739,,232,E8,INX
A73A,0,048 249,30 F9,BMI -7   --> &A735
A73C,,250,FA,PLX
A73D,5,132 053,84 35,STY &35
A73F,1,165 049,A5 31,LDA &31
A741,0,048 129,30 81,BMI -127   --> &A6C4
A743,L,076 251 129,4C FB 81,JMP &81FB
A746,,032 218 150,20 DA 96,JSR &96DA
A749,,032 242 163,20 F2 A3,JSR &A3F2
A74C,,240 002,F0 02,BEQ 2   --> &A750
A74E,,016 022,10 16,BPL 22   --> &A766
A750,,000,00,BRK
A751,,022,16,EQUB &16
A752,L,076,4C,xxx Invalid Code
A753,o,111,6F,xxx Invalid Code
A754,g,103,67,xxx Invalid Code
A755,ra,032 114 097,20 72 61,JSR &6172
A758,nge,110 103 101,6E 67 65,ROR &6567
A75B,,000,00,BRK
A75C,,021,15,EQUB &15
A75D,-,045,2D,xxx Invalid Code
A75E,ve,118 101,76 65,"ROR &65,X"
A760,ro,032 114 111,20 72 6F,JSR &6F72
A763,o,111,6F,xxx Invalid Code
A764,t,116,74,xxx Invalid Code
A765,,000,00,EQUB &00
A766,v,032 118 165,20 76 A5,JSR &A576
A769,,160 128,A0 80,LDY#&80
A76B,;,132 059,84 3B,STY &3B
A76D,=,132 061,84 3D,STY &3D
A76F,,200,C8,INY
A770,<,132 060,84 3C,STY &3C
A772,0,166 048,A6 30,LDX &30
```

A774,,240 006,F0 06,BEQ 6   --> &A77C
A776,1,165 049,A5 31,LDA &31
A778,,201 181,C9 B5,CMP#&B5
A77A,,144 002,90 02,BCC 2   --> &A77E
A77C,,232,E8,INX
A77D,,136,88,DEY
A77E,,218,DA,PHX
A77F,0,132 048,84 30,STY &30
A781,,032 146 166,20 92 A6,JSR &A692
A784,{,169 123,A9 7B,LDA#&7B
A786,,032 019 165,20 13 A5,JSR &A513
A789,Q,162 081,A2 51,LDX#&51
A78B,o,169 111,A9 6F,LDA#&6F
A78D,,160 002,A0 02,LDY#&02
A78F,a,032 097 168,20 61 A8,JSR &A861
A792,{,169 123,A9 7B,LDA#&7B
A794,,032 161 169,20 A1 A9,JSR &A9A1
A797,,032 166 166,20 A6 A6,JSR &A6A6
A79A,,032 141 166,20 8D A6,JSR &A68D
A79D,,032 017 165,20 11 A5,JSR &A511
A7A0,h,104,68,PLA
A7A1,8,056,38,SEC
A7A2,,233 129,E9 81,SBC#&81
A7A4,,032 213 129,20 D5 81,JSR &81D5
A7A7,L,169 076,A9 4C,LDA#&4C
A7A9,,032 212 169,20 D4 A9,JSR &A9D4
A7AC,,032 146 165,20 92 A5,JSR &A592
A7AF,,032 141 166,20 8D A6,JSR &A68D
A7B2,,169 255,A9 FF,LDA#&FF
A7B4,`,096,60,RTS
A7B5,,032 218 150,20 DA 96,JSR &96DA
A7B8,,032 242 163,20 F2 A3,JSR &A3F2
A7BB,,240 245,F0 F5,BEQ -11   --> &A7B2
A7BD,0,048 156,30 9C,BMI -100   --> &A75B
A7BF,0,165 048,A5 30,LDA &30
A7C1,J,074,4A,LSR A
A7C2,,008,08,PHP
A7C3,iA,105 065,69 41,ADC#&41
A7C5,0,133 048,85 30,STA &30
A7C7,(,040,28,PLP
A7C8,,144 010,90 0A,BCC 10   --> &A7D4
A7CA,F1,070 049,46 31,LSR &31
A7CC,f2,102 050,66 32,ROR &32

A7CE,f3,102 051,66 33,ROR &33
A7D0,f4,102 052,66 34,ROR &34
A7D2,f5,102 053,66 35,ROR &35
A7D4,p,032 112 165,20 70 A5,JSR &A570
A7D7,dC,100 067,64 43,STZ &43
A7D9,dD,100 068,64 44,STZ &44
A7DB,dE,100 069,64 45,STZ &45
A7DD,dF,100 070,64 46,STZ &46
A7DF,@,169 064,A9 40,LDA#&40
A7E1,=,133 061,85 3D,STA &3D
A7E3,B,133 066,85 42,STA &42
A7E5,,162 251,A2 FB,LDX#&FB
A7E7,,160 016,A0 10,LDY#&10
A7E9,8,056,38,SEC
A7EA,1,165 049,A5 31,LDA &31
A7EC,@,233 064,E9 40,SBC#&40
A7EE,1,133 049,85 31,STA &31
A7F0,,152,98,TYA
A7F1,UB,085 066,55 42,"EOR &42,X"
A7F3,G,149 071,95 47,"STA &47,X"
A7F5,1,165 049,A5 31,LDA &31
A7F7,B,197 066,C5 42,CMP &42
A7F9,,208 013,D0 0D,BNE 13   --> &A808
A7FB,,218,DA,PHX
A7FC,,162 252,A2 FC,LDX#&FC
A7FE,6,181 054,B5 36,"LDA &36,X"
A800,G,213 071,D5 47,"CMP &47,X"
A802,,208 003,D0 03,BNE 3   --> &A807
A804,,232,E8,INX
A805,,208 247,D0 F7,BNE -9   --> &A7FE
A807,,250,FA,PLX
A808,),144 041,90 29,BCC 41   --> &A833
A80A,5,165 053,A5 35,LDA &35
A80C,F,229 070,E5 46,SBC &46
A80E,5,133 053,85 35,STA &35
A810,4,165 052,A5 34,LDA &34
A812,E,229 069,E5 45,SBC &45
A814,4,133 052,85 34,STA &34
A816,3,165 051,A5 33,LDA &33
A818,D,229 068,E5 44,SBC &44
A81A,3,133 051,85 33,STA &33
A81C,2,165 050,A5 32,LDA &32
A81E,C,229 067,E5 43,SBC &43

```
A820,2,133 050,85 32,STA &32
A822,1,165 049,A5 31,LDA &31
A824,B,229 066,E5 42,SBC &42
A826,1,133 049,85 31,STA &31
A828,,152,98,TYA
A829,,010,0A,ASL A
A82A,,144 011,90 0B,BCC 11   --> &A837
A82C,,026,1A,INC A
A82D,UA,085 065,55 41,"EOR &41,X"
A82F,A,149 065,95 41,"STA &41,X"
A831,F,149 070,95 46,"STA &46,X"
A833,B,181 066,B5 42,"LDA &42,X"
A835,,128 004,80 04,BRA 4   --> &A83B
A837,UB,085 066,55 42,"EOR &42,X"
A839,B,149 066,95 42,"STA &42,X"
A83B,G,149 071,95 47,"STA &47,X"
A83D,5,006 053,06 35,ASL &35
A83F,&4,038 052,26 34,ROL &34
A841,&3,038 051,26 33,ROL &33
A843,&2,038 050,26 32,ROL &32
A845,&1,038 049,26 31,ROL &31
A847,,152,98,TYA
A848,J,074,4A,LSR A
A849,,168,A8,TAY
A84A,,144 164,90 A4,BCC -92   --> &A7F0
A84C,,160 128,A0 80,LDY#&80
A84E,,232,E8,INX
A84F,,208 159,D0 9F,BNE -97   --> &A7F0
A851,S,032 083 131,20 53 83,JSR &8353
A854,1,165 049,A5 31,LDA &31
A856,0,048 003,30 03,BMI 3   --> &A85B
A858,,032 251 129,20 FB 81,JSR &81FB
A85B,,032 149 166,20 95 A6,JSR &A695
A85E,,169 255,A9 FF,LDA#&FF
A860,`,096,60,RTS
A861,G,132 071,84 47,STY &47
A863,L,134 076,86 4C,STX &4C
A865,0,166 048,A6 30,LDX &30
A867,@,224 064,E0 40,CPX#&40
A869,+,144 043,90 2B,BCC 43   --> &A896
A86B,,032 233 165,20 E9 A5,JSR &A5E9
A86E,,032 017 165,20 11 A5,JSR &A511
A871,L,165 076,A5 4C,LDA &4C
```

A873,,032 139 165,20 8B A5,JSR &A58B
A876,,032 141 166,20 8D A6,JSR &A68D
A879,,032 134 168,20 86 A8,JSR &A886
A87C,,032 146 165,20 92 A5,JSR &A592
A87F,,032 141 166,20 8D A6,JSR &A68D
A882,G,198 071,C6 47,DEC &47
A884,,208 243,D0 F3,BNE -13 --> &A879
A886,,169 191,A9 BF,LDA#&BF
A888,K,133 075,85 4B,STA &4B
A88A,,032 127 165,20 7F A5,JSR &A57F
A88D,,032 238 165,20 EE A5,JSR &A5EE
A890,,032 127 165,20 7F A5,JSR &A57F
A893,L,076 141 166,4C 8D A6,JMP &A68D
A896,,032 139 165,20 8B A5,JSR &A58B
A899,LA,076 065 165,4C 41 A5,JMP &A541
A89C,,032 161 168,20 A1 A8,JSR &A8A1
A89F,@,128 064,80 40,BRA 64 --> &A8E1
A8A1,,032 218 150,20 DA 96,JSR &96DA
A8A4,.,165 046,A5 2E,LDA &2E
A8A6,,016 007,10 07,BPL 7 --> &A8AF
A8A8,d.,100 046,64 2E,STZ &2E
A8AA,,032 175 168,20 AF A8,JSR &A8AF
A8AD,#,128 035,80 23,BRA 35 --> &A8D2
A8AF,,032 013 165,20 0D A5,JSR &A50D
A8B2,),032 041 169,20 29 A9,JSR &A929
A8B5,1,165 049,A5 31,LDA &31
A8B7,,240 005,F0 05,BEQ 5 --> &A8BE
A8B9,,032 179 165,20 B3 A5,JSR &A5B3
A8BC,,128 008,80 08,BRA 8 --> &A8C6
A8BE,.,169 046,A9 2E,LDA#&2E
A8C0,L,076 150 168,4C 96 A8,JMP &A896
A8C3,,032 218 150,20 DA 96,JSR &96DA
A8C6,,032 242 163,20 F2 A3,JSR &A3F2
A8C9,[,240 091,F0 5B,BEQ 91 --> &A926
A8CB,,016 008,10 08,BPL 8 --> &A8D5
A8CD,d.,100 046,64 2E,STZ &2E
A8CF,,032 213 168,20 D5 A8,JSR &A8D5
A8D2,.,133 046,85 2E,STA &2E
A8D4,`,096,60,RTS
A8D5,0,165 048,A5 30,LDA &30
A8D7,,201 129,C9 81,CMP#&81
A8D9,,144 015,90 0F,BCC 15 --> &A8EA
A8DB,,032 233 165,20 E9 A5,JSR &A5E9

A8DE,,032 234 168,20 EA A8,JSR &A8EA
A8E1,,032 137 165,20 89 A5,JSR &A589
A8E4,,032 138 166,20 8A A6,JSR &A68A
A8E7,,169 255,A9 FF,LDA#&FF
A8E9,`,096,60,RTS
A8EA,0,165 048,A5 30,LDA &30
A8EC,s,201 115,C9 73,CMP#&73
A8EE,6,144 054,90 36,BCC 54 --> &A926
A8F0,,032 013 165,20 0D A5,JSR &A50D
A8F3,v,032 118 165,20 76 A5,JSR &A576
A8F6,,169 128,A9 80,LDA#&80
A8F8,<,133 060,85 3C,STA &3C
A8FA,=,133 061,85 3D,STA &3D
A8FC,;,133 059,85 3B,STA &3B
A8FE,,032 146 166,20 92 A6,JSR &A692
A901,,162 151,A2 97,LDX#&97
A903,,169 201,A9 C9,LDA#&C9
A905,,160 004,A0 04,LDY#&04
A907,a,032 097 168,20 61 A8,JSR &A861
A90A,L,076 159 169,4C 9F A9,JMP &A99F
A90D,,024,18,CLC
A90E,,008,08,PHP
A90F,:,032 058 169,20 3A A9,JSR &A93A
A912,(,040,28,PLP
A913,,144 002,90 02,BCC 2 --> &A917
A915,I,230 073,E6 49,INC &49
A917,I,165 073,A5 49,LDA &49
A919,,137 002,89 02,BIT#&02
A91B,,240 006,F0 06,BEQ 6 --> &A923
A91D,#,032 035 169,20 23 A9,JSR &A923
A920,L,076 202 172,4C CA AC,JMP &ACCA
A923,J,074,4A,LSR A
A924,,176 003,B0 03,BCS 3 --> &A929
A926,,169 255,A9 FF,LDA#&FF
A928,`,096,60,RTS
A929,,032 017 165,20 11 A5,JSR &A511
A92C,,032 166 166,20 A6 A6,JSR &A6A6
A92F,,169 146,A9 92,LDA#&92
A931,,032 139 165,20 8B A5,JSR &A58B
A934,,032 138 166,20 8A A6,JSR &A68A
A937,L,076 184 167,4C B8 A7,JMP &A7B8
A93A,,032 218 150,20 DA 96,JSR &96DA
A93D,0,165 048,A5 30,LDA &30

A93F,,201 152,C9 98,CMP#&98
A941,j,176 106,B0 6A,BCS 106 --> &A9AD
A943,,032 017 165,20 11 A5,JSR &A511
A946,,032 137 165,20 89 A5,JSR &A589
A949,,032 224 164,20 E0 A4,JSR &A4E0
A94C,.,165 046,A5 2E,LDA &2E
A94E,;,133 059,85 3B,STA &3B
A950,<,198 060,C6 3C,DEC &3C
A952,,032 146 166,20 92 A6,JSR &A692
A955,3,169 051,A9 33,LDA#&33
A957,,032 212 169,20 D4 A9,JSR &A9D4
A95A,,032 195 150,20 C3 96,JSR &96C3
A95D,I,133 073,85 49,STA &49
A95F,+,005 043,05 2B,ORA &2B
A961,",",005 044,05 2C,ORA &2C
A963,(,240 040,F0 28,BEQ 40 --> &A98D
A965,,032 137 129,20 89 81,JSR &8189
A968,q,169 113,A9 71,LDA#&71
A96A,,032 019 165,20 13 A5,JSR &A513
A96D,$,169 036,A9 24,LDA#&24
A96F,,032 212 169,20 D4 A9,JSR &A9D4
A972,,032 146 165,20 92 A5,JSR &A592
A975,,032 141 166,20 8D A6,JSR &A68D
A978,,032 025 165,20 19 A5,JSR &A519
A97B,q,169 113,A9 71,LDA#&71
A97D,;,032 059 165,20 3B A5,JSR &A53B
A980,),169 041,A9 29,LDA#&29
A982,,032 212 169,20 D4 A9,JSR &A9D4
A985,,032 146 165,20 92 A5,JSR &A592
A988,,032 141 166,20 8D A6,JSR &A68D
A98B,,128 003,80 03,BRA 3 --> &A990
A98D,9,032 057 165,20 39 A5,JSR &A539
A990,,032 013 165,20 0D A5,JSR &A50D
A993,,032 166 166,20 A6 A6,JSR &A6A6
A996,t,162 116,A2 74,LDX#&74
A998,,169 146,A9 92,LDA#&92
A99A,,160 002,A0 02,LDY#&02
A99C,a,032 097 168,20 61 A8,JSR &A861
A99F,v,169 118,A9 76,LDA#&76
A9A1,,160 004,A0 04,LDY#&04
A9A3,K,132 075,84 4B,STY &4B
A9A5,J,133 074,85 4A,STA &4A
A9A7,,032 166 166,20 A6 A6,JSR &A6A6

A9AA,,169 255,A9 FF,LDA#&FF
A9AC,`,096,60,RTS
A9AD,,000,00,BRK
A9AE,,023,17,EQUB &17
A9AF,Ac,065 099,41 63,"EOR (&63,X)"
A9B1,c,099,63,xxx Invalid Code
A9B2,ur,117 114,75 72,"ADC &72,X"
A9B4,acy,097 099 121,61 63 79,"ADC (&7963,X)"
A9B7,lo,032 108 111,20 6C 6F,JSR &6F6C
A9BA,s,115,73,xxx Invalid Code
A9BB,t,116,74,xxx Invalid Code
A9BC,,000,00,BRK
A9BD,,024,18,EQUB &18
A9BE,Ex,069 120,45 78,EOR &78
A9C0,p,112 032,70 20,BVS 32 --> &A9E2
A9C2,ra,114 097,72 61,ADC (&61)
A9C4,nge,110 103 101,6E 67 65,ROR &6567
A9C7,,000,00,EQUB &00
A9C8,,032 218 150,20 DA 96,JSR &96DA
A9CB,8,169 056,A9 38,LDA#&38
A9CD,,128 005,80 05,BRA 5 --> &A9D4
A9CF,F,032 070 167,20 46 A7,JSR &A746
A9D2,B,169 066,A9 42,LDA#&42
A9D4,,160 191,A0 BF,LDY#&BF
A9D6,,128 203,80 CB,BRA -53 --> &A9A3
A9D8,,032 218 150,20 DA 96,JSR &96DA
A9DB,=,169 061,A9 3D,LDA#&3D
A9DD,,128 245,80 F5,BRA -11 --> &A9D4
A9DF,,032 218 150,20 DA 96,JSR &96DA
A9E2,0,165 048,A5 30,LDA &30
A9E4,,201 135,C9 87,CMP#&87
A9E6,,144 015,90 0F,BCC 15 --> &A9F7
A9E8,,208 006,D0 06,BNE 6 --> &A9F0
A9EA,1,164 049,A4 31,LDY &31
A9EC,,192 179,C0 B3,CPY#&B3
A9EE,,144 007,90 07,BCC 7 --> &A9F7
A9F0,.,165 046,A5 2E,LDA &2E
A9F2,,016 200,10 C8,BPL -56 --> &A9BC
A9F4,L,076 180 166,4C B4 A6,JMP &A6B4
A9F7,,032 224 130,20 E0 82,JSR &82E0
A9FA,,162 206,A2 CE,LDX#&CE
A9FC,,169 246,A9 F6,LDA#&F6
A9FE,,160 003,A0 03,LDY#&03

```
AA00,a,032 097 168,20 61 A8,JSR &A861
AA03,,032 013 165,20 0D A5,JSR &A50D
AA06,G,169 071,A9 47,LDA#&47
AA08,,032 150 168,20 96 A8,JSR &A896
AA0B,I,165 073,A5 49,LDA &49
AA0D,,032 190 165,20 BE A5,JSR &A5BE
AA10,,128 141,80 8D,BRA -115   --> &A99F
AA12,,032 180 150,20 B4 96,JSR &96B4
AA15,,169 129,A9 81,LDA#&81
AA17,*,166 042,A6 2A,LDX &2A
AA19,+,164 043,A4 2B,LDY &2B
AA1B,L,076 244 255,4C F4 FF,JMP &FFF4
AA1E,,032 030 131,20 1E 83,JSR &831E
AA21,d.,100 046,64 2E,STZ &2E
AA23,d/,100 047,64 2F,STZ &2F
AA25,d5,100 053,64 35,STZ &35
AA27,,169 128,A9 80,LDA#&80
AA29,0,133 048,85 30,STA &30
AA2B,,160 000,A0 00,LDY#&00
AA2D,,162 003,A2 03,LDX#&03
AA2F,Y,089 013 000,59 0D 00,"EOR &000D,Y"
AA32,1,149 049,95 31,"STA &31,X"
AA34,,200,C8,INY
AA35,,202,CA,DEX
AA36,,016 247,10 F7,BPL -9   --> &AA2F
AA38,LT,076 084 168,4C 54 A8,JMP &A854
AA3B,,230 027,E6 1B,INC &1B
AA3D,,032 167 150,20 A7 96,JSR &96A7
AA40,-,165 045,A5 2D,LDA &2D
AA42,0%,048 037,30 25,BMI 37   --> &AA69
AA44,",",005 044,05 2C,ORA &2C
AA46,+,005 043,05 2B,ORA &2B
AA48,,208 008,D0 08,BNE 8   --> &AA52
AA4A,*,165 042,A5 2A,LDA &2A
AA4C,,240 211,F0 D3,BEQ -45   --> &AA21
AA4E,,201 001,C9 01,CMP#&01
AA50,,240 204,F0 CC,BEQ -52   --> &AA1E
AA52,,032 133 129,20 85 81,JSR &8185
AA55,,032 250 187,20 FA BB,JSR &BBFA
AA58,,032 030 170,20 1E AA,JSR &AA1E
AA5B,,032 232 187,20 E8 BB,JSR &BBE8
AA5E,,032 207 166,20 CF A6,JSR &A6CF
AA61,,032 195 150,20 C3 96,JSR &96C3
```

```
AA64,,032 239 190,20 EF BE,JSR &BEEF
AA67,',128 039,80 27,BRA 39   --> &AA90
AA69,,162 013,A2 0D,LDX#&0D
AA6B,,032 198 189,20 C6 BD,JSR &BDC6
AA6E,@,169 064,A9 40,LDA#&40
AA70,,133 017,85 11,STA &11
AA72,`,096,60,RTS
AA73,,164 027,A4 1B,LDY &1B
AA75,,177 025,B1 19,"LDA (&19),Y"
AA77,(,201 040,C9 28,CMP#&28
AA79,,240 192,F0 C0,BEQ -64   --> &AA3B
AA7B,,032 030 131,20 1E 83,JSR &831E
AA7E,,162 013,A2 0D,LDX#&0D
AA80,,181 000,B5 00,"LDA &00,X"
AA82,*,133 042,85 2A,STA &2A
AA84,,181 001,B5 01,"LDA &01,X"
AA86,+,133 043,85 2B,STA &2B
AA88,,181 002,B5 02,"LDA &02,X"
AA8A,",",133 044,85 2C,STA &2C
AA8C,,181 003,B5 03,"LDA &03,X"
AA8E,-,133 045,85 2D,STA &2D
AA90,@,169 064,A9 40,LDA#&40
AA92,`,096,60,RTS
AA93,,032 180 150,20 B4 96,JSR &96B4
AA96,,162 003,A2 03,LDX#&03
AA98,*,181 042,B5 2A,"LDA &2A,X"
AA9A,I,073 255,49 FF,EOR#&FF
AA9C,*,149 042,95 2A,"STA &2A,X"
AA9E,,202,CA,DEX
AA9F,,016 247,10 F7,BPL -9   --> &AA98
AAA1,,128 237,80 ED,BRA -19   --> &AA90
AAA3,,032 188 170,20 BC AA,JSR &AABC
AAA6,*,134 042,86 2A,STX &2A
AAA8,`,096,60,RTS
AAA9,,032 180 150,20 B4 96,JSR &96B4
AAAC,,032 004 147,20 04 93,JSR &9304
AAAF,*,133 042,85 2A,STA &2A
AAB1,+,134 043,86 2B,STX &2B
AAB3,",",132 044,84 2C,STY &2C
AAB5,,008,08,PHP
AAB6,h,104,68,PLA
AAB7,-,133 045,85 2D,STA &2D
AAB9,,216,D8,CLD
```

AABA,,128 212,80 D4,BRA -44 --> &AA90
AABC,,169 134,A9 86,LDA#&86
AABE,,032 244 255,20 F4 FF,JSR &FFF4
AAC1,,152,98,TYA
AAC2,L,076 024 174,4C 18 AE,JMP &AE18
AAC5,,169 002,A9 02,LDA#&02
AAC7,,128 002,80 02,BRA 2 --> &AACB
AAC9,,169 000,A9 00,LDA#&00
AACB,H,072,48,PHA
AACC,J,032 074 186,20 4A BA,JSR &BA4A
AACF,*,162 042,A2 2A,LDX#&2A
AAD1,h,104,68,PLA
AAD2,,032 218 255,20 DA FF,JSR &FFDA
AAD5,,128 185,80 B9,BRA -71 --> &AA90
AAD7,J,032 074 186,20 4A BA,JSR &BA4A
AADA,,032 215 255,20 D7 FF,JSR &FFD7
AADD,,128 227,80 E3,BRA -29 --> &AAC2
AADF,@,169 064,A9 40,LDA#&40
AAE1,,128 006,80 06,BRA 6 --> &AAE9
AAE3,,169 128,A9 80,LDA#&80
AAE5,,128 002,80 02,BRA 2 --> &AAE9
AAE7,,169 192,A9 C0,LDA#&C0
AAE9,H,072,48,PHA
AAEA,6,032 054 173,20 36 AD,JSR &AD36
AAED,,208 013,D0 0D,BNE 13 --> &AAFC
AAEF,+,032 043 190,20 2B BE,JSR &BE2B
AAF2,,162 000,A2 00,LDX#&00
AAF4,,160 006,A0 06,LDY#&06
AAF6,h,104,68,PLA
AAF7,,032 206 255,20 CE FF,JSR &FFCE
AAFA,,128 198,80 C6,BRA -58 --> &AAC2
AAFC,L,076 146 144,4C 92 90,JMP &9092
AAFF,,032 190 168,20 BE A8,JSR &A8BE
AB02,0,230 048,E6 30,INC &30
AB04,`,096,60,RTS
AB05,6,032 054 173,20 36 AD,JSR &AD36
AB08,<,208 060,D0 3C,BNE 60 --> &AB46
AB0A,6,230 054,E6 36,INC &36
AB0C,6,164 054,A4 36,LDY &36
AB0E,,169 013,A9 0D,LDA#&0D
AB10,,153 255 005,99 FF 05,"STA &05FF,Y"
AB13,Q,032 081 188,20 51 BC,JSR &BC51
AB16,,165 025,A5 19,LDA &19

AB18,H,072,48,PHA
AB19,,165 026,A5 1A,LDA &1A
AB1B,H,072,48,PHA
AB1C,,165 027,A5 1B,LDA &1B
AB1E,H,072,48,PHA
AB1F,,164 004,A4 04,LDY &04
AB21,,166 005,A6 05,LDX &05
AB23,,200,C8,INY
AB24,,132 025,84 19,STY &19
AB26,7,132 055,84 37,STY &37
AB28,,208 001,D0 01,BNE 1 --> &AB2B
AB2A,,232,E8,INX
AB2B,,134 026,86 1A,STX &1A
AB2D,8,134 056,86 38,STX &38
AB2F,,032 031 142,20 1F 8E,JSR &8E1F
AB32,d,100 027,64 1B,STZ &1B
AB34,;,032 059 157,20 3B 9D,JSR &9D3B
AB37,,032 225 188,20 E1 BC,JSR &BCE1
AB3A,h,104,68,PLA
AB3B,,133 027,85 1B,STA &1B
AB3D,h,104,68,PLA
AB3E,,133 026,85 1A,STA &1A
AB40,h,104,68,PLA
AB41,,133 025,85 19,STA &19
AB43,',165 039,A5 27,LDA &27
AB45,`,096,60,RTS
AB46,L,076 146 144,4C 92 90,JMP &9092
AB49,6,032 054 173,20 36 AD,JSR &AD36
AB4C,,208 248,D0 F8,BNE -8 --> &AB46
AB4E,6,166 054,A6 36,LDX &36
AB50,,158 000 006,9E 00 06,"STZ &0600,X"
AB53,,165 025,A5 19,LDA &19
AB55,H,072,48,PHA
AB56,,165 026,A5 1A,LDA &1A
AB58,H,072,48,PHA
AB59,,165 027,A5 1B,LDA &1B
AB5B,H,072,48,PHA
AB5C,d,100 027,64 1B,STZ &1B
AB5E,d,100 025,64 19,STZ &19
AB60,,169 006,A9 06,LDA#&06
AB62,,133 026,85 1A,STA &1A
AB64,,032 213 142,20 D5 8E,JSR &8ED5
AB67,-,201 045,C9 2D,CMP#&2D

AB69,,240 014,F0 0E,BEQ 14   --> &AB79
AB6B,+,201 043,C9 2B,CMP#&2B
AB6D,,208 003,D0 03,BNE 3   --> &AB72
AB6F,,032 213 142,20 D5 8E,JSR &8ED5
AB72,,198 027,C6 1B,DEC &1B
AB74,,032 225 162,20 E1 A2,JSR &A2E1
AB77,,128 013,80 0D,BRA 13   --> &AB86
AB79,,032 213 142,20 D5 8E,JSR &8ED5
AB7C,,198 027,C6 1B,DEC &1B
AB7E,,032 225 162,20 E1 A2,JSR &A2E1
AB81,,144 003,90 03,BCC 3   --> &AB86
AB83,,032 218 172,20 DA AC,JSR &ACDA
AB86,',133 039,85 27,STA &27
AB88,,128 176,80 B0,BRA -80   --> &AB3A
AB8A,6,032 054 173,20 36 AD,JSR &AD36
AB8D,=,240 061,F0 3D,BEQ 61   --> &ABCC
AB8F,!,016 033,10 21,BPL 33   --> &ABB2
AB91,.,165 046,A5 2E,LDA &2E
AB93,,008,08,PHP
AB94,u,032 117 130,20 75 82,JSR &8275
AB97,(,040,28,PLP
AB98,,016 019,10 13,BPL 19   --> &ABAD
AB9A,=,165 061,A5 3D,LDA &3D
AB9C,>,005 062,05 3E,ORA &3E
AB9E,?,005 063,05 3F,ORA &3F
ABA0,@,005 064,05 40,ORA &40
ABA2,,240 009,F0 09,BEQ 9   --> &ABAD
ABA4,,032 200 130,20 C8 82,JSR &82C8
ABA7,,032 013 131,20 0D 83,JSR &830D
ABAA,,032 200 130,20 C8 82,JSR &82C8
ABAD,,032 198 150,20 C6 96,JSR &96C6
ABB0,@,169 064,A9 40,LDA#&40
ABB2,`,096,60,RTS
ABB3,6,032 054 173,20 36 AD,JSR &AD36
ABB6,,208 020,D0 14,BNE 20   --> &ABCC
ABB8,6,165 054,A5 36,LDA &36
ABBA,,240 031,F0 1F,BEQ 31   --> &ABDB
ABBC,,173 000 006,AD 00 06,LDA &0600
ABBF,L,076 024 174,4C 18 AE,JMP &AE18
ABC2,,032 018 170,20 12 AA,JSR &AA12
ABC5,,152,98,TYA
ABC6,,208 019,D0 13,BNE 19   --> &ABDB
ABC8,,138,8A,TXA

ABC9,L,076 026 174,4C 1A AE,JMP &AE1A
ABCC,L,076 146 144,4C 92 90,JMP &9092
ABCF,J,032 074 186,20 4A BA,JSR &BA4A
ABD2,,170,AA,TAX
ABD3,,169 127,A9 7F,LDA#&7F
ABD5,,032 244 255,20 F4 FF,JSR &FFF4
ABD8,,138,8A,TXA
ABD9,,240 002,F0 02,BEQ 2 --> &ABDD
ABDB,,162 255,A2 FF,LDX#&FF
ABDD,*,134 042,86 2A,STX &2A
ABDF,+,134 043,86 2B,STX &2B
ABE1,",",134 044,86 2C,STX &2C
ABE3,-,134 045,86 2D,STX &2D
ABE5,@,169 064,A9 40,LDA#&40
ABE7,`,096,60,RTS
ABE8,,162 000,A2 00,LDX#&00
ABEA,,128 241,80 F1,BRA -15 --> &ABDD
ABEC,,032 242 163,20 F2 A3,JSR &A3F2
ABEF,,240 247,F0 F7,BEQ -9 --> &ABE8
ABF1,,016 023,10 17,BPL 23 --> &AC0A
ABF3,,128 230,80 E6,BRA -26 --> &ABDB
ABF5,6,032 054 173,20 36 AD,JSR &AD36
ABF8,,240 210,F0 D2,BEQ -46 --> &ABCC
ABFA,0,048 240,30 F0,BMI -16 --> &ABEC
ABFC,-,165 045,A5 2D,LDA &2D
ABFE,",",005 044,05 2C,ORA &2C
AC00,+,005 043,05 2B,ORA &2B
AC02,*,005 042,05 2A,ORA &2A
AC04,,240 223,F0 DF,BEQ -33 --> &ABE5
AC06,-,165 045,A5 2D,LDA &2D
AC08,0,048 209,30 D1,BMI -47 --> &ABDB
AC0A,,169 001,A9 01,LDA#&01
AC0C,,128 177,80 B1,BRA -79 --> &ABBF
AC0E,,032 175 150,20 AF 96,JSR &96AF
AC11,&,032 038 188,20 26 BC,JSR &BC26
AC14,,032 241 142,20 F1 8E,JSR &8EF1
AC17,,032 167 150,20 A7 96,JSR &96A7
AC1A,*,165 042,A5 2A,LDA &2A
AC1C,H,072,48,PHA
AC1D,+,166 043,A6 2B,LDX &2B
AC1F,,032 230 188,20 E6 BC,JSR &BCE6
AC22,-,134 045,86 2D,STX &2D
AC24,h,104,68,PLA

AC25,",",133 044,85 2C,STA &2C
AC27,,160 000,A0 00,LDY#&00
AC29,*,162 042,A2 2A,LDX#&2A
AC2B,,169 009,A9 09,LDA#&09
AC2D,,032 241 255,20 F1 FF,JSR &FFF1
AC30,.,165 046,A5 2E,LDA &2E
AC32,0,048 167,30 A7,BMI -89 --> &ABDB
AC34,,128 214,80 D6,BRA -42 --> &AC0C
AC36,;,032 059 157,20 3B 9D,JSR &9D3B
AC39,,208 145,D0 91,BNE -111 --> &ABCC
AC3B,",",224 044,E0 2C,CPX#&2C
AC3D,,208 024,D0 18,BNE 24 --> &AC57
AC3F,,230 027,E6 1B,INC &1B
AC41,Q,032 081 188,20 51 BC,JSR &BC51
AC44,;,032 059 157,20 3B 9D,JSR &9D3B
AC47,,208 131,D0 83,BNE -125 --> &ABCC
AC49,,169 001,A9 01,LDA#&01
AC4B,*,133 042,85 2A,STA &2A
AC4D,,230 027,E6 1B,INC &1B
AC4F,),224 041,E0 29,CPX#&29
AC51,,240 013,F0 0D,BEQ 13 --> &AC60
AC53,",",224 044,E0 2C,CPX#&2C
AC55,,240 003,F0 03,BEQ 3 --> &AC5A
AC57,L,076 246 142,4C F6 8E,JMP &8EF6
AC5A,,032 164 150,20 A4 96,JSR &96A4
AC5D,,032 210 188,20 D2 BC,JSR &BCD2
AC60,*,166 042,A6 2A,LDX &2A
AC62,,208 002,D0 02,BNE 2 --> &AC66
AC64,,162 001,A2 01,LDX#&01
AC66,*,134 042,86 2A,STX &2A
AC68,,138,8A,TXA
AC69,,202,CA,DEX
AC6A,-,134 045,86 2D,STX &2D
AC6C,,024,18,CLC
AC6D,e,101 004,65 04,ADC &04
AC6F,7,133 055,85 37,STA &37
AC71,,169 000,A9 00,LDA#&00
AC73,e,101 005,65 05,ADC &05
AC75,8,133 056,85 38,STA &38
AC77,,178 004,B2 04,LDA (&04)
AC79,8,056,38,SEC
AC7A,-,229 045,E5 2D,SBC &2D
AC7C,!,144 033,90 21,BCC 33 --> &AC9F

```
AC7E,6,229 054,E5 36,SBC &36
AC80,,144 029,90 1D,BCC 29   --> &AC9F
AC82,i,105 000,69 00,ADC#&00
AC84,+,133 043,85 2B,STA &2B
AC86,,032 225 188,20 E1 BC,JSR &BCE1
AC89,,160 000,A0 00,LDY#&00
AC8B,6,166 054,A6 36,LDX &36
AC8D,,240 011,F0 0B,BEQ 11   --> &AC9A
AC8F,7,177 055,B1 37,"LDA (&37),Y"
AC91,,217 000 006,D9 00 06,"CMP &0600,Y"
AC94,,208 016,D0 10,BNE 16   --> &ACA6
AC96,,200,C8,INY
AC97,,202,CA,DEX
AC98,,208 245,D0 F5,BNE -11   --> &AC8F
AC9A,*,165 042,A5 2A,LDA &2A
AC9C,L,076 024 174,4C 18 AE,JMP &AE18
AC9F,,032 225 188,20 E1 BC,JSR &BCE1
ACA2,,169 000,A9 00,LDA#&00
ACA4,,128 246,80 F6,BRA -10   --> &AC9C
ACA6,*,230 042,E6 2A,INC &2A
ACA8,+,198 043,C6 2B,DEC &2B
ACAA,,240 246,F0 F6,BEQ -10   --> &ACA2
ACAC,7,230 055,E6 37,INC &37
ACAE,,208 217,D0 D9,BNE -39   --> &AC89
ACB0,8,230 056,E6 38,INC &38
ACB2,,128 213,80 D5,BRA -43   --> &AC89
ACB4,L,076 146 144,4C 92 90,JMP &9092
ACB7,6,032 054 173,20 36 AD,JSR &AD36
ACBA,,240 248,F0 F8,BEQ -8   --> &ACB4
ACBC,0,048 006,30 06,BMI 6   --> &ACC4
ACBE,$-,036 045,24 2D,BIT &2D
ACC0,0,048 028,30 1C,BMI 28   --> &ACDE
ACC2,1,128 049,80 31,BRA 49   --> &ACF5
ACC4,d.,100 046,64 2E,STZ &2E
ACC6,`,096,60,RTS
ACC7,,032 138 166,20 8A A6,JSR &A68A
ACCA,1,165 049,A5 31,LDA &31
ACCC,,240 006,F0 06,BEQ 6   --> &ACD4
ACCE,.,165 046,A5 2E,LDA &2E
ACD0,I,073 128,49 80,EOR#&80
ACD2,.,133 046,85 2E,STA &2E
ACD4,,169 255,A9 FF,LDA#&FF
ACD6,`,096,60,RTS
```

ACD7,L,032 076 173,20 4C AD,JSR &AD4C
ACDA,,240 216,F0 D8,BEQ -40 --> &ACB4
ACDC,0,048 236,30 EC,BMI -20 --> &ACCA
ACDE,8,056,38,SEC
ACDF,,169 000,A9 00,LDA#&00
ACE1,,168,A8,TAY
ACE2,*,229 042,E5 2A,SBC &2A
ACE4,*,133 042,85 2A,STA &2A
ACE6,,152,98,TYA
ACE7,+,229 043,E5 2B,SBC &2B
ACE9,+,133 043,85 2B,STA &2B
ACEB,,152,98,TYA
ACEC,",",229 044,E5 2C,SBC &2C
ACEE,",",133 044,85 2C,STA &2C
ACF0,,152,98,TYA
ACF1,-,229 045,E5 2D,SBC &2D
ACF3,-,133 045,85 2D,STA &2D
ACF5,@,169 064,A9 40,LDA#&40
ACF7,`,096,60,RTS
ACF8,,032 213 142,20 D5 8E,JSR &8ED5
ACFB,"""",201 034,C9 22,CMP#&22
ACFD,,240 026,F0 1A,BEQ 26 --> &AD19
ACFF,,162 000,A2 00,LDX#&00
AD01,,177 025,B1 19,"LDA (&19),Y"
AD03,,157 000 006,9D 00 06,"STA &0600,X"
AD06,,200,C8,INY
AD07,,232,E8,INX
AD08,,201 013,C9 0D,CMP#&0D
AD0A,,240 004,F0 04,BEQ 4 --> &AD10
AD0C,",",201 044,C9 2C,CMP#&2C
AD0E,,208 241,D0 F1,BNE -15 --> &AD01
AD10,,136,88,DEY
AD11,,202,CA,DEX
AD12,6,134 054,86 36,STX &36
AD14,,132 027,84 1B,STY &1B
AD16,,169 000,A9 00,LDA#&00
AD18,`,096,60,RTS
AD19,,162 000,A2 00,LDX#&00
AD1B,,200,C8,INY
AD1C,,177 025,B1 19,"LDA (&19),Y"
AD1E,,201 013,C9 0D,CMP#&0D
AD20,,240 017,F0 11,BEQ 17 --> &AD33
AD22,,157 000 006,9D 00 06,"STA &0600,X"

AD25,,200,C8,INY
AD26,,232,E8,INX
AD27,"""",201 034,C9 22,CMP#&22
AD29,,208 241,D0 F1,BNE -15 --> &AD1C
AD2B,,177 025,B1 19,"LDA (&19),Y"
AD2D,"""",201 034,C9 22,CMP#&22
AD2F,,240 234,F0 EA,BEQ -22 --> &AD1B
AD31,,208 222,D0 DE,BNE -34 --> &AD11
AD33,L,076 148 146,4C 94 92,JMP &9294
AD36,,164 027,A4 1B,LDY &1B
AD38,,230 027,E6 1B,INC &1B
AD3A,,177 025,B1 19,"LDA (&19),Y"
AD3C,,201 032,C9 20,CMP#&20
AD3E,,240 246,F0 F6,BEQ -10 --> &AD36
AD40,-,201 045,C9 2D,CMP#&2D
AD42,,240 147,F0 93,BEQ -109 --> &ACD7
AD44,"""",201 034,C9 22,CMP#&22
AD46,,240 209,F0 D1,BEQ -47 --> &AD19
AD48,+,201 043,C9 2B,CMP#&2B
AD4A,,208 003,D0 03,BNE 3 --> &AD4F
AD4C,,032 213 142,20 D5 8E,JSR &8ED5
AD4F,,201 142,C9 8E,CMP#&8E
AD51,,144 007,90 07,BCC 7 --> &AD5A
AD53,,201 198,C9 C6,CMP#&C6
AD55,5,176 053,B0 35,BCS 53 --> &AD8C
AD57,L,076 025 144,4C 19 90,JMP &9019
AD5A,?,201 063,C9 3F,CMP#&3F
AD5C,,176 012,B0 0C,BCS 12 --> &AD6A
AD5E,.,201 046,C9 2E,CMP#&2E
AD60,,176 018,B0 12,BCS 18 --> &AD74
AD62,&,201 038,C9 26,CMP#&26
AD64,Q,240 081,F0 51,BEQ 81 --> &ADB7
AD66,(,201 040,C9 28,CMP#&28
AD68,B,240 066,F0 42,BEQ 66 --> &ADAC
AD6A,,198 027,C6 1B,DEC &1B
AD6C,,032 009 153,20 09 99,JSR &9909
AD6F,,240 009,F0 09,BEQ 9 --> &AD7A
AD71,L,076 160 177,4C A0 B1,JMP &B1A0
AD74,,032 225 162,20 E1 A2,JSR &A2E1
AD77,,144 019,90 13,BCC 19 --> &AD8C
AD79,`,096,60,RTS
AD7A,(,165 040,A5 28,LDA &28
AD7C,),041 002,29 02,AND#&02

AD7E,,208 012,D0 0C,BNE 12   --> &AD8C
AD80,,176 010,B0 0A,BCS 10   --> &AD8C
AD82,,134 027,86 1B,STX &1B
AD84,@,173 064 004,AD 40 04,LDA &0440
AD87,A,172 065 004,AC 41 04,LDY &0441
AD8A,k,128 107,80 6B,BRA 107   --> &ADF7
AD8C,,000,00,BRK
AD8D,,026,1A,EQUB &1A
AD8E,No,078 111 032,4E 6F 20,LSR &206F
AD91,s,115,73,xxx Invalid Code
AD92,uc,117 099,75 63,"ADC &63,X"
AD94,h,104,68,PLA
AD95,va,032 118 097,20 76 61,JSR &6176
AD98,ri,114 105,72 69,ADC (&69)
AD9A,abl,097 098 108,61 62 6C,"ADC (&6C62,X)"
AD9D,e,101,65,xxx Invalid Code
AD9E,,000,00,BRK
AD9F,,027,1B,EQUB &1B
ADA0,),141 041 000,8D 29 00,STA &0029
ADA3,Ba,028 066 097,1C 42 61,TRB &6142
ADA6,d,100 032,64 20,STZ &20
ADA8,H,072,48,PHA
ADA9,ex,101 120,65 78,ADC &78
ADAB,,000,00,EQUB &00
ADAC,;,032 059 157,20 3B 9D,JSR &9D3B
ADAF,,230 027,E6 1B,INC &1B
ADB1,),224 041,E0 29,CPX#&29
ADB3,,208 233,D0 E9,BNE -23   --> &AD9E
ADB5,,168,A8,TAY
ADB6,`,096,60,RTS
ADB7,,032 232 171,20 E8 AB,JSR &ABE8
ADBA,,200,C8,INY
ADBB,,177 025,B1 19,"LDA (&19),Y"
ADBD,0,201 048,C9 30,CMP#&30
ADBF,#,144 035,90 23,BCC 35   --> &ADE4
ADC1,:,201 058,C9 3A,CMP#&3A
ADC3,,144 010,90 0A,BCC 10   --> &ADCF
ADC5,7,233 055,E9 37,SBC#&37
ADC7,,201 010,C9 0A,CMP#&0A
ADC9,,144 025,90 19,BCC 25   --> &ADE4
ADCB,,201 016,C9 10,CMP#&10
ADCD,,176 021,B0 15,BCS 21   --> &ADE4
ADCF,,010,0A,ASL A

ADD0,,010,0A,ASL A
ADD1,,010,0A,ASL A
ADD2,,010,0A,ASL A
ADD3,,162 003,A2 03,LDX#&03
ADD5,,010,0A,ASL A
ADD6,&*,038 042,26 2A,ROL &2A
ADD8,&+,038 043,26 2B,ROL &2B
ADDA,"&,",038 044,26 2C,ROL &2C
ADDC,&-,038 045,26 2D,ROL &2D
ADDE,,202,CA,DEX
ADDF,,016 244,10 F4,BPL -12 --> &ADD5
ADE1,,200,C8,INY
ADE2,,208 215,D0 D7,BNE -41 --> &ADBB
ADE4,,138,8A,TXA
ADE5,,016 187,10 BB,BPL -69 --> &ADA2
ADE7,,132 027,84 1B,STY &1B
ADE9,@,169 064,A9 40,LDA#&40
ADEB,`,096,60,RTS
ADEC,,032 180 150,20 B4 96,JSR &96B4
ADEF,*,166 042,A6 2A,LDX &2A
ADF1,,169 128,A9 80,LDA#&80
ADF3,,032 244 255,20 F4 FF,JSR &FFF4
ADF6,,138,8A,TXA
ADF7,!,128 033,80 21,BRA 33 --> &AE1A
ADF9,,200,C8,INY
ADFA,,177 025,B1 19,"LDA (&19),Y"
ADFC,P,201 080,C9 50,CMP#&50
ADFE,,208 140,D0 8C,BNE -116 --> &AD8C
AE00,,230 027,E6 1B,INC &1B
AE02,,165 018,A5 12,LDA &12
AE04,,164 019,A4 13,LDY &13
AE06,,128 018,80 12,BRA 18 --> &AE1A
AE08,,164 024,A4 18,LDY &18
AE0A,,169 000,A9 00,LDA#&00
AE0C,,128 012,80 0C,BRA 12 --> &AE1A
AE0E,L,076 146 144,4C 92 90,JMP &9092
AE11,6,032 054 173,20 36 AD,JSR &AD36
AE14,,208 248,D0 F8,BNE -8 --> &AE0E
AE16,6,165 054,A5 36,LDA &36
AE18,,160 000,A0 00,LDY#&00
AE1A,*,133 042,85 2A,STA &2A
AE1C,+,132 043,84 2B,STY &2B
AE1E,"d,",100 044,64 2C,STZ &2C

```
AE20,d-,100 045,64 2D,STZ &2D
AE22,@,169 064,A9 40,LDA#&40
AE24,`,096,60,RTS
AE25,,165 030,A5 1E,LDA &1E
AE27,,128 239,80 EF,BRA -17   --> &AE18
AE29,,165 000,A5 00,LDA &00
AE2B,,164 001,A4 01,LDY &01
AE2D,,128 235,80 EB,BRA -21   --> &AE1A
AE2F,,165 006,A5 06,LDA &06
AE31,,164 007,A4 07,LDY &07
AE33,,128 229,80 E5,BRA -27   --> &AE1A
AE35,,164 009,A4 09,LDY &09
AE37,,165 008,A5 08,LDA &08
AE39,,128 223,80 DF,BRA -33   --> &AE1A
AE3B,,178 253,B2 FD,LDA (&FD)
AE3D,,128 217,80 D9,BRA -39   --> &AE18
AE3F,,032 224 255,20 E0 FF,JSR &FFE0
AE42,,128 212,80 D4,BRA -44   --> &AE18
AE44,,200,C8,INY
AE45,,177 025,B1 19,"LDA (&19),Y"
AE47,$,201 036,C9 24,CMP#&24
AE49,,240 012,F0 0C,BEQ 12   --> &AE57
AE4B,*,162 042,A2 2A,LDX#&2A
AE4D,,160 000,A0 00,LDY#&00
AE4F,,169 001,A9 01,LDA#&01
AE51,,032 241 255,20 F1 FF,JSR &FFF1
AE54,@,169 064,A9 40,LDA#&40
AE56,`,096,60,RTS
AE57,,230 027,E6 1B,INC &1B
AE59,,169 014,A9 0E,LDA#&0E
AE5B,,162 000,A2 00,LDX#&00
AE5D,,160 006,A0 06,LDY#&06
AE5F,,156 000 006,9C 00 06,STZ &0600
AE62,,032 241 255,20 F1 FF,JSR &FFF1
AE65,,169 024,A9 18,LDA#&18
AE67,&,128 038,80 26,BRA 38   --> &AE8F
AE69,,032 224 255,20 E0 FF,JSR &FFE0
AE6C,,141 000 006,8D 00 06,STA &0600
AE6F,,169 001,A9 01,LDA#&01
AE71,,128 028,80 1C,BRA 28   --> &AE8F
AE73,,024,18,CLC
AE74,,008,08,PHP
AE75,;,032 059 157,20 3B 9D,JSR &9D3B
```

AE78,E,208 069,D0 45,BNE 69   --> &AEBF
AE7A,",",224 044,E0 2C,CPX#&2C
AE7C,D,208 068,D0 44,BNE 68   --> &AEC2
AE7E,,230 027,E6 1B,INC &1B
AE80,,032 164 150,20 A4 96,JSR &96A4
AE83,,032 210 188,20 D2 BC,JSR &BCD2
AE86,(,040,28,PLP
AE87,,176 011,B0 0B,BCS 11   --> &AE94
AE89,*,165 042,A5 2A,LDA &2A
AE8B,6,197 054,C5 36,CMP &36
AE8D,,176 002,B0 02,BCS 2   --> &AE91
AE8F,6,133 054,85 36,STA &36
AE91,,169 000,A9 00,LDA#&00
AE93,`,096,60,RTS
AE94,6,165 054,A5 36,LDA &36
AE96,*,229 042,E5 2A,SBC &2A
AE98,,144 247,90 F7,BCC -9   --> &AE91
AE9A,,240 247,F0 F7,BEQ -9   --> &AE93
AE9C,,170,AA,TAX
AE9D,*,165 042,A5 2A,LDA &2A
AE9F,6,133 054,85 36,STA &36
AEA1,,240 240,F0 F0,BEQ -16   --> &AE93
AEA3,,160 000,A0 00,LDY#&00
AEA5,,189 000 006,BD 00 06,"LDA &0600,X"
AEA8,,153 000 006,99 00 06,"STA &0600,Y"
AEAB,,232,E8,INX
AEAC,,200,C8,INY
AEAD,*,198 042,C6 2A,DEC &2A
AEAF,,208 244,D0 F4,BNE -12   --> &AEA5
AEB1,,128 222,80 DE,BRA -34   --> &AE91
AEB3,,032 018 170,20 12 AA,JSR &AA12
AEB6,,138,8A,TXA
AEB7,,192 000,C0 00,CPY#&00
AEB9,,240 177,F0 B1,BEQ -79   --> &AE6C
AEBB,,169 000,A9 00,LDA#&00
AEBD,,128 208,80 D0,BRA -48   --> &AE8F
AEBF,L,076 146 144,4C 92 90,JMP &9092
AEC2,L,076 246 142,4C F6 8E,JMP &8EF6
AEC5,;,032 059 157,20 3B 9D,JSR &9D3B
AEC8,,208 245,D0 F5,BNE -11   --> &AEBF
AECA,",",224 044,E0 2C,CPX#&2C
AECC,,208 244,D0 F4,BNE -12   --> &AEC2
AECE,Q,032 081 188,20 51 BC,JSR &BC51

AED1,,230 027,E6 1B,INC &1B
AED3,,032 175 150,20 AF 96,JSR &96AF
AED6,*,165 042,A5 2A,LDA &2A
AED8,H,072,48,PHA
AED9,,169 255,A9 FF,LDA#&FF
AEDB,*,133 042,85 2A,STA &2A
AEDD,,230 027,E6 1B,INC &1B
AEDF,),224 041,E0 29,CPX#&29
AEE1,,240 007,F0 07,BEQ 7 --> &AEEA
AEE3,",",224 044,E0 2C,CPX#&2C
AEE5,,208 219,D0 DB,BNE -37 --> &AEC2
AEE7,,032 167 150,20 A7 96,JSR &96A7
AEEA,,032 210 188,20 D2 BC,JSR &BCD2
AEED,h,104,68,PLA
AEEE,,168,A8,TAY
AEEF,,024,18,CLC
AEF0,,240 006,F0 06,BEQ 6 --> &AEF8
AEF2,6,229 054,E5 36,SBC &36
AEF4,,176 197,B0 C5,BCS -59 --> &AEBB
AEF6,,136,88,DEY
AEF7,,152,98,TYA
AEF8,",",133 044,85 2C,STA &2C
AEFA,,170,AA,TAX
AEFB,,160 000,A0 00,LDY#&00
AEFD,6,165 054,A5 36,LDA &36
AEFF,8,056,38,SEC
AF00,",",229 044,E5 2C,SBC &2C
AF02,*,197 042,C5 2A,CMP &2A
AF04,,176 002,B0 02,BCS 2 --> &AF08
AF06,*,133 042,85 2A,STA &2A
AF08,*,165 042,A5 2A,LDA &2A
AF0A,,240 175,F0 AF,BEQ -81 --> &AEBB
AF0C,,189 000 006,BD 00 06,"LDA &0600,X"
AF0F,,153 000 006,99 00 06,"STA &0600,Y"
AF12,,200,C8,INY
AF13,,232,E8,INX
AF14,*,196 042,C4 2A,CPY &2A
AF16,,208 244,D0 F4,BNE -12 --> &AF0C
AF18,6,132 054,84 36,STY &36
AF1A,^,128 094,80 5E,BRA 94 --> &AF7A
AF1C,,032 213 142,20 D5 8E,JSR &8ED5
AF1F,,160 255,A0 FF,LDY#&FF
AF21,~,201 126,C9 7E,CMP#&7E

AF23,,240 004,F0 04,BEQ 4   --> &AF29
AF25,,160 000,A0 00,LDY#&00
AF27,,198 027,C6 1B,DEC &1B
AF29,Z,090,5A,PHY
AF2A,6,032 054 173,20 36 AD,JSR &AD36
AF2D,,240 021,F0 15,BEQ 21   --> &AF44
AF2F,,168,A8,TAY
AF30,h,104,68,PLA
AF31,,133 021,85 15,STA &15
AF33,,173 003 004,AD 03 04,LDA &0403
AF36,,208 007,D0 07,BNE 7   --> &AF3F
AF38,7,133 055,85 37,STA &37
AF3A,2,032 050 161,20 32 A1,JSR &A132
AF3D,;,128 059,80 3B,BRA 59   --> &AF7A
AF3F,,032 024 161,20 18 A1,JSR &A118
AF42,6,128 054,80 36,BRA 54   --> &AF7A
AF44,L,076 146 144,4C 92 90,JMP &9092
AF47,,032 175 150,20 AF 96,JSR &96AF
AF4A,&,032 038 188,20 26 BC,JSR &BC26
AF4D,,032 241 142,20 F1 8E,JSR &8EF1
AF50,,032 172 173,20 AC AD,JSR &ADAC
AF53,,208 239,D0 EF,BNE -17   --> &AF44
AF55,,032 230 188,20 E6 BC,JSR &BCE6
AF58,6,164 054,A4 36,LDY &36
AF5A,,240 030,F0 1E,BEQ 30   --> &AF7A
AF5C,*,165 042,A5 2A,LDA &2A
AF5E,,240 029,F0 1D,BEQ 29   --> &AF7D
AF60,*,198 042,C6 2A,DEC &2A
AF62,,240 022,F0 16,BEQ 22   --> &AF7A
AF64,,162 000,A2 00,LDX#&00
AF66,,189 000 006,BD 00 06,"LDA &0600,X"
AF69,,153 000 006,99 00 06,"STA &0600,Y"
AF6C,,232,E8,INX
AF6D,,200,C8,INY
AF6E,,240 016,F0 10,BEQ 16   --> &AF80
AF70,6,228 054,E4 36,CPX &36
AF72,,144 242,90 F2,BCC -14   --> &AF66
AF74,*,198 042,C6 2A,DEC &2A
AF76,,208 236,D0 EC,BNE -20   --> &AF64
AF78,6,132 054,84 36,STY &36
AF7A,,169 000,A9 00,LDA#&00
AF7C,`,096,60,RTS
AF7D,6,133 054,85 36,STA &36

AF7F,`,096,60,RTS
AF80,L,076 016 158,4C 10 9E,JMP &9E10
AF83,h,104,68,PLA
AF84,,133 012,85 0C,STA &0C
AF86,h,104,68,PLA
AF87,,133 011,85 0B,STA &0B
AF89,,000,00,BRK
AF8A,,029,1D,EQUB &1D
AF8B,No,078 111,4E 6F,xxx Invalid Code
AF8D,su,032 115 117,20 73 75,JSR &7573
AF90,c,099,63,xxx Invalid Code
AF91,h,104,68,PLA
AF92,/,032 164 047,20 A4 2F,JSR &2FA4
AF95,,242,F2,xxx Invalid Code
AF96,,000,00,EQUB &00
AF97,,165 024,A5 18,LDA &18
AF99,,133 012,85 0C,STA &0C
AF9B,d,100 011,64 0B,STZ &0B
AF9D,,160 001,A0 01,LDY#&01
AF9F,,177 011,B1 0B,"LDA (&0B),Y"
AFA1,0,048 224,30 E0,BMI -32 --> &AF83
AFA3,,160 003,A0 03,LDY#&03
AFA5,,200,C8,INY
AFA6,,177 011,B1 0B,"LDA (&0B),Y"
AFA8,,201 032,C9 20,CMP#&20
AFAA,,240 249,F0 F9,BEQ -7 --> &AFA5
AFAC,,201 221,C9 DD,CMP#&DD
AFAE,,240 015,F0 0F,BEQ 15 --> &AFBF
AFB0,,160 003,A0 03,LDY#&03
AFB2,,177 011,B1 0B,"LDA (&0B),Y"
AFB4,,024,18,CLC
AFB5,e,101 011,65 0B,ADC &0B
AFB7,,133 011,85 0B,STA &0B
AFB9,,144 226,90 E2,BCC -30 --> &AF9D
AFBB,,230 012,E6 0C,INC &0C
AFBD,,128 222,80 DE,BRA -34 --> &AF9D
AFBF,,200,C8,INY
AFC0,,132 010,84 0A,STY &0A
AFC2,,032 224 142,20 E0 8E,JSR &8EE0
AFC5,,152,98,TYA
AFC6,,170,AA,TAX
AFC7,,024,18,CLC
AFC8,e,101 011,65 0B,ADC &0B

AFCA,,164 012,A4 0C,LDY &0C
AFCC,,144 002,90 02,BCC 2   --> &AFD0
AFCE,,200,C8,INY
AFCF,,024,18,CLC
AFD0,,233 000,E9 00,SBC#&00
AFD2,<,133 060,85 3C,STA &3C
AFD4,,152,98,TYA
AFD5,,233 000,E9 00,SBC#&00
AFD7,=,133 061,85 3D,STA &3D
AFD9,,160 001,A0 01,LDY#&01
AFDB,,232,E8,INX
AFDC,<,177 060,B1 3C,"LDA (&3C),Y"
AFDE,7,209 055,D1 37,"CMP (&37),Y"
AFE0,,208 206,D0 CE,BNE -50   --> &AFB0
AFE2,,200,C8,INY
AFE3,9,196 057,C4 39,CPY &39
AFE5,,208 244,D0 F4,BNE -12   --> &AFDB
AFE7,<,177 060,B1 3C,"LDA (&3C),Y"
AFE9,,032 132 141,20 84 8D,JSR &8D84
AFEC,,176 194,B0 C2,BCS -62   --> &AFB0
AFEE,,138,8A,TXA
AFEF,,168,A8,TAY
AFF0,,032 188 155,20 BC 9B,JSR &9BBC
AFF3,E,032 069 152,20 45 98,JSR &9845
AFF6,,162 001,A2 01,LDX#&01
AFF8,,032 131 152,20 83 98,JSR &9883
AFFB,,165 011,A5 0B,LDA &0B
AFFD,,146 002,92 02,STA (&02)
AFFF,,160 001,A0 01,LDY#&01
B001,,165 012,A5 0C,LDA &0C

BASIC IV ROM Routines

Disassembly B000 to C000

B001		165 012	A5 0C	LDA &0C
B003		145 002	91 02	STA (&02),Y
B005		200	C8	INY
B006		032 139 152	20 8B 98	JSR &988B
B009	Lr	076 114 176	4C 72 B0	JMP &B072
B00C		000	00	BRK
B00D		030	1E	EQUB &1E
B00E	Ba	066 097	42 61	xxx Invalid Code
B010	d	100 032	64 20	STZ &20
B012	c	099	63	xxx Invalid Code
B013	all	097 108 108	61 6C 6C	ADC (&6C6C,X)
B016		000	00	EQUB &00
B017		169 164	A9 A4	LDA#&A4
B019	'	133 039	85 27	STA &27
B01B		186	BA	TSX

```
B01C        138          8A          TXA
B01D        024          18          CLC
B01E    e   101 004      65 04       ADC &04
B020        032 030 189  20 1E BD     JSR &BD1E
B023        138          8A          TXA
B024        146 004      92 04       STA (&04)
B026        160 000      A0 00       LDY#&00
B028        232          E8          INX
B029        200          C8          INY
B02A        189 000 001  BD 00 01     LDA &0100,X
B02D        145 004      91 04       STA (&04),Y
B02F        224 255      E0 FF       CPX#&FF
B031        208 245      D0 F5       BNE -11 --> &B028
B033        154          9A          TXS
B034    '   165 039      A5 27       LDA &27
B036    H   072          48          PHA
B037        165 010      A5 0A       LDA &0A
B039    H   072          48          PHA
B03A        165 011      A5 0B       LDA &0B
B03C    H   072          48          PHA
B03D        165 012      A5 0C       LDA &0C
B03F    H   072          48          PHA
B040        165 027      A5 1B       LDA &1B
B042        170          AA          TAX
B043        024          18          CLC
B044    e   101 025      65 19       ADC &19
B046        164 026      A4 1A       LDY &1A
B048        144 002      90 02       BCC 2 --> &B04C
B04A        200          C8          INY
B04B        024          18          CLC
B04C        233 001      E9 01       SBC#&01
B04E    7   133 055      85 37       STA &37
B050        152          98          TYA
B051        233 000      E9 00       SBC#&00
B053    8   133 056      85 38       STA &38
```

B055		160 002	A0 02	LDY#&02
B057		032 248 154	20 F8 9A	JSR &9AF8
B05A		192 002	C0 02	CPY#&02
B05C		240 174	F0 AE	BEQ -82 --> &B00C
B05E		134 027	86 1B	STX &1B
B060	u	032 117 128	20 75 80	JSR &8075
B063		208 003	D0 03	BNE 3 --> &B068
B065	L	076 151 175	4C 97 AF	JMP &AF97
B068	*	178 042	B2 2A	LDA (&2A)
B06A		133 011	85 0B	STA &0B
B06C		160 001	A0 01	LDY#&01
B06E	*	177 042	B1 2A	LDA (&2A),Y
B070		133 012	85 0C	STA &0C
B072		169 000	A9 00	LDA#&00
B074	H	072	48	PHA
B075	d	100 010	64 0A	STZ &0A
B077		032 224 142	20 E0 8E	JSR &8EE0
B07A	(	201 040	C9 28	CMP#&28
B07C	M	240 077	F0 4D	BEQ 77 --> &B0CB
B07E		198 010	C6 0A	DEC &0A
B080		165 027	A5 1B	LDA &1B
B082	H	072	48	PHA
B083		165 025	A5 19	LDA &19
B085	H	072	48	PHA
B086		165 026	A5 1A	LDA &1A
B088	H	072	48	PHA
B089		032 011 144	20 0B 90	JSR &900B
B08C	h	104	68	PLA
B08D		133 026	85 1A	STA &1A
B08F	h	104	68	PLA
B090		133 025	85 19	STA &19
B092	h	104	68	PLA
B093		133 027	85 1B	STA &1B
B095	h	104	68	PLA
B096		240 012	F0 0C	BEQ 12 --> &B0A4

B098	?	133 063	85 3F	STA &3F
B09A		032 006 189	20 06 BD	JSR &BD06
B09D	j	032 106 188	20 6A BC	JSR &BC6A
B0A0	?	198 063	C6 3F	DEC &3F
B0A2		208 246	D0 F6	BNE -10 --> &B09A
B0A4	h	104	68	PLA
B0A5		133 012	85 0C	STA &0C
B0A7	h	104	68	PLA
B0A8		133 011	85 0B	STA &0B
B0AA	h	104	68	PLA
B0AB		133 010	85 0A	STA &0A
B0AD	h	104	68	PLA
B0AE		178 004	B2 04	LDA (&04)
B0B0		170	AA	TAX
B0B1		154	9A	TXS
B0B2		160 000	A0 00	LDY#&00
B0B4		200	C8	INY
B0B5		232	E8	INX
B0B6		177 004	B1 04	LDA (&04),Y
B0B8		157 000 001	9D 00 01	STA &0100,X
B0BB		224 255	E0 FF	CPX#&FF
B0BD		208 245	D0 F5	BNE -11 --> &B0B4
B0BF		152	98	TYA
B0C0	e	101 004	65 04	ADC &04
B0C2		133 004	85 04	STA &04
B0C4		144 002	90 02	BCC 2 --> &B0C8
B0C6		230 005	E6 05	INC &05
B0C8	'	165 039	A5 27	LDA &27
B0CA	`	096	60	RTS
B0CB		165 027	A5 1B	LDA &1B
B0CD	H	072	48	PHA
B0CE		165 025	A5 19	LDA &19
B0D0	H	072	48	PHA
B0D1		165 026	A5 1A	LDA &1A
B0D3	H	072	48	PHA

B0D4		032 174 152	20 AE 98	JSR &98AE
B0D7	R	240 082	F0 52	BEQ 82 --> &B12B
B0D9		165 027	A5 1B	LDA &1B
B0DB		133 010	85 0A	STA &0A
B0DD	h	104	68	PLA
B0DE		133 026	85 1A	STA &1A
B0E0	h	104	68	PLA
B0E1		133 025	85 19	STA &19
B0E3	h	104	68	PLA
B0E4		133 027	85 1B	STA &1B
B0E6		250	FA	PLX
B0E7	,	165 044	A5 2C	LDA &2C
B0E9	H	072	48	PHA
B0EA	+	165 043	A5 2B	LDA &2B
B0EC	H	072	48	PHA
B0ED	*	165 042	A5 2A	LDA &2A
B0EF	H	072	48	PHA
B0F0		232	E8	INX
B0F1		218	DA	PHX
B0F2		032 129 177	20 81 B1	JSR &B181
B0F5		032 229 140	20 E5 8C	JSR &8CE5
B0F8		240 209	F0 D1	BEQ -47 --> &B0CB
B0FA	)	201 041	C9 29	CMP#&29
B0FC	-	208 045	D0 2D	BNE 45 --> &B12B
B0FE		169 000	A9 00	LDA#&00
B100	H	072	48	PHA
B101		032 213 142	20 D5 8E	JSR &8ED5
B104	(	201 040	C9 28	CMP#&28
B106	#	208 035	D0 23	BNE 35 --> &B12B
B108	;	032 059 157	20 3B 9D	JSR &9D3B
B10B	"	032 034 188	20 22 BC	JSR &BC22
B10E	'	165 039	A5 27	LDA &27
B110	-	133 045	85 2D	STA &2D
B112	&	032 038 188	20 26 BC	JSR &BC26
B115		250	FA	PLX

B116		232	E8	INX
B117		218	DA	PHX
B118		032 235 142	20 EB 8E	JSR &8EEB
B11B		240 235	F0 EB	BEQ -21 --> &B108
B11D	)	201 041	C9 29	CMP#&29
B11F		208 010	D0 0A	BNE 10 --> &B12B
B121	h	104	68	PLA
B122	h	104	68	PLA
B123	L	133 076	85 4C	STA &4C
B125	M	133 077	85 4D	STA &4D
B127	L	228 076	E4 4C	CPX &4C
B129		240 021	F0 15	BEQ 21 --> &B140
B12B		162 251	A2 FB	LDX#&FB
B12D		154	9A	TXS
B12E	h	104	68	PLA
B12F		133 012	85 0C	STA &0C
B131	h	104	68	PLA
B132		133 011	85 0B	STA &0B
B134		000	00	BRK
B135		031	1F	EQUB &1F
B136	Ar	065 114	41 72	EOR (&72,X)
B138	g	103	67	xxx Invalid Code
B139	um	117 109	75 6D	ADC &6D,X
B13B	en	101 110	65 6E	ADC &6E
B13D	ts	116 115	74 73	STZ &73,X
B13F		000	00	EQUB &00
B140		032 230 188	20 E6 BC	JSR &BCE6
B143	h	104	68	PLA
B144	*	133 042	85 2A	STA &2A
B146	h	104	68	PLA
B147	+	133 043	85 2B	STA &2B
B149	h	104	68	PLA
B14A	,	133 044	85 2C	STA &2C
B14C	0	048 031	30 1F	BMI 31 --> &B16D
B14E	-	165 045	A5 2D	LDA &2D

B150		240 217	F0 D9	BEQ -39 --> &B12B
B152	'	133 039	85 27	STA &27
B154	7	162 055	A2 37	LDX#&37
B156		032 198 189	20 C6 BD	JSR &BDC6
B159	'	165 039	A5 27	LDA &27
B15B		016 008	10 08	BPL 8 --> &B165
B15D		032 232 187	20 E8 BB	JSR &BBE8
B160	A	032 065 165	20 41 A5	JSR &A541
B163		128 003	80 03	BRA 3 --> &B168
B165		032 230 188	20 E6 BC	JSR &BCE6
B168	8	032 056 179	20 38 B3	JSR &B338
B16B		128 010	80 0A	BRA 10 --> &B177
B16D	-	165 045	A5 2D	LDA &2D
B16F		208 186	D0 BA	BNE -70 --> &B12B
B171		032 210 188	20 D2 BC	JSR &BCD2
B174		032 174 144	20 AE 90	JSR &90AE
B177	L	198 076	C6 4C	DEC &4C
B179		208 197	D0 C5	BNE -59 --> &B140
B17B	M	165 077	A5 4D	LDA &4D
B17D	H	072	48	PHA
B17E	L	076 128 176	4C 80 B0	JMP &B080
B181	,	164 044	A4 2C	LDY &2C
B183		192 005	C0 05	CPY#&05
B185		176 005	B0 05	BCS 5 --> &B18C
B187	7	162 055	A2 37	LDX#&37
B189		032 198 189	20 C6 BD	JSR &BDC6
B18C		032 160 177	20 A0 B1	JSR &B1A0
B18F		008	08	PHP
B190	"	032 034 188	20 22 BC	JSR &BC22
B193	(	040	28	PLP
B194		240 007	F0 07	BEQ 7 --> &B19D
B196	0	048 005	30 05	BMI 5 --> &B19D
B198	7	162 055	A2 37	LDX#&37
B19A		032 128 170	20 80 AA	JSR &AA80
B19D	L&	076 038 188	4C 26 BC	JMP &BC26

B1A0	,	164 044	A4 2C	LDY &2C
B1A2	0S	048 083	30 53	BMI 83 --> &B1F7
B1A4		240 028	F0 1C	BEQ 28 --> &B1C2
B1A6		192 005	C0 05	CPY#&05
B1A8		240 029	F0 1D	BEQ 29 --> &B1C7
B1AA		160 003	A0 03	LDY#&03
B1AC	*	177 042	B1 2A	LDA (&2A),Y
B1AE	-	133 045	85 2D	STA &2D
B1B0		136	88	DEY
B1B1	*	177 042	B1 2A	LDA (&2A),Y
B1B3	,	133 044	85 2C	STA &2C
B1B5		136	88	DEY
B1B6	*	177 042	B1 2A	LDA (&2A),Y
B1B8		170	AA	TAX
B1B9	*	178 042	B2 2A	LDA (&2A)
B1BB	*	133 042	85 2A	STA &2A
B1BD	+	134 043	86 2B	STX &2B
B1BF	@	169 064	A9 40	LDA#&40
B1C1	`	096	60	RTS
B1C2	*	177 042	B1 2A	LDA (&2A),Y
B1C4	L	076 026 174	4C 1A AE	JMP &AE1A
B1C7	d5	100 053	64 35	STZ &35
B1C9	d/	100 047	64 2F	STZ &2F
B1CB		136	88	DEY
B1CC	*	177 042	B1 2A	LDA (&2A),Y
B1CE	4	133 052	85 34	STA &34
B1D0		136	88	DEY
B1D1	*	177 042	B1 2A	LDA (&2A),Y
B1D3	3	133 051	85 33	STA &33
B1D5		136	88	DEY
B1D6	*	177 042	B1 2A	LDA (&2A),Y
B1D8	2	133 050	85 32	STA &32
B1DA		136	88	DEY
B1DB	*	177 042	B1 2A	LDA (&2A),Y
B1DD	.	133 046	85 2E	STA &2E

```
B1DF          168          A8          TAY
B1E0    *     178 042      B2 2A       LDA (&2A)
B1E2    0     133 048      85 30       STA &30
B1E4          208 009      D0 09       BNE 9 --> &B1EF
B1E6          152          98          TYA
B1E7    2     005 050      05 32       ORA &32
B1E9    3     005 051      05 33       ORA &33
B1EB    4     005 052      05 34       ORA &34
B1ED          240 003      F0 03       BEQ 3 --> &B1F2
B1EF          152          98          TYA
B1F0          009 128      09 80       ORA#&80
B1F2    1     133 049      85 31       STA &31
B1F4          169 255      A9 FF       LDA#&FF
B1F6    `     096          60          RTS
B1F7          192 128      C0 80       CPY#&80
B1F9          240 030      F0 1E       BEQ 30 --> &B219
B1FB          160 003      A0 03       LDY#&03
B1FD    *     177 042      B1 2A       LDA (&2A),Y
B1FF    6     133 054      85 36       STA &36
B201          240 021      F0 15       BEQ 21 --> &B218
B203          160 001      A0 01       LDY#&01
B205    *     177 042      B1 2A       LDA (&2A),Y
B207    8     133 056      85 38       STA &38
B209    *     178 042      B2 2A       LDA (&2A)
B20B    7     133 055      85 37       STA &37
B20D    6     164 054      A4 36       LDY &36
B20F          136          88          DEY
B210    7     177 055      B1 37       LDA (&37),Y
B212          153 000 006  99 00 06    STA &0600,Y
B215          152          98          TYA
B216          208 247      D0 F7       BNE -9 --> &B20F
B218    `     096          60          RTS
B219    +     165 043      A5 2B       LDA &2B
B21B          240 021      F0 15       BEQ 21 --> &B232
B21D          160 000      A0 00       LDY#&00
```

B21F	*	177 042	B1 2A	LDA (&2A),Y
B221		153 000 006	99 00 06	STA &0600,Y
B224	I	073 013	49 0D	EOR#&0D
B226		240 004	F0 04	BEQ 4 --> &B22C
B228		200	C8	INY
B229		208 244	D0 F4	BNE -12 --> &B21F
B22B		152	98	TYA
B22C	6	132 054	84 36	STY &36
B22E	`	096	60	RTS
B22F		032 180 150	20 B4 96	JSR &96B4
B232	*	165 042	A5 2A	LDA &2A
B234	Ll	076 108 174	4C 6C AE	JMP &AE6C
B237		164 010	A4 0A	LDY &0A
B239		240 001	F0 01	BEQ 1 --> &B23C
B23B		136	88	DEY
B23C		032 188 155	20 BC 9B	JSR &9BBC
B23F	d	100 008	64 08	STZ &08
B241	d	100 009	64 09	STZ &09
B243		166 024	A6 18	LDX &18
B245	8	134 056	86 38	STX &38
B247	d7	100 055	64 37	STZ &37
B249		164 012	A4 0C	LDY &0C
B24B		192 007	C0 07	CPY#&07
B24D	(	240 040	F0 28	BEQ 40 --> &B277
B24F		166 011	A6 0B	LDX &0B
B251		032 160 141	20 A0 8D	JSR &8DA0
B254		201 013	C9 0D	CMP#&0D
B256		208 024	D0 18	BNE 24 --> &B270
B258	7	228 055	E4 37	CPX &37
B25A		152	98	TYA
B25B	8	229 056	E5 38	SBC &38
B25D		144 024	90 18	BCC 24 --> &B277
B25F		032 160 141	20 A0 8D	JSR &8DA0
B262		009 000	09 00	ORA#&00
B264	0	048 017	30 11	BMI 17 --> &B277

B266		133 009	85 09	STA &09
B268		032 160 141	20 A0 8D	JSR &8DA0
B26B		133 008	85 08	STA &08
B26D		032 160 141	20 A0 8D	JSR &8DA0
B270	7	228 055	E4 37	CPX &37
B272		152	98	TYA
B273	8	229 056	E5 38	SBC &38
B275		176 218	B0 DA	BCS -38 --> &B251
B277	`	096	60	RTS
B278		162 255	A2 FF	LDX#&FF
B27A	(	134 040	86 28	STX &28
B27C		154	9A	TXS
B27D		232	E8	INX
B27E		160 000	A0 00	LDY#&00
B280		169 218	A9 DA	LDA#&DA
B282		032 244 255	20 F4 FF	JSR &FFF4
B285	~	169 126	A9 7E	LDA#&7E
B287		032 244 255	20 F4 FF	JSR &FFF4
B28A	7	032 055 178	20 37 B2	JSR &B237
B28D	d	100 032	64 20	STZ &20
B28F		178 253	B2 FD	LDA (&FD)
B291		208 003	D0 03	BNE 3 --> &B296
B293		032 166 178	20 A6 B2	JSR &B2A6
B296		165 022	A5 16	LDA &16
B298		133 011	85 0B	STA &0B
B29A		165 023	A5 17	LDA &17
B29C		133 012	85 0C	STA &0C
B29E	d	100 010	64 0A	STZ &0A
B2A0		032 207 187	20 CF BB	JSR &BBCF
B2A3	L	076 011 144	4C 0B 90	JMP &900B
B2A6		169 175	A9 AF	LDA#&AF
B2A8		133 022	85 16	STA &16
B2AA		169 178	A9 B2	LDA#&B2
B2AC		133 023	85 17	STA &17
B2AE	`	096	60	RTS

B2AF	:	246 058	F6 3A	INC &3A,X
B2B1		231	E7	xxx Invalid Code
B2B2	"	158 241 034	9E F1 22	STZ &22F1,X
B2B5	at	032 097 116	20 61 74	JSR &7461
B2B8	li	032 108 105	20 6C 69	JSR &696C
B2BB	ne	110 101 032	6E 65 20	ROR &2065
B2BE	"	034	22	xxx Invalid Code
B2BF	;	059	3B	xxx Invalid Code
B2C0	:	158 058 224	9E 3A E0	STZ &E03A,X
B2C3		139	8B	xxx Invalid Code
B2C4	:	241 058	F1 3A	SBC (&3A),Y
B2C6		224 013	E0 0D	CPX#&0D
B2C8	o	032 111 146	20 6F 92	JSR &926F
B2CB		162 003	A2 03	LDX#&03
B2CD	*	165 042	A5 2A	LDA &2A
B2CF	H	072	48	PHA
B2D0	+	165 043	A5 2B	LDA &2B
B2D2	H	072	48	PHA
B2D3		218	DA	PHX
B2D4		032 172 150	20 AC 96	JSR &96AC
B2D7		250	FA	PLX
B2D8		202	CA	DEX
B2D9		208 242	D0 F2	BNE -14 --> &B2CD
B2DB		032 150 155	20 96 9B	JSR &9B96
B2DE	*	165 042	A5 2A	LDA &2A
B2E0	=	133 061	85 3D	STA &3D
B2E2	+	165 043	A5 2B	LDA &2B
B2E4	>	133 062	85 3E	STA &3E
B2E6		160 007	A0 07	LDY#&07
B2E8		162 005	A2 05	LDX#&05
B2EA		128 027	80 1B	BRA 27 --> &B307
B2EC	o	032 111 146	20 6F 92	JSR &926F
B2EF		162 013	A2 0D	LDX#&0D
B2F1	*	165 042	A5 2A	LDA &2A
B2F3	H	072	48	PHA

B2F4		218	DA	PHX
B2F5		032 172 150	20 AC 96	JSR &96AC
B2F8		250	FA	PLX
B2F9		202	CA	DEX
B2FA		208 245	D0 F5	BNE -11 --> &B2F1
B2FC		032 150 155	20 96 9B	JSR &9B96
B2FF	*	165 042	A5 2A	LDA &2A
B301	D	133 068	85 44	STA &44
B303		162 012	A2 0C	LDX#&0C
B305		160 008	A0 08	LDY#&08
B307	h	104	68	PLA
B308	7	149 055	95 37	STA &37,X
B30A		202	CA	DEX
B30B		016 250	10 FA	BPL -6 --> &B307
B30D		152	98	TYA
B30E	7	162 055	A2 37	LDX#&37
B310		160 000	A0 00	LDY#&00
B312		032 241 255	20 F1 FF	JSR &FFF1
B315		128 011	80 0B	BRA 11 --> &B322
B317	o	032 111 146	20 6F 92	JSR &926F
B31A		032 150 155	20 96 9B	JSR &9B96
B31D	*	164 042	A4 2A	LDY &2A
B31F		136	88	DEY
B320	#	132 035	84 23	STY &23
B322	L	076 005 144	4C 05 90	JMP &9005
B325	L	076 146 144	4C 92 90	JMP &9092
B328	;	032 059 157	20 3B 9D	JSR &9D3B
B32B	z	122	7A	PLY
B32C		250	FA	PLX
B32D	h	104	68	PLA
B32E	9	133 057	85 39	STA &39
B330	h	104	68	PLA
B331	8	133 056	85 38	STA &38
B333	h	104	68	PLA
B334	7	133 055	85 37	STA &37

B336		218	DA	PHX
B337	Z	090	5A	PHY
B338	9	165 057	A5 39	LDA &39
B33A		201 005	C9 05	CMP#&05
B33C	"	240 034	F0 22	BEQ 34 --> &B360
B33E	'	165 039	A5 27	LDA &27
B340		240 227	F0 E3	BEQ -29 --> &B325
B342		016 003	10 03	BPL 3 --> &B347
B344		032 195 150	20 C3 96	JSR &96C3
B347	*	165 042	A5 2A	LDA &2A
B349	7	146 055	92 37	STA (&37)
B34B	9	165 057	A5 39	LDA &39
B34D		240 016	F0 10	BEQ 16 --> &B35F
B34F	+	165 043	A5 2B	LDA &2B
B351		160 001	A0 01	LDY#&01
B353	7	145 055	91 37	STA (&37),Y
B355	,	165 044	A5 2C	LDA &2C
B357		200	C8	INY
B358	7	145 055	91 37	STA (&37),Y
B35A	-	165 045	A5 2D	LDA &2D
B35C		200	C8	INY
B35D	7	145 055	91 37	STA (&37),Y
B35F	`	096	60	RTS
B360	'	165 039	A5 27	LDA &27
B362		240 193	F0 C1	BEQ -63 --> &B325
B364	0	048 003	30 03	BMI 3 --> &B369
B366		032 133 129	20 85 81	JSR &8185
B369	0	165 048	A5 30	LDA &30
B36B	7	146 055	92 37	STA (&37)
B36D		160 001	A0 01	LDY#&01
B36F	.	165 046	A5 2E	LDA &2E
B371	E1	069 049	45 31	EOR &31
B373	)	041 128	29 80	AND#&80
B375	E1	069 049	45 31	EOR &31
B377	7	145 055	91 37	STA (&37),Y

B379		200	C8	INY
B37A	2	165 050	A5 32	LDA &32
B37C	7	145 055	91 37	STA (&37),Y
B37E		200	C8	INY
B37F	3	165 051	A5 33	LDA &33
B381	7	145 055	91 37	STA (&37),Y
B383		200	C8	INY
B384	4	165 052	A5 34	LDA &34
B386	7	145 055	91 37	STA (&37),Y
B388	`	096	60	RTS
B389	ED	069 068	45 44	EOR &44
B38B	IT	073 084	49 54	EOR#&54
B38D	12	032 049 050	20 31 32	JSR &3231
B390	,2	044 050 013	2C 32 0D	BIT &0D32
B393		032 172 187	20 AC BB	JSR &BBAC
B396		169 128	A9 80	LDA#&80
B398		133 031	85 1F	STA &1F
B39A	d;	100 059	64 3B	STZ &3B
B39C	d<	100 060	64 3C	STZ &3C
B39E		032 232 171	20 E8 AB	JSR &ABE8
B3A1		032 030 155	20 1E 9B	JSR &9B1E
B3A4		008	08	PHP
B3A5	&	032 038 188	20 26 BC	JSR &BC26
B3A8		032 219 171	20 DB AB	JSR &ABDB
B3AB	F+	070 043	46 2B	LSR &2B
B3AD	(	040	28	PLP
B3AE		144 015	90 0F	BCC 15 --> &B3BF
B3B0		032 229 140	20 E5 8C	JSR &8CE5
B3B3		240 017	F0 11	BEQ 17 --> &B3C6
B3B5		032 230 188	20 E6 BC	JSR &BCE6
B3B8	&	032 038 188	20 26 BC	JSR &BC26
B3BB		198 010	C6 0A	DEC &0A
B3BD		128 010	80 0A	BRA 10 --> &B3C9
B3BF		032 229 140	20 E5 8C	JSR &8CE5
B3C2		240 002	F0 02	BEQ 2 --> &B3C6

B3C4		198 010	C6 0A	DEC &0A
B3C6		032 030 155	20 1E 9B	JSR &9B1E
B3C9	1	162 049	A2 31	LDX#&31
B3CB		032 198 189	20 C6 BD	JSR &BDC6
B3CE		032 224 142	20 E0 8E	JSR &8EE0
B3D1		201 231	C9 E7	CMP#&E7
B3D3		208 030	D0 1E	BNE 30 --> &B3F3
B3D5		032 224 142	20 E0 8E	JSR &8EE0
B3D8		032 188 155	20 BC 9B	JSR &9BBC
B3DB		128 025	80 19	BRA 25 --> &B3F6
B3DD		200	C8	INY
B3DE		177 011	B1 0B	LDA (&0B),Y
B3E0	O	201 079	C9 4F	CMP#&4F
B3E2		208 182	D0 B6	BNE -74 --> &B39A
B3E4		230 010	E6 0A	INC &0A
B3E6	o	032 111 146	20 6F 92	JSR &926F
B3E9		032 166 155	20 A6 9B	JSR &9BA6
B3EC	*	165 042	A5 2A	LDA &2A
B3EE		133 031	85 1F	STA &1F
B3F0	L	076 134 143	4C 86 8F	JMP &8F86
B3F3		032 176 155	20 B0 9B	JSR &9BB0
B3F6		165 011	A5 0B	LDA &0B
B3F8		133 025	85 19	STA &19
B3FA		032 229 189	20 E5 BD	JSR &BDE5
B3FD		032 230 188	20 E6 BC	JSR &BCE6
B400		032 205 128	20 CD 80	JSR &80CD
B403	=	165 061	A5 3D	LDA &3D
B405		133 011	85 0B	STA &0B
B407	>	165 062	A5 3E	LDA &3E
B409		133 012	85 0C	STA &0C
B40B		176 027	B0 1B	BCS 27 --> &B428
B40D		136	88	DEY
B40E		128 015	80 0F	BRA 15 --> &B41F
B410		032 148 189	20 94 BD	JSR &BD94
B413	$	036 031	24 1F	BIT &1F

```
B415    0     048 005          30 05       BMI 5 --> &B41C
B417          169 010          A9 0A       LDA#&0A
B419          032 238 255      20 EE FF    JSR &FFEE
B41C          032 188 155      20 BC 9B    JSR &9BBC
B41F          177 011          B1 0B       LDA (&0B),Y
B421    +     133 043          85 2B       STA &2B
B423          200              C8          INY
B424          177 011          B1 0B       LDA (&0B),Y
B426    *     133 042          85 2A       STA &2A
B428    *     165 042          A5 2A       LDA &2A
B42A          024              18          CLC
B42B    1     229 049          E5 31       SBC &31
B42D    +     165 043          A5 2B       LDA &2B
B42F    2     229 050          E5 32       SBC &32
B431          144 011          90 0B       BCC 11 --> &B43E
B433    $     036 031          24 1F       BIT &1F
B435          016 185          10 B9       BPL -71 --> &B3F0
B437          162 137          A2 89       LDX#&89
B439          160 179          A0 B3       LDY#&B3
B43B    L     076 247 255      4C F7 FF    JMP &FFF7
B43E    dL    100 076          64 4C       STZ &4C
B440    dM    100 077          64 4D       STZ &4D
B442          160 004          A0 04       LDY#&04
B444          132 010          84 0A       STY &0A
B446          132 027          84 1B       STY &1B
B448    $;    036 059          24 3B       BIT &3B
B44A          016 002          10 02       BPL 2 --> &B44E
B44C    d;    100 059          64 3B       STZ &3B
B44E    $<    036 060          24 3C       BIT &3C
B450          016 002          10 02       BPL 2 --> &B454
B452    d<    100 060          64 3C       STZ &3C
B454          177 011          B1 0B       LDA (&0B),Y
B456          201 013          C9 0D       CMP#&0D
B458    7     240 055          F0 37       BEQ 55 --> &B491
B45A          201 244          C9 F4       CMP#&F4
```

B45C		240 006	F0 06	BEQ 6 --> &B464
B45E	"	201 034	C9 22	CMP#&22
B460		208 004	D0 04	BNE 4 --> &B466
B462	EL	069 076	45 4C	EOR &4C
B464	L	133 076	85 4C	STA &4C
B466	L	166 076	A6 4C	LDX &4C
B468		208 012	D0 0C	BNE 12 --> &B476
B46A		201 237	C9 ED	CMP#&ED
B46C		208 002	D0 02	BNE 2 --> &B470
B46E	;	198 059	C6 3B	DEC &3B
B470		201 253	C9 FD	CMP#&FD
B472		208 002	D0 02	BNE 2 --> &B476
B474	<	198 060	C6 3C	DEC &3C
B476		166 025	A6 19	LDX &19
B478		189 000 007	BD 00 07	LDA &0700,X
B47B		201 013	C9 0D	CMP#&0D
B47D		240 010	F0 0A	BEQ 10 --> &B489
B47F		209 011	D1 0B	CMP (&0B),Y
B481		208 008	D0 08	BNE 8 --> &B48B
B483		200	C8	INY
B484		232	E8	INX
B485		128 241	80 F1	BRA -15 --> &B478
B487		128 135	80 87	BRA -121 --> &B410
B489	M	133 077	85 4D	STA &4D
B48B		230 027	E6 1B	INC &1B
B48D		164 027	A4 1B	LDY &1B
B48F		128 195	80 C3	BRA -61 --> &B454
B491	M	165 077	A5 4D	LDA &4D
B493		240 135	F0 87	BEQ -121 --> &B41C
B495		032 133 160	20 85 A0	JSR &A085
B498		169 001	A9 01	LDA#&01
B49A		232	E8	INX
B49B	8	056	38	SEC
B49C		032 180 189	20 B4 BD	JSR &BDB4
B49F	;	166 059	A6 3B	LDX &3B

```
B4A1        169 002        A9 02       LDA#&02
B4A3        032 179 189    20 B3 BD    JSR &BDB3
B4A6    <   166 060        A6 3C       LDX &3C
B4A8        169 004        A9 04       LDA#&04
B4AA        032 179 189    20 B3 BD    JSR &BDB3
B4AD    dL  100 076        64 4C       STZ &4C
B4AF        164 010        A4 0A       LDY &0A
B4B1        177 011        B1 0B       LDA (&0B),Y
B4B3        201 013        C9 0D       CMP#&0D
B4B5        240 208        F0 D0       BEQ -48 --> &B487
B4B7    "   201 034        C9 22       CMP#&22
B4B9        208 012        D0 0C       BNE 12 --> &B4C7
B4BB    EL  069 076        45 4C       EOR &4C
B4BD    L   133 076        85 4C       STA &4C
B4BF    "   169 034        A9 22       LDA#&22
B4C1        032 148 189    20 94 BD    JSR &BD94
B4C4        200            C8          INY
B4C5        128 234        80 EA       BRA -22 --> &B4B1
B4C7    L   166 076        A6 4C       LDX &4C
B4C9        208 246        D0 F6       BNE -10 --> &B4C1
B4CB        201 141        C9 8D       CMP#&8D
B4CD        208 010        D0 0A       BNE 10 --> &B4D9
B4CF    *   032 042 155    20 2A 9B    JSR &9B2A
B4D2        132 010        84 0A       STY &0A
B4D4        032 129 160    20 81 A0    JSR &A081
B4D7        128 214        80 D6       BRA -42 --> &B4AF
B4D9        201 227        C9 E3       CMP#&E3
B4DB        208 002        D0 02       BNE 2 --> &B4DF
B4DD    ;   230 059        E6 3B       INC &3B
B4DF        201 245        C9 F5       CMP#&F5
B4E1        208 002        D0 02       BNE 2 --> &B4E5
B4E3    <   230 060        E6 3C       INC &3C
B4E5        201 244        C9 F4       CMP#&F4
B4E7        208 002        D0 02       BNE 2 --> &B4EB
B4E9    L   133 076        85 4C       STA &4C
```

B4EB	7	032 055 189	20 37 BD	JSR &BD37
B4EE		200	C8	INY
B4EF		128 192	80 C0	BRA -64 --> &B4B1
B4F1		032 245 152	20 F5 98	JSR &98F5
B4F4		208 009	D0 09	BNE 9 --> &B4FF
B4F6	&	166 038	A6 26	LDX &26
B4F8	8	240 056	F0 38	BEQ 56 --> &B532
B4FA	=	176 061	B0 3D	BCS 61 --> &B539
B4FC	Li	076 105 155	4C 69 9B	JMP &9B69
B4FF		176 251	B0 FB	BCS -5 --> &B4FC
B501	&	166 038	A6 26	LDX &26
B503	-	240 045	F0 2D	BEQ 45 --> &B532
B505	*	165 042	A5 2A	LDA &2A
B507		221 025 005	DD 19 05	CMP &0519,X
B50A		208 014	D0 0E	BNE 14 --> &B51A
B50C	+	165 043	A5 2B	LDA &2B
B50E		221 026 005	DD 1A 05	CMP &051A,X
B511		208 007	D0 07	BNE 7 --> &B51A
B513	,	165 044	A5 2C	LDA &2C
B515		221 027 005	DD 1B 05	CMP &051B,X
B518		240 031	F0 1F	BEQ 31 --> &B539
B51A		138	8A	TXA
B51B	8	056	38	SEC
B51C		233 015	E9 0F	SBC#&0F
B51E		170	AA	TAX
B51F	&	134 038	86 26	STX &26
B521		208 226	D0 E2	BNE -30 --> &B505
B523		000	00	BRK
B524	!	033	21	EQUB &21
B525	C	067	43	xxx Invalid Code
B526	an'	097 110 039	61 6E 27	ADC (&276E,X)
B529	t	116 032	74 20	STZ &20,X
B52B	mat	109 097 116	6D 61 74	ADC &7461
B52E	c	099	63	xxx Invalid Code
B52F	h	104	68	PLA

B530		032 227	20 E3	xxx Invalid Code
B532		000	00	BRK
B533		032	20	EQUB &20
B534	No	078 111	4E 6F	xxx Invalid Code
B536		032 227	20 E3	xxx Invalid Code
B538		000	00	EQUB &00
B539		189 025 005	BD 19 05	LDA &0519,X
B53C	*	133 042	85 2A	STA &2A
B53E		189 026 005	BD 1A 05	LDA &051A,X
B541	+	133 043	85 2B	STA &2B
B543		188 027 005	BC 1B 05	LDY &051B,X
B546		192 005	C0 05	CPY#&05
B548	v	240 118	F0 76	BEQ 118 --> &B5C0
B54A	*	178 042	B2 2A	LDA (&2A)
B54C	}	125 028 005	7D 1C 05	ADC &051C,X
B54F	*	146 042	92 2A	STA (&2A)
B551	7	133 055	85 37	STA &37
B553		160 001	A0 01	LDY#&01
B555	*	177 042	B1 2A	LDA (&2A),Y
B557	}	125 029 005	7D 1D 05	ADC &051D,X
B55A	*	145 042	91 2A	STA (&2A),Y
B55C	8	133 056	85 38	STA &38
B55E		200	C8	INY
B55F	*	177 042	B1 2A	LDA (&2A),Y
B561	}	125 030 005	7D 1E 05	ADC &051E,X
B564	*	145 042	91 2A	STA (&2A),Y
B566	9	133 057	85 39	STA &39
B568		200	C8	INY
B569	*	177 042	B1 2A	LDA (&2A),Y
B56B	}	125 031 005	7D 1F 05	ADC &051F,X
B56E	*	145 042	91 2A	STA (&2A),Y
B570		168	A8	TAY
B571	7	165 055	A5 37	LDA &37
B573	8	056	38	SEC
B574	!	253 033 005	FD 21 05	SBC &0521,X

B577	7	133 055	85 37	STA &37
B579	8	165 056	A5 38	LDA &38
B57B	"	253 034 005	FD 22 05	SBC &0522,X
B57E	7	004 055	04 37	TSB &37
B580	9	165 057	A5 39	LDA &39
B582	#	253 035 005	FD 23 05	SBC &0523,X
B585	7	004 055	04 37	TSB &37
B587		152	98	TYA
B588	$	253 036 005	FD 24 05	SBC &0524,X
B58B	7	005 055	05 37	ORA &37
B58D		240 015	F0 0F	BEQ 15 --> &B59E
B58F		152	98	TYA
B590	]	093 031 005	5D 1F 05	EOR &051F,X
B593	]$	093 036 005	5D 24 05	EOR &0524,X
B596		016 004	10 04	BPL 4 --> &B59C
B598		176 004	B0 04	BCS 4 --> &B59E
B59A		128 018	80 12	BRA 18 --> &B5AE
B59C		176 016	B0 10	BCS 16 --> &B5AE
B59E	&	188 038 005	BC 26 05	LDY &0526,X
B5A1	'	189 039 005	BD 27 05	LDA &0527,X
B5A4		132 011	84 0B	STY &0B
B5A6		133 012	85 0C	STA &0C
B5A8		032 198 155	20 C6 9B	JSR &9BC6
B5AB	L	076 011 144	4C 0B 90	JMP &900B
B5AE		138	8A	TXA
B5AF	8	056	38	SEC
B5B0		233 015	E9 0F	SBC#&0F
B5B2	&	133 038	85 26	STA &26
B5B4		164 027	A4 1B	LDY &1B
B5B6		132 010	84 0A	STY &0A
B5B8		032 229 140	20 E5 8C	JSR &8CE5
B5BB	8	208 056	D0 38	BNE 56 --> &B5F5
B5BD	L	076 241 180	4C F1 B4	JMP &B4F1
B5C0		032 199 177	20 C7 B1	JSR &B1C7
B5C3		138	8A	TXA

B5C4		024	18	CLC
B5C5	i	105 028	69 1C	ADC#&1C
B5C7	J	133 074	85 4A	STA &4A
B5C9		169 005	A9 05	LDA#&05
B5CB	K	133 075	85 4B	STA &4B
B5CD		032 141 166	20 8D A6	JSR &A68D
B5D0	*	165 042	A5 2A	LDA &2A
B5D2	7	133 055	85 37	STA &37
B5D4	+	165 043	A5 2B	LDA &2B
B5D6	8	133 056	85 38	STA &38
B5D8	i	032 105 179	20 69 B3	JSR &B369
B5DB	&	166 038	A6 26	LDX &26
B5DD		138	8A	TXA
B5DE		024	18	CLC
B5DF	i!	105 033	69 21	ADC#&21
B5E1	J	133 074	85 4A	STA &4A
B5E3		032 143 156	20 8F 9C	JSR &9C8F
B5E6		240 182	F0 B6	BEQ -74 --> &B59E
B5E8		189 029 005	BD 1D 05	LDA &051D,X
B5EB	0	048 004	30 04	BMI 4 --> &B5F1
B5ED		176 175	B0 AF	BCS -81 --> &B59E
B5EF		128 189	80 BD	BRA -67 --> &B5AE
B5F1		144 171	90 AB	BCC -85 --> &B59E
B5F3		128 185	80 B9	BRA -71 --> &B5AE
B5F5	L	076 000 144	4C 00 90	JMP &9000
B5F8		000	00	BRK
B5F9	"	034	22	EQUB &22
B5FA		227	E3	xxx Invalid Code
B5FB	va	032 118 097	20 76 61	JSR &6176
B5FE	ri	114 105	72 69	ADC (&69)
B600	abl	097 098 108	61 62 6C	ADC (&6C62,X)
B603	e	101	65	xxx Invalid Code
B604		000	00	BRK
B605	#	035	23	EQUB &23
B606	T	084	54	xxx Invalid Code

B607	o	111	6F	xxx Invalid Code
B608	o	111	6F	xxx Invalid Code
B609	ma	032 109 097	20 6D 61	JSR &616D
B60C	ny	110 121 032	6E 79 20	ROR &2079
B60F		227	E3	xxx Invalid Code
B610	s	115	73	xxx Invalid Code
B611		000	00	BRK
B612	$	036	24	EQUB &24
B613	N	078	4E	xxx Invalid Code
B614	o	111	6F	xxx Invalid Code
B615		032 184	20 B8	xxx Invalid Code
B617		000	00	EQUB &00
B618		032 174 152	20 AE 98	JSR &98AE
B61B		240 219	F0 DB	BEQ -37 --> &B5F8
B61D		176 217	B0 D9	BCS -39 --> &B5F8
B61F	C	032 067 188	20 43 BC	JSR &BC43
B622		032 134 155	20 86 9B	JSR &9B86
B625	(	032 040 179	20 28 B3	JSR &B328
B628		032 213 142	20 D5 8E	JSR &8ED5
B62B		201 184	C9 B8	CMP#&B8
B62D		208 226	D0 E2	BNE -30 --> &B611
B62F	&	164 038	A4 26	LDY &26
B631		192 150	C0 96	CPY#&96
B633		176 207	B0 CF	BCS -49 --> &B604
B635		152	98	TYA
B636	i	105 015	69 0F	ADC#&0F
B638	&	133 038	85 26	STA &26
B63A	7	165 055	A5 37	LDA &37
B63C	(	153 040 005	99 28 05	STA &0528,Y
B63F	8	165 056	A5 38	LDA &38
B641	)	153 041 005	99 29 05	STA &0529,Y
B644	9	165 057	A5 39	LDA &39
B646	*	153 042 005	99 2A 05	STA &052A,Y
B649		201 005	C9 05	CMP#&05
B64B	T	240 084	F0 54	BEQ 84 --> &B6A1

```
B64D        032 175 150     20 AF 96    JSR &96AF
B650  &     164 038         A4 26       LDY &26
B652  *     165 042         A5 2A       LDA &2A
B654  !     153 033 005     99 21 05    STA &0521,Y
B657  +     165 043         A5 2B       LDA &2B
B659  "     153 034 005     99 22 05    STA &0522,Y
B65C  ,     165 044         A5 2C       LDA &2C
B65E  #     153 035 005     99 23 05    STA &0523,Y
B661  -     165 045         A5 2D       LDA &2D
B663  $     153 036 005     99 24 05    STA &0524,Y
B666        169 001         A9 01       LDA#&01
B668        032 024 174     20 18 AE    JSR &AE18
B66B        032 213 142     20 D5 8E    JSR &8ED5
B66E        201 136         C9 88       CMP#&88
B670        208 005         D0 05       BNE 5 --> &B677
B672        032 175 150     20 AF 96    JSR &96AF
B675        164 027         A4 1B       LDY &1B
B677        132 010         84 0A       STY &0A
B679  &     164 038         A4 26       LDY &26
B67B  *     165 042         A5 2A       LDA &2A
B67D        153 028 005     99 1C 05    STA &051C,Y
B680  +     165 043         A5 2B       LDA &2B
B682        153 029 005     99 1D 05    STA &051D,Y
B685  ,     165 044         A5 2C       LDA &2C
B687        153 030 005     99 1E 05    STA &051E,Y
B68A  -     165 045         A5 2D       LDA &2D
B68C        153 031 005     99 1F 05    STA &051F,Y
B68F        032 207 155     20 CF 9B    JSR &9BCF
B692  &     164 038         A4 26       LDY &26
B694        165 011         A5 0B       LDA &0B
B696  &     153 038 005     99 26 05    STA &0526,Y
B699        165 012         A5 0C       LDA &0C
B69B  '     153 039 005     99 27 05    STA &0527,Y
B69E  L     076 011 144     4C 0B 90    JMP &900B
B6A1  ;     032 059 157     20 3B 9D    JSR &9D3B
```

B6A4		032 221 150	20 DD 96	JSR &96DD
B6A7	&	165 038	A5 26	LDA &26
B6A9		024	18	CLC
B6AA	i!	105 033	69 21	ADC#&21
B6AC	J	133 074	85 4A	STA &4A
B6AE		169 005	A9 05	LDA#&05
B6B0	K	133 075	85 4B	STA &4B
B6B2		032 025 165	20 19 A5	JSR &A519
B6B5		032 216 165	20 D8 A5	JSR &A5D8
B6B8		032 213 142	20 D5 8E	JSR &8ED5
B6BB		201 136	C9 88	CMP#&88
B6BD		208 008	D0 08	BNE 8 --> &B6C7
B6BF	;	032 059 157	20 3B 9D	JSR &9D3B
B6C2		032 221 150	20 DD 96	JSR &96DD
B6C5		164 027	A4 1B	LDY &1B
B6C7		132 010	84 0A	STY &0A
B6C9	&	165 038	A5 26	LDA &26
B6CB		024	18	CLC
B6CC	i	105 028	69 1C	ADC#&1C
B6CE	J	133 074	85 4A	STA &4A
B6D0		169 005	A9 05	LDA#&05
B6D2	K	133 075	85 4B	STA &4B
B6D4		032 025 165	20 19 A5	JSR &A519
B6D7		128 182	80 B6	BRA -74 --> &B68F
B6D9	*	032 042 184	20 2A B8	JSR &B82A
B6DC		032 166 155	20 A6 9B	JSR &9BA6
B6DF	%	164 037	A4 25	LDY &25
B6E1		192 026	C0 1A	CPY#&1A
B6E3		176 014	B0 0E	BCS 14 --> &B6F3
B6E5		165 011	A5 0B	LDA &0B
B6E7		153 204 005	99 CC 05	STA &05CC,Y
B6EA		165 012	A5 0C	LDA &0C
B6EC		153 230 005	99 E6 05	STA &05E6,Y
B6EF	%	230 037	E6 25	INC &25
B6F1	0	128 048	80 30	BRA 48 --> &B723

```
B6F3        000          00          BRK
B6F4   %    037          25          EQUB &25
B6F5   T    084          54          xxx Invalid Code
B6F6   o    111          6F          xxx Invalid Code
B6F7   o    111          6F          xxx Invalid Code
B6F8   ma   032 109 097  20 6D 61    JSR &616D
B6FB   ny   110 121 032  6E 79 20    ROR &2079
B6FE   s    228 115      E4 73       CPX &73
B700        000          00          BRK
B701   &    038          26          EQUB &26
B702   N    078          4E          xxx Invalid Code
B703   o    111          6F          xxx Invalid Code
B704        032 228      20 E4       xxx Invalid Code
B706        000          00          EQUB &00
B707        032 166 155  20 A6 9B    JSR &9BA6
B70A   %    166 037      A6 25       LDX &25
B70C        240 242      F0 F2       BEQ -14 --> &B700
B70E   %    198 037      C6 25       DEC &25
B710        188 203 005  BC CB 05    LDY &05CB,X
B713        189 229 005  BD E5 05    LDA &05E5,X
B716        132 011      84 0B       STY &0B
B718        133 012      85 0C       STA &0C
B71A   L    076 005 144  4C 05 90    JMP &9005
B71D   *    032 042 184  20 2A B8    JSR &B82A
B720        032 166 155  20 A6 9B    JSR &9BA6
B723        165 032      A5 20       LDA &20
B725        240 003      F0 03       BEQ 3 --> &B72A
B727   K    032 075 156  20 4B 9C    JSR &9C4B
B72A        160 004      A0 04       LDY#&04
B72C        132 010      84 0A       STY &0A
B72E   =    164 061      A4 3D       LDY &3D
B730   >    165 062      A5 3E       LDA &3E
B732        132 011      84 0B       STY &0B
B734        133 012      85 0C       STA &0C
B736   L    076 011 144  4C 0B 90    JMP &900B
```

B739		032 166 155	20 A6 9B	JSR &9BA6
B73C		032 166 178	20 A6 B2	JSR &B2A6
B73F		128 217	80 D9	BRA -39 --> &B71A
B741		032 224 142	20 E0 8E	JSR &8EE0
B744		201 135	C9 87	CMP#&87
B746		240 241	F0 F1	BEQ -15 --> &B739
B748		164 010	A4 0A	LDY &0A
B74A		136	88	DEY
B74B		032 188 155	20 BC 9B	JSR &9BBC
B74E	d	100 010	64 0A	STZ &0A
B750		165 011	A5 0B	LDA &0B
B752		133 022	85 16	STA &16
B754		165 012	A5 0C	LDA &0C
B756		133 023	85 17	STA &17
B758	L	076 174 143	4C AE 8F	JMP &8FAE
B75B		032 224 142	20 E0 8E	JSR &8EE0
B75E		201 133	C9 85	CMP#&85
B760		240 223	F0 DF	BEQ -33 --> &B741
B762		198 010	C6 0A	DEC &0A
B764	o	032 111 146	20 6F 92	JSR &926F
B767		224 242	E0 F2	CPX#&F2
B769		240 009	F0 09	BEQ 9 --> &B774
B76B		200	C8	INY
B76C		224 229	E0 E5	CPX#&E5
B76E		240 004	F0 04	BEQ 4 --> &B774
B770		224 228	E0 E4	CPX#&E4
B772	v	208 118	D0 76	BNE 118 --> &B7EA
B774		218	DA	PHX
B775	+	165 043	A5 2B	LDA &2B
B777	,	005 044	05 2C	ORA &2C
B779	-	005 045	05 2D	ORA &2D
B77B	X	208 088	D0 58	BNE 88 --> &B7D5
B77D	*	198 042	C6 2A	DEC &2A
B77F	5	240 053	F0 35	BEQ 53 --> &B7B6
B781	0R	048 082	30 52	BMI 82 --> &B7D5

B783		177 011	B1 0B	LDA (&0B),Y
B785		201 013	C9 0D	CMP#&0D
B787	L	240 076	F0 4C	BEQ 76 --> &B7D5
B789	:	201 058	C9 3A	CMP#&3A
B78B	H	240 072	F0 48	BEQ 72 --> &B7D5
B78D		201 139	C9 8B	CMP#&8B
B78F	D	240 068	F0 44	BEQ 68 --> &B7D5
B791		200	C8	INY
B792	"	201 034	C9 22	CMP#&22
B794		208 004	D0 04	BNE 4 --> &B79A
B796	E+	069 043	45 2B	EOR &2B
B798	+	133 043	85 2B	STA &2B
B79A	+	166 043	A6 2B	LDX &2B
B79C		208 229	D0 E5	BNE -27 --> &B783
B79E	)	201 041	C9 29	CMP#&29
B7A0		208 002	D0 02	BNE 2 --> &B7A4
B7A2	,	198 044	C6 2C	DEC &2C
B7A4	(	201 040	C9 28	CMP#&28
B7A6		208 002	D0 02	BNE 2 --> &B7AA
B7A8	,	230 044	E6 2C	INC &2C
B7AA	,	201 044	C9 2C	CMP#&2C
B7AC		208 213	D0 D5	BNE -43 --> &B783
B7AE	,	166 044	A6 2C	LDX &2C
B7B0		208 209	D0 D1	BNE -47 --> &B783
B7B2	*	198 042	C6 2A	DEC &2A
B7B4		208 205	D0 CD	BNE -51 --> &B783
B7B6	h	104	68	PLA
B7B7		201 242	C9 F2	CMP#&F2
B7B9	H	240 072	F0 48	BEQ 72 --> &B803
B7BB		132 010	84 0A	STY &0A
B7BD		201 228	C9 E4	CMP#&E4
B7BF		240 009	F0 09	BEQ 9 --> &B7CA
B7C1	*	032 042 184	20 2A B8	JSR &B82A
B7C4		032 198 155	20 C6 9B	JSR &9BC6
B7C7	L#	076 035 183	4C 23 B7	JMP &B723

```
B7CA    *     032 042 184     20 2A B8     JSR &B82A
B7CD          164 010         A4 0A        LDY &0A
B7CF          032 029 184     20 1D B8     JSR &B81D
B7D2    L     076 220 182     4C DC B6     JMP &B6DC
B7D5    h     104             68           PLA
B7D6          177 011         B1 0B        LDA (&0B),Y
B7D8          200             C8           INY
B7D9          201 139         C9 8B        CMP#&8B
B7DB    :     240 058         F0 3A        BEQ 58 --> &B817
B7DD          201 013         C9 0D        CMP#&0D
B7DF          208 245         D0 F5        BNE -11 --> &B7D6
B7E1          000             00           BRK
B7E2    (     040             28           EQUB &28
B7E3    r     238 032 114     EE 20 72     INC &7220
B7E6    ang   097 110 103     61 6E 67     ADC (&676E,X)
B7E9    e     101             65           xxx Invalid Code
B7EA          000             00           BRK
B7EB    '     039             27           EQUB &27
B7EC    s     238 032 115     EE 20 73     INC &7320
B7EF    ynt   121 110 116     79 6E 74     ADC &746E,Y
B7F2    ax    097 120         61 78        xxx Invalid Code
B7F4          000             00           BRK
B7F5    )     041             29           EQUB &29
B7F6    N     078             4E           xxx Invalid Code
B7F7    o     111             6F           xxx Invalid Code
B7F8    su    032 115 117     20 73 75     JSR &7573
B7FB    c     099             63           xxx Invalid Code
B7FC    h     104             68           PLA
B7FD    li    032 108 105     20 6C 69     JSR &696C
B800    ne    110 101         6E 65        xxx Invalid Code
B802          000             00           EQUB &00
B803          132 027         84 1B        STY &1B
B805          032 213 142     20 D5 8E     JSR &8ED5
B808          201 242         C9 F2        CMP#&F2
B80A          208 222         D0 DE        BNE -34 --> &B7EA
```

B80C		032 025 176	20 19 B0	JSR &B019
B80F		164 027	A4 1B	LDY &1B
B811		032 029 184	20 1D B8	JSR &B81D
B814	L	076 002 144	4C 02 90	JMP &9002
B817		132 010	84 0A	STY &0A
B819	L)	076 041 156	4C 29 9C	JMP &9C29
B81C		200	C8	INY
B81D		177 011	B1 0B	LDA (&0B),Y
B81F		201 013	C9 0D	CMP#&0D
B821		240 004	F0 04	BEQ 4 --> &B827
B823	:	201 058	C9 3A	CMP#&3A
B825		208 245	D0 F5	BNE -11 --> &B81C
B827		132 010	84 0A	STY &0A
B829	`	096	60	RTS
B82A		032 030 155	20 1E 9B	JSR &9B1E
B82D		176 007	B0 07	BCS 7 --> &B836
B82F	o	032 111 146	20 6F 92	JSR &926F
B832		169 128	A9 80	LDA#&80
B834	+	020 043	14 2B	TRB &2B
B836		032 205 128	20 CD 80	JSR &80CD
B839		144 185	90 B9	BCC -71 --> &B7F4
B83B	`	096	60	RTS
B83C	L	076 146 144	4C 92 90	JMP &9092
B83F	Li	076 105 155	4C 69 9B	JMP &9B69
B842		132 010	84 0A	STY &0A
B844	L	076 002 144	4C 02 90	JMP &9002
B847	<	032 060 186	20 3C BA	JSR &BA3C
B84A	L	132 076	84 4C	STY &4C
B84C	u	032 117 146	20 75 92	JSR &9275
B84F		032 229 140	20 E5 8C	JSR &8CE5
B852		208 238	D0 EE	BNE -18 --> &B842
B854	L	165 076	A5 4C	LDA &4C
B856	H	072	48	PHA
B857		032 174 152	20 AE 98	JSR &98AE
B85A		240 227	F0 E3	BEQ -29 --> &B83F

B85C	u	032 117 146	20 75 92	JSR &9275
B85F	h	104	68	PLA
B860	L	133 076	85 4C	STA &4C
B862		008	08	PHP
B863	&	032 038 188	20 26 BC	JSR &BC26
B866	L	164 076	A4 4C	LDY &4C
B868		032 215 255	20 D7 FF	JSR &FFD7
B86B	'	133 039	85 27	STA &27
B86D	(	040	28	PLP
B86E		144 026	90 1A	BCC 26 --> &B88A
B870	'	165 039	A5 27	LDA &27
B872		208 200	D0 C8	BNE -56 --> &B83C
B874		032 215 255	20 D7 FF	JSR &FFD7
B877	6	133 054	85 36	STA &36
B879		170	AA	TAX
B87A		240 009	F0 09	BEQ 9 --> &B885
B87C		032 215 255	20 D7 FF	JSR &FFD7
B87F		157 255 005	9D FF 05	STA &05FF,X
B882		202	CA	DEX
B883		208 247	D0 F7	BNE -9 --> &B87C
B885		032 171 144	20 AB 90	JSR &90AB
B888		128 197	80 C5	BRA -59 --> &B84F
B88A	'	165 039	A5 27	LDA &27
B88C		240 174	F0 AE	BEQ -82 --> &B83C
B88E	0	048 012	30 0C	BMI 12 --> &B89C
B890		162 003	A2 03	LDX#&03
B892		032 215 255	20 D7 FF	JSR &FFD7
B895	*	149 042	95 2A	STA &2A,X
B897		202	CA	DEX
B898		016 248	10 F8	BPL -8 --> &B892
B89A		128 014	80 0E	BRA 14 --> &B8AA
B89C		162 004	A2 04	LDX#&04
B89E		032 215 255	20 D7 FF	JSR &FFD7
B8A1	l	157 108 004	9D 6C 04	STA &046C,X
B8A4		202	CA	DEX

B8A5		016 247	10 F7	BPL -9 --> &B89E
B8A7	9	032 057 165	20 39 A5	JSR &A539
B8AA		032 006 189	20 06 BD	JSR &BD06
B8AD	8	032 056 179	20 38 B3	JSR &B338
B8B0		128 157	80 9D	BRA -99 --> &B84F
B8B2	h	104	68	PLA
B8B3	h	104	68	PLA
B8B4		128 142	80 8E	BRA -114 --> &B844
B8B6		032 223 140	20 DF 8C	JSR &8CDF
B8B9		240 140	F0 8C	BEQ -116 --> &B847
B8BB		201 134	C9 86	CMP#&86
B8BD		240 003	F0 03	BEQ 3 --> &B8C2
B8BF		198 010	C6 0A	DEC &0A
B8C1		024	18	CLC
B8C2	fL	102 076	66 4C	ROR &4C
B8C4	FL	070 076	46 4C	LSR &4C
B8C6		169 255	A9 FF	LDA#&FF
B8C8	M	133 077	85 4D	STA &4D
B8CA		032 153 146	20 99 92	JSR &9299
B8CD		176 010	B0 0A	BCS 10 --> &B8D9
B8CF		032 153 146	20 99 92	JSR &9299
B8D2		144 251	90 FB	BCC -5 --> &B8CF
B8D4		162 255	A2 FF	LDX#&FF
B8D6	M	134 077	86 4D	STX &4D
B8D8		024	18	CLC
B8D9		008	08	PHP
B8DA	L	006 076	06 4C	ASL &4C
B8DC	(	040	28	PLP
B8DD	fL	102 076	66 4C	ROR &4C
B8DF	,	201 044	C9 2C	CMP#&2C
B8E1		240 231	F0 E7	BEQ -25 --> &B8CA
B8E3	;	201 059	C9 3B	CMP#&3B
B8E5		240 227	F0 E3	BEQ -29 --> &B8CA
B8E7		198 010	C6 0A	DEC &0A
B8E9	L	165 076	A5 4C	LDA &4C

B8EB	H	072	48	PHA
B8EC	M	165 077	A5 4D	LDA &4D
B8EE	H	072	48	PHA
B8EF		032 174 152	20 AE 98	JSR &98AE
B8F2		240 190	F0 BE	BEQ -66 --> &B8B2
B8F4	h	104	68	PLA
B8F5	M	133 077	85 4D	STA &4D
B8F7	h	104	68	PLA
B8F8	L	133 076	85 4C	STA &4C
B8FA	u	032 117 146	20 75 92	JSR &9275
B8FD		008	08	PHP
B8FE	$L	036 076	24 4C	BIT &4C
B900	p	112 006	70 06	BVS 6 --> &B908
B902	M	165 077	A5 4D	LDA &4D
B904		201 255	C9 FF	CMP#&FF
B906		208 023	D0 17	BNE 23 --> &B91F
B908	$L	036 076	24 4C	BIT &4C
B90A		016 005	10 05	BPL 5 --> &B911
B90C	?	169 063	A9 3F	LDA#&3F
B90E		032 238 255	20 EE FF	JSR &FFEE
B911	p	032 112 186	20 70 BA	JSR &BA70
B914	6	132 054	84 36	STY &36
B916	L	006 076	06 4C	ASL &4C
B918		024	18	CLC
B919	fL	102 076	66 4C	ROR &4C
B91B	$L	036 076	24 4C	BIT &4C
B91D	p	112 025	70 19	BVS 25 --> &B938
B91F		133 027	85 1B	STA &1B
B921	d	100 025	64 19	STZ &19
B923		169 006	A9 06	LDA#&06
B925		133 026	85 1A	STA &1A
B927		032 248 172	20 F8 AC	JSR &ACF8
B92A		032 235 142	20 EB 8E	JSR &8EEB
B92D		240 006	F0 06	BEQ 6 --> &B935
B92F		201 013	C9 0D	CMP#&0D

```
B931        208 247        D0 F7       BNE -9 --> &B92A
B933        160 254        A0 FE       LDY#&FE
B935        200            C8          INY
B936   M    132 077        84 4D       STY &4D
B938   (    040            28          PLP
B939        176 011        B0 0B       BCS 11 --> &B946
B93B   C    032 067 188    20 43 BC    JSR &BC43
B93E   N    032 078 171    20 4E AB    JSR &AB4E
B941   +    032 043 179    20 2B B3    JSR &B32B
B944        128 132        80 84       BRA -124 --> &B8CA
B946   d'   100 039        64 27       STZ &27
B948        032 174 144    20 AE 90    JSR &90AE
B94B        128 247        80 F7       BRA -9 --> &B944
B94D   d=   100 061        64 3D       STZ &3D
B94F        164 024        A4 18       LDY &18
B951   >    132 062        84 3E       STY &3E
B953        032 224 142    20 E0 8E    JSR &8EE0
B956        198 010        C6 0A       DEC &0A
B958   :    201 058        C9 3A       CMP#&3A
B95A        240 011        F0 0B       BEQ 11 --> &B967
B95C        201 013        C9 0D       CMP#&0D
B95E        240 007        F0 07       BEQ 7 --> &B967
B960        201 139        C9 8B       CMP#&8B
B962        240 003        F0 03       BEQ 3 --> &B967
B964   *    032 042 184    20 2A B8    JSR &B82A
B967        032 166 155    20 A6 9B    JSR &9BA6
B96A   =    165 061        A5 3D       LDA &3D
B96C        133 028        85 1C       STA &1C
B96E   >    165 062        A5 3E       LDA &3E
B970        133 029        85 1D       STA &1D
B972   L    076 005 144    4C 05 90    JMP &9005
B975        032 229 140    20 E5 8C    JSR &8CE5
B978        240 003        F0 03       BEQ 3 --> &B97D
B97A   L    076 000 144    4C 00 90    JMP &9000
B97D        032 174 152    20 AE 98    JSR &98AE
```

B980		240 243	F0 F3	BEQ -13 --> &B975
B982		176 011	B0 0B	BCS 11 --> &B98F
B984		032 172 185	20 AC B9	JSR &B9AC
B987	C	032 067 188	20 43 BC	JSR &BC43
B98A	(	032 040 179	20 28 B3	JSR &B328
B98D		128 014	80 0E	BRA 14 --> &B99D
B98F		032 172 185	20 AC B9	JSR &B9AC
B992	&	032 038 188	20 26 BC	JSR &BC26
B995		032 248 172	20 F8 AC	JSR &ACF8
B998	'	133 039	85 27	STA &27
B99A		032 171 144	20 AB 90	JSR &90AB
B99D		024	18	CLC
B99E		165 027	A5 1B	LDA &1B
B9A0	e	101 025	65 19	ADC &19
B9A2		133 028	85 1C	STA &1C
B9A4		165 026	A5 1A	LDA &1A
B9A6	i	105 000	69 00	ADC#&00
B9A8		133 029	85 1D	STA &1D
B9AA		128 201	80 C9	BRA -55 --> &B975
B9AC	u	032 117 146	20 75 92	JSR &9275
B9AF		165 028	A5 1C	LDA &1C
B9B1		133 025	85 19	STA &19
B9B3		165 029	A5 1D	LDA &1D
B9B5		133 026	85 1A	STA &1A
B9B7	d	100 027	64 1B	STZ &1B
B9B9		032 235 142	20 EB 8E	JSR &8EEB
B9BC	X	240 088	F0 58	BEQ 88 --> &BA16
B9BE		201 220	C9 DC	CMP#&DC
B9C0	T	240 084	F0 54	BEQ 84 --> &BA16
B9C2		201 013	C9 0D	CMP#&0D
B9C4		240 009	F0 09	BEQ 9 --> &B9CF
B9C6		032 235 142	20 EB 8E	JSR &8EEB
B9C9	K	240 075	F0 4B	BEQ 75 --> &BA16
B9CB		201 013	C9 0D	CMP#&0D
B9CD		208 247	D0 F7	BNE -9 --> &B9C6

B9CF		164 027	A4 1B	LDY &1B
B9D1		177 025	B1 19	LDA (&19),Y
B9D3	0	048 028	30 1C	BMI 28 --> &B9F1
B9D5		200	C8	INY
B9D6		200	C8	INY
B9D7		177 025	B1 19	LDA (&19),Y
B9D9		170	AA	TAX
B9DA		200	C8	INY
B9DB		177 025	B1 19	LDA (&19),Y
B9DD		201 032	C9 20	CMP#&20
B9DF		240 249	F0 F9	BEQ -7 --> &B9DA
B9E1		201 220	C9 DC	CMP#&DC
B9E3	.	240 046	F0 2E	BEQ 46 --> &BA13
B9E5		138	8A	TXA
B9E6		024	18	CLC
B9E7	e	101 025	65 19	ADC &19
B9E9		133 025	85 19	STA &19
B9EB		144 226	90 E2	BCC -30 --> &B9CF
B9ED		230 026	E6 1A	INC &1A
B9EF		128 222	80 DE	BRA -34 --> &B9CF
B9F1		000	00	BRK
B9F2	*	042	2A	EQUB &2A
B9F3	O	079	4F	xxx Invalid Code
B9F4	ut	117 116	75 74	ADC &74,X
B9F6	of	032 111 102	20 6F 66	JSR &666F
B9F9		032 220	20 DC	xxx Invalid Code
B9FB		000	00	BRK
B9FC	+	043	2B	EQUB &2B
B9FD	No	078 111 032	4E 6F 20	LSR &206F
BA00		245	F5	xxx Invalid Code
BA01		000	00	BRK
BA02	-	045	2D	EQUB &2D
BA03	#	141 035	8D 23	xxx Invalid Code
BA05		000	00	BRK
BA06	,	044	2C	EQUB &2C

BA07	To	084 111	54 6F	xxx Invalid Code
BA09	o	111	6F	xxx Invalid Code
BA0A	ma	032 109 097	20 6D 61	JSR &616D
BA0D	ny	110 121 032	6E 79 20	ROR &2079
BA10	s	245 115	F5 73	SBC &73,X
BA12		000	00	EQUB &00
BA13		200	C8	INY
BA14		132 027	84 1B	STY &1B
BA16	`	096	60	RTS
BA17	/	032 047 157	20 2F 9D	JSR &9D2F
BA1A		032 145 155	20 91 9B	JSR &9B91
BA1D		032 188 150	20 BC 96	JSR &96BC
BA20	$	166 036	A6 24	LDX &24
BA22		240 215	F0 D7	BEQ -41 --> &B9FB
BA24	*	165 042	A5 2A	LDA &2A
BA26	+	005 043	05 2B	ORA &2B
BA28	,	005 044	05 2C	ORA &2C
BA2A	-	005 045	05 2D	ORA &2D
BA2C		240 005	F0 05	BEQ 5 --> &BA33
BA2E	$	198 036	C6 24	DEC &24
BA30	L	076 005 144	4C 05 90	JMP &9005
BA33		188 255 004	BC FF 04	LDY &04FF,X
BA36		189 019 005	BD 13 05	LDA &0513,X
BA39	L2	076 050 183	4C 32 B7	JMP &B732
BA3C		198 010	C6 0A	DEC &0A
BA3E		165 010	A5 0A	LDA &0A
BA40		133 027	85 1B	STA &1B
BA42		165 011	A5 0B	LDA &0B
BA44		133 025	85 19	STA &19
BA46		165 012	A5 0C	LDA &0C
BA48		133 026	85 1A	STA &1A
BA4A		032 213 142	20 D5 8E	JSR &8ED5
BA4D	#	201 035	C9 23	CMP#&23
BA4F		208 176	D0 B0	BNE -80 --> &BA01
BA51		032 180 150	20 B4 96	JSR &96B4

BA54	*	164 042	A4 2A	LDY &2A
BA56		152	98	TYA
BA57	`	096	60	RTS
BA58	$	166 036	A6 24	LDX &24
BA5A		224 020	E0 14	CPX#&14
BA5C		176 167	B0 A7	BCS -89 --> &BA05
BA5E		032 188 155	20 BC 9B	JSR &9BBC
BA61		165 011	A5 0B	LDA &0B
BA63		157 000 005	9D 00 05	STA &0500,X
BA66		165 012	A5 0C	LDA &0C
BA68		157 020 005	9D 14 05	STA &0514,X
BA6B	$	230 036	E6 24	INC &24
BA6D	L	076 011 144	4C 0B 90	JMP &900B
BA70		169 006	A9 06	LDA#&06
BA72		128 002	80 02	BRA 2 --> &BA76
BA74		169 007	A9 07	LDA#&07
BA76	d7	100 055	64 37	STZ &37
BA78	8	133 056	85 38	STA &38
BA7A		169 238	A9 EE	LDA#&EE
BA7C	9	133 057	85 39	STA &39
BA7E		169 032	A9 20	LDA#&20
BA80	:	133 058	85 3A	STA &3A
BA82		160 255	A0 FF	LDY#&FF
BA84	;	132 059	84 3B	STY &3B
BA86		200	C8	INY
BA87	7	162 055	A2 37	LDX#&37
BA89		152	98	TYA
BA8A		032 241 255	20 F1 FF	JSR &FFF1
BA8D		144 006	90 06	BCC 6 --> &BA95
BA8F	L}	076 125 155	4C 7D 9B	JMP &9B7D
BA92		032 231 255	20 E7 FF	JSR &FFE7
BA95	d	100 030	64 1E	STZ &1E
BA97	`	096	60	RTS
BA98		032 205 128	20 CD 80	JSR &80CD
BA9B	M	144 077	90 4D	BCC 77 --> &BAEA

```
BA9D    =    165 061        A5 3D      LDA &3D
BA9F    7    133 055        85 37      STA &37
BAA1         133 018        85 12      STA &12
BAA3    >    165 062        A5 3E      LDA &3E
BAA5    8    133 056        85 38      STA &38
BAA7         133 019        85 13      STA &13
BAA9         160 003        A0 03      LDY#&03
BAAB    7    177 055        B1 37      LDA (&37),Y
BAAD         024            18         CLC
BAAE    e7   101 055        65 37      ADC &37
BAB0    7    133 055        85 37      STA &37
BAB2         144 002        90 02      BCC 2 --> &BAB6
BAB4    8    230 056        E6 38      INC &38
BAB6         160 000        A0 00      LDY#&00
BAB8    7    177 055        B1 37      LDA (&37),Y
BABA         145 018        91 12      STA (&12),Y
BABC         201 013        C9 0D      CMP#&0D
BABE         208 019        D0 13      BNE 19 --> &BAD3
BAC0         200            C8         INY
BAC1         208 004        D0 04      BNE 4 --> &BAC7
BAC3    8    230 056        E6 38      INC &38
BAC5         230 019        E6 13      INC &13
BAC7    7    177 055        B1 37      LDA (&37),Y
BAC9         145 018        91 12      STA (&12),Y
BACB    0    048 015        30 0F      BMI 15 --> &BADC
BACD         032 223 186    20 DF BA   JSR &BADF
BAD0         032 223 186    20 DF BA   JSR &BADF
BAD3         200            C8         INY
BAD4         208 226        D0 E2      BNE -30 --> &BAB8
BAD6    8    230 056        E6 38      INC &38
BAD8         230 019        E6 13      INC &13
BADA         128 220        80 DC      BRA -36 --> &BAB8
BADC    L    076 005 190    4C 05 BE   JMP &BE05
BADF         200            C8         INY
BAE0         208 004        D0 04      BNE 4 --> &BAE6
```

BAE2		230 019	E6 13	INC &13
BAE4	8	230 056	E6 38	INC &38
BAE6	7	177 055	B1 37	LDA (&37),Y
BAE8		145 018	91 12	STA (&12),Y
BAEA	`	096	60	RTS
BAEB		162 255	A2 FF	LDX#&FF
BAED	(	134 040	86 28	STX &28
BAEF	<	134 060	86 3C	STX &3C
BAF1		032 207 187	20 CF BB	JSR &BBCF
BAF4		165 011	A5 0B	LDA &0B
BAF6	7	133 055	85 37	STA &37
BAF8		165 012	A5 0C	LDA &0C
BAFA	8	133 056	85 38	STA &38
BAFC	d;	100 059	64 3B	STZ &3B
BAFE	d	100 010	64 0A	STZ &0A
BB00		032 178 141	20 B2 8D	JSR &8DB2
BB03		032 030 155	20 1E 9B	JSR &9B1E
BB06		144 226	90 E2	BCC -30 --> &BAEA
BB08		165 031	A5 1F	LDA &1F
BB0A		240 009	F0 09	BEQ 9 --> &BB15
BB0C		185 000 007	B9 00 07	LDA &0700,Y
BB0F		200	C8	INY
BB10		201 032	C9 20	CMP#&20
BB12		240 248	F0 F8	BEQ -8 --> &BB0C
BB14		136	88	DEY
BB15	;	132 059	84 3B	STY &3B
BB17		032 152 186	20 98 BA	JSR &BA98
BB1A		160 007	A0 07	LDY#&07
BB1C	<	132 060	84 3C	STY &3C
BB1E		160 000	A0 00	LDY#&00
BB20		169 013	A9 0D	LDA#&0D
BB22	;	210 059	D2 3B	CMP (&3B)
BB24		240 196	F0 C4	BEQ -60 --> &BAEA
BB26		200	C8	INY
BB27	;	209 059	D1 3B	CMP (&3B),Y

BB29		208 251	D0 FB	BNE -5 --> &BB26
BB2B		169 032	A9 20	LDA#&20
BB2D		136	88	DEY
BB2E		240 004	F0 04	BEQ 4 --> &BB34
BB30	;	209 059	D1 3B	CMP (&3B),Y
BB32		240 249	F0 F9	BEQ -7 --> &BB2D
BB34		200	C8	INY
BB35		169 013	A9 0D	LDA#&0D
BB37	;	145 059	91 3B	STA (&3B),Y
BB39		200	C8	INY
BB3A		200	C8	INY
BB3B		200	C8	INY
BB3C		200	C8	INY
BB3D	?	132 063	84 3F	STY &3F
BB3F		165 018	A5 12	LDA &12
BB41	9	133 057	85 39	STA &39
BB43		165 019	A5 13	LDA &13
BB45	:	133 058	85 3A	STA &3A
BB47		032 004 190	20 04 BE	JSR &BE04
BB4A	7	133 055	85 37	STA &37
BB4C		165 019	A5 13	LDA &13
BB4E	8	133 056	85 38	STA &38
BB50		136	88	DEY
BB51		165 006	A5 06	LDA &06
BB53		197 018	C5 12	CMP &12
BB55		165 007	A5 07	LDA &07
BB57		229 019	E5 13	SBC &13
BB59		176 016	B0 10	BCS 16 --> &BB6B
BB5B		032 229 189	20 E5 BD	JSR &BDE5
BB5E		032 172 187	20 AC BB	JSR &BBAC
BB61		000	00	BRK
BB62		000	00	EQUB &00
BB63		134 032	86 20	STX &20
BB65	s	115	73	xxx Invalid Code
BB66	pa	112 097	70 61	BVS 97 --> &BBC9

BB68	c	099	63	xxx Invalid Code
BB69	e	101	65	xxx Invalid Code
BB6A		000	00	EQUB &00
BB6B	9	177 057	B1 39	LDA (&39),Y
BB6D	7	145 055	91 37	STA (&37),Y
BB6F		152	98	TYA
BB70		208 004	D0 04	BNE 4 --> &BB76
BB72	:	198 058	C6 3A	DEC &3A
BB74	8	198 056	C6 38	DEC &38
BB76		136	88	DEY
BB77		152	98	TYA
BB78	e9	101 057	65 39	ADC &39
BB7A	:	166 058	A6 3A	LDX &3A
BB7C		144 001	90 01	BCC 1 --> &BB7F
BB7E		232	E8	INX
BB7F	=	197 061	C5 3D	CMP &3D
BB81		138	8A	TXA
BB82	>	229 062	E5 3E	SBC &3E
BB84		176 229	B0 E5	BCS -27 --> &BB6B
BB86		160 001	A0 01	LDY#&01
BB88	+	165 043	A5 2B	LDA &2B
BB8A	=	145 061	91 3D	STA (&3D),Y
BB8C		200	C8	INY
BB8D	*	165 042	A5 2A	LDA &2A
BB8F	=	145 061	91 3D	STA (&3D),Y
BB91		200	C8	INY
BB92	?	165 063	A5 3F	LDA &3F
BB94	=	145 061	91 3D	STA (&3D),Y
BB96	8	056	38	SEC
BB97		152	98	TYA
BB98	e=	101 061	65 3D	ADC &3D
BB9A	=	133 061	85 3D	STA &3D
BB9C		144 002	90 02	BCC 2 --> &BBA0
BB9E	>	230 062	E6 3E	INC &3E
BBA0		160 255	A0 FF	LDY#&FF

BBA2		200	C8	INY
BBA3	;	177 059	B1 3B	LDA (&3B),Y
BBA5	=	145 061	91 3D	STA (&3D),Y
BBA7		201 013	C9 0D	CMP#&0D
BBA9		208 247	D0 F7	BNE -9 --> &BBA2
BBAB	`	096	60	RTS
BBAC		165 018	A5 12	LDA &12
BBAE		133 000	85 00	STA &00
BBB0		133 002	85 02	STA &02
BBB2		165 019	A5 13	LDA &13
BBB4		133 001	85 01	STA &01
BBB6		133 003	85 03	STA &03
BBB8		032 207 187	20 CF BB	JSR &BBCF
BBBB		162 016	A2 10	LDX#&10
BBBD		189 019 191	BD 13 BF	LDA &BF13,X
BBC0		157 239 007	9D EF 07	STA &07EF,X
BBC3		202	CA	DEX
BBC4		208 247	D0 F7	BNE -9 --> &BBBD
BBC6		162 128	A2 80	LDX#&80
BBC8		158 127 004	9E 7F 04	STZ &047F,X
BBCB		202	CA	DEX
BBCC		208 250	D0 FA	BNE -6 --> &BBC8
BBCE	`	096	60	RTS
BBCF		165 024	A5 18	LDA &18
BBD1		133 029	85 1D	STA &1D
BBD3		165 006	A5 06	LDA &06
BBD5		133 004	85 04	STA &04
BBD7		165 007	A5 07	LDA &07
BBD9		133 005	85 05	STA &05
BBDB		169 128	A9 80	LDA#&80
BBDD		020 031	14 1F	TRB &1F
BBDF	d$	100 036	64 24	STZ &24
BBE1	d&	100 038	64 26	STZ &26
BBE3	d%	100 037	64 25	STZ &25
BBE5	d	100 028	64 1C	STZ &1C

BBE7	`	096	60	RTS
BBE8		165 004	A5 04	LDA &04
BBEA		024	18	CLC
BBEB	J	133 074	85 4A	STA &4A
BBED	i	105 005	69 05	ADC#&05
BBEF		133 004	85 04	STA &04
BBF1		165 005	A5 05	LDA &05
BBF3	K	133 075	85 4B	STA &4B
BBF5	i	105 000	69 00	ADC#&00
BBF7		133 005	85 05	STA &05
BBF9	`	096	60	RTS
BBFA		165 004	A5 04	LDA &04
BBFC	8	056	38	SEC
BBFD		233 005	E9 05	SBC#&05
BBFF		032 030 189	20 1E BD	JSR &BD1E
BC02	0	165 048	A5 30	LDA &30
BC04		146 004	92 04	STA (&04)
BC06		160 001	A0 01	LDY#&01
BC08	.	165 046	A5 2E	LDA &2E
BC0A	E1	069 049	45 31	EOR &31
BC0C	)	041 128	29 80	AND#&80
BC0E	E1	069 049	45 31	EOR &31
BC10		145 004	91 04	STA (&04),Y
BC12		200	C8	INY
BC13	2	165 050	A5 32	LDA &32
BC15		145 004	91 04	STA (&04),Y
BC17		200	C8	INY
BC18	3	165 051	A5 33	LDA &33
BC1A		145 004	91 04	STA (&04),Y
BC1C		200	C8	INY
BC1D	4	165 052	A5 34	LDA &34
BC1F		145 004	91 04	STA (&04),Y
BC21	`	096	60	RTS
BC22	-	240 045	F0 2D	BEQ 45 --> &BC51
BC24	0	048 212	30 D4	BMI -44 --> &BBFA

BC26		165 004	A5 04	LDA &04
BC28	8	056	38	SEC
BC29		233 004	E9 04	SBC#&04
BC2B		032 030 189	20 1E BD	JSR &BD1E
BC2E		160 003	A0 03	LDY#&03
BC30	-	165 045	A5 2D	LDA &2D
BC32		145 004	91 04	STA (&04),Y
BC34		136	88	DEY
BC35	,	165 044	A5 2C	LDA &2C
BC37		145 004	91 04	STA (&04),Y
BC39		136	88	DEY
BC3A	+	165 043	A5 2B	LDA &2B
BC3C		145 004	91 04	STA (&04),Y
BC3E	*	165 042	A5 2A	LDA &2A
BC40		146 004	92 04	STA (&04)
BC42	`	096	60	RTS
BC43	z	122	7A	PLY
BC44		250	FA	PLX
BC45	*	165 042	A5 2A	LDA &2A
BC47	H	072	48	PHA
BC48	+	165 043	A5 2B	LDA &2B
BC4A	H	072	48	PHA
BC4B	,	165 044	A5 2C	LDA &2C
BC4D	H	072	48	PHA
BC4E		218	DA	PHX
BC4F	Z	090	5A	PHY
BC50	`	096	60	RTS
BC51		024	18	CLC
BC52		165 004	A5 04	LDA &04
BC54	6	229 054	E5 36	SBC &36
BC56		032 030 189	20 1E BD	JSR &BD1E
BC59	6	164 054	A4 36	LDY &36
BC5B		240 008	F0 08	BEQ 8 --> &BC65
BC5D		185 255 005	B9 FF 05	LDA &05FF,Y
BC60		145 004	91 04	STA (&04),Y

```
BC62        136         88          DEY
BC63        208 248     D0 F8       BNE -8 --> &BC5D
BC65  6     165 054     A5 36       LDA &36
BC67        146 004     92 04       STA (&04)
BC69  `     096         60          RTS
BC6A  9     165 057     A5 39       LDA &39
BC6C        201 128     C9 80       CMP#&80
BC6E  %     240 037     F0 25       BEQ 37 --> &BC95
BC70  8     144 056     90 38       BCC 56 --> &BCAA
BC72        178 004     B2 04       LDA (&04)
BC74        170         AA          TAX
BC75        240 022     F0 16       BEQ 22 --> &BC8D
BC77  7     178 055     B2 37       LDA (&37)
BC79        233 001     E9 01       SBC#&01
BC7B  9     133 057     85 39       STA &39
BC7D        160 001     A0 01       LDY#&01
BC7F  7     177 055     B1 37       LDA (&37),Y
BC81        233 000     E9 00       SBC#&00
BC83  :     133 058     85 3A       STA &3A
BC85        177 004     B1 04       LDA (&04),Y
BC87  9     145 057     91 39       STA (&39),Y
BC89        200         C8          INY
BC8A        202         CA          DEX
BC8B        208 248     D0 F8       BNE -8 --> &BC85
BC8D        178 004     B2 04       LDA (&04)
BC8F        160 003     A0 03       LDY#&03
BC91  7     145 055     91 37       STA (&37),Y
BC93  L     128 076     80 4C       BRA 76 --> &BCE1
BC95        178 004     B2 04       LDA (&04)
BC97        170         AA          TAX
BC98        240 012     F0 0C       BEQ 12 --> &BCA6
BC9A        160 001     A0 01       LDY#&01
BC9C        177 004     B1 04       LDA (&04),Y
BC9E        136         88          DEY
BC9F  7     145 055     91 37       STA (&37),Y
```

BCA1		200	C8	INY
BCA2		200	C8	INY
BCA3		202	CA	DEX
BCA4		208 246	D0 F6	BNE -10 --> &BC9C
BCA6		169 013	A9 0D	LDA#&0D
BCA8		208 231	D0 E7	BNE -25 --> &BC91
BCAA		178 004	B2 04	LDA (&04)
BCAC	7	146 055	92 37	STA (&37)
BCAE		160 004	A0 04	LDY#&04
BCB0	9	165 057	A5 39	LDA &39
BCB2		240 026	F0 1A	BEQ 26 --> &BCCE
BCB4		160 001	A0 01	LDY#&01
BCB6		177 004	B1 04	LDA (&04),Y
BCB8	7	145 055	91 37	STA (&37),Y
BCBA		200	C8	INY
BCBB		177 004	B1 04	LDA (&04),Y
BCBD	7	145 055	91 37	STA (&37),Y
BCBF		200	C8	INY
BCC0		177 004	B1 04	LDA (&04),Y
BCC2	7	145 055	91 37	STA (&37),Y
BCC4		200	C8	INY
BCC5	9	196 057	C4 39	CPY &39
BCC7		176 005	B0 05	BCS 5 --> &BCCE
BCC9		177 004	B1 04	LDA (&04),Y
BCCB	7	145 055	91 37	STA (&37),Y
BCCD		200	C8	INY
BCCE		152	98	TYA
BCCF		024	18	CLC
BCD0	+	128 043	80 2B	BRA 43 --> &BCFD
BCD2		178 004	B2 04	LDA (&04)
BCD4	6	133 054	85 36	STA &36
BCD6		240 011	F0 0B	BEQ 11 --> &BCE3
BCD8		168	A8	TAY
BCD9		177 004	B1 04	LDA (&04),Y
BCDB		153 255 005	99 FF 05	STA &05FF,Y

BCDE		136	88	DEY
BCDF		208 248	D0 F8	BNE -8 --> &BCD9
BCE1		178 004	B2 04	LDA (&04)
BCE3	8	056	38	SEC
BCE4		128 023	80 17	BRA 23 --> &BCFD
BCE6		160 003	A0 03	LDY#&03
BCE8		177 004	B1 04	LDA (&04),Y
BCEA	-	133 045	85 2D	STA &2D
BCEC		136	88	DEY
BCED		177 004	B1 04	LDA (&04),Y
BCEF	,	133 044	85 2C	STA &2C
BCF1		136	88	DEY
BCF2		177 004	B1 04	LDA (&04),Y
BCF4	+	133 043	85 2B	STA &2B
BCF6		178 004	B2 04	LDA (&04)
BCF8	*	133 042	85 2A	STA &2A
BCFA		024	18	CLC
BCFB		169 004	A9 04	LDA#&04
BCFD	e	101 004	65 04	ADC &04
BCFF		133 004	85 04	STA &04
BD01		144 002	90 02	BCC 2 --> &BD05
BD03		230 005	E6 05	INC &05
BD05	`	096	60	RTS
BD06	7	162 055	A2 37	LDX#&37
BD08		160 003	A0 03	LDY#&03
BD0A		177 004	B1 04	LDA (&04),Y
BD0C		149 003	95 03	STA &03,X
BD0E		136	88	DEY
BD0F		177 004	B1 04	LDA (&04),Y
BD11		149 002	95 02	STA &02,X
BD13		136	88	DEY
BD14		177 004	B1 04	LDA (&04),Y
BD16		149 001	95 01	STA &01,X
BD18		178 004	B2 04	LDA (&04)
BD1A		149 000	95 00	STA &00,X

BD1C		128 220	80 DC	BRA -36 --> &BCFA
BD1E		133 004	85 04	STA &04
BD20		176 002	B0 02	BCS 2 --> &BD24
BD22		198 005	C6 05	DEC &05
BD24		164 005	A4 05	LDY &05
BD26		196 003	C4 03	CPY &03
BD28		144 010	90 0A	BCC 10 --> &BD34
BD2A		208 004	D0 04	BNE 4 --> &BD30
BD2C		197 002	C5 02	CMP &02
BD2E		144 004	90 04	BCC 4 --> &BD34
BD30	`	096	60	RTS
BD31		032 172 187	20 AC BB	JSR &BBAC
BD34	L	076 161 144	4C A1 90	JMP &90A1
BD37	7	133 055	85 37	STA &37
BD39		201 128	C9 80	CMP#&80
BD3B	W	144 087	90 57	BCC 87 --> &BD94
BD3D	V	169 086	A9 56	LDA#&56
BD3F	8	133 056	85 38	STA &38
BD41		169 132	A9 84	LDA#&84
BD43	9	133 057	85 39	STA &39
BD45	Z	090	5A	PHY
BD46		160 000	A0 00	LDY#&00
BD48		200	C8	INY
BD49	8	177 056	B1 38	LDA (&38),Y
BD4B		016 251	10 FB	BPL -5 --> &BD48
BD4D	7	197 055	C5 37	CMP &37
BD4F		240 013	F0 0D	BEQ 13 --> &BD5E
BD51		200	C8	INY
BD52		152	98	TYA
BD53	8	056	38	SEC
BD54	e8	101 056	65 38	ADC &38
BD56	8	133 056	85 38	STA &38
BD58		144 236	90 EC	BCC -20 --> &BD46
BD5A	9	230 057	E6 39	INC &39
BD5C		128 232	80 E8	BRA -24 --> &BD46

BD5E		160 000	A0 00	LDY#&00
BD60	8	177 056	B1 38	LDA (&38),Y
BD62	0	048 006	30 06	BMI 6 --> &BD6A
BD64		032 148 189	20 94 BD	JSR &BD94
BD67		200	C8	INY
BD68		208 246	D0 F6	BNE -10 --> &BD60
BD6A	z	122	7A	PLY
BD6B	`	096	60	RTS
BD6C	H	072	48	PHA
BD6D	J	074	4A	LSR A
BD6E	J	074	4A	LSR A
BD6F	J	074	4A	LSR A
BD70	J	074	4A	LSR A
BD71	w	032 119 189	20 77 BD	JSR &BD77
BD74	h	104	68	PLA
BD75	)	041 015	29 0F	AND#&0F
BD77		201 010	C9 0A	CMP#&0A
BD79		144 002	90 02	BCC 2 --> &BD7D
BD7B	i	105 006	69 06	ADC#&06
BD7D	i0	105 048	69 30	ADC#&30
BD7F	H	072	48	PHA
BD80	#	165 035	A5 23	LDA &23
BD82		197 030	C5 1E	CMP &1E
BD84		176 003	B0 03	BCS 3 --> &BD89
BD86		032 146 186	20 92 BA	JSR &BA92
BD89	h	104	68	PLA
BD8A		230 030	E6 1E	INC &1E
BD8C	l	108 014 002	6C 0E 02	JMP (&020E)
BD8F	l	032 108 189	20 6C BD	JSR &BD6C
BD92		169 032	A9 20	LDA#&20
BD94	$	036 031	24 1F	BIT &1F
BD96	0	048 010	30 0A	BMI 10 --> &BDA2
BD98		201 013	C9 0D	CMP#&0D
BD9A		208 227	D0 E3	BNE -29 --> &BD7F
BD9C		032 238 255	20 EE FF	JSR &FFEE

BD9F	L	076 149 186	4C 95 BA	JMP &BA95
BDA2		146 002	92 02	STA (&02)
BDA4		230 002	E6 02	INC &02
BDA6		208 029	D0 1D	BNE 29 --> &BDC5
BDA8		230 003	E6 03	INC &03
BDAA	H	072	48	PHA
BDAB		165 003	A5 03	LDA &03
BDAD	E	069 007	45 07	EOR &07
BDAF		240 128	F0 80	BEQ -128 --> &BD31
BDB1	h	104	68	PLA
BDB2	`	096	60	RTS
BDB3		024	18	CLC
BDB4	%	037 031	25 1F	AND &1F
BDB6		240 013	F0 0D	BEQ 13 --> &BDC5
BDB8		138	8A	TXA
BDB9	0	048 010	30 0A	BMI 10 --> &BDC5
BDBB	*	042	2A	ROL A
BDBC		170	AA	TAX
BDBD		240 006	F0 06	BEQ 6 --> &BDC5
BDBF		032 146 189	20 92 BD	JSR &BD92
BDC2		202	CA	DEX
BDC3		208 250	D0 FA	BNE -6 --> &BDBF
BDC5	`	096	60	RTS
BDC6	*	165 042	A5 2A	LDA &2A
BDC8		149 000	95 00	STA &00,X
BDCA	+	165 043	A5 2B	LDA &2B
BDCC		149 001	95 01	STA &01,X
BDCE	,	165 044	A5 2C	LDA &2C
BDD0		149 002	95 02	STA &02,X
BDD2	-	165 045	A5 2D	LDA &2D
BDD4		149 003	95 03	STA &03,X
BDD6	`	096	60	RTS
BDD7	A	032 065 190	20 41 BE	JSR &BE41
BDDA	d=	100 061	64 3D	STZ &3D
BDDC		160 000	A0 00	LDY#&00

BDDE		169 255	A9 FF	LDA#&FF
BDE0	7	162 055	A2 37	LDX#&37
BDE2		032 221 255	20 DD FF	JSR &FFDD
BDE5		165 024	A5 18	LDA &18
BDE7		133 019	85 13	STA &13
BDE9	d	100 018	64 12	STZ &12
BDEB		160 001	A0 01	LDY#&01
BDED		178 018	B2 12	LDA (&12)
BDEF		201 013	C9 0D	CMP#&0D
BDF1		208 030	D0 1E	BNE 30 --> &BE11
BDF3		177 018	B1 12	LDA (&12),Y
BDF5	0	048 012	30 0C	BMI 12 --> &BE03
BDF7		160 003	A0 03	LDY#&03
BDF9		177 018	B1 12	LDA (&12),Y
BDFB		240 020	F0 14	BEQ 20 --> &BE11
BDFD		024	18	CLC
BDFE		032 006 190	20 06 BE	JSR &BE06
BE01		128 234	80 EA	BRA -22 --> &BDED
BE03		200	C8	INY
BE04		024	18	CLC
BE05		152	98	TYA
BE06	e	101 018	65 12	ADC &12
BE08		133 018	85 12	STA &12
BE0A		144 002	90 02	BCC 2 --> &BE0E
BE0C		230 019	E6 13	INC &13
BE0E		160 001	A0 01	LDY#&01
BE10	`	096	60	RTS
BE11		032 207 190	20 CF BE	JSR &BECF
BE14	Ba	013 066 097	0D 42 61	ORA &6142
BE17	d	100 032	64 20	STZ &20
BE19	pr	112 114	70 72	BVS 114 --> &BE8D
BE1B	o	111	6F	xxx Invalid Code
BE1C	g	103	67	xxx Invalid Code
BE1D	ra	114 097	72 61	ADC (&61)
BE1F	m	109 013 234	6D 0D EA	ADC &EA0D

BE22	L	076 134 143	4C 86 8F	JMP &8F86
BE25	d7	100 055	64 37	STZ &37
BE27		169 006	A9 06	LDA#&06
BE29	8	133 056	85 38	STA &38
BE2B	6	164 054	A4 36	LDY &36
BE2D		169 013	A9 0D	LDA#&0D
BE2F		153 000 006	99 00 06	STA &0600,Y
BE32	`	096	60	RTS
BE33	L	076 146 144	4C 92 90	JMP &9092
BE36	/	032 047 157	20 2F 9D	JSR &9D2F
BE39		208 248	D0 F8	BNE -8 --> &BE33
BE3B	%	032 037 190	20 25 BE	JSR &BE25
BE3E	L	076 145 155	4C 91 9B	JMP &9B91
BE41	6	032 054 190	20 36 BE	JSR &BE36
BE44		136	88	DEY
BE45	9	132 057	84 39	STY &39
BE47		165 024	A5 18	LDA &18
BE49	:	133 058	85 3A	STA &3A
BE4B		169 130	A9 82	LDA#&82
BE4D		032 244 255	20 F4 FF	JSR &FFF4
BE50	;	134 059	86 3B	STX &3B
BE52	<	132 060	84 3C	STY &3C
BE54	`	096	60	RTS
BE55		032 229 189	20 E5 BD	JSR &BDE5
BE58	A	032 065 190	20 41 BE	JSR &BE41
BE5B	?	134 063	86 3F	STX &3F
BE5D	@	132 064	84 40	STY &40
BE5F	C	134 067	86 43	STX &43
BE61	D	132 068	84 44	STY &44
BE63	G	134 071	86 47	STX &47
BE65	H	132 072	84 48	STY &48
BE67	dA	100 065	64 41	STZ &41
BE69		166 018	A6 12	LDX &12
BE6B	E	134 069	86 45	STX &45
BE6D		166 019	A6 13	LDX &13

BE6F	F	134 070	86 46	STX &46
BE71	+	162 043	A2 2B	LDX#&2B
BE73	=	134 061	86 3D	STX &3D
BE75		162 128	A2 80	LDX#&80
BE77	>	134 062	86 3E	STX &3E
BE79		166 024	A6 18	LDX &18
BE7B	B	134 066	86 42	STX &42
BE7D		169 000	A9 00	LDA#&00
BE7F		168	A8	TAY
BE80	7	162 055	A2 37	LDX#&37
BE82		032 221 255	20 DD FF	JSR &FFDD
BE85	$	128 036	80 24	BRA 36 --> &BEAB
BE87	6	032 054 190	20 36 BE	JSR &BE36
BE8A		162 000	A2 00	LDX#&00
BE8C		160 006	A0 06	LDY#&06
BE8E		032 247 255	20 F7 FF	JSR &FFF7
BE91		128 024	80 18	BRA 24 --> &BEAB
BE93		169 003	A9 03	LDA#&03
BE95		128 002	80 02	BRA 2 --> &BE99
BE97		169 001	A9 01	LDA#&01
BE99	H	072	48	PHA
BE9A	>	032 062 186	20 3E BA	JSR &BA3E
BE9D	Z	090	5A	PHY
BE9E	R	032 082 155	20 52 9B	JSR &9B52
BEA1		032 188 150	20 BC 96	JSR &96BC
BEA4	z	122	7A	PLY
BEA5	*	162 042	A2 2A	LDX#&2A
BEA7	h	104	68	PLA
BEA8		032 218 255	20 DA FF	JSR &FFDA
BEAB	L	076 005 144	4C 05 90	JMP &9005
BEAE	>	032 062 186	20 3E BA	JSR &BA3E
BEB1		032 150 155	20 96 9B	JSR &9B96
BEB4	*	164 042	A4 2A	LDY &2A
BEB6		169 000	A9 00	LDA#&00
BEB8		032 206 255	20 CE FF	JSR &FFCE

BEBB		128 238	80 EE	BRA -18 --> &BEAB
BEBD	>	032 062 186	20 3E BA	JSR &BA3E
BEC0	H	072	48	PHA
BEC1		032 172 150	20 AC 96	JSR &96AC
BEC4		032 150 155	20 96 9B	JSR &9B96
BEC7	z	122	7A	PLY
BEC8	*	165 042	A5 2A	LDA &2A
BECA		032 212 255	20 D4 FF	JSR &FFD4
BECD		128 220	80 DC	BRA -36 --> &BEAB
BECF	h	104	68	PLA
BED0	7	133 055	85 37	STA &37
BED2	h	104	68	PLA
BED3	8	133 056	85 38	STA &38
BED5		128 003	80 03	BRA 3 --> &BEDA
BED7		032 227 255	20 E3 FF	JSR &FFE3
BEDA		032 169 141	20 A9 8D	JSR &8DA9
BEDD		016 248	10 F8	BPL -8 --> &BED7
BEDF	l7	108 055 000	6C 37 00	JMP (&0037)
BEE2		169 005	A9 05	LDA#&05
BEE4		218	DA	PHX
BEE5	*	162 042	A2 2A	LDX#&2A
BEE7		160 000	A0 00	LDY#&00
BEE9		032 241 255	20 F1 FF	JSR &FFF1
BEEC		250	FA	PLX
BEED	.	165 046	A5 2E	LDA &2E
BEEF	*	230 042	E6 2A	INC &2A
BEF1		208 010	D0 0A	BNE 10 --> &BEFD
BEF3	+	230 043	E6 2B	INC &2B
BEF5		208 006	D0 06	BNE 6 --> &BEFD
BEF7	,	230 044	E6 2C	INC &2C
BEF9		208 002	D0 02	BNE 2 --> &BEFD
BEFB	-	230 045	E6 2D	INC &2D
BEFD	`	096	60	RTS
BEFE		169 013	A9 0D	LDA#&0D
BF00		164 024	A4 18	LDY &18

BF02		132 019	84 13	STY &13
BF04	d	100 018	64 12	STZ &12
BF06	d	100 032	64 20	STZ &20
BF08		146 018	92 12	STA (&12)
BF0A		169 255	A9 FF	LDA#&FF
BF0C		160 001	A0 01	LDY#&01
BF0E		145 018	91 12	STA (&12),Y
BF10		200	C8	INY
BF11		132 018	84 12	STY &12
BF13	`	096	60	RTS
BF14		184	B8	CLV
BF15		167	A7	xxx Invalid Code
BF16		238 165 166	EE A5 A6	INC &A6A5
BF19		166 141	A6 8D	LDX &8D
BF1B		166 202	A6 CA	LDX &CA
BF1D	A	172 065 165	AC 41 A5	LDY &A541
BF20	J	025 165 074	19 A5 4A	ORA &4AA5,Y
BF23	.	046 129 201	2E 81 C9	ROL &C981
BF26		016 000	10 00	BPL 0 --> &BF28
BF28		000	00	BRK
BF29	o	111	6F	xxx Invalid Code
BF2A	w	021 119	15 77	ORA &77,X
BF2C	z	122	7A	PLY
BF2D	a I	097 129 073	61 81 49	ADC (&4981,X)
BF30		015	0F	xxx Invalid Code
BF31		218	DA	PHX
BF32		162 128	A2 80	LDX#&80
BF34	"	034	22	xxx Invalid Code
BF35	n	249 131 110	F9 83 6E	SBC &6E83,Y
BF38	{	123	7B	xxx Invalid Code
BF39	5	014 250 053	0E FA 35	ASL &35FA
BF3C		018 134	12 86	ORA (&86)
BF3E	e.	101 046	65 2E	ADC &2E
BF40		224 211	E0 D3	CPX#&D3
BF42		127	7F	xxx Invalid Code

BF43	^[	094 091 216	5E 5B D8	LSR &D85B,X
BF46		170	AA	TAX
BF47		130	82	xxx Invalid Code
BF48	- T	045 248 084	2D F8 54	AND &54F8
BF4B	X	088	58	CLI
BF4C	1	128 049	80 31	BRA 49 --> &BF7F
BF4E	r	114 023	72 17	ADC (&17)
BF50		248	F8	SED
BF51		128 011	80 0B	BRA 11 --> &BF5E
BF53		215	D7	xxx Invalid Code
BF54	P)	080 041	50 29	BVC 41 --> &BF7F
BF56	| |	124 210 124	7C D2 7C	JMP (&7CD2,X)
BF59		134 005	86 05	STX &05
BF5B		128 021	80 15	BRA 21 --> &BF72
BF5D	R	082 182	52 B6	EOR (&B6)
BF5F	6|	054 124	36 7C	ROL &7C,X
BF61	6	153 152 054	99 98 36	STA &3698,Y
BF64		004 128	04 80	TSB &80
BF66	@	064	40	RTI
BF67		000	00	BRK
BF68		001 016	01 10	ORA (&10,X)
BF6A		127	7F	xxx Invalid Code
BF6B	*	042	2A	ROL A
BF6C		170	AA	TAX
BF6D		170	AA	TAX
BF6E		227	E3	xxx Invalid Code
BF6F		127	7F	xxx Invalid Code
BF70		255	FF	xxx Invalid Code
BF71		255	FF	xxx Invalid Code
BF72		255	FF	xxx Invalid Code
BF73		255	FF	xxx Invalid Code
BF74	z	122	7A	PLY
BF75		195	C3	xxx Invalid Code
BF76		030 024 190	1E 18 BE	ASL &BE18,X
BF79	s	115	73	xxx Invalid Code

BF7A	aqU	097 113 085	61 71 55	ADC (&5571,X)
BF7D	-{	045 123 140	2D 7B 8C	AND &8C7B
BF80		155	9B	xxx Invalid Code
BF81		145 136	91 88	STA (&88),Y
BF83	w	119	77	xxx Invalid Code
BF84	+	043	2B	xxx Invalid Code
BF85		164 196	A4 C4	LDY &C4
BF87	S	083	53	xxx Invalid Code
BF88	|L	124 076 204	7C 4C CC	JMP (&CC4C,X)
BF8B		202	CA	DEX
BF8C		183	B7	xxx Invalid Code
BF8D	~	126 170 170	7E AA AA	ROR &AAAA,X
BF90		170	AA	TAX
BF91		166 129	A6 81	LDX &81
BF93		000	00	BRK
BF94		000	00	BRK
BF95		000	00	BRK
BF96		000	00	BRK
BF97	}	125 163 242	7D A3 F2	ADC &F2A3,X
BF9A		239	EF	xxx Invalid Code
BF9B	D	068	44	xxx Invalid Code
BF9C	~	126 031 001	7E 1F 01	ROR &011F,X
BF9F	M	161 077	A1 4D	LDA (&4D,X)
BFA1		127	7F	xxx Invalid Code
BFA2	am	097 109 244	61 6D F4	ADC (&F46D,X)
BFA5	?	063	3F	xxx Invalid Code
BFA6	~\	126 092 145	7E 5C 91	ROR &915C,X
BFA9	#	035	23	xxx Invalid Code
BFAA	~v	172 126 118	AC 7E 76	LDY &767E
BFAD		184	B8	CLV
BFAE	}	141 026 125	8D 1A 7D	STA &7D1A
BFB1	>	029 062 171	1D 3E AB	ORA &AB3E,X
BFB4	,	044 129 009	2C 81 09	BIT &0981
BFB7	A	065 129	41 81	EOR (&81,X)
BFB9		210 128	D2 80	CMP (&80)

BFBB	t	116 223	74 DF	STZ &DF,X
BFBD		189 032 128	BD 20 80	LDA &8020,X
BFC0		131	83	xxx Invalid Code
BFC1		139	8B	xxx Invalid Code
BFC2		031	1F	xxx Invalid Code
BFC3		181 127	B5 7F	LDA &7F,X
BFC5		130	82	xxx Invalid Code
BFC6	Y	089 173 171	59 AD AB	EOR &ABAD,Y
BFC9	m	128 109	80 6D	BRA 109 --> &C038
BFCB	c	099	63	xxx Invalid Code
BFCC	8	056	38	SEC
BFCD	,}	044 125 017	2C 7D 11	BIT &117D
BFD0		212	D4	xxx Invalid Code
BFD1		177 209	B1 D1	LDA (&D1),Y
BFD3	yh	121 104 188	79 68 BC	ADC &BC68,Y
BFD6	O	079	4F	xxx Invalid Code
BFD7	Yu	089 117 005	59 75 05	EOR &0575,Y
BFDA	, 9	044 158 057	2C 9E 39	BIT &399E
BFDD	{	123	7B	xxx Invalid Code
BFDE		008	08	PHP
BFDF		136	88	DEY
BFE0	;	059	3B	xxx Invalid Code
BFE1	l	166 108	A6 6C	LDX &6C
BFE3	1	049 207	31 CF	AND (&CF,Y)
BFE5		209 140	D1 8C	CMP (&8C),Y
BFE7	}*	125 042 170	7D 2A AA	ADC &AA2A,X
BFEA		170	AA	TAX
BFEB		137 127	89 7F	BIT#&7F
BFED		255	FF	xxx Invalid Code
BFEE		255	FF	xxx Invalid Code
BFEF		255	FF	xxx Invalid Code
BFF0		232	E8	INX
BFF1		129 000	81 00	STA (&00,X)
BFF3		000	00	BRK
BFF4		000	00	BRK

BFF5		000	00	BRK
BFF6		129 000	81 00	STA (&00,X)
BFF8		000	00	BRK
BFF9		000	00	BRK
BFFA		000	00	BRK
BFFB	Ro	082 111	52 6F	EOR (&6F)
BFFD	g	103	67	xxx Invalid Code
BFFE	er	101 114	65 72	ADC &72

```
B001      165 012     A5 0C     LDA &0C
B003      145 002     91 02     STA (&02),Y
B005      200         C8        INY
B006      032 139 152 20 8B 98  JSR &988B
B009 Lr   076 114 176 4C 72 B0  JMP &B072
B00C      000         00        BRK
B00D      030         1E        EQUB &1E
B00E Ba   066 097     42 61     xxx Invalid Code
B010 d    100 032     64 20     STZ &20
B012 c    099         63        xxx Invalid Code
B013 all  097 108 108 61 6C 6C  ADC (&6C6C,X)
B016      000         00        EQUB &00
B017      169 164     A9 A4     LDA#&A4
B019  '   133 039     85 27     STA &27
B01B      186         BA        TSX
B01C      138         8A        TXA
B01D      024         18        CLC
B01E e    101 004     65 04     ADC &04
B020      032 030 189 20 1E BD  JSR &BD1E
B023      138         8A        TXA
B024      146 004     92 04     STA (&04)
B026      160 000     A0 00     LDY#&00
B028      232         E8        INX
B029      200         C8        INY
B02A      189 000 001 BD 00 01  LDA &0100,X
B02D      145 004     91 04     STA (&04),Y
B02F      224 255     E0 FF     CPX#&FF
B031      208 245     D0 F5     BNE -11   --> &B028
B033      154         9A        TXS
B034  '   165 039     A5 27     LDA &27
B036 H    072         48        PHA
B037      165 010     A5 0A     LDA &0A
B039 H    072         48        PHA
B03A      165 011     A5 0B     LDA &0B
B03C H    072         48        PHA
B03D      165 012     A5 0C     LDA &0C
B03F H    072         48        PHA
B040      165 027     A5 1B     LDA &1B
B042      170         AA        TAX
B043      024         18        CLC
B044 e    101 025     65 19     ADC &19
B046      164 026     A4 1A     LDY &1A
```

```
B048      144 002     90 02     BCC 2   --> &B04C
B04A      200         C8        INY
B04B      024         18        CLC
B04C      233 001     E9 01     SBC#&01
B04E  7   133 055     85 37     STA &37
B050      152         98        TYA
B051      233 000     E9 00     SBC#&00
B053  8   133 056     85 38     STA &38
B055      160 002     A0 02     LDY#&02
B057      032 248 154 20 F8 9A  JSR &9AF8
B05A      192 002     C0 02     CPY#&02
B05C      240 174     F0 AE     BEQ -82   --> &B00C
B05E      134 027     86 1B     STX &1B
B060  u   032 117 128 20 75 80  JSR &8075
B063      208 003     D0 03     BNE 3   --> &B068
B065 L    076 151 175 4C 97 AF  JMP &AF97
B068  *   178 042     B2 2A     LDA (&2A)
B06A      133 011     85 0B     STA &0B
B06C      160 001     A0 01     LDY#&01
B06E  *   177 042     B1 2A     LDA (&2A),Y
B070      133 012     85 0C     STA &0C
B072      169 000     A9 00     LDA#&00
B074 H    072         48        PHA
B075 d    100 010     64 0A     STZ &0A
B077      032 224 142 20 E0 8E  JSR &8EE0
B07A  (   201 040     C9 28     CMP#&28
B07C  M   240 077     F0 4D     BEQ 77   --> &B0CB
B07E      198 010     C6 0A     DEC &0A
B080      165 027     A5 1B     LDA &1B
B082 H    072         48        PHA
B083      165 025     A5 19     LDA &19
B085 H    072         48        PHA
B086      165 026     A5 1A     LDA &1A
B088 H    072         48        PHA
B089      032 011 144 20 0B 90  JSR &900B
B08C h    104         68        PLA
B08D      133 026     85 1A     STA &1A
B08F h    104         68        PLA
B090      133 025     85 19     STA &19
B092 h    104         68        PLA
B093      133 027     85 1B     STA &1B
B095 h    104         68        PLA
B096      240 012     F0 0C     BEQ 12   --> &B0A4
```

```
B098  ?   133 063     85 3F     STA &3F
B09A      032 006 189 20 06 BD  JSR &BD06
B09D  j   032 106 188 20 6A BC  JSR &BC6A
B0A0  ?   198 063     C6 3F     DEC &3F
B0A2      208 246     D0 F6     BNE -10   --> &B09A
B0A4 h    104         68        PLA
B0A5      133 012     85 0C     STA &0C
B0A7 h    104         68        PLA
B0A8      133 011     85 0B     STA &0B
B0AA h    104         68        PLA
B0AB      133 010     85 0A     STA &0A
B0AD h    104         68        PLA
B0AE      178 004     B2 04     LDA (&04)
B0B0      170         AA        TAX
B0B1      154         9A        TXS
B0B2      160 000     A0 00     LDY#&00
B0B4      200         C8        INY
B0B5      232         E8        INX
B0B6      177 004     B1 04     LDA (&04),Y
B0B8      157 000 001 9D 00 01  STA &0100,X
B0BB      224 255     E0 FF     CPX#&FF
B0BD      208 245     D0 F5     BNE -11   --> &B0B4
B0BF      152         98        TYA
B0C0 e    101 004     65 04     ADC &04
B0C2      133 004     85 04     STA &04
B0C4      144 002     90 02     BCC 2   --> &B0C8
B0C6      230 005     E6 05     INC &05
B0C8  '   165 039     A5 27     LDA &27
B0CA `    096         60        RTS
B0CB      165 027     A5 1B     LDA &1B
B0CD H    072         48        PHA
B0CE      165 025     A5 19     LDA &19
B0D0 H    072         48        PHA
B0D1      165 026     A5 1A     LDA &1A
B0D3 H    072         48        PHA
B0D4      032 174 152 20 AE 98  JSR &98AE
B0D7  R   240 082     F0 52     BEQ 82   --> &B12B
B0D9      165 027     A5 1B     LDA &1B
B0DB      133 010     85 0A     STA &0A
B0DD h    104         68        PLA
B0DE      133 026     85 1A     STA &1A
B0E0 h    104         68        PLA
B0E1      133 025     85 19     STA &19
```

```
B0E3 h   104       68       PLA
B0E4     133 027   85 1B    STA &1B
B0E6     250       FA       PLX
B0E7  ,  165 044   A5 2C    LDA &2C
B0E9 H   072       48       PHA
B0EA  +  165 043   A5 2B    LDA &2B
B0EC H   072       48       PHA
B0ED  *  165 042   A5 2A    LDA &2A
B0EF H   072       48       PHA
B0F0     232       E8       INX
B0F1     218       DA       PHX
B0F2     032 129 177 20 81 B1  JSR &B181
B0F5     032 229 140 20 E5 8C  JSR &8CE5
B0F8     240 209   F0 D1    BEQ -47   --> &B0CB
B0FA  )  201 041   C9 29    CMP#&29
B0FC  -  208 045   D0 2D    BNE 45   --> &B12B
B0FE     169 000   A9 00    LDA#&00
B100 H   072       48       PHA
B101     032 213 142 20 D5 8E  JSR &8ED5
B104  (  201 040   C9 28    CMP#&28
B106  #  208 035   D0 23    BNE 35   --> &B12B
B108  ;  032 059 157 20 3B 9D  JSR &9D3B
B10B  "  032 034 188 20 22 BC  JSR &BC22
B10E  '  165 039   A5 27    LDA &27
B110  -  133 045   85 2D    STA &2D
B112  &  032 038 188 20 26 BC  JSR &BC26
B115     250       FA       PLX
B116     232       E8       INX
B117     218       DA       PHX
B118     032 235 142 20 EB 8E  JSR &8EEB
B11B     240 235   F0 EB    BEQ -21   --> &B108
B11D  )  201 041   C9 29    CMP#&29
B11F     208 010   D0 0A    BNE 10   --> &B12B
B121 h   104       68       PLA
B122 h   104       68       PLA
B123  L  133 076   85 4C    STA &4C
B125  M  133 077   85 4D    STA &4D
B127  L  228 076   E4 4C    CPX &4C
B129     240 021   F0 15    BEQ 21   --> &B140
B12B     162 251   A2 FB    LDX#&FB
B12D     154       9A       TXS
B12E h   104       68       PLA
B12F     133 012   85 0C    STA &0C
```

```
B131 h    104         68        PLA
B132      133 011     85 0B     STA &0B
B134      000         00        BRK
B135      031         1F        EQUB &1F
B136 Ar   065 114     41 72     EOR (&72,X)
B138 g    103         67        xxx Invalid Code
B139 um   117 109     75 6D     ADC &6D,X
B13B en   101 110     65 6E     ADC &6E
B13D ts   116 115     74 73     STZ &73,X
B13F      000         00        EQUB &00
B140      032 230 188 20 E6 BC  JSR &BCE6
B143 h    104         68        PLA
B144  *   133 042     85 2A     STA &2A
B146 h    104         68        PLA
B147  +   133 043     85 2B     STA &2B
B149 h    104         68        PLA
B14A  ,   133 044     85 2C     STA &2C
B14C 0    048 031     30 1F     BMI 31   --> &B16D
B14E  -   165 045     A5 2D     LDA &2D
B150      240 217     F0 D9     BEQ -39   --> &B12B
B152  '   133 039     85 27     STA &27
B154  7   162 055     A2 37     LDX#&37
B156      032 198 189 20 C6 BD  JSR &BDC6
B159  '   165 039     A5 27     LDA &27
B15B      016 008     10 08     BPL 8   --> &B165
B15D      032 232 187 20 E8 BB  JSR &BBE8
B160  A   032 065 165 20 41 A5  JSR &A541
B163      128 003     80 03     BRA 3   --> &B168
B165      032 230 188 20 E6 BC  JSR &BCE6
B168  8   032 056 179 20 38 B3  JSR &B338
B16B      128 010     80 0A     BRA 10   --> &B177
B16D  -   165 045     A5 2D     LDA &2D
B16F      208 186     D0 BA     BNE -70   --> &B12B
B171      032 210 188 20 D2 BC  JSR &BCD2
B174      032 174 144 20 AE 90  JSR &90AE
B177  L   198 076     C6 4C     DEC &4C
B179      208 197     D0 C5     BNE -59   --> &B140
B17B  M   165 077     A5 4D     LDA &4D
B17D H    072         48        PHA
B17E L    076 128 176 4C 80 B0  JMP &B080
B181  ,   164 044     A4 2C     LDY &2C
B183      192 005     C0 05     CPY#&05
B185      176 005     B0 05     BCS 5   --> &B18C
```

```
B187  7   162 055     A2 37     LDX#&37
B189      032 198 189 20 C6 BD  JSR &BDC6
B18C      032 160 177 20 A0 B1  JSR &B1A0
B18F      008         08        PHP
B190  "   032 034 188 20 22 BC  JSR &BC22
B193 (    040         28        PLP
B194      240 007     F0 07     BEQ 7   --> &B19D
B196 0    048 005     30 05     BMI 5   --> &B19D
B198  7   162 055     A2 37     LDX#&37
B19A      032 128 170 20 80 AA  JSR &AA80
B19D L&   076 038 188 4C 26 BC  JMP &BC26
B1A0  ,   164 044     A4 2C     LDY &2C
B1A2 0S   048 083     30 53     BMI 83   --> &B1F7
B1A4      240 028     F0 1C     BEQ 28   --> &B1C2
B1A6      192 005     C0 05     CPY#&05
B1A8      240 029     F0 1D     BEQ 29   --> &B1C7
B1AA      160 003     A0 03     LDY#&03
B1AC  *   177 042     B1 2A     LDA (&2A),Y
B1AE  -   133 045     85 2D     STA &2D
B1B0      136         88        DEY
B1B1  *   177 042     B1 2A     LDA (&2A),Y
B1B3  ,   133 044     85 2C     STA &2C
B1B5      136         88        DEY
B1B6  *   177 042     B1 2A     LDA (&2A),Y
B1B8      170         AA        TAX
B1B9  *   178 042     B2 2A     LDA (&2A)
B1BB  *   133 042     85 2A     STA &2A
B1BD  +   134 043     86 2B     STX &2B
B1BF  @   169 064     A9 40     LDA#&40
B1C1 `    096         60        RTS
B1C2  *   177 042     B1 2A     LDA (&2A),Y
B1C4 L    076 026 174 4C 1A AE  JMP &AE1A
B1C7 d5   100 053     64 35     STZ &35
B1C9 d/   100 047     64 2F     STZ &2F
B1CB      136         88        DEY
B1CC  *   177 042     B1 2A     LDA (&2A),Y
B1CE  4   133 052     85 34     STA &34
B1D0      136         88        DEY
B1D1  *   177 042     B1 2A     LDA (&2A),Y
B1D3  3   133 051     85 33     STA &33
B1D5      136         88        DEY
B1D6  *   177 042     B1 2A     LDA (&2A),Y
B1D8  2   133 050     85 32     STA &32
```

```
B1DA      136         88        DEY
B1DB  *   177 042     B1 2A     LDA (&2A),Y
B1DD  .   133 046     85 2E     STA &2E
B1DF      168         A8        TAY
B1E0  *   178 042     B2 2A     LDA (&2A)
B1E2  0   133 048     85 30     STA &30
B1E4      208 009     D0 09     BNE 9   --> &B1EF
B1E6      152         98        TYA
B1E7  2   005 050     05 32     ORA &32
B1E9  3   005 051     05 33     ORA &33
B1EB  4   005 052     05 34     ORA &34
B1ED      240 003     F0 03     BEQ 3   --> &B1F2
B1EF      152         98        TYA
B1F0      009 128     09 80     ORA#&80
B1F2  1   133 049     85 31     STA &31
B1F4      169 255     A9 FF     LDA#&FF
B1F6 `    096         60        RTS
B1F7      192 128     C0 80     CPY#&80
B1F9      240 030     F0 1E     BEQ 30   --> &B219
B1FB      160 003     A0 03     LDY#&03
B1FD  *   177 042     B1 2A     LDA (&2A),Y
B1FF  6   133 054     85 36     STA &36
B201      240 021     F0 15     BEQ 21   --> &B218
B203      160 001     A0 01     LDY#&01
B205  *   177 042     B1 2A     LDA (&2A),Y
B207  8   133 056     85 38     STA &38
B209  *   178 042     B2 2A     LDA (&2A)
B20B  7   133 055     85 37     STA &37
B20D  6   164 054     A4 36     LDY &36
B20F      136         88        DEY
B210  7   177 055     B1 37     LDA (&37),Y
B212      153 000 006 99 00 06  STA &0600,Y
B215      152         98        TYA
B216      208 247     D0 F7     BNE -9   --> &B20F
B218 `    096         60        RTS
B219  +   165 043     A5 2B     LDA &2B
B21B      240 021     F0 15     BEQ 21   --> &B232
B21D      160 000     A0 00     LDY#&00
B21F  *   177 042     B1 2A     LDA (&2A),Y
B221      153 000 006 99 00 06  STA &0600,Y
B224 I    073 013     49 0D     EOR#&0D
B226      240 004     F0 04     BEQ 4   --> &B22C
B228      200         C8        INY
```

```
B229      208 244     D0 F4     BNE -12   --> &B21F
B22B      152         98        TYA
B22C  6   132 054     84 36     STY &36
B22E `    096         60        RTS
B22F      032 180 150 20 B4 96  JSR &96B4
B232  *   165 042     A5 2A     LDA &2A
B234 Ll   076 108 174 4C 6C AE  JMP &AE6C
B237      164 010     A4 0A     LDY &0A
B239      240 001     F0 01     BEQ 1   --> &B23C
B23B      136         88        DEY
B23C      032 188 155 20 BC 9B  JSR &9BBC
B23F d    100 008     64 08     STZ &08
B241 d    100 009     64 09     STZ &09
B243      166 024     A6 18     LDX &18
B245  8   134 056     86 38     STX &38
B247 d7   100 055     64 37     STZ &37
B249      164 012     A4 0C     LDY &0C
B24B      192 007     C0 07     CPY#&07
B24D  (   240 040     F0 28     BEQ 40   --> &B277
B24F      166 011     A6 0B     LDX &0B
B251      032 160 141 20 A0 8D  JSR &8DA0
B254      201 013     C9 0D     CMP#&0D
B256      208 024     D0 18     BNE 24   --> &B270
B258  7   228 055     E4 37     CPX &37
B25A      152         98        TYA
B25B  8   229 056     E5 38     SBC &38
B25D      144 024     90 18     BCC 24   --> &B277
B25F      032 160 141 20 A0 8D  JSR &8DA0
B262      009 000     09 00     ORA#&00
B264 0    048 017     30 11     BMI 17   --> &B277
B266      133 009     85 09     STA &09
B268      032 160 141 20 A0 8D  JSR &8DA0
B26B      133 008     85 08     STA &08
B26D      032 160 141 20 A0 8D  JSR &8DA0
B270  7   228 055     E4 37     CPX &37
B272      152         98        TYA
B273  8   229 056     E5 38     SBC &38
B275      176 218     B0 DA     BCS -38   --> &B251
B277 `    096         60        RTS
B278      162 255     A2 FF     LDX#&FF
B27A  (   134 040     86 28     STX &28
B27C      154         9A        TXS
B27D      232         E8        INX
```

```
B27E      160 000     A0 00     LDY#&00
B280      169 218     A9 DA     LDA#&DA
B282      032 244 255 20 F4 FF  JSR &FFF4
B285  ~   169 126     A9 7E     LDA#&7E
B287      032 244 255 20 F4 FF  JSR &FFF4
B28A  7   032 055 178 20 37 B2  JSR &B237
B28D d    100 032     64 20     STZ &20
B28F      178 253     B2 FD     LDA (&FD)
B291      208 003     D0 03     BNE 3   --> &B296
B293      032 166 178 20 A6 B2  JSR &B2A6
B296      165 022     A5 16     LDA &16
B298      133 011     85 0B     STA &0B
B29A      165 023     A5 17     LDA &17
B29C      133 012     85 0C     STA &0C
B29E d    100 010     64 0A     STZ &0A
B2A0      032 207 187 20 CF BB  JSR &BBCF
B2A3 L    076 011 144 4C 0B 90  JMP &900B
B2A6      169 175     A9 AF     LDA#&AF
B2A8      133 022     85 16     STA &16
B2AA      169 178     A9 B2     LDA#&B2
B2AC      133 023     85 17     STA &17
B2AE `    096         60        RTS
B2AF  :   246 058     F6 3A     INC &3A,X
B2B1      231         E7        xxx Invalid Code
B2B2   "  158 241 034 9E F1 22  STZ &22F1,X
B2B5  at  032 097 116 20 61 74  JSR &7461
B2B8  li  032 108 105 20 6C 69  JSR &696C
B2BB ne   110 101 032 6E 65 20  ROR &2065
B2BE "    034         22        xxx Invalid Code
B2BF ;    059         3B        xxx Invalid Code
B2C0  :   158 058 224 9E 3A E0  STZ &E03A,X
B2C3      139         8B        xxx Invalid Code
B2C4  :   241 058     F1 3A     SBC (&3A),Y
B2C6      224 013     E0 0D     CPX#&0D
B2C8  o   032 111 146 20 6F 92  JSR &926F
B2CB      162 003     A2 03     LDX#&03
B2CD  *   165 042     A5 2A     LDA &2A
B2CF H    072         48        PHA
B2D0  +   165 043     A5 2B     LDA &2B
B2D2 H    072         48        PHA
B2D3      218         DA        PHX
B2D4      032 172 150 20 AC 96  JSR &96AC
B2D7      250         FA        PLX
```

```
B2D8      202         CA        DEX
B2D9      208 242     D0 F2     BNE -14   --> &B2CD
B2DB      032 150 155 20 96 9B  JSR &9B96
B2DE  *   165 042     A5 2A     LDA &2A
B2E0  =   133 061     85 3D     STA &3D
B2E2  +   165 043     A5 2B     LDA &2B
B2E4  >   133 062     85 3E     STA &3E
B2E6      160 007     A0 07     LDY#&07
B2E8      162 005     A2 05     LDX#&05
B2EA      128 027     80 1B     BRA 27   --> &B307
B2EC  o   032 111 146 20 6F 92  JSR &926F
B2EF      162 013     A2 0D     LDX#&0D
B2F1  *   165 042     A5 2A     LDA &2A
B2F3 H    072         48        PHA
B2F4      218         DA        PHX
B2F5      032 172 150 20 AC 96  JSR &96AC
B2F8      250         FA        PLX
B2F9      202         CA        DEX
B2FA      208 245     D0 F5     BNE -11   --> &B2F1
B2FC      032 150 155 20 96 9B  JSR &9B96
B2FF  *   165 042     A5 2A     LDA &2A
B301  D   133 068     85 44     STA &44
B303      162 012     A2 0C     LDX#&0C
B305      160 008     A0 08     LDY#&08
B307 h    104         68        PLA
B308  7   149 055     95 37     STA &37,X
B30A      202         CA        DEX
B30B      016 250     10 FA     BPL -6   --> &B307
B30D      152         98        TYA
B30E  7   162 055     A2 37     LDX#&37
B310      160 000     A0 00     LDY#&00
B312      032 241 255 20 F1 FF  JSR &FFF1
B315      128 011     80 0B     BRA 11   --> &B322
B317  o   032 111 146 20 6F 92  JSR &926F
B31A      032 150 155 20 96 9B  JSR &9B96
B31D  *   164 042     A4 2A     LDY &2A
B31F      136         88        DEY
B320  #   132 035     84 23     STY &23
B322 L    076 005 144 4C 05 90  JMP &9005
B325 L    076 146 144 4C 92 90  JMP &9092
B328  ;   032 059 157 20 3B 9D  JSR &9D3B
B32B z    122         7A        PLY
B32C      250         FA        PLX
```

```
B32D h    104         68        PLA
B32E  9   133 057     85 39     STA &39
B330 h    104         68        PLA
B331  8   133 056     85 38     STA &38
B333 h    104         68        PLA
B334  7   133 055     85 37     STA &37
B336      218         DA        PHX
B337 Z    090         5A        PHY
B338  9   165 057     A5 39     LDA &39
B33A      201 005     C9 05     CMP#&05
B33C  "   240 034     F0 22     BEQ 34   --> &B360
B33E  '   165 039     A5 27     LDA &27
B340      240 227     F0 E3     BEQ -29   --> &B325
B342      016 003     10 03     BPL 3   --> &B347
B344      032 195 150 20 C3 96  JSR &96C3
B347  *   165 042     A5 2A     LDA &2A
B349  7   146 055     92 37     STA (&37)
B34B  9   165 057     A5 39     LDA &39
B34D      240 016     F0 10     BEQ 16   --> &B35F
B34F  +   165 043     A5 2B     LDA &2B
B351      160 001     A0 01     LDY#&01
B353  7   145 055     91 37     STA (&37),Y
B355  ,   165 044     A5 2C     LDA &2C
B357      200         C8        INY
B358  7   145 055     91 37     STA (&37),Y
B35A  -   165 045     A5 2D     LDA &2D
B35C      200         C8        INY
B35D  7   145 055     91 37     STA (&37),Y
B35F `    096         60        RTS
B360  '   165 039     A5 27     LDA &27
B362      240 193     F0 C1     BEQ -63   --> &B325
B364 0    048 003     30 03     BMI 3   --> &B369
B366      032 133 129 20 85 81  JSR &8185
B369  0   165 048     A5 30     LDA &30
B36B  7   146 055     92 37     STA (&37)
B36D      160 001     A0 01     LDY#&01
B36F  .   165 046     A5 2E     LDA &2E
B371 E1   069 049     45 31     EOR &31
B373 )    041 128     29 80     AND#&80
B375 E1   069 049     45 31     EOR &31
B377  7   145 055     91 37     STA (&37),Y
B379      200         C8        INY
B37A  2   165 050     A5 32     LDA &32
```

```
B37C  7   145 055     91 37     STA (&37),Y
B37E      200         C8        INY
B37F  3   165 051     A5 33     LDA &33
B381  7   145 055     91 37     STA (&37),Y
B383      200         C8        INY
B384  4   165 052     A5 34     LDA &34
B386  7   145 055     91 37     STA (&37),Y
B388 `    096         60        RTS
B389 ED   069 068     45 44     EOR &44
B38B IT   073 084     49 54     EOR#&54
B38D  12  032 049 050 20 31 32  JSR &3231
B390 ,2   044 050 013 2C 32 0D  BIT &0D32
B393      032 172 187 20 AC BB  JSR &BBAC
B396      169 128     A9 80     LDA#&80
B398      133 031     85 1F     STA &1F
B39A d;   100 059     64 3B     STZ &3B
B39C d<   100 060     64 3C     STZ &3C
B39E      032 232 171 20 E8 AB  JSR &ABE8
B3A1      032 030 155 20 1E 9B  JSR &9B1E
B3A4      008         08        PHP
B3A5  &   032 038 188 20 26 BC  JSR &BC26
B3A8      032 219 171 20 DB AB  JSR &ABDB
B3AB F+   070 043     46 2B     LSR &2B
B3AD (    040         28        PLP
B3AE      144 015     90 0F     BCC 15   --> &B3BF
B3B0      032 229 140 20 E5 8C  JSR &8CE5
B3B3      240 017     F0 11     BEQ 17   --> &B3C6
B3B5      032 230 188 20 E6 BC  JSR &BCE6
B3B8  &   032 038 188 20 26 BC  JSR &BC26
B3BB      198 010     C6 0A     DEC &0A
B3BD      128 010     80 0A     BRA 10   --> &B3C9
B3BF      032 229 140 20 E5 8C  JSR &8CE5
B3C2      240 002     F0 02     BEQ 2    --> &B3C6
B3C4      198 010     C6 0A     DEC &0A
B3C6      032 030 155 20 1E 9B  JSR &9B1E
B3C9  1   162 049     A2 31     LDX#&31
B3CB      032 198 189 20 C6 BD  JSR &BDC6
B3CE      032 224 142 20 E0 8E  JSR &8EE0
B3D1      201 231     C9 E7     CMP#&E7
B3D3      208 030     D0 1E     BNE 30   --> &B3F3
B3D5      032 224 142 20 E0 8E  JSR &8EE0
B3D8      032 188 155 20 BC 9B  JSR &9BBC
B3DB      128 025     80 19     BRA 25   --> &B3F6
```

```
B3DD      200         C8        INY
B3DE      177 011     B1 0B     LDA (&0B),Y
B3E0  O   201 079     C9 4F     CMP#&4F
B3E2      208 182     D0 B6     BNE -74   --> &B39A
B3E4      230 010     E6 0A     INC &0A
B3E6  o   032 111 146 20 6F 92  JSR &926F
B3E9      032 166 155 20 A6 9B  JSR &9BA6
B3EC  *   165 042     A5 2A     LDA &2A
B3EE      133 031     85 1F     STA &1F
B3F0 L    076 134 143 4C 86 8F  JMP &8F86
B3F3      032 176 155 20 B0 9B  JSR &9BB0
B3F6      165 011     A5 0B     LDA &0B
B3F8      133 025     85 19     STA &19
B3FA      032 229 189 20 E5 BD  JSR &BDE5
B3FD      032 230 188 20 E6 BC  JSR &BCE6
B400      032 205 128 20 CD 80  JSR &80CD
B403  =   165 061     A5 3D     LDA &3D
B405      133 011     85 0B     STA &0B
B407  >   165 062     A5 3E     LDA &3E
B409      133 012     85 0C     STA &0C
B40B      176 027     B0 1B     BCS 27   --> &B428
B40D      136         88        DEY
B40E      128 015     80 0F     BRA 15   --> &B41F
B410      032 148 189 20 94 BD  JSR &BD94
B413 $    036 031     24 1F     BIT &1F
B415 0    048 005     30 05     BMI 5   --> &B41C
B417      169 010     A9 0A     LDA#&0A
B419      032 238 255 20 EE FF  JSR &FFEE
B41C      032 188 155 20 BC 9B  JSR &9BBC
B41F      177 011     B1 0B     LDA (&0B),Y
B421  +   133 043     85 2B     STA &2B
B423      200         C8        INY
B424      177 011     B1 0B     LDA (&0B),Y
B426  *   133 042     85 2A     STA &2A
B428  *   165 042     A5 2A     LDA &2A
B42A      024         18        CLC
B42B  1   229 049     E5 31     SBC &31
B42D  +   165 043     A5 2B     LDA &2B
B42F  2   229 050     E5 32     SBC &32
B431      144 011     90 0B     BCC 11   --> &B43E
B433 $    036 031     24 1F     BIT &1F
B435      016 185     10 B9     BPL -71   --> &B3F0
B437      162 137     A2 89     LDX#&89
```

```
B439      160 179     A0 B3     LDY#&B3
B43B L    076 247 255 4C F7 FF  JMP &FFF7
B43E dL   100 076     64 4C     STZ &4C
B440 dM   100 077     64 4D     STZ &4D
B442      160 004     A0 04     LDY#&04
B444      132 010     84 0A     STY &0A
B446      132 027     84 1B     STY &1B
B448 $;   036 059     24 3B     BIT &3B
B44A      016 002     10 02     BPL 2   --> &B44E
B44C d;   100 059     64 3B     STZ &3B
B44E $<   036 060     24 3C     BIT &3C
B450      016 002     10 02     BPL 2   --> &B454
B452 d<   100 060     64 3C     STZ &3C
B454      177 011     B1 0B     LDA (&0B),Y
B456      201 013     C9 0D     CMP#&0D
B458  7   240 055     F0 37     BEQ 55   --> &B491
B45A      201 244     C9 F4     CMP#&F4
B45C      240 006     F0 06     BEQ 6   --> &B464
B45E  "   201 034     C9 22     CMP#&22
B460      208 004     D0 04     BNE 4   --> &B466
B462 EL   069 076     45 4C     EOR &4C
B464  L   133 076     85 4C     STA &4C
B466  L   166 076     A6 4C     LDX &4C
B468      208 012     D0 0C     BNE 12   --> &B476
B46A      201 237     C9 ED     CMP#&ED
B46C      208 002     D0 02     BNE 2   --> &B470
B46E  ;   198 059     C6 3B     DEC &3B
B470      201 253     C9 FD     CMP#&FD
B472      208 002     D0 02     BNE 2   --> &B476
B474  <   198 060     C6 3C     DEC &3C
B476      166 025     A6 19     LDX &19
B478      189 000 007 BD 00 07  LDA &0700,X
B47B      201 013     C9 0D     CMP#&0D
B47D      240 010     F0 0A     BEQ 10   --> &B489
B47F      209 011     D1 0B     CMP (&0B),Y
B481      208 008     D0 08     BNE 8   --> &B48B
B483      200         C8        INY
B484      232         E8        INX
B485      128 241     80 F1     BRA -15   --> &B478
B487      128 135     80 87     BRA -121   --> &B410
B489  M   133 077     85 4D     STA &4D
B48B      230 027     E6 1B     INC &1B
B48D      164 027     A4 1B     LDY &1B
```

```
B48F      128 195     80 C3     BRA -61   --> &B454
B491  M   165 077     A5 4D     LDA &4D
B493      240 135     F0 87     BEQ -121   --> &B41C
B495      032 133 160 20 85 A0  JSR &A085
B498      169 001     A9 01     LDA#&01
B49A      232         E8        INX
B49B 8    056         38        SEC
B49C      032 180 189 20 B4 BD  JSR &BDB4
B49F  ;   166 059     A6 3B     LDX &3B
B4A1      169 002     A9 02     LDA#&02
B4A3      032 179 189 20 B3 BD  JSR &BDB3
B4A6  <   166 060     A6 3C     LDX &3C
B4A8      169 004     A9 04     LDA#&04
B4AA      032 179 189 20 B3 BD  JSR &BDB3
B4AD dL   100 076     64 4C     STZ &4C
B4AF      164 010     A4 0A     LDY &0A
B4B1      177 011     B1 0B     LDA (&0B),Y
B4B3      201 013     C9 0D     CMP#&0D
B4B5      240 208     F0 D0     BEQ -48   --> &B487
B4B7  "   201 034     C9 22     CMP#&22
B4B9      208 012     D0 0C     BNE 12   --> &B4C7
B4BB EL   069 076     45 4C     EOR &4C
B4BD  L   133 076     85 4C     STA &4C
B4BF  "   169 034     A9 22     LDA#&22
B4C1      032 148 189 20 94 BD  JSR &BD94
B4C4      200         C8        INY
B4C5      128 234     80 EA     BRA -22   --> &B4B1
B4C7  L   166 076     A6 4C     LDX &4C
B4C9      208 246     D0 F6     BNE -10   --> &B4C1
B4CB      201 141     C9 8D     CMP#&8D
B4CD      208 010     D0 0A     BNE 10   --> &B4D9
B4CF  *   032 042 155 20 2A 9B  JSR &9B2A
B4D2      132 010     84 0A     STY &0A
B4D4      032 129 160 20 81 A0  JSR &A081
B4D7      128 214     80 D6     BRA -42   --> &B4AF
B4D9      201 227     C9 E3     CMP#&E3
B4DB      208 002     D0 02     BNE 2   --> &B4DF
B4DD  ;   230 059     E6 3B     INC &3B
B4DF      201 245     C9 F5     CMP#&F5
B4E1      208 002     D0 02     BNE 2   --> &B4E5
B4E3  <   230 060     E6 3C     INC &3C
B4E5      201 244     C9 F4     CMP#&F4
B4E7      208 002     D0 02     BNE 2   --> &B4EB
```

```
B4E9  L   133 076     85 4C     STA &4C
B4EB  7   032 055 189 20 37 BD  JSR &BD37
B4EE      200         C8        INY
B4EF      128 192     80 C0     BRA -64   --> &B4B1
B4F1      032 245 152 20 F5 98  JSR &98F5
B4F4      208 009     D0 09     BNE 9   --> &B4FF
B4F6  &   166 038     A6 26     LDX &26
B4F8  8   240 056     F0 38     BEQ 56   --> &B532
B4FA  =   176 061     B0 3D     BCS 61   --> &B539
B4FC Li   076 105 155 4C 69 9B  JMP &9B69
B4FF      176 251     B0 FB     BCS -5   --> &B4FC
B501  &   166 038     A6 26     LDX &26
B503  -   240 045     F0 2D     BEQ 45   --> &B532
B505  *   165 042     A5 2A     LDA &2A
B507      221 025 005 DD 19 05  CMP &0519,X
B50A      208 014     D0 0E     BNE 14   --> &B51A
B50C  +   165 043     A5 2B     LDA &2B
B50E      221 026 005 DD 1A 05  CMP &051A,X
B511      208 007     D0 07     BNE 7   --> &B51A
B513  ,   165 044     A5 2C     LDA &2C
B515      221 027 005 DD 1B 05  CMP &051B,X
B518      240 031     F0 1F     BEQ 31   --> &B539
B51A      138         8A        TXA
B51B 8    056         38        SEC
B51C      233 015     E9 0F     SBC#&0F
B51E      170         AA        TAX
B51F  &   134 038     86 26     STX &26
B521      208 226     D0 E2     BNE -30   --> &B505
B523      000         00        BRK
B524 !    033         21        EQUB &21
B525 C    067         43        xxx Invalid Code
B526 an'  097 110 039 61 6E 27  ADC (&276E,X)
B529 t    116 032     74 20     STZ &20,X
B52B mat  109 097 116 6D 61 74  ADC &7461
B52E c    099         63        xxx Invalid Code
B52F h    104         68        PLA
B530      032 227     20 E3     xxx Invalid Code
B532      000         00        BRK
B533      032         20        EQUB &20
B534 No   078 111     4E 6F     xxx Invalid Code
B536      032 227     20 E3     xxx Invalid Code
B538      000         00        EQUB &00
B539      189 025 005 BD 19 05  LDA &0519,X
```

```
B53C  *   133 042     85 2A     STA &2A
B53E      189 026 005 BD 1A 05  LDA &051A,X
B541  +   133 043     85 2B     STA &2B
B543      188 027 005 BC 1B 05  LDY &051B,X
B546      192 005     C0 05     CPY#&05
B548  v   240 118     F0 76     BEQ 118   --> &B5C0
B54A  *   178 042     B2 2A     LDA (&2A)
B54C }    125 028 005 7D 1C 05  ADC &051C,X
B54F  *   146 042     92 2A     STA (&2A)
B551  7   133 055     85 37     STA &37
B553      160 001     A0 01     LDY#&01
B555  *   177 042     B1 2A     LDA (&2A),Y
B557 }    125 029 005 7D 1D 05  ADC &051D,X
B55A  *   145 042     91 2A     STA (&2A),Y
B55C  8   133 056     85 38     STA &38
B55E      200         C8        INY
B55F  *   177 042     B1 2A     LDA (&2A),Y
B561 }    125 030 005 7D 1E 05  ADC &051E,X
B564  *   145 042     91 2A     STA (&2A),Y
B566  9   133 057     85 39     STA &39
B568      200         C8        INY
B569  *   177 042     B1 2A     LDA (&2A),Y
B56B }    125 031 005 7D 1F 05  ADC &051F,X
B56E  *   145 042     91 2A     STA (&2A),Y
B570      168         A8        TAY
B571  7   165 055     A5 37     LDA &37
B573 8    056         38        SEC
B574  !   253 033 005 FD 21 05  SBC &0521,X
B577  7   133 055     85 37     STA &37
B579  8   165 056     A5 38     LDA &38
B57B  "   253 034 005 FD 22 05  SBC &0522,X
B57E  7   004 055     04 37     TSB &37
B580  9   165 057     A5 39     LDA &39
B582  #   253 035 005 FD 23 05  SBC &0523,X
B585  7   004 055     04 37     TSB &37
B587      152         98        TYA
B588  $   253 036 005 FD 24 05  SBC &0524,X
B58B  7   005 055     05 37     ORA &37
B58D      240 015     F0 0F     BEQ 15   --> &B59E
B58F      152         98        TYA
B590 ]    093 031 005 5D 1F 05  EOR &051F,X
B593 ]$   093 036 005 5D 24 05  EOR &0524,X
B596      016 004     10 04     BPL 4   --> &B59C
```

```
B598      176 004     B0 04     BCS 4   --> &B59E
B59A      128 018     80 12     BRA 18   --> &B5AE
B59C      176 016     B0 10     BCS 16   --> &B5AE
B59E  &   188 038 005 BC 26 05  LDY &0526,X
B5A1  '   189 039 005 BD 27 05  LDA &0527,X
B5A4      132 011     84 0B     STY &0B
B5A6      133 012     85 0C     STA &0C
B5A8      032 198 155 20 C6 9B  JSR &9BC6
B5AB L    076 011 144 4C 0B 90  JMP &900B
B5AE      138         8A        TXA
B5AF 8    056         38        SEC
B5B0      233 015     E9 0F     SBC#&0F
B5B2  &   133 038     85 26     STA &26
B5B4      164 027     A4 1B     LDY &1B
B5B6      132 010     84 0A     STY &0A
B5B8      032 229 140 20 E5 8C  JSR &8CE5
B5BB  8   208 056     D0 38     BNE 56   --> &B5F5
B5BD L    076 241 180 4C F1 B4  JMP &B4F1
B5C0      032 199 177 20 C7 B1  JSR &B1C7
B5C3      138         8A        TXA
B5C4      024         18        CLC
B5C5 i    105 028     69 1C     ADC#&1C
B5C7  J   133 074     85 4A     STA &4A
B5C9      169 005     A9 05     LDA#&05
B5CB  K   133 075     85 4B     STA &4B
B5CD      032 141 166 20 8D A6  JSR &A68D
B5D0  *   165 042     A5 2A     LDA &2A
B5D2  7   133 055     85 37     STA &37
B5D4  +   165 043     A5 2B     LDA &2B
B5D6  8   133 056     85 38     STA &38
B5D8  i   032 105 179 20 69 B3  JSR &B369
B5DB  &   166 038     A6 26     LDX &26
B5DD      138         8A        TXA
B5DE      024         18        CLC
B5DF i!   105 033     69 21     ADC#&21
B5E1  J   133 074     85 4A     STA &4A
B5E3      032 143 156 20 8F 9C  JSR &9C8F
B5E6      240 182     F0 B6     BEQ -74   --> &B59E
B5E8      189 029 005 BD 1D 05  LDA &051D,X
B5EB 0    048 004     30 04     BMI 4   --> &B5F1
B5ED      176 175     B0 AF     BCS -81   --> &B59E
B5EF      128 189     80 BD     BRA -67   --> &B5AE
B5F1      144 171     90 AB     BCC -85   --> &B59E
```

```
B5F3      128 185     80 B9     BRA -71   --> &B5AE
B5F5 L    076 000 144 4C 00 90  JMP &9000
B5F8      000         00        BRK
B5F9 "    034         22        EQUB &22
B5FA      227         E3        xxx Invalid Code
B5FB  va  032 118 097 20 76 61  JSR &6176
B5FE ri   114 105     72 69     ADC (&69)
B600 abl  097 098 108 61 62 6C  ADC (&6C62,X)
B603 e    101         65        xxx Invalid Code
B604      000         00        BRK
B605 #    035         23        EQUB &23
B606 T    084         54        xxx Invalid Code
B607 o    111         6F        xxx Invalid Code
B608 o    111         6F        xxx Invalid Code
B609  ma  032 109 097 20 6D 61  JSR &616D
B60C ny   110 121 032 6E 79 20  ROR &2079
B60F      227         E3        xxx Invalid Code
B610 s    115         73        xxx Invalid Code
B611      000         00        BRK
B612 $    036         24        EQUB &24
B613 N    078         4E        xxx Invalid Code
B614 o    111         6F        xxx Invalid Code
B615      032 184     20 B8     xxx Invalid Code
B617      000         00        EQUB &00
B618      032 174 152 20 AE 98  JSR &98AE
B61B      240 219     F0 DB     BEQ -37   --> &B5F8
B61D      176 217     B0 D9     BCS -39   --> &B5F8
B61F  C   032 067 188 20 43 BC  JSR &BC43
B622      032 134 155 20 86 9B  JSR &9B86
B625  (   032 040 179 20 28 B3  JSR &B328
B628      032 213 142 20 D5 8E  JSR &8ED5
B62B      201 184     C9 B8     CMP#&B8
B62D      208 226     D0 E2     BNE -30   --> &B611
B62F  &   164 038     A4 26     LDY &26
B631      192 150     C0 96     CPY#&96
B633      176 207     B0 CF     BCS -49   --> &B604
B635      152         98        TYA
B636 i    105 015     69 0F     ADC#&0F
B638  &   133 038     85 26     STA &26
B63A  7   165 055     A5 37     LDA &37
B63C  (   153 040 005 99 28 05  STA &0528,Y
B63F  8   165 056     A5 38     LDA &38
B641  )   153 041 005 99 29 05  STA &0529,Y
```

```
B644  9   165 057     A5 39     LDA &39
B646  *   153 042 005 99 2A 05  STA &052A,Y
B649      201 005     C9 05     CMP#&05
B64B  T   240 084     F0 54     BEQ 84   --> &B6A1
B64D      032 175 150 20 AF 96  JSR &96AF
B650  &   164 038     A4 26     LDY &26
B652  *   165 042     A5 2A     LDA &2A
B654  !   153 033 005 99 21 05  STA &0521,Y
B657  +   165 043     A5 2B     LDA &2B
B659  "   153 034 005 99 22 05  STA &0522,Y
B65C  ,   165 044     A5 2C     LDA &2C
B65E  #   153 035 005 99 23 05  STA &0523,Y
B661  -   165 045     A5 2D     LDA &2D
B663  $   153 036 005 99 24 05  STA &0524,Y
B666      169 001     A9 01     LDA#&01
B668      032 024 174 20 18 AE  JSR &AE18
B66B      032 213 142 20 D5 8E  JSR &8ED5
B66E      201 136     C9 88     CMP#&88
B670      208 005     D0 05     BNE 5   --> &B677
B672      032 175 150 20 AF 96  JSR &96AF
B675      164 027     A4 1B     LDY &1B
B677      132 010     84 0A     STY &0A
B679  &   164 038     A4 26     LDY &26
B67B  *   165 042     A5 2A     LDA &2A
B67D      153 028 005 99 1C 05  STA &051C,Y
B680  +   165 043     A5 2B     LDA &2B
B682      153 029 005 99 1D 05  STA &051D,Y
B685  ,   165 044     A5 2C     LDA &2C
B687      153 030 005 99 1E 05  STA &051E,Y
B68A  -   165 045     A5 2D     LDA &2D
B68C      153 031 005 99 1F 05  STA &051F,Y
B68F      032 207 155 20 CF 9B  JSR &9BCF
B692  &   164 038     A4 26     LDY &26
B694      165 011     A5 0B     LDA &0B
B696  &   153 038 005 99 26 05  STA &0526,Y
B699      165 012     A5 0C     LDA &0C
B69B  '   153 039 005 99 27 05  STA &0527,Y
B69E L    076 011 144 4C 0B 90  JMP &900B
B6A1  ;   032 059 157 20 3B 9D  JSR &9D3B
B6A4      032 221 150 20 DD 96  JSR &96DD
B6A7  &   165 038     A5 26     LDA &26
B6A9      024         18        CLC
B6AA i!   105 033     69 21     ADC#&21
```

```
B6AC  J   133 074     85 4A     STA &4A
B6AE      169 005     A9 05     LDA#&05
B6B0  K   133 075     85 4B     STA &4B
B6B2      032 025 165 20 19 A5  JSR &A519
B6B5      032 216 165 20 D8 A5  JSR &A5D8
B6B8      032 213 142 20 D5 8E  JSR &8ED5
B6BB      201 136     C9 88     CMP#&88
B6BD      208 008     D0 08     BNE 8   --> &B6C7
B6BF  ;   032 059 157 20 3B 9D  JSR &9D3B
B6C2      032 221 150 20 DD 96  JSR &96DD
B6C5      164 027     A4 1B     LDY &1B
B6C7      132 010     84 0A     STY &0A
B6C9  &   165 038     A5 26     LDA &26
B6CB      024         18        CLC
B6CC i    105 028     69 1C     ADC#&1C
B6CE  J   133 074     85 4A     STA &4A
B6D0      169 005     A9 05     LDA#&05
B6D2  K   133 075     85 4B     STA &4B
B6D4      032 025 165 20 19 A5  JSR &A519
B6D7      128 182     80 B6     BRA -74   --> &B68F
B6D9  *   032 042 184 20 2A B8  JSR &B82A
B6DC      032 166 155 20 A6 9B  JSR &9BA6
B6DF  %   164 037     A4 25     LDY &25
B6E1      192 026     C0 1A     CPY#&1A
B6E3      176 014     B0 0E     BCS 14   --> &B6F3
B6E5      165 011     A5 0B     LDA &0B
B6E7      153 204 005 99 CC 05  STA &05CC,Y
B6EA      165 012     A5 0C     LDA &0C
B6EC      153 230 005 99 E6 05  STA &05E6,Y
B6EF  %   230 037     E6 25     INC &25
B6F1  0   128 048     80 30     BRA 48   --> &B723
B6F3      000         00        BRK
B6F4 %    037         25        EQUB &25
B6F5 T    084         54        xxx Invalid Code
B6F6 o    111         6F        xxx Invalid Code
B6F7 o    111         6F        xxx Invalid Code
B6F8  ma  032 109 097 20 6D 61  JSR &616D
B6FB ny   110 121 032 6E 79 20  ROR &2079
B6FE  s   228 115     E4 73     CPX &73
B700      000         00        BRK
B701 &    038         26        EQUB &26
B702 N    078         4E        xxx Invalid Code
B703 o    111         6F        xxx Invalid Code
```

```
B704      032 228     20 E4     xxx Invalid Code
B706      000         00        EQUB &00
B707      032 166 155 20 A6 9B  JSR &9BA6
B70A  %   166 037     A6 25     LDX &25
B70C      240 242     F0 F2     BEQ -14   --> &B700
B70E  %   198 037     C6 25     DEC &25
B710      188 203 005 BC CB 05  LDY &05CB,X
B713      189 229 005 BD E5 05  LDA &05E5,X
B716      132 011     84 0B     STY &0B
B718      133 012     85 0C     STA &0C
B71A L    076 005 144 4C 05 90  JMP &9005
B71D  *   032 042 184 20 2A B8  JSR &B82A
B720      032 166 155 20 A6 9B  JSR &9BA6
B723      165 032     A5 20     LDA &20
B725      240 003     F0 03     BEQ 3   --> &B72A
B727  K   032 075 156 20 4B 9C  JSR &9C4B
B72A      160 004     A0 04     LDY#&04
B72C      132 010     84 0A     STY &0A
B72E  =   164 061     A4 3D     LDY &3D
B730  >   165 062     A5 3E     LDA &3E
B732      132 011     84 0B     STY &0B
B734      133 012     85 0C     STA &0C
B736 L    076 011 144 4C 0B 90  JMP &900B
B739      032 166 155 20 A6 9B  JSR &9BA6
B73C      032 166 178 20 A6 B2  JSR &B2A6
B73F      128 217     80 D9     BRA -39   --> &B71A
B741      032 224 142 20 E0 8E  JSR &8EE0
B744      201 135     C9 87     CMP#&87
B746      240 241     F0 F1     BEQ -15   --> &B739
B748      164 010     A4 0A     LDY &0A
B74A      136         88        DEY
B74B      032 188 155 20 BC 9B  JSR &9BBC
B74E d    100 010     64 0A     STZ &0A
B750      165 011     A5 0B     LDA &0B
B752      133 022     85 16     STA &16
B754      165 012     A5 0C     LDA &0C
B756      133 023     85 17     STA &17
B758 L    076 174 143 4C AE 8F  JMP &8FAE
B75B      032 224 142 20 E0 8E  JSR &8EE0
B75E      201 133     C9 85     CMP#&85
B760      240 223     F0 DF     BEQ -33   --> &B741
B762      198 010     C6 0A     DEC &0A
B764  o   032 111 146 20 6F 92  JSR &926F
```

```
B767      224 242     E0 F2     CPX#&F2
B769      240 009     F0 09     BEQ 9   --> &B774
B76B      200         C8        INY
B76C      224 229     E0 E5     CPX#&E5
B76E      240 004     F0 04     BEQ 4   --> &B774
B770      224 228     E0 E4     CPX#&E4
B772  v   208 118     D0 76     BNE 118   --> &B7EA
B774      218         DA        PHX
B775  +   165 043     A5 2B     LDA &2B
B777  ,   005 044     05 2C     ORA &2C
B779  -   005 045     05 2D     ORA &2D
B77B  X   208 088     D0 58     BNE 88   --> &B7D5
B77D  *   198 042     C6 2A     DEC &2A
B77F  5   240 053     F0 35     BEQ 53   --> &B7B6
B781 0R   048 082     30 52     BMI 82   --> &B7D5
B783      177 011     B1 0B     LDA (&0B),Y
B785      201 013     C9 0D     CMP#&0D
B787  L   240 076     F0 4C     BEQ 76   --> &B7D5
B789  :   201 058     C9 3A     CMP#&3A
B78B  H   240 072     F0 48     BEQ 72   --> &B7D5
B78D      201 139     C9 8B     CMP#&8B
B78F  D   240 068     F0 44     BEQ 68   --> &B7D5
B791      200         C8        INY
B792  "   201 034     C9 22     CMP#&22
B794      208 004     D0 04     BNE 4   --> &B79A
B796 E+   069 043     45 2B     EOR &2B
B798  +   133 043     85 2B     STA &2B
B79A  +   166 043     A6 2B     LDX &2B
B79C      208 229     D0 E5     BNE -27   --> &B783
B79E  )   201 041     C9 29     CMP#&29
B7A0      208 002     D0 02     BNE 2   --> &B7A4
B7A2  ,   198 044     C6 2C     DEC &2C
B7A4  (   201 040     C9 28     CMP#&28
B7A6      208 002     D0 02     BNE 2   --> &B7AA
B7A8  ,   230 044     E6 2C     INC &2C
B7AA  ,   201 044     C9 2C     CMP#&2C
B7AC      208 213     D0 D5     BNE -43   --> &B783
B7AE  ,   166 044     A6 2C     LDX &2C
B7B0      208 209     D0 D1     BNE -47   --> &B783
B7B2  *   198 042     C6 2A     DEC &2A
B7B4      208 205     D0 CD     BNE -51   --> &B783
B7B6 h    104         68        PLA
B7B7      201 242     C9 F2     CMP#&F2
```

```
B7B9  H   240 072     F0 48     BEQ 72   --> &B803
B7BB      132 010     84 0A     STY &0A
B7BD      201 228     C9 E4     CMP#&E4
B7BF      240 009     F0 09     BEQ 9   --> &B7CA
B7C1  *   032 042 184 20 2A B8  JSR &B82A
B7C4      032 198 155 20 C6 9B  JSR &9BC6
B7C7 L#   076 035 183 4C 23 B7  JMP &B723
B7CA  *   032 042 184 20 2A B8  JSR &B82A
B7CD      164 010     A4 0A     LDY &0A
B7CF      032 029 184 20 1D B8  JSR &B81D
B7D2 L    076 220 182 4C DC B6  JMP &B6DC
B7D5 h    104         68        PLA
B7D6      177 011     B1 0B     LDA (&0B),Y
B7D8      200         C8        INY
B7D9      201 139     C9 8B     CMP#&8B
B7DB  :   240 058     F0 3A     BEQ 58   --> &B817
B7DD      201 013     C9 0D     CMP#&0D
B7DF      208 245     D0 F5     BNE -11   --> &B7D6
B7E1      000         00        BRK
B7E2 (    040         28        EQUB &28
B7E3   r  238 032 114 EE 20 72  INC &7220
B7E6 ang  097 110 103 61 6E 67  ADC (&676E,X)
B7E9 e    101         65        xxx Invalid Code
B7EA      000         00        BRK
B7EB '    039         27        EQUB &27
B7EC   s  238 032 115 EE 20 73  INC &7320
B7EF ynt  121 110 116 79 6E 74  ADC &746E,Y
B7F2 ax   097 120     61 78     xxx Invalid Code
B7F4      000         00        BRK
B7F5 )    041         29        EQUB &29
B7F6 N    078         4E        xxx Invalid Code
B7F7 o    111         6F        xxx Invalid Code
B7F8  su  032 115 117 20 73 75  JSR &7573
B7FB c    099         63        xxx Invalid Code
B7FC h    104         68        PLA
B7FD  li  032 108 105 20 6C 69  JSR &696C
B800 ne   110 101     6E 65     xxx Invalid Code
B802      000         00        EQUB &00
B803      132 027     84 1B     STY &1B
B805      032 213 142 20 D5 8E  JSR &8ED5
B808      201 242     C9 F2     CMP#&F2
B80A      208 222     D0 DE     BNE -34   --> &B7EA
B80C      032 025 176 20 19 B0  JSR &B019
```

```
B80F      164 027     A4 1B     LDY &1B
B811      032 029 184 20 1D B8  JSR &B81D
B814 L    076 002 144 4C 02 90  JMP &9002
B817      132 010     84 0A     STY &0A
B819 L)   076 041 156 4C 29 9C  JMP &9C29
B81C      200         C8        INY
B81D      177 011     B1 0B     LDA (&0B),Y
B81F      201 013     C9 0D     CMP#&0D
B821      240 004     F0 04     BEQ 4   --> &B827
B823  :   201 058     C9 3A     CMP#&3A
B825      208 245     D0 F5     BNE -11   --> &B81C
B827      132 010     84 0A     STY &0A
B829 `    096         60        RTS
B82A      032 030 155 20 1E 9B  JSR &9B1E
B82D      176 007     B0 07     BCS 7   --> &B836
B82F  o   032 111 146 20 6F 92  JSR &926F
B832      169 128     A9 80     LDA#&80
B834  +   020 043     14 2B     TRB &2B
B836      032 205 128 20 CD 80  JSR &80CD
B839      144 185     90 B9     BCC -71   --> &B7F4
B83B `    096         60        RTS
B83C L    076 146 144 4C 92 90  JMP &9092
B83F Li   076 105 155 4C 69 9B  JMP &9B69
B842      132 010     84 0A     STY &0A
B844 L    076 002 144 4C 02 90  JMP &9002
B847  <   032 060 186 20 3C BA  JSR &BA3C
B84A  L   132 076     84 4C     STY &4C
B84C  u   032 117 146 20 75 92  JSR &9275
B84F      032 229 140 20 E5 8C  JSR &8CE5
B852      208 238     D0 EE     BNE -18   --> &B842
B854  L   165 076     A5 4C     LDA &4C
B856 H    072         48        PHA
B857      032 174 152 20 AE 98  JSR &98AE
B85A      240 227     F0 E3     BEQ -29   --> &B83F
B85C  u   032 117 146 20 75 92  JSR &9275
B85F h    104         68        PLA
B860  L   133 076     85 4C     STA &4C
B862      008         08        PHP
B863  &   032 038 188 20 26 BC  JSR &BC26
B866  L   164 076     A4 4C     LDY &4C
B868      032 215 255 20 D7 FF  JSR &FFD7
B86B  '   133 039     85 27     STA &27
B86D (    040         28        PLP
```

```
B86E      144 026     90 1A     BCC 26   --> &B88A
B870  '   165 039     A5 27     LDA &27
B872      208 200     D0 C8     BNE -56   --> &B83C
B874      032 215 255 20 D7 FF  JSR &FFD7
B877  6   133 054     85 36     STA &36
B879      170         AA        TAX
B87A      240 009     F0 09     BEQ 9   --> &B885
B87C      032 215 255 20 D7 FF  JSR &FFD7
B87F      157 255 005 9D FF 05  STA &05FF,X
B882      202         CA        DEX
B883      208 247     D0 F7     BNE -9   --> &B87C
B885      032 171 144 20 AB 90  JSR &90AB
B888      128 197     80 C5     BRA -59   --> &B84F
B88A  '   165 039     A5 27     LDA &27
B88C      240 174     F0 AE     BEQ -82   --> &B83C
B88E 0    048 012     30 0C     BMI 12   --> &B89C
B890      162 003     A2 03     LDX#&03
B892      032 215 255 20 D7 FF  JSR &FFD7
B895  *   149 042     95 2A     STA &2A,X
B897      202         CA        DEX
B898      016 248     10 F8     BPL -8   --> &B892
B89A      128 014     80 0E     BRA 14   --> &B8AA
B89C      162 004     A2 04     LDX#&04
B89E      032 215 255 20 D7 FF  JSR &FFD7
B8A1  l   157 108 004 9D 6C 04  STA &046C,X
B8A4      202         CA        DEX
B8A5      016 247     10 F7     BPL -9   --> &B89E
B8A7  9   032 057 165 20 39 A5  JSR &A539
B8AA      032 006 189 20 06 BD  JSR &BD06
B8AD  8   032 056 179 20 38 B3  JSR &B338
B8B0      128 157     80 9D     BRA -99   --> &B84F
B8B2 h    104         68        PLA
B8B3 h    104         68        PLA
B8B4      128 142     80 8E     BRA -114   --> &B844
B8B6      032 223 140 20 DF 8C  JSR &8CDF
B8B9      240 140     F0 8C     BEQ -116   --> &B847
B8BB      201 134     C9 86     CMP#&86
B8BD      240 003     F0 03     BEQ 3   --> &B8C2
B8BF      198 010     C6 0A     DEC &0A
B8C1      024         18        CLC
B8C2 fL   102 076     66 4C     ROR &4C
B8C4 FL   070 076     46 4C     LSR &4C
B8C6      169 255     A9 FF     LDA#&FF
```

```
B8C8  M   133 077     85 4D     STA &4D
B8CA      032 153 146 20 99 92  JSR &9299
B8CD      176 010     B0 0A     BCS 10   --> &B8D9
B8CF      032 153 146 20 99 92  JSR &9299
B8D2      144 251     90 FB     BCC -5   --> &B8CF
B8D4      162 255     A2 FF     LDX#&FF
B8D6  M   134 077     86 4D     STX &4D
B8D8      024         18        CLC
B8D9      008         08        PHP
B8DA  L   006 076     06 4C     ASL &4C
B8DC (    040         28        PLP
B8DD fL   102 076     66 4C     ROR &4C
B8DF  ,   201 044     C9 2C     CMP#&2C
B8E1      240 231     F0 E7     BEQ -25   --> &B8CA
B8E3  ;   201 059     C9 3B     CMP#&3B
B8E5      240 227     F0 E3     BEQ -29   --> &B8CA
B8E7      198 010     C6 0A     DEC &0A
B8E9  L   165 076     A5 4C     LDA &4C
B8EB H    072         48        PHA
B8EC  M   165 077     A5 4D     LDA &4D
B8EE H    072         48        PHA
B8EF      032 174 152 20 AE 98  JSR &98AE
B8F2      240 190     F0 BE     BEQ -66   --> &B8B2
B8F4 h    104         68        PLA
B8F5  M   133 077     85 4D     STA &4D
B8F7 h    104         68        PLA
B8F8  L   133 076     85 4C     STA &4C
B8FA  u   032 117 146 20 75 92  JSR &9275
B8FD      008         08        PHP
B8FE $L   036 076     24 4C     BIT &4C
B900 p    112 006     70 06     BVS 6   --> &B908
B902  M   165 077     A5 4D     LDA &4D
B904      201 255     C9 FF     CMP#&FF
B906      208 023     D0 17     BNE 23   --> &B91F
B908 $L   036 076     24 4C     BIT &4C
B90A      016 005     10 05     BPL 5   --> &B911
B90C  ?   169 063     A9 3F     LDA#&3F
B90E      032 238 255 20 EE FF  JSR &FFEE
B911  p   032 112 186 20 70 BA  JSR &BA70
B914  6   132 054     84 36     STY &36
B916  L   006 076     06 4C     ASL &4C
B918      024         18        CLC
B919 fL   102 076     66 4C     ROR &4C
```

```
B91B $L   036 076     24 4C     BIT &4C
B91D p    112 025     70 19     BVS 25   --> &B938
B91F      133 027     85 1B     STA &1B
B921 d    100 025     64 19     STZ &19
B923      169 006     A9 06     LDA#&06
B925      133 026     85 1A     STA &1A
B927      032 248 172 20 F8 AC  JSR &ACF8
B92A      032 235 142 20 EB 8E  JSR &8EEB
B92D      240 006     F0 06     BEQ 6   --> &B935
B92F      201 013     C9 0D     CMP#&0D
B931      208 247     D0 F7     BNE -9   --> &B92A
B933      160 254     A0 FE     LDY#&FE
B935      200         C8        INY
B936  M   132 077     84 4D     STY &4D
B938 (    040         28        PLP
B939      176 011     B0 0B     BCS 11   --> &B946
B93B  C   032 067 188 20 43 BC  JSR &BC43
B93E  N   032 078 171 20 4E AB  JSR &AB4E
B941  +   032 043 179 20 2B B3  JSR &B32B
B944      128 132     80 84     BRA -124   --> &B8CA
B946 d'   100 039     64 27     STZ &27
B948      032 174 144 20 AE 90  JSR &90AE
B94B      128 247     80 F7     BRA -9   --> &B944
B94D d=   100 061     64 3D     STZ &3D
B94F      164 024     A4 18     LDY &18
B951  >   132 062     84 3E     STY &3E
B953      032 224 142 20 E0 8E  JSR &8EE0
B956      198 010     C6 0A     DEC &0A
B958  :   201 058     C9 3A     CMP#&3A
B95A      240 011     F0 0B     BEQ 11   --> &B967
B95C      201 013     C9 0D     CMP#&0D
B95E      240 007     F0 07     BEQ 7   --> &B967
B960      201 139     C9 8B     CMP#&8B
B962      240 003     F0 03     BEQ 3   --> &B967
B964  *   032 042 184 20 2A B8  JSR &B82A
B967      032 166 155 20 A6 9B  JSR &9BA6
B96A  =   165 061     A5 3D     LDA &3D
B96C      133 028     85 1C     STA &1C
B96E  >   165 062     A5 3E     LDA &3E
B970      133 029     85 1D     STA &1D
B972 L    076 005 144 4C 05 90  JMP &9005
B975      032 229 140 20 E5 8C  JSR &8CE5
B978      240 003     F0 03     BEQ 3   --> &B97D
```

```
B97A L    076 000 144 4C 00 90  JMP &9000
B97D      032 174 152 20 AE 98  JSR &98AE
B980      240 243     F0 F3     BEQ -13   --> &B975
B982      176 011     B0 0B     BCS 11   --> &B98F
B984      032 172 185 20 AC B9  JSR &B9AC
B987  C   032 067 188 20 43 BC  JSR &BC43
B98A  (   032 040 179 20 28 B3  JSR &B328
B98D      128 014     80 0E     BRA 14   --> &B99D
B98F      032 172 185 20 AC B9  JSR &B9AC
B992  &   032 038 188 20 26 BC  JSR &BC26
B995      032 248 172 20 F8 AC  JSR &ACF8
B998  '   133 039     85 27     STA &27
B99A      032 171 144 20 AB 90  JSR &90AB
B99D      024         18        CLC
B99E      165 027     A5 1B     LDA &1B
B9A0 e    101 025     65 19     ADC &19
B9A2      133 028     85 1C     STA &1C
B9A4      165 026     A5 1A     LDA &1A
B9A6 i    105 000     69 00     ADC#&00
B9A8      133 029     85 1D     STA &1D
B9AA      128 201     80 C9     BRA -55   --> &B975
B9AC  u   032 117 146 20 75 92  JSR &9275
B9AF      165 028     A5 1C     LDA &1C
B9B1      133 025     85 19     STA &19
B9B3      165 029     A5 1D     LDA &1D
B9B5      133 026     85 1A     STA &1A
B9B7 d    100 027     64 1B     STZ &1B
B9B9      032 235 142 20 EB 8E  JSR &8EEB
B9BC  X   240 088     F0 58     BEQ 88   --> &BA16
B9BE      201 220     C9 DC     CMP#&DC
B9C0  T   240 084     F0 54     BEQ 84   --> &BA16
B9C2      201 013     C9 0D     CMP#&0D
B9C4      240 009     F0 09     BEQ 9   --> &B9CF
B9C6      032 235 142 20 EB 8E  JSR &8EEB
B9C9  K   240 075     F0 4B     BEQ 75   --> &BA16
B9CB      201 013     C9 0D     CMP#&0D
B9CD      208 247     D0 F7     BNE -9   --> &B9C6
B9CF      164 027     A4 1B     LDY &1B
B9D1      177 025     B1 19     LDA (&19),Y
B9D3 0    048 028     30 1C     BMI 28   --> &B9F1
B9D5      200         C8        INY
B9D6      200         C8        INY
B9D7      177 025     B1 19     LDA (&19),Y
```

```
B9D9     170        AA       TAX
B9DA     200        C8       INY
B9DB     177 025    B1 19    LDA (&19),Y
B9DD     201 032    C9 20    CMP#&20
B9DF     240 249    F0 F9    BEQ -7   --> &B9DA
B9E1     201 220    C9 DC    CMP#&DC
B9E3 .   240 046    F0 2E    BEQ 46   --> &BA13
B9E5     138        8A       TXA
B9E6     024        18       CLC
B9E7 e   101 025    65 19    ADC &19
B9E9     133 025    85 19    STA &19
B9EB     144 226    90 E2    BCC -30   --> &B9CF
B9ED     230 026    E6 1A    INC &1A
B9EF     128 222    80 DE    BRA -34   --> &B9CF
B9F1     000        00       BRK
B9F2 *   042        2A       EQUB &2A
B9F3 O   079        4F       xxx Invalid Code
B9F4 ut  117 116    75 74    ADC &74,X
B9F6  of 032 111 102 20 6F 66  JSR &666F
B9F9     032 220    20 DC    xxx Invalid Code
B9FB     000        00       BRK
B9FC +   043        2B       EQUB &2B
B9FD No  078 111 032 4E 6F 20  LSR &206F
BA00     245        F5       xxx Invalid Code
BA01     000        00       BRK
BA02 -   045        2D       EQUB &2D
BA03  #  141 035    8D 23    xxx Invalid Code
BA05     000        00       BRK
BA06 ,   044        2C       EQUB &2C
BA07 To  084 111    54 6F    xxx Invalid Code
BA09 o   111        6F       xxx Invalid Code
BA0A  ma 032 109 097 20 6D 61  JSR &616D
BA0D ny  110 121 032 6E 79 20  ROR &2079
BA10  s  245 115    F5 73    SBC &73,X
BA12     000        00       EQUB &00
BA13     200        C8       INY
BA14     132 027    84 1B    STY &1B
BA16 `   096        60       RTS
BA17  /  032 047 157 20 2F 9D  JSR &9D2F
BA1A     032 145 155 20 91 9B  JSR &9B91
BA1D     032 188 150 20 BC 96  JSR &96BC
BA20  $  166 036    A6 24    LDX &24
BA22     240 215    F0 D7    BEQ -41   --> &B9FB
```

```
BA24  *   165 042     A5 2A     LDA &2A
BA26  +   005 043     05 2B     ORA &2B
BA28  ,   005 044     05 2C     ORA &2C
BA2A  -   005 045     05 2D     ORA &2D
BA2C      240 005     F0 05     BEQ 5   --> &BA33
BA2E  $   198 036     C6 24     DEC &24
BA30 L    076 005 144 4C 05 90  JMP &9005
BA33      188 255 004 BC FF 04  LDY &04FF,X
BA36      189 019 005 BD 13 05  LDA &0513,X
BA39 L2   076 050 183 4C 32 B7  JMP &B732
BA3C      198 010     C6 0A     DEC &0A
BA3E      165 010     A5 0A     LDA &0A
BA40      133 027     85 1B     STA &1B
BA42      165 011     A5 0B     LDA &0B
BA44      133 025     85 19     STA &19
BA46      165 012     A5 0C     LDA &0C
BA48      133 026     85 1A     STA &1A
BA4A      032 213 142 20 D5 8E  JSR &8ED5
BA4D  #   201 035     C9 23     CMP#&23
BA4F      208 176     D0 B0     BNE -80   --> &BA01
BA51      032 180 150 20 B4 96  JSR &96B4
BA54  *   164 042     A4 2A     LDY &2A
BA56      152         98        TYA
BA57  `   096         60        RTS
BA58  $   166 036     A6 24     LDX &24
BA5A      224 020     E0 14     CPX#&14
BA5C      176 167     B0 A7     BCS -89   --> &BA05
BA5E      032 188 155 20 BC 9B  JSR &9BBC
BA61      165 011     A5 0B     LDA &0B
BA63      157 000 005 9D 00 05  STA &0500,X
BA66      165 012     A5 0C     LDA &0C
BA68      157 020 005 9D 14 05  STA &0514,X
BA6B  $   230 036     E6 24     INC &24
BA6D L    076 011 144 4C 0B 90  JMP &900B
BA70      169 006     A9 06     LDA#&06
BA72      128 002     80 02     BRA 2   --> &BA76
BA74      169 007     A9 07     LDA#&07
BA76 d7   100 055     64 37     STZ &37
BA78  8   133 056     85 38     STA &38
BA7A      169 238     A9 EE     LDA#&EE
BA7C  9   133 057     85 39     STA &39
BA7E      169 032     A9 20     LDA#&20
BA80  :   133 058     85 3A     STA &3A
```

```
BA82      160 255     A0 FF     LDY#&FF
BA84  ;   132 059     84 3B     STY &3B
BA86      200         C8        INY
BA87  7   162 055     A2 37     LDX#&37
BA89      152         98        TYA
BA8A      032 241 255 20 F1 FF  JSR &FFF1
BA8D      144 006     90 06     BCC 6   --> &BA95
BA8F L}   076 125 155 4C 7D 9B  JMP &9B7D
BA92      032 231 255 20 E7 FF  JSR &FFE7
BA95 d    100 030     64 1E     STZ &1E
BA97 `    096         60        RTS
BA98      032 205 128 20 CD 80  JSR &80CD
BA9B  M   144 077     90 4D     BCC 77   --> &BAEA
BA9D  =   165 061     A5 3D     LDA &3D
BA9F  7   133 055     85 37     STA &37
BAA1      133 018     85 12     STA &12
BAA3  >   165 062     A5 3E     LDA &3E
BAA5  8   133 056     85 38     STA &38
BAA7      133 019     85 13     STA &13
BAA9      160 003     A0 03     LDY#&03
BAAB  7   177 055     B1 37     LDA (&37),Y
BAAD      024         18        CLC
BAAE e7   101 055     65 37     ADC &37
BAB0  7   133 055     85 37     STA &37
BAB2      144 002     90 02     BCC 2   --> &BAB6
BAB4  8   230 056     E6 38     INC &38
BAB6      160 000     A0 00     LDY#&00
BAB8  7   177 055     B1 37     LDA (&37),Y
BABA      145 018     91 12     STA (&12),Y
BABC      201 013     C9 0D     CMP#&0D
BABE      208 019     D0 13     BNE 19   --> &BAD3
BAC0      200         C8        INY
BAC1      208 004     D0 04     BNE 4   --> &BAC7
BAC3  8   230 056     E6 38     INC &38
BAC5      230 019     E6 13     INC &13
BAC7  7   177 055     B1 37     LDA (&37),Y
BAC9      145 018     91 12     STA (&12),Y
BACB 0    048 015     30 0F     BMI 15   --> &BADC
BACD      032 223 186 20 DF BA  JSR &BADF
BAD0      032 223 186 20 DF BA  JSR &BADF
BAD3      200         C8        INY
BAD4      208 226     D0 E2     BNE -30   --> &BAB8
BAD6  8   230 056     E6 38     INC &38
```

```
BAD8      230 019     E6 13     INC &13
BADA      128 220     80 DC     BRA -36   --> &BAB8
BADC L    076 005 190 4C 05 BE  JMP &BE05
BADF      200         C8        INY
BAE0      208 004     D0 04     BNE 4   --> &BAE6
BAE2      230 019     E6 13     INC &13
BAE4  8   230 056     E6 38     INC &38
BAE6  7   177 055     B1 37     LDA (&37),Y
BAE8      145 018     91 12     STA (&12),Y
BAEA `    096         60        RTS
BAEB      162 255     A2 FF     LDX#&FF
BAED  (   134 040     86 28     STX &28
BAEF  <   134 060     86 3C     STX &3C
BAF1      032 207 187 20 CF BB  JSR &BBCF
BAF4      165 011     A5 0B     LDA &0B
BAF6  7   133 055     85 37     STA &37
BAF8      165 012     A5 0C     LDA &0C
BAFA  8   133 056     85 38     STA &38
BAFC d;   100 059     64 3B     STZ &3B
BAFE d    100 010     64 0A     STZ &0A
BB00      032 178 141 20 B2 8D  JSR &8DB2
BB03      032 030 155 20 1E 9B  JSR &9B1E
BB06      144 226     90 E2     BCC -30   --> &BAEA
BB08      165 031     A5 1F     LDA &1F
BB0A      240 009     F0 09     BEQ 9   --> &BB15
BB0C      185 000 007 B9 00 07  LDA &0700,Y
BB0F      200         C8        INY
BB10      201 032     C9 20     CMP#&20
BB12      240 248     F0 F8     BEQ -8   --> &BB0C
BB14      136         88        DEY
BB15  ;   132 059     84 3B     STY &3B
BB17      032 152 186 20 98 BA  JSR &BA98
BB1A      160 007     A0 07     LDY#&07
BB1C  <   132 060     84 3C     STY &3C
BB1E      160 000     A0 00     LDY#&00
BB20      169 013     A9 0D     LDA#&0D
BB22  ;   210 059     D2 3B     CMP (&3B)
BB24      240 196     F0 C4     BEQ -60   --> &BAEA
BB26      200         C8        INY
BB27  ;   209 059     D1 3B     CMP (&3B),Y
BB29      208 251     D0 FB     BNE -5   --> &BB26
BB2B      169 032     A9 20     LDA#&20
BB2D      136         88        DEY
```

```
BB2E      240 004     F0 04     BEQ 4   --> &BB34
BB30  ;   209 059     D1 3B     CMP (&3B),Y
BB32      240 249     F0 F9     BEQ -7   --> &BB2D
BB34      200         C8        INY
BB35      169 013     A9 0D     LDA#&0D
BB37  ;   145 059     91 3B     STA (&3B),Y
BB39      200         C8        INY
BB3A      200         C8        INY
BB3B      200         C8        INY
BB3C      200         C8        INY
BB3D  ?   132 063     84 3F     STY &3F
BB3F      165 018     A5 12     LDA &12
BB41  9   133 057     85 39     STA &39
BB43      165 019     A5 13     LDA &13
BB45  :   133 058     85 3A     STA &3A
BB47      032 004 190 20 04 BE  JSR &BE04
BB4A  7   133 055     85 37     STA &37
BB4C      165 019     A5 13     LDA &13
BB4E  8   133 056     85 38     STA &38
BB50      136         88        DEY
BB51      165 006     A5 06     LDA &06
BB53      197 018     C5 12     CMP &12
BB55      165 007     A5 07     LDA &07
BB57      229 019     E5 13     SBC &13
BB59      176 016     B0 10     BCS 16   --> &BB6B
BB5B      032 229 189 20 E5 BD  JSR &BDE5
BB5E      032 172 187 20 AC BB  JSR &BBAC
BB61      000         00        BRK
BB62      000         00        EQUB &00
BB63      134 032     86 20     STX &20
BB65 s    115         73        xxx Invalid Code
BB66 pa   112 097     70 61     BVS 97   --> &BBC9
BB68 c    099         63        xxx Invalid Code
BB69 e    101         65        xxx Invalid Code
BB6A      000         00        EQUB &00
BB6B  9   177 057     B1 39     LDA (&39),Y
BB6D  7   145 055     91 37     STA (&37),Y
BB6F      152         98        TYA
BB70      208 004     D0 04     BNE 4   --> &BB76
BB72  :   198 058     C6 3A     DEC &3A
BB74  8   198 056     C6 38     DEC &38
BB76      136         88        DEY
BB77      152         98        TYA
```

```
BB78 e9   101 057     65 39     ADC &39
BB7A  :   166 058     A6 3A     LDX &3A
BB7C      144 001     90 01     BCC 1   --> &BB7F
BB7E      232         E8        INX
BB7F  =   197 061     C5 3D     CMP &3D
BB81      138         8A        TXA
BB82  >   229 062     E5 3E     SBC &3E
BB84      176 229     B0 E5     BCS -27   --> &BB6B
BB86      160 001     A0 01     LDY#&01
BB88  +   165 043     A5 2B     LDA &2B
BB8A  =   145 061     91 3D     STA (&3D),Y
BB8C      200         C8        INY
BB8D  *   165 042     A5 2A     LDA &2A
BB8F  =   145 061     91 3D     STA (&3D),Y
BB91      200         C8        INY
BB92  ?   165 063     A5 3F     LDA &3F
BB94  =   145 061     91 3D     STA (&3D),Y
BB96 8    056         38        SEC
BB97      152         98        TYA
BB98 e=   101 061     65 3D     ADC &3D
BB9A  =   133 061     85 3D     STA &3D
BB9C      144 002     90 02     BCC 2   --> &BBA0
BB9E  >   230 062     E6 3E     INC &3E
BBA0      160 255     A0 FF     LDY#&FF
BBA2      200         C8        INY
BBA3  ;   177 059     B1 3B     LDA (&3B),Y
BBA5  =   145 061     91 3D     STA (&3D),Y
BBA7      201 013     C9 0D     CMP#&0D
BBA9      208 247     D0 F7     BNE -9   --> &BBA2
BBAB `    096         60        RTS
BBAC      165 018     A5 12     LDA &12
BBAE      133 000     85 00     STA &00
BBB0      133 002     85 02     STA &02
BBB2      165 019     A5 13     LDA &13
BBB4      133 001     85 01     STA &01
BBB6      133 003     85 03     STA &03
BBB8      032 207 187 20 CF BB  JSR &BBCF
BBBB      162 016     A2 10     LDX#&10
BBBD      189 019 191 BD 13 BF  LDA &BF13,X
BBC0      157 239 007 9D EF 07  STA &07EF,X
BBC3      202         CA        DEX
BBC4      208 247     D0 F7     BNE -9   --> &BBBD
BBC6      162 128     A2 80     LDX#&80
```

```
BBC8      158 127 004 9E 7F 04  STZ &047F,X
BBCB      202         CA        DEX
BBCC      208 250     D0 FA     BNE -6   --> &BBC8
BBCE `    096         60        RTS
BBCF      165 024     A5 18     LDA &18
BBD1      133 029     85 1D     STA &1D
BBD3      165 006     A5 06     LDA &06
BBD5      133 004     85 04     STA &04
BBD7      165 007     A5 07     LDA &07
BBD9      133 005     85 05     STA &05
BBDB      169 128     A9 80     LDA#&80
BBDD      020 031     14 1F     TRB &1F
BBDF d$   100 036     64 24     STZ &24
BBE1 d&   100 038     64 26     STZ &26
BBE3 d%   100 037     64 25     STZ &25
BBE5 d    100 028     64 1C     STZ &1C
BBE7 `    096         60        RTS
BBE8      165 004     A5 04     LDA &04
BBEA      024         18        CLC
BBEB  J   133 074     85 4A     STA &4A
BBED i    105 005     69 05     ADC#&05
BBEF      133 004     85 04     STA &04
BBF1      165 005     A5 05     LDA &05
BBF3  K   133 075     85 4B     STA &4B
BBF5 i    105 000     69 00     ADC#&00
BBF7      133 005     85 05     STA &05
BBF9 `    096         60        RTS
BBFA      165 004     A5 04     LDA &04
BBFC 8    056         38        SEC
BBFD      233 005     E9 05     SBC#&05
BBFF      032 030 189 20 1E BD  JSR &BD1E
BC02  0   165 048     A5 30     LDA &30
BC04      146 004     92 04     STA (&04)
BC06      160 001     A0 01     LDY#&01
BC08  .   165 046     A5 2E     LDA &2E
BC0A E1   069 049     45 31     EOR &31
BC0C )    041 128     29 80     AND#&80
BC0E E1   069 049     45 31     EOR &31
BC10      145 004     91 04     STA (&04),Y
BC12      200         C8        INY
BC13  2   165 050     A5 32     LDA &32
BC15      145 004     91 04     STA (&04),Y
BC17      200         C8        INY
```

```
BC18  3   165 051     A5 33     LDA &33
BC1A      145 004     91 04     STA (&04),Y
BC1C      200         C8        INY
BC1D  4   165 052     A5 34     LDA &34
BC1F      145 004     91 04     STA (&04),Y
BC21 `    096         60        RTS
BC22  -   240 045     F0 2D     BEQ 45   --> &BC51
BC24 0    048 212     30 D4     BMI -44   --> &BBFA
BC26      165 004     A5 04     LDA &04
BC28 8    056         38        SEC
BC29      233 004     E9 04     SBC#&04
BC2B      032 030 189 20 1E BD  JSR &BD1E
BC2E      160 003     A0 03     LDY#&03
BC30  -   165 045     A5 2D     LDA &2D
BC32      145 004     91 04     STA (&04),Y
BC34      136         88        DEY
BC35  ,   165 044     A5 2C     LDA &2C
BC37      145 004     91 04     STA (&04),Y
BC39      136         88        DEY
BC3A  +   165 043     A5 2B     LDA &2B
BC3C      145 004     91 04     STA (&04),Y
BC3E  *   165 042     A5 2A     LDA &2A
BC40      146 004     92 04     STA (&04)
BC42 `    096         60        RTS
BC43 z    122         7A        PLY
BC44      250         FA        PLX
BC45  *   165 042     A5 2A     LDA &2A
BC47 H    072         48        PHA
BC48  +   165 043     A5 2B     LDA &2B
BC4A H    072         48        PHA
BC4B  ,   165 044     A5 2C     LDA &2C
BC4D H    072         48        PHA
BC4E      218         DA        PHX
BC4F Z    090         5A        PHY
BC50 `    096         60        RTS
BC51      024         18        CLC
BC52      165 004     A5 04     LDA &04
BC54  6   229 054     E5 36     SBC &36
BC56      032 030 189 20 1E BD  JSR &BD1E
BC59  6   164 054     A4 36     LDY &36
BC5B      240 008     F0 08     BEQ 8   --> &BC65
BC5D      185 255 005 B9 FF 05  LDA &05FF,Y
BC60      145 004     91 04     STA (&04),Y
```

```
BC62      136         88        DEY
BC63      208 248     D0 F8     BNE -8   --> &BC5D
BC65  6   165 054     A5 36     LDA &36
BC67      146 004     92 04     STA (&04)
BC69 `    096         60        RTS
BC6A  9   165 057     A5 39     LDA &39
BC6C      201 128     C9 80     CMP#&80
BC6E  %   240 037     F0 25     BEQ 37   --> &BC95
BC70  8   144 056     90 38     BCC 56   --> &BCAA
BC72      178 004     B2 04     LDA (&04)
BC74      170         AA        TAX
BC75      240 022     F0 16     BEQ 22   --> &BC8D
BC77  7   178 055     B2 37     LDA (&37)
BC79      233 001     E9 01     SBC#&01
BC7B  9   133 057     85 39     STA &39
BC7D      160 001     A0 01     LDY#&01
BC7F  7   177 055     B1 37     LDA (&37),Y
BC81      233 000     E9 00     SBC#&00
BC83  :   133 058     85 3A     STA &3A
BC85      177 004     B1 04     LDA (&04),Y
BC87  9   145 057     91 39     STA (&39),Y
BC89      200         C8        INY
BC8A      202         CA        DEX
BC8B      208 248     D0 F8     BNE -8   --> &BC85
BC8D      178 004     B2 04     LDA (&04)
BC8F      160 003     A0 03     LDY#&03
BC91  7   145 055     91 37     STA (&37),Y
BC93  L   128 076     80 4C     BRA 76   --> &BCE1
BC95      178 004     B2 04     LDA (&04)
BC97      170         AA        TAX
BC98      240 012     F0 0C     BEQ 12   --> &BCA6
BC9A      160 001     A0 01     LDY#&01
BC9C      177 004     B1 04     LDA (&04),Y
BC9E      136         88        DEY
BC9F  7   145 055     91 37     STA (&37),Y
BCA1      200         C8        INY
BCA2      200         C8        INY
BCA3      202         CA        DEX
BCA4      208 246     D0 F6     BNE -10  --> &BC9C
BCA6      169 013     A9 0D     LDA#&0D
BCA8      208 231     D0 E7     BNE -25  --> &BC91
BCAA      178 004     B2 04     LDA (&04)
BCAC  7   146 055     92 37     STA (&37)
```

```
BCAE      160 004     A0 04     LDY#&04
BCB0  9   165 057     A5 39     LDA &39
BCB2      240 026     F0 1A     BEQ 26   --> &BCCE
BCB4      160 001     A0 01     LDY#&01
BCB6      177 004     B1 04     LDA (&04),Y
BCB8  7   145 055     91 37     STA (&37),Y
BCBA      200         C8        INY
BCBB      177 004     B1 04     LDA (&04),Y
BCBD  7   145 055     91 37     STA (&37),Y
BCBF      200         C8        INY
BCC0      177 004     B1 04     LDA (&04),Y
BCC2  7   145 055     91 37     STA (&37),Y
BCC4      200         C8        INY
BCC5  9   196 057     C4 39     CPY &39
BCC7      176 005     B0 05     BCS 5   --> &BCCE
BCC9      177 004     B1 04     LDA (&04),Y
BCCB  7   145 055     91 37     STA (&37),Y
BCCD      200         C8        INY
BCCE      152         98        TYA
BCCF      024         18        CLC
BCD0  +   128 043     80 2B     BRA 43   --> &BCFD
BCD2      178 004     B2 04     LDA (&04)
BCD4  6   133 054     85 36     STA &36
BCD6      240 011     F0 0B     BEQ 11   --> &BCE3
BCD8      168         A8        TAY
BCD9      177 004     B1 04     LDA (&04),Y
BCDB      153 255 005 99 FF 05  STA &05FF,Y
BCDE      136         88        DEY
BCDF      208 248     D0 F8     BNE -8   --> &BCD9
BCE1      178 004     B2 04     LDA (&04)
BCE3 8    056         38        SEC
BCE4      128 023     80 17     BRA 23   --> &BCFD
BCE6      160 003     A0 03     LDY#&03
BCE8      177 004     B1 04     LDA (&04),Y
BCEA  -   133 045     85 2D     STA &2D
BCEC      136         88        DEY
BCED      177 004     B1 04     LDA (&04),Y
BCEF  ,   133 044     85 2C     STA &2C
BCF1      136         88        DEY
BCF2      177 004     B1 04     LDA (&04),Y
BCF4  +   133 043     85 2B     STA &2B
BCF6      178 004     B2 04     LDA (&04)
BCF8  *   133 042     85 2A     STA &2A
```

```
BCFA      024         18        CLC
BCFB      169 004     A9 04     LDA#&04
BCFD e    101 004     65 04     ADC &04
BCFF      133 004     85 04     STA &04
BD01      144 002     90 02     BCC 2   --> &BD05
BD03      230 005     E6 05     INC &05
BD05 `    096         60        RTS
BD06  7   162 055     A2 37     LDX#&37
BD08      160 003     A0 03     LDY#&03
BD0A      177 004     B1 04     LDA (&04),Y
BD0C      149 003     95 03     STA &03,X
BD0E      136         88        DEY
BD0F      177 004     B1 04     LDA (&04),Y
BD11      149 002     95 02     STA &02,X
BD13      136         88        DEY
BD14      177 004     B1 04     LDA (&04),Y
BD16      149 001     95 01     STA &01,X
BD18      178 004     B2 04     LDA (&04)
BD1A      149 000     95 00     STA &00,X
BD1C      128 220     80 DC     BRA -36   --> &BCFA
BD1E      133 004     85 04     STA &04
BD20      176 002     B0 02     BCS 2   --> &BD24
BD22      198 005     C6 05     DEC &05
BD24      164 005     A4 05     LDY &05
BD26      196 003     C4 03     CPY &03
BD28      144 010     90 0A     BCC 10   --> &BD34
BD2A      208 004     D0 04     BNE 4   --> &BD30
BD2C      197 002     C5 02     CMP &02
BD2E      144 004     90 04     BCC 4   --> &BD34
BD30 `    096         60        RTS
BD31      032 172 187 20 AC BB  JSR &BBAC
BD34 L    076 161 144 4C A1 90  JMP &90A1
BD37  7   133 055     85 37     STA &37
BD39      201 128     C9 80     CMP#&80
BD3B  W   144 087     90 57     BCC 87   --> &BD94
BD3D  V   169 086     A9 56     LDA#&56
BD3F  8   133 056     85 38     STA &38
BD41      169 132     A9 84     LDA#&84
BD43  9   133 057     85 39     STA &39
BD45 Z    090         5A        PHY
BD46      160 000     A0 00     LDY#&00
BD48      200         C8        INY
BD49  8   177 056     B1 38     LDA (&38),Y
```

```
BD4B      016 251     10 FB     BPL -5   --> &BD48
BD4D  7   197 055     C5 37     CMP &37
BD4F      240 013     F0 0D     BEQ 13   --> &BD5E
BD51      200         C8        INY
BD52      152         98        TYA
BD53 8    056         38        SEC
BD54 e8   101 056     65 38     ADC &38
BD56  8   133 056     85 38     STA &38
BD58      144 236     90 EC     BCC -20   --> &BD46
BD5A  9   230 057     E6 39     INC &39
BD5C      128 232     80 E8     BRA -24   --> &BD46
BD5E      160 000     A0 00     LDY#&00
BD60  8   177 056     B1 38     LDA (&38),Y
BD62 0    048 006     30 06     BMI 6   --> &BD6A
BD64      032 148 189 20 94 BD  JSR &BD94
BD67      200         C8        INY
BD68      208 246     D0 F6     BNE -10   --> &BD60
BD6A z    122         7A        PLY
BD6B `    096         60        RTS
BD6C H    072         48        PHA
BD6D J    074         4A        LSR A
BD6E J    074         4A        LSR A
BD6F J    074         4A        LSR A
BD70 J    074         4A        LSR A
BD71  w   032 119 189 20 77 BD  JSR &BD77
BD74 h    104         68        PLA
BD75 )    041 015     29 0F     AND#&0F
BD77      201 010     C9 0A     CMP#&0A
BD79      144 002     90 02     BCC 2   --> &BD7D
BD7B i    105 006     69 06     ADC#&06
BD7D i0   105 048     69 30     ADC#&30
BD7F H    072         48        PHA
BD80  #   165 035     A5 23     LDA &23
BD82      197 030     C5 1E     CMP &1E
BD84      176 003     B0 03     BCS 3   --> &BD89
BD86      032 146 186 20 92 BA  JSR &BA92
BD89 h    104         68        PLA
BD8A      230 030     E6 1E     INC &1E
BD8C l    108 014 002 6C 0E 02  JMP (&020E)
BD8F  l   032 108 189 20 6C BD  JSR &BD6C
BD92      169 032     A9 20     LDA#&20
BD94 $    036 031     24 1F     BIT &1F
BD96 0    048 010     30 0A     BMI 10   --> &BDA2
```

```
BD98      201 013     C9 0D     CMP#&0D
BD9A      208 227     D0 E3     BNE -29   --> &BD7F
BD9C      032 238 255 20 EE FF  JSR &FFEE
BD9F L    076 149 186 4C 95 BA  JMP &BA95
BDA2      146 002     92 02     STA (&02)
BDA4      230 002     E6 02     INC &02
BDA6      208 029     D0 1D     BNE 29   --> &BDC5
BDA8      230 003     E6 03     INC &03
BDAA H    072         48        PHA
BDAB      165 003     A5 03     LDA &03
BDAD E    069 007     45 07     EOR &07
BDAF      240 128     F0 80     BEQ 128   --> &BE31
BDB1 h    104         68        PLA
BDB2 `    096         60        RTS
BDB3      024         18        CLC
BDB4 %    037 031     25 1F     AND &1F
BDB6      240 013     F0 0D     BEQ 13   --> &BDC5
BDB8      138         8A        TXA
BDB9 0    048 010     30 0A     BMI 10   --> &BDC5
BDBB *    042         2A        ROL A
BDBC      170         AA        TAX
BDBD      240 006     F0 06     BEQ 6   --> &BDC5
BDBF      032 146 189 20 92 BD  JSR &BD92
BDC2      202         CA        DEX
BDC3      208 250     D0 FA     BNE -6   --> &BDBF
BDC5 `    096         60        RTS
BDC6  *   165 042     A5 2A     LDA &2A
BDC8      149 000     95 00     STA &00,X
BDCA  +   165 043     A5 2B     LDA &2B
BDCC      149 001     95 01     STA &01,X
BDCE  ,   165 044     A5 2C     LDA &2C
BDD0      149 002     95 02     STA &02,X
BDD2  -   165 045     A5 2D     LDA &2D
BDD4      149 003     95 03     STA &03,X
BDD6 `    096         60        RTS
BDD7  A   032 065 190 20 41 BE  JSR &BE41
BDDA d=   100 061     64 3D     STZ &3D
BDDC      160 000     A0 00     LDY#&00
BDDE      169 255     A9 FF     LDA#&FF
BDE0  7   162 055     A2 37     LDX#&37
BDE2      032 221 255 20 DD FF  JSR &FFDD
BDE5      165 024     A5 18     LDA &18
BDE7      133 019     85 13     STA &13
```

```
BDE9 d    100 018     64 12     STZ &12
BDEB      160 001     A0 01     LDY#&01
BDED      178 018     B2 12     LDA (&12)
BDEF      201 013     C9 0D     CMP#&0D
BDF1      208 030     D0 1E     BNE 30   --> &BE11
BDF3      177 018     B1 12     LDA (&12),Y
BDF5 0    048 012     30 0C     BMI 12   --> &BE03
BDF7      160 003     A0 03     LDY#&03
BDF9      177 018     B1 12     LDA (&12),Y
BDFB      240 020     F0 14     BEQ 20   --> &BE11
BDFD      024         18        CLC
BDFE      032 006 190 20 06 BE  JSR &BE06
BE01      128 234     80 EA     BRA -22   --> &BDED
BE03      200         C8        INY
BE04      024         18        CLC
BE05      152         98        TYA
BE06 e    101 018     65 12     ADC &12
BE08      133 018     85 12     STA &12
BE0A      144 002     90 02     BCC 2   --> &BE0E
BE0C      230 019     E6 13     INC &13
BE0E      160 001     A0 01     LDY#&01
BE10 `    096         60        RTS
BE11      032 207 190 20 CF BE  JSR &BECF
BE14  Ba  013 066 097 0D 42 61  ORA &6142
BE17 d    100 032     64 20     STZ &20
BE19 pr   112 114     70 72     BVS 114   --> &BE8D
BE1B o    111         6F        xxx Invalid Code
BE1C g    103         67        xxx Invalid Code
BE1D ra   114 097     72 61     ADC (&61)
BE1F m    109 013 234 6D 0D EA  ADC &EA0D
BE22 L    076 134 143 4C 86 8F  JMP &8F86
BE25 d7   100 055     64 37     STZ &37
BE27      169 006     A9 06     LDA#&06
BE29  8   133 056     85 38     STA &38
BE2B  6   164 054     A4 36     LDY &36
BE2D      169 013     A9 0D     LDA#&0D
BE2F      153 000 006 99 00 06  STA &0600,Y
BE32 `    096         60        RTS
BE33 L    076 146 144 4C 92 90  JMP &9092
BE36  /   032 047 157 20 2F 9D  JSR &9D2F
BE39      208 248     D0 F8     BNE -8   --> &BE33
BE3B  %   032 037 190 20 25 BE  JSR &BE25
BE3E L    076 145 155 4C 91 9B  JMP &9B91
```

```
BE41  6   032 054 190 20 36 BE  JSR &BE36
BE44      136         88        DEY
BE45  9   132 057     84 39     STY &39
BE47      165 024     A5 18     LDA &18
BE49  :   133 058     85 3A     STA &3A
BE4B      169 130     A9 82     LDA#&82
BE4D      032 244 255 20 F4 FF  JSR &FFF4
BE50  ;   134 059     86 3B     STX &3B
BE52  <   132 060     84 3C     STY &3C
BE54 `    096         60        RTS
BE55      032 229 189 20 E5 BD  JSR &BDE5
BE58  A   032 065 190 20 41 BE  JSR &BE41
BE5B  ?   134 063     86 3F     STX &3F
BE5D  @   132 064     84 40     STY &40
BE5F  C   134 067     86 43     STX &43
BE61  D   132 068     84 44     STY &44
BE63  G   134 071     86 47     STX &47
BE65  H   132 072     84 48     STY &48
BE67 dA   100 065     64 41     STZ &41
BE69      166 018     A6 12     LDX &12
BE6B  E   134 069     86 45     STX &45
BE6D      166 019     A6 13     LDX &13
BE6F  F   134 070     86 46     STX &46
BE71  +   162 043     A2 2B     LDX#&2B
BE73  =   134 061     86 3D     STX &3D
BE75      162 128     A2 80     LDX#&80
BE77  >   134 062     86 3E     STX &3E
BE79      166 024     A6 18     LDX &18
BE7B  B   134 066     86 42     STX &42
BE7D      169 000     A9 00     LDA#&00
BE7F      168         A8        TAY
BE80  7   162 055     A2 37     LDX#&37
BE82      032 221 255 20 DD FF  JSR &FFDD
BE85  $   128 036     80 24     BRA 36   --> &BEAB
BE87  6   032 054 190 20 36 BE  JSR &BE36
BE8A      162 000     A2 00     LDX#&00
BE8C      160 006     A0 06     LDY#&06
BE8E      032 247 255 20 F7 FF  JSR &FFF7
BE91      128 024     80 18     BRA 24   --> &BEAB
BE93      169 003     A9 03     LDA#&03
BE95      128 002     80 02     BRA 2   --> &BE99
BE97      169 001     A9 01     LDA#&01
BE99 H    072         48        PHA
```

```
BE9A  >   032 062 186 20 3E BA  JSR &BA3E
BE9D Z    090         5A        PHY
BE9E  R   032 082 155 20 52 9B  JSR &9B52
BEA1      032 188 150 20 BC 96  JSR &96BC
BEA4 z    122         7A        PLY
BEA5  *   162 042     A2 2A     LDX#&2A
BEA7 h    104         68        PLA
BEA8      032 218 255 20 DA FF  JSR &FFDA
BEAB L    076 005 144 4C 05 90  JMP &9005
BEAE  >   032 062 186 20 3E BA  JSR &BA3E
BEB1      032 150 155 20 96 9B  JSR &9B96
BEB4  *   164 042     A4 2A     LDY &2A
BEB6      169 000     A9 00     LDA#&00
BEB8      032 206 255 20 CE FF  JSR &FFCE
BEBB      128 238     80 EE     BRA -18   --> &BEAB
BEBD  >   032 062 186 20 3E BA  JSR &BA3E
BEC0 H    072         48        PHA
BEC1      032 172 150 20 AC 96  JSR &96AC
BEC4      032 150 155 20 96 9B  JSR &9B96
BEC7 z    122         7A        PLY
BEC8  *   165 042     A5 2A     LDA &2A
BECA      032 212 255 20 D4 FF  JSR &FFD4
BECD      128 220     80 DC     BRA -36   --> &BEAB
BECF h    104         68        PLA
BED0  7   133 055     85 37     STA &37
BED2 h    104         68        PLA
BED3  8   133 056     85 38     STA &38
BED5      128 003     80 03     BRA 3   --> &BEDA
BED7      032 227 255 20 E3 FF  JSR &FFE3
BEDA      032 169 141 20 A9 8D  JSR &8DA9
BEDD      016 248     10 F8     BPL -8   --> &BED7
BEDF l7   108 055 000 6C 37 00  JMP (&0037)
BEE2      169 005     A9 05     LDA#&05
BEE4      218         DA        PHX
BEE5  *   162 042     A2 2A     LDX#&2A
BEE7      160 000     A0 00     LDY#&00
BEE9      032 241 255 20 F1 FF  JSR &FFF1
BEEC      250         FA        PLX
BEED  .   165 046     A5 2E     LDA &2E
BEEF  *   230 042     E6 2A     INC &2A
BEF1      208 010     D0 0A     BNE 10   --> &BEFD
BEF3  +   230 043     E6 2B     INC &2B
BEF5      208 006     D0 06     BNE 6   --> &BEFD
```

```
BEF7  ,   230 044     E6 2C     INC &2C
BEF9      208 002     D0 02     BNE 2   --> &BEFD
BEFB  -   230 045     E6 2D     INC &2D
BEFD `    096         60        RTS
BEFE      169 013     A9 0D     LDA#&0D
BF00      164 024     A4 18     LDY &18
BF02      132 019     84 13     STY &13
BF04 d    100 018     64 12     STZ &12
BF06 d    100 032     64 20     STZ &20
BF08      146 018     92 12     STA (&12)
BF0A      169 255     A9 FF     LDA#&FF
BF0C      160 001     A0 01     LDY#&01
BF0E      145 018     91 12     STA (&12),Y
BF10      200         C8        INY
BF11      132 018     84 12     STY &12
BF13 `    096         60        RTS
BF14      184         B8        CLV
BF15      167         A7        xxx Invalid Code
BF16      238 165 166 EE A5 A6  INC &A6A5
BF19      166 141     A6 8D     LDX &8D
BF1B      166 202     A6 CA     LDX &CA
BF1D  A   172 065 165 AC 41 A5  LDY &A541
BF20   J  025 165 074 19 A5 4A  ORA &4AA5,Y
BF23 .    046 129 201 2E 81 C9  ROL &C981
BF26      016 000     10 00     BPL 0   --> &BF28
BF28      000         00        BRK
BF29 o    111         6F        xxx Invalid Code
BF2A  w   021 119     15 77     ORA &77,X
BF2C z    122         7A        PLY
BF2D a I  097 129 073 61 81 49  ADC (&4981,X)
BF30      015         0F        xxx Invalid Code
BF31      218         DA        PHX
BF32      162 128     A2 80     LDX#&80
BF34 "    034         22        xxx Invalid Code
BF35   n  249 131 110 F9 83 6E  SBC &6E83,Y
BF38 {    123         7B        xxx Invalid Code
BF39   5  014 250 053 0E FA 35  ASL &35FA
BF3C      018 134     12 86     ORA (&86)
BF3E e.   101 046     65 2E     ADC &2E
BF40      224 211     E0 D3     CPX#&D3
BF42      127         7F        xxx Invalid Code
BF43 ^[   094 091 216 5E 5B D8  LSR &D85B,X
BF46      170         AA        TAX
```

```
BF47      130         82        xxx Invalid Code
BF48 - T  045 248 084 2D F8 54  AND &54F8
BF4B X    088         58        CLI
BF4C  1   128 049     80 31     BRA 49   --> &BF7F
BF4E r    114 023     72 17     ADC (&17)
BF50      248         F8        SED
BF51      128 011     80 0B     BRA 11   --> &BF5E
BF53      215         D7        xxx Invalid Code
BF54 P)   080 041     50 29     BVC 41   --> &BF7F
BF56 | |  124 210 124 7C D2 7C  JMP (&7CD2,X)
BF59      134 005     86 05     STX &05
BF5B      128 021     80 15     BRA 21   --> &BF72
BF5D R    082 182     52 B6     EOR (&B6)
BF5F 6|   054 124     36 7C     ROL &7C,X
BF61   6  153 152 054 99 98 36  STA &3698,Y
BF64      004 128     04 80     TSB &80
BF66 @    064         40        RTI
BF67      000         00        BRK
BF68      001 016     01 10     ORA (&10,X)
BF6A      127         7F        xxx Invalid Code
BF6B *    042         2A        ROL A
BF6C      170         AA        TAX
BF6D      170         AA        TAX
BF6E      227         E3        xxx Invalid Code
BF6F      127         7F        xxx Invalid Code
BF70      255         FF        xxx Invalid Code
BF71      255         FF        xxx Invalid Code
BF72      255         FF        xxx Invalid Code
BF73      255         FF        xxx Invalid Code
BF74 z    122         7A        PLY
BF75      195         C3        xxx Invalid Code
BF76      030 024 190 1E 18 BE  ASL &BE18,X
BF79 s    115         73        xxx Invalid Code
BF7A aqU  097 113 085 61 71 55  ADC (&5571,X)
BF7D -{   045 123 140 2D 7B 8C  AND &8C7B
BF80      155         9B        xxx Invalid Code
BF81      145 136     91 88     STA (&88),Y
BF83 w    119         77        xxx Invalid Code
BF84 +    043         2B        xxx Invalid Code
BF85      164 196     A4 C4     LDY &C4
BF87 S    083         53        xxx Invalid Code
BF88 |L   124 076 204 7C 4C CC  JMP (&CC4C,X)
BF8B      202         CA        DEX
```

```
BF8C      183         B7        xxx Invalid Code
BF8D ~    126 170 170 7E AA AA  ROR &AAAA,X
BF90      170         AA        TAX
BF91      166 129     A6 81     LDX &81
BF93      000         00        BRK
BF94      000         00        BRK
BF95      000         00        BRK
BF96      000         00        BRK
BF97 }    125 163 242 7D A3 F2  ADC &F2A3,X
BF9A      239         EF        xxx Invalid Code
BF9B D    068         44        xxx Invalid Code
BF9C ~    126 031 001 7E 1F 01  ROR &011F,X
BF9F  M   161 077     A1 4D     LDA (&4D,X)
BFA1      127         7F        xxx Invalid Code
BFA2 am   097 109 244 61 6D F4  ADC (&F46D,X)
BFA5 ?    063         3F        xxx Invalid Code
BFA6 ~\   126 092 145 7E 5C 91  ROR &915C,X
BFA9 #    035         23        xxx Invalid Code
BFAA  ~v  172 126 118 AC 7E 76  LDY &767E
BFAD      184         B8        CLV
BFAE   }  141 026 125 8D 1A 7D  STA &7D1A
BFB1  >   029 062 171 1D 3E AB  ORA &AB3E,X
BFB4 ,    044 129 009 2C 81 09  BIT &0981
BFB7 A    065 129     41 81     EOR (&81,X)
BFB9      210 128     D2 80     CMP (&80)
BFBB t    116 223     74 DF     STZ &DF,X
BFBD      189 032 128 BD 20 80  LDA &8020,X
BFC0      131         83        xxx Invalid Code
BFC1      139         8B        xxx Invalid Code
BFC2      031         1F        xxx Invalid Code
BFC3      181 127     B5 7F     LDA &7F,X
BFC5      130         82        xxx Invalid Code
BFC6 Y    089 173 171 59 AD AB  EOR &ABAD,Y
BFC9  m   128 109     80 6D     BRA 109   --> &C038
BFCB c    099         63        xxx Invalid Code
BFCC 8    056         38        SEC
BFCD ,}   044 125 017 2C 7D 11  BIT &117D
BFD0      212         D4        xxx Invalid Code
BFD1      177 209     B1 D1     LDA (&D1),Y
BFD3 yh   121 104 188 79 68 BC  ADC &BC68,Y
BFD6 O    079         4F        xxx Invalid Code
BFD7 Yu   089 117 005 59 75 05  EOR &0575,Y
BFDA , 9  044 158 057 2C 9E 39  BIT &399E
```

```
BFDD {    123        7B       xxx Invalid Code
BFDE      008        08       PHP
BFDF      136        88       DEY
BFE0 ;    059        3B       xxx Invalid Code
BFE1  l   166 108    A6 6C    LDX &6C
BFE3 1    049 207    31 CF    AND (&CF,Y)
BFE5      209 140    D1 8C    CMP (&8C),Y
BFE7 }*   125 042 170 7D 2A AA  ADC &AA2A,X
BFEA      170        AA       TAX
BFEB      137 127    89 7F    BIT#&7F
BFED      255        FF       xxx Invalid Code
BFEE      255        FF       xxx Invalid Code
BFEF      255        FF       xxx Invalid Code
BFF0      232        E8       INX
BFF1      129 000    81 00    STA (&00,X)
BFF3      000        00       BRK
BFF4      000        00       BRK
BFF5      000        00       BRK
BFF6      129 000    81 00    STA (&00,X)
BFF8      000        00       BRK
BFF9      000        00       BRK
BFFA      000        00       BRK
BFFB Ro   082 111    52 6F    EOR (&6F)
BFFD g    103        67       xxx Invalid Code
BFFE er   101 114    65 72    ADC &72
```

B001,,165 012,A5 0C,LDA &0C
B003,,145 002,91 02,"STA (&02),Y"
B005,,200,C8,INY
B006,,032 139 152,20 8B 98,JSR &988B
B009,Lr,076 114 176,4C 72 B0,JMP &B072
B00C,,000,00,BRK
B00D,,030,1E,EQUB &1E
B00E,Ba,066 097,42 61,xxx Invalid Code
B010,d,100 032,64 20,STZ &20
B012,c,099,63,xxx Invalid Code
B013,all,097 108 108,61 6C 6C,"ADC (&6C6C,X)"
B016,,000,00,EQUB &00
B017,,169 164,A9 A4,LDA#&A4
B019,',133 039,85 27,STA &27
B01B,,186,BA,TSX
B01C,,138,8A,TXA
B01D,,024,18,CLC
B01E,e,101 004,65 04,ADC &04
B020,,032 030 189,20 1E BD,JSR &BD1E
B023,,138,8A,TXA
B024,,146 004,92 04,STA (&04)
B026,,160 000,A0 00,LDY#&00
B028,,232,E8,INX
B029,,200,C8,INY
B02A,,189 000 001,BD 00 01,"LDA &0100,X"
B02D,,145 004,91 04,"STA (&04),Y"
B02F,,224 255,E0 FF,CPX#&FF
B031,,208 245,D0 F5,BNE -11 --> &B028
B033,,154,9A,TXS
B034,',165 039,A5 27,LDA &27
B036,H,072,48,PHA
B037,,165 010,A5 0A,LDA &0A
B039,H,072,48,PHA
B03A,,165 011,A5 0B,LDA &0B
B03C,H,072,48,PHA
B03D,,165 012,A5 0C,LDA &0C
B03F,H,072,48,PHA
B040,,165 027,A5 1B,LDA &1B
B042,,170,AA,TAX
B043,,024,18,CLC
B044,e,101 025,65 19,ADC &19
B046,,164 026,A4 1A,LDY &1A

B048,,144 002,90 02,BCC 2   --> &B04C
B04A,,200,C8,INY
B04B,,024,18,CLC
B04C,,233 001,E9 01,SBC#&01
B04E,7,133 055,85 37,STA &37
B050,,152,98,TYA
B051,,233 000,E9 00,SBC#&00
B053,8,133 056,85 38,STA &38
B055,,160 002,A0 02,LDY#&02
B057,,032 248 154,20 F8 9A,JSR &9AF8
B05A,,192 002,C0 02,CPY#&02
B05C,,240 174,F0 AE,BEQ -82   --> &B00C
B05E,,134 027,86 1B,STX &1B
B060,u,032 117 128,20 75 80,JSR &8075
B063,,208 003,D0 03,BNE 3   --> &B068
B065,L,076 151 175,4C 97 AF,JMP &AF97
B068,*,178 042,B2 2A,LDA (&2A)
B06A,,133 011,85 0B,STA &0B
B06C,,160 001,A0 01,LDY#&01
B06E,*,177 042,B1 2A,"LDA (&2A),Y"
B070,,133 012,85 0C,STA &0C
B072,,169 000,A9 00,LDA#&00
B074,H,072,48,PHA
B075,d,100 010,64 0A,STZ &0A
B077,,032 224 142,20 E0 8E,JSR &8EE0
B07A,(,201 040,C9 28,CMP#&28
B07C,M,240 077,F0 4D,BEQ 77   --> &B0CB
B07E,,198 010,C6 0A,DEC &0A
B080,,165 027,A5 1B,LDA &1B
B082,H,072,48,PHA
B083,,165 025,A5 19,LDA &19
B085,H,072,48,PHA
B086,,165 026,A5 1A,LDA &1A
B088,H,072,48,PHA
B089,,032 011 144,20 0B 90,JSR &900B
B08C,h,104,68,PLA
B08D,,133 026,85 1A,STA &1A
B08F,h,104,68,PLA
B090,,133 025,85 19,STA &19
B092,h,104,68,PLA
B093,,133 027,85 1B,STA &1B
B095,h,104,68,PLA
B096,,240 012,F0 0C,BEQ 12   --> &B0A4

```
B098,?,133 063,85 3F,STA &3F
B09A,,032 006 189,20 06 BD,JSR &BD06
B09D,j,032 106 188,20 6A BC,JSR &BC6A
B0A0,?,198 063,C6 3F,DEC &3F
B0A2,,208 246,D0 F6,BNE -10   --> &B09A
B0A4,h,104,68,PLA
B0A5,,133 012,85 0C,STA &0C
B0A7,h,104,68,PLA
B0A8,,133 011,85 0B,STA &0B
B0AA,h,104,68,PLA
B0AB,,133 010,85 0A,STA &0A
B0AD,h,104,68,PLA
B0AE,,178 004,B2 04,LDA (&04)
B0B0,,170,AA,TAX
B0B1,,154,9A,TXS
B0B2,,160 000,A0 00,LDY#&00
B0B4,,200,C8,INY
B0B5,,232,E8,INX
B0B6,,177 004,B1 04,"LDA (&04),Y"
B0B8,,157 000 001,9D 00 01,"STA &0100,X"
B0BB,,224 255,E0 FF,CPX#&FF
B0BD,,208 245,D0 F5,BNE -11   --> &B0B4
B0BF,,152,98,TYA
B0C0,e,101 004,65 04,ADC &04
B0C2,,133 004,85 04,STA &04
B0C4,,144 002,90 02,BCC 2   --> &B0C8
B0C6,,230 005,E6 05,INC &05
B0C8,',165 039,A5 27,LDA &27
B0CA,`,096,60,RTS
B0CB,,165 027,A5 1B,LDA &1B
B0CD,H,072,48,PHA
B0CE,,165 025,A5 19,LDA &19
B0D0,H,072,48,PHA
B0D1,,165 026,A5 1A,LDA &1A
B0D3,H,072,48,PHA
B0D4,,032 174 152,20 AE 98,JSR &98AE
B0D7,R,240 082,F0 52,BEQ 82   --> &B12B
B0D9,,165 027,A5 1B,LDA &1B
B0DB,,133 010,85 0A,STA &0A
B0DD,h,104,68,PLA
B0DE,,133 026,85 1A,STA &1A
B0E0,h,104,68,PLA
B0E1,,133 025,85 19,STA &19
```

B0E3,h,104,68,PLA
B0E4,,133 027,85 1B,STA &1B
B0E6,,250,FA,PLX
B0E7,",",165 044,A5 2C,LDA &2C
B0E9,H,072,48,PHA
B0EA,+,165 043,A5 2B,LDA &2B
B0EC,H,072,48,PHA
B0ED,*,165 042,A5 2A,LDA &2A
B0EF,H,072,48,PHA
B0F0,,232,E8,INX
B0F1,,218,DA,PHX
B0F2,,032 129 177,20 81 B1,JSR &B181
B0F5,,032 229 140,20 E5 8C,JSR &8CE5
B0F8,,240 209,F0 D1,BEQ -47 --> &B0CB
B0FA,),201 041,C9 29,CMP#&29
B0FC,-,208 045,D0 2D,BNE 45 --> &B12B
B0FE,,169 000,A9 00,LDA#&00
B100,H,072,48,PHA
B101,,032 213 142,20 D5 8E,JSR &8ED5
B104,(,201 040,C9 28,CMP#&28
B106,#,208 035,D0 23,BNE 35 --> &B12B
B108,;,032 059 157,20 3B 9D,JSR &9D3B
B10B,"""",032 034 188,20 22 BC,JSR &BC22
B10E,',165 039,A5 27,LDA &27
B110,-,133 045,85 2D,STA &2D
B112,&,032 038 188,20 26 BC,JSR &BC26
B115,,250,FA,PLX
B116,,232,E8,INX
B117,,218,DA,PHX
B118,,032 235 142,20 EB 8E,JSR &8EEB
B11B,,240 235,F0 EB,BEQ -21 --> &B108
B11D,),201 041,C9 29,CMP#&29
B11F,,208 010,D0 0A,BNE 10 --> &B12B
B121,h,104,68,PLA
B122,h,104,68,PLA
B123,L,133 076,85 4C,STA &4C
B125,M,133 077,85 4D,STA &4D
B127,L,228 076,E4 4C,CPX &4C
B129,,240 021,F0 15,BEQ 21 --> &B140
B12B,,162 251,A2 FB,LDX#&FB
B12D,,154,9A,TXS
B12E,h,104,68,PLA
B12F,,133 012,85 0C,STA &0C

B131,h,104,68,PLA
B132,,133 011,85 0B,STA &0B
B134,,000,00,BRK
B135,,031,1F,EQUB &1F
B136,Ar,065 114,41 72,"EOR (&72,X)"
B138,g,103,67,xxx Invalid Code
B139,um,117 109,75 6D,"ADC &6D,X"
B13B,en,101 110,65 6E,ADC &6E
B13D,ts,116 115,74 73,"STZ &73,X"
B13F,,000,00,EQUB &00
B140,,032 230 188,20 E6 BC,JSR &BCE6
B143,h,104,68,PLA
B144,*,133 042,85 2A,STA &2A
B146,h,104,68,PLA
B147,+,133 043,85 2B,STA &2B
B149,h,104,68,PLA
B14A,",",133 044,85 2C,STA &2C
B14C,0,048 031,30 1F,BMI 31 --> &B16D
B14E,-,165 045,A5 2D,LDA &2D
B150,,240 217,F0 D9,BEQ -39 --> &B12B
B152,',133 039,85 27,STA &27
B154,7,162 055,A2 37,LDX#&37
B156,,032 198 189,20 C6 BD,JSR &BDC6
B159,',165 039,A5 27,LDA &27
B15B,,016 008,10 08,BPL 8 --> &B165
B15D,,032 232 187,20 E8 BB,JSR &BBE8
B160,A,032 065 165,20 41 A5,JSR &A541
B163,,128 003,80 03,BRA 3 --> &B168
B165,,032 230 188,20 E6 BC,JSR &BCE6
B168,8,032 056 179,20 38 B3,JSR &B338
B16B,,128 010,80 0A,BRA 10 --> &B177
B16D,-,165 045,A5 2D,LDA &2D
B16F,,208 186,D0 BA,BNE -70 --> &B12B
B171,,032 210 188,20 D2 BC,JSR &BCD2
B174,,032 174 144,20 AE 90,JSR &90AE
B177,L,198 076,C6 4C,DEC &4C
B179,,208 197,D0 C5,BNE -59 --> &B140
B17B,M,165 077,A5 4D,LDA &4D
B17D,H,072,48,PHA
B17E,L,076 128 176,4C 80 B0,JMP &B080
B181,",",164 044,A4 2C,LDY &2C
B183,,192 005,C0 05,CPY#&05
B185,,176 005,B0 05,BCS 5 --> &B18C

B187,7,162 055,A2 37,LDX#&37
B189,,032 198 189,20 C6 BD,JSR &BDC6
B18C,,032 160 177,20 A0 B1,JSR &B1A0
B18F,,008,08,PHP
B190,"""",032 034 188,20 22 BC,JSR &BC22
B193,(,040,28,PLP
B194,,240 007,F0 07,BEQ 7   --> &B19D
B196,0,048 005,30 05,BMI 5   --> &B19D
B198,7,162 055,A2 37,LDX#&37
B19A,,032 128 170,20 80 AA,JSR &AA80
B19D,L&,076 038 188,4C 26 BC,JMP &BC26
B1A0,",",164 044,A4 2C,LDY &2C
B1A2,0S,048 083,30 53,BMI 83   --> &B1F7
B1A4,,240 028,F0 1C,BEQ 28   --> &B1C2
B1A6,,192 005,C0 05,CPY#&05
B1A8,,240 029,F0 1D,BEQ 29   --> &B1C7
B1AA,,160 003,A0 03,LDY#&03
B1AC,*,177 042,B1 2A,"LDA (&2A),Y"
B1AE,-,133 045,85 2D,STA &2D
B1B0,,136,88,DEY
B1B1,*,177 042,B1 2A,"LDA (&2A),Y"
B1B3,",",133 044,85 2C,STA &2C
B1B5,,136,88,DEY
B1B6,*,177 042,B1 2A,"LDA (&2A),Y"
B1B8,,170,AA,TAX
B1B9,*,178 042,B2 2A,LDA (&2A)
B1BB,*,133 042,85 2A,STA &2A
B1BD,+,134 043,86 2B,STX &2B
B1BF,@,169 064,A9 40,LDA#&40
B1C1,`,096,60,RTS
B1C2,*,177 042,B1 2A,"LDA (&2A),Y"
B1C4,L,076 026 174,4C 1A AE,JMP &AE1A
B1C7,d5,100 053,64 35,STZ &35
B1C9,d/,100 047,64 2F,STZ &2F
B1CB,,136,88,DEY
B1CC,*,177 042,B1 2A,"LDA (&2A),Y"
B1CE,4,133 052,85 34,STA &34
B1D0,,136,88,DEY
B1D1,*,177 042,B1 2A,"LDA (&2A),Y"
B1D3,3,133 051,85 33,STA &33
B1D5,,136,88,DEY
B1D6,*,177 042,B1 2A,"LDA (&2A),Y"
B1D8,2,133 050,85 32,STA &32

```
B1DA,,136,88,DEY
B1DB,*,177 042,B1 2A,"LDA (&2A),Y"
B1DD,.,133 046,85 2E,STA &2E
B1DF,,168,A8,TAY
B1E0,*,178 042,B2 2A,LDA (&2A)
B1E2,0,133 048,85 30,STA &30
B1E4,,208 009,D0 09,BNE 9   --> &B1EF
B1E6,,152,98,TYA
B1E7,2,005 050,05 32,ORA &32
B1E9,3,005 051,05 33,ORA &33
B1EB,4,005 052,05 34,ORA &34
B1ED,,240 003,F0 03,BEQ 3   --> &B1F2
B1EF,,152,98,TYA
B1F0,,009 128,09 80,ORA#&80
B1F2,1,133 049,85 31,STA &31
B1F4,,169 255,A9 FF,LDA#&FF
B1F6,`,096,60,RTS
B1F7,,192 128,C0 80,CPY#&80
B1F9,,240 030,F0 1E,BEQ 30   --> &B219
B1FB,,160 003,A0 03,LDY#&03
B1FD,*,177 042,B1 2A,"LDA (&2A),Y"
B1FF,6,133 054,85 36,STA &36
B201,,240 021,F0 15,BEQ 21   --> &B218
B203,,160 001,A0 01,LDY#&01
B205,*,177 042,B1 2A,"LDA (&2A),Y"
B207,8,133 056,85 38,STA &38
B209,*,178 042,B2 2A,LDA (&2A)
B20B,7,133 055,85 37,STA &37
B20D,6,164 054,A4 36,LDY &36
B20F,,136,88,DEY
B210,7,177 055,B1 37,"LDA (&37),Y"
B212,,153 000 006,99 00 06,"STA &0600,Y"
B215,,152,98,TYA
B216,,208 247,D0 F7,BNE -9   --> &B20F
B218,`,096,60,RTS
B219,+,165 043,A5 2B,LDA &2B
B21B,,240 021,F0 15,BEQ 21   --> &B232
B21D,,160 000,A0 00,LDY#&00
B21F,*,177 042,B1 2A,"LDA (&2A),Y"
B221,,153 000 006,99 00 06,"STA &0600,Y"
B224,I,073 013,49 0D,EOR#&0D
B226,,240 004,F0 04,BEQ 4   --> &B22C
B228,,200,C8,INY
```

B229,,208 244,D0 F4,BNE -12   --> &B21F
B22B,,152,98,TYA
B22C,6,132 054,84 36,STY &36
B22E,`,096,60,RTS
B22F,,032 180 150,20 B4 96,JSR &96B4
B232,*,165 042,A5 2A,LDA &2A
B234,Ll,076 108 174,4C 6C AE,JMP &AE6C
B237,,164 010,A4 0A,LDY &0A
B239,,240 001,F0 01,BEQ 1   --> &B23C
B23B,,136,88,DEY
B23C,,032 188 155,20 BC 9B,JSR &9BBC
B23F,d,100 008,64 08,STZ &08
B241,d,100 009,64 09,STZ &09
B243,,166 024,A6 18,LDX &18
B245,8,134 056,86 38,STX &38
B247,d7,100 055,64 37,STZ &37
B249,,164 012,A4 0C,LDY &0C
B24B,,192 007,C0 07,CPY#&07
B24D,(,240 040,F0 28,BEQ 40   --> &B277
B24F,,166 011,A6 0B,LDX &0B
B251,,032 160 141,20 A0 8D,JSR &8DA0
B254,,201 013,C9 0D,CMP#&0D
B256,,208 024,D0 18,BNE 24   --> &B270
B258,7,228 055,E4 37,CPX &37
B25A,,152,98,TYA
B25B,8,229 056,E5 38,SBC &38
B25D,,144 024,90 18,BCC 24   --> &B277
B25F,,032 160 141,20 A0 8D,JSR &8DA0
B262,,009 000,09 00,ORA#&00
B264,0,048 017,30 11,BMI 17   --> &B277
B266,,133 009,85 09,STA &09
B268,,032 160 141,20 A0 8D,JSR &8DA0
B26B,,133 008,85 08,STA &08
B26D,,032 160 141,20 A0 8D,JSR &8DA0
B270,7,228 055,E4 37,CPX &37
B272,,152,98,TYA
B273,8,229 056,E5 38,SBC &38
B275,,176 218,B0 DA,BCS -38   --> &B251
B277,`,096,60,RTS
B278,,162 255,A2 FF,LDX#&FF
B27A,(,134 040,86 28,STX &28
B27C,,154,9A,TXS
B27D,,232,E8,INX

B27E,,160 000,A0 00,LDY#&00
B280,,169 218,A9 DA,LDA#&DA
B282,,032 244 255,20 F4 FF,JSR &FFF4
B285,~,169 126,A9 7E,LDA#&7E
B287,,032 244 255,20 F4 FF,JSR &FFF4
B28A,7,032 055 178,20 37 B2,JSR &B237
B28D,d,100 032,64 20,STZ &20
B28F,,178 253,B2 FD,LDA (&FD)
B291,,208 003,D0 03,BNE 3 --> &B296
B293,,032 166 178,20 A6 B2,JSR &B2A6
B296,,165 022,A5 16,LDA &16
B298,,133 011,85 0B,STA &0B
B29A,,165 023,A5 17,LDA &17
B29C,,133 012,85 0C,STA &0C
B29E,d,100 010,64 0A,STZ &0A
B2A0,,032 207 187,20 CF BB,JSR &BBCF
B2A3,L,076 011 144,4C 0B 90,JMP &900B
B2A6,,169 175,A9 AF,LDA#&AF
B2A8,,133 022,85 16,STA &16
B2AA,,169 178,A9 B2,LDA#&B2
B2AC,,133 023,85 17,STA &17
B2AE,`,096,60,RTS
B2AF,:,246 058,F6 3A,"INC &3A,X"
B2B1,,231,E7,xxx Invalid Code
B2B2,"""",158 241 034,9E F1 22,"STZ &22F1,X"
B2B5,at,032 097 116,20 61 74,JSR &7461
B2B8,li,032 108 105,20 6C 69,JSR &696C
B2BB,ne,110 101 032,6E 65 20,ROR &2065
B2BE,"""",034,22,xxx Invalid Code
B2BF,;,059,3B,xxx Invalid Code
B2C0,:,158 058 224,9E 3A E0,"STZ &E03A,X"
B2C3,,139,8B,xxx Invalid Code
B2C4,:,241 058,F1 3A,"SBC (&3A),Y"
B2C6,,224 013,E0 0D,CPX#&0D
B2C8,o,032 111 146,20 6F 92,JSR &926F
B2CB,,162 003,A2 03,LDX#&03
B2CD,*,165 042,A5 2A,LDA &2A
B2CF,H,072,48,PHA
B2D0,+,165 043,A5 2B,LDA &2B
B2D2,H,072,48,PHA
B2D3,,218,DA,PHX
B2D4,,032 172 150,20 AC 96,JSR &96AC
B2D7,,250,FA,PLX

B2D8,,202,CA,DEX
B2D9,,208 242,D0 F2,BNE -14 --> &B2CD
B2DB,,032 150 155,20 96 9B,JSR &9B96
B2DE,*,165 042,A5 2A,LDA &2A
B2E0,=,133 061,85 3D,STA &3D
B2E2,+,165 043,A5 2B,LDA &2B
B2E4,>,133 062,85 3E,STA &3E
B2E6,,160 007,A0 07,LDY#&07
B2E8,,162 005,A2 05,LDX#&05
B2EA,,128 027,80 1B,BRA 27 --> &B307
B2EC,o,032 111 146,20 6F 92,JSR &926F
B2EF,,162 013,A2 0D,LDX#&0D
B2F1,*,165 042,A5 2A,LDA &2A
B2F3,H,072,48,PHA
B2F4,,218,DA,PHX
B2F5,,032 172 150,20 AC 96,JSR &96AC
B2F8,,250,FA,PLX
B2F9,,202,CA,DEX
B2FA,,208 245,D0 F5,BNE -11 --> &B2F1
B2FC,,032 150 155,20 96 9B,JSR &9B96
B2FF,*,165 042,A5 2A,LDA &2A
B301,D,133 068,85 44,STA &44
B303,,162 012,A2 0C,LDX#&0C
B305,,160 008,A0 08,LDY#&08
B307,h,104,68,PLA
B308,7,149 055,95 37,"STA &37,X"
B30A,,202,CA,DEX
B30B,,016 250,10 FA,BPL -6 --> &B307
B30D,,152,98,TYA
B30E,7,162 055,A2 37,LDX#&37
B310,,160 000,A0 00,LDY#&00
B312,,032 241 255,20 F1 FF,JSR &FFF1
B315,,128 011,80 0B,BRA 11 --> &B322
B317,o,032 111 146,20 6F 92,JSR &926F
B31A,,032 150 155,20 96 9B,JSR &9B96
B31D,*,164 042,A4 2A,LDY &2A
B31F,,136,88,DEY
B320,#,132 035,84 23,STY &23
B322,L,076 005 144,4C 05 90,JMP &9005
B325,L,076 146 144,4C 92 90,JMP &9092
B328,;,032 059 157,20 3B 9D,JSR &9D3B
B32B,z,122,7A,PLY
B32C,,250,FA,PLX

B32D,h,104,68,PLA
B32E,9,133 057,85 39,STA &39
B330,h,104,68,PLA
B331,8,133 056,85 38,STA &38
B333,h,104,68,PLA
B334,7,133 055,85 37,STA &37
B336,,218,DA,PHX
B337,Z,090,5A,PHY
B338,9,165 057,A5 39,LDA &39
B33A,,201 005,C9 05,CMP#&05
B33C,"""",240 034,F0 22,BEQ 34 --> &B360
B33E,',165 039,A5 27,LDA &27
B340,,240 227,F0 E3,BEQ -29 --> &B325
B342,,016 003,10 03,BPL 3 --> &B347
B344,,032 195 150,20 C3 96,JSR &96C3
B347,*,165 042,A5 2A,LDA &2A
B349,7,146 055,92 37,STA (&37)
B34B,9,165 057,A5 39,LDA &39
B34D,,240 016,F0 10,BEQ 16 --> &B35F
B34F,+,165 043,A5 2B,LDA &2B
B351,,160 001,A0 01,LDY#&01
B353,7,145 055,91 37,"STA (&37),Y"
B355,",",165 044,A5 2C,LDA &2C
B357,,200,C8,INY
B358,7,145 055,91 37,"STA (&37),Y"
B35A,-,165 045,A5 2D,LDA &2D
B35C,,200,C8,INY
B35D,7,145 055,91 37,"STA (&37),Y"
B35F,`,096,60,RTS
B360,',165 039,A5 27,LDA &27
B362,,240 193,F0 C1,BEQ -63 --> &B325
B364,0,048 003,30 03,BMI 3 --> &B369
B366,,032 133 129,20 85 81,JSR &8185
B369,0,165 048,A5 30,LDA &30
B36B,7,146 055,92 37,STA (&37)
B36D,,160 001,A0 01,LDY#&01
B36F,.,165 046,A5 2E,LDA &2E
B371,E1,069 049,45 31,EOR &31
B373,),041 128,29 80,AND#&80
B375,E1,069 049,45 31,EOR &31
B377,7,145 055,91 37,"STA (&37),Y"
B379,,200,C8,INY
B37A,2,165 050,A5 32,LDA &32

B37C,7,145 055,91 37,"STA (&37),Y"
B37E,,200,C8,INY
B37F,3,165 051,A5 33,LDA &33
B381,7,145 055,91 37,"STA (&37),Y"
B383,,200,C8,INY
B384,4,165 052,A5 34,LDA &34
B386,7,145 055,91 37,"STA (&37),Y"
B388,`,096,60,RTS
B389,ED,069 068,45 44,EOR &44
B38B,IT,073 084,49 54,EOR#&54
B38D,12,032 049 050,20 31 32,JSR &3231
B390,",2",044 050 013,2C 32 0D,BIT &0D32
B393,,032 172 187,20 AC BB,JSR &BBAC
B396,,169 128,A9 80,LDA#&80
B398,,133 031,85 1F,STA &1F
B39A,d;,100 059,64 3B,STZ &3B
B39C,d<,100 060,64 3C,STZ &3C
B39E,,032 232 171,20 E8 AB,JSR &ABE8
B3A1,,032 030 155,20 1E 9B,JSR &9B1E
B3A4,,008,08,PHP
B3A5,&,032 038 188,20 26 BC,JSR &BC26
B3A8,,032 219 171,20 DB AB,JSR &ABDB
B3AB,F+,070 043,46 2B,LSR &2B
B3AD,(,040,28,PLP
B3AE,,144 015,90 0F,BCC 15 --> &B3BF
B3B0,,032 229 140,20 E5 8C,JSR &8CE5
B3B3,,240 017,F0 11,BEQ 17 --> &B3C6
B3B5,,032 230 188,20 E6 BC,JSR &BCE6
B3B8,&,032 038 188,20 26 BC,JSR &BC26
B3BB,,198 010,C6 0A,DEC &0A
B3BD,,128 010,80 0A,BRA 10 --> &B3C9
B3BF,,032 229 140,20 E5 8C,JSR &8CE5
B3C2,,240 002,F0 02,BEQ 2 --> &B3C6
B3C4,,198 010,C6 0A,DEC &0A
B3C6,,032 030 155,20 1E 9B,JSR &9B1E
B3C9,1,162 049,A2 31,LDX#&31
B3CB,,032 198 189,20 C6 BD,JSR &BDC6
B3CE,,032 224 142,20 E0 8E,JSR &8EE0
B3D1,,201 231,C9 E7,CMP#&E7
B3D3,,208 030,D0 1E,BNE 30 --> &B3F3
B3D5,,032 224 142,20 E0 8E,JSR &8EE0
B3D8,,032 188 155,20 BC 9B,JSR &9BBC
B3DB,,128 025,80 19,BRA 25 --> &B3F6

```
B3DD,,200,C8,INY
B3DE,,177 011,B1 0B,"LDA (&0B),Y"
B3E0,O,201 079,C9 4F,CMP#&4F
B3E2,,208 182,D0 B6,BNE -74   --> &B39A
B3E4,,230 010,E6 0A,INC &0A
B3E6,o,032 111 146,20 6F 92,JSR &926F
B3E9,,032 166 155,20 A6 9B,JSR &9BA6
B3EC,*,165 042,A5 2A,LDA &2A
B3EE,,133 031,85 1F,STA &1F
B3F0,L,076 134 143,4C 86 8F,JMP &8F86
B3F3,,032 176 155,20 B0 9B,JSR &9BB0
B3F6,,165 011,A5 0B,LDA &0B
B3F8,,133 025,85 19,STA &19
B3FA,,032 229 189,20 E5 BD,JSR &BDE5
B3FD,,032 230 188,20 E6 BC,JSR &BCE6
B400,,032 205 128,20 CD 80,JSR &80CD
B403,=,165 061,A5 3D,LDA &3D
B405,,133 011,85 0B,STA &0B
B407,>,165 062,A5 3E,LDA &3E
B409,,133 012,85 0C,STA &0C
B40B,,176 027,B0 1B,BCS 27   --> &B428
B40D,,136,88,DEY
B40E,,128 015,80 0F,BRA 15   --> &B41F
B410,,032 148 189,20 94 BD,JSR &BD94
B413,$,036 031,24 1F,BIT &1F
B415,0,048 005,30 05,BMI 5   --> &B41C
B417,,169 010,A9 0A,LDA#&0A
B419,,032 238 255,20 EE FF,JSR &FFEE
B41C,,032 188 155,20 BC 9B,JSR &9BBC
B41F,,177 011,B1 0B,"LDA (&0B),Y"
B421,+,133 043,85 2B,STA &2B
B423,,200,C8,INY
B424,,177 011,B1 0B,"LDA (&0B),Y"
B426,*,133 042,85 2A,STA &2A
B428,*,165 042,A5 2A,LDA &2A
B42A,,024,18,CLC
B42B,1,229 049,E5 31,SBC &31
B42D,+,165 043,A5 2B,LDA &2B
B42F,2,229 050,E5 32,SBC &32
B431,,144 011,90 0B,BCC 11   --> &B43E
B433,$,036 031,24 1F,BIT &1F
B435,,016 185,10 B9,BPL -71   --> &B3F0
B437,,162 137,A2 89,LDX#&89
```

```
B439,,160 179,A0 B3,LDY#&B3
B43B,L,076 247 255,4C F7 FF,JMP &FFF7
B43E,dL,100 076,64 4C,STZ &4C
B440,dM,100 077,64 4D,STZ &4D
B442,,160 004,A0 04,LDY#&04
B444,,132 010,84 0A,STY &0A
B446,,132 027,84 1B,STY &1B
B448,$;,036 059,24 3B,BIT &3B
B44A,,016 002,10 02,BPL 2   --> &B44E
B44C,d;,100 059,64 3B,STZ &3B
B44E,$<,036 060,24 3C,BIT &3C
B450,,016 002,10 02,BPL 2   --> &B454
B452,d<,100 060,64 3C,STZ &3C
B454,,177 011,B1 0B,"LDA (&0B),Y"
B456,,201 013,C9 0D,CMP#&0D
B458,7,240 055,F0 37,BEQ 55   --> &B491
B45A,,201 244,C9 F4,CMP#&F4
B45C,,240 006,F0 06,BEQ 6   --> &B464
B45E,"""",201 034,C9 22,CMP#&22
B460,,208 004,D0 04,BNE 4   --> &B466
B462,EL,069 076,45 4C,EOR &4C
B464,L,133 076,85 4C,STA &4C
B466,L,166 076,A6 4C,LDX &4C
B468,,208 012,D0 0C,BNE 12   --> &B476
B46A,,201 237,C9 ED,CMP#&ED
B46C,,208 002,D0 02,BNE 2   --> &B470
B46E,;,198 059,C6 3B,DEC &3B
B470,,201 253,C9 FD,CMP#&FD
B472,,208 002,D0 02,BNE 2   --> &B476
B474,<,198 060,C6 3C,DEC &3C
B476,,166 025,A6 19,LDX &19
B478,,189 000 007,BD 00 07,"LDA &0700,X"
B47B,,201 013,C9 0D,CMP#&0D
B47D,,240 010,F0 0A,BEQ 10   --> &B489
B47F,,209 011,D1 0B,"CMP (&0B),Y"
B481,,208 008,D0 08,BNE 8   --> &B48B
B483,,200,C8,INY
B484,,232,E8,INX
B485,,128 241,80 F1,BRA -15   --> &B478
B487,,128 135,80 87,BRA -121   --> &B410
B489,M,133 077,85 4D,STA &4D
B48B,,230 027,E6 1B,INC &1B
B48D,,164 027,A4 1B,LDY &1B
```

B48F,,128 195,80 C3,BRA -61   --> &B454
B491,M,165 077,A5 4D,LDA &4D
B493,,240 135,F0 87,BEQ -121   --> &B41C
B495,,032 133 160,20 85 A0,JSR &A085
B498,,169 001,A9 01,LDA#&01
B49A,,232,E8,INX
B49B,8,056,38,SEC
B49C,,032 180 189,20 B4 BD,JSR &BDB4
B49F,;,166 059,A6 3B,LDX &3B
B4A1,,169 002,A9 02,LDA#&02
B4A3,,032 179 189,20 B3 BD,JSR &BDB3
B4A6,<,166 060,A6 3C,LDX &3C
B4A8,,169 004,A9 04,LDA#&04
B4AA,,032 179 189,20 B3 BD,JSR &BDB3
B4AD,dL,100 076,64 4C,STZ &4C
B4AF,,164 010,A4 0A,LDY &0A
B4B1,,177 011,B1 0B,"LDA (&0B),Y"
B4B3,,201 013,C9 0D,CMP#&0D
B4B5,,240 208,F0 D0,BEQ -48   --> &B487
B4B7,"""",201 034,C9 22,CMP#&22
B4B9,,208 012,D0 0C,BNE 12   --> &B4C7
B4BB,EL,069 076,45 4C,EOR &4C
B4BD,L,133 076,85 4C,STA &4C
B4BF,"""",169 034,A9 22,LDA#&22
B4C1,,032 148 189,20 94 BD,JSR &BD94
B4C4,,200,C8,INY
B4C5,,128 234,80 EA,BRA -22   --> &B4B1
B4C7,L,166 076,A6 4C,LDX &4C
B4C9,,208 246,D0 F6,BNE -10   --> &B4C1
B4CB,,201 141,C9 8D,CMP#&8D
B4CD,,208 010,D0 0A,BNE 10   --> &B4D9
B4CF,*,032 042 155,20 2A 9B,JSR &9B2A
B4D2,,132 010,84 0A,STY &0A
B4D4,,032 129 160,20 81 A0,JSR &A081
B4D7,,128 214,80 D6,BRA -42   --> &B4AF
B4D9,,201 227,C9 E3,CMP#&E3
B4DB,,208 002,D0 02,BNE 2   --> &B4DF
B4DD,;,230 059,E6 3B,INC &3B
B4DF,,201 245,C9 F5,CMP#&F5
B4E1,,208 002,D0 02,BNE 2   --> &B4E5
B4E3,<,230 060,E6 3C,INC &3C
B4E5,,201 244,C9 F4,CMP#&F4
B4E7,,208 002,D0 02,BNE 2   --> &B4EB

B4E9,L,133 076,85 4C,STA &4C
B4EB,7,032 055 189,20 37 BD,JSR &BD37
B4EE,,200,C8,INY
B4EF,,128 192,80 C0,BRA -64 --> &B4B1
B4F1,,032 245 152,20 F5 98,JSR &98F5
B4F4,,208 009,D0 09,BNE 9 --> &B4FF
B4F6,&,166 038,A6 26,LDX &26
B4F8,8,240 056,F0 38,BEQ 56 --> &B532
B4FA,=,176 061,B0 3D,BCS 61 --> &B539
B4FC,Li,076 105 155,4C 69 9B,JMP &9B69
B4FF,,176 251,B0 FB,BCS -5 --> &B4FC
B501,&,166 038,A6 26,LDX &26
B503,-,240 045,F0 2D,BEQ 45 --> &B532
B505,*,165 042,A5 2A,LDA &2A
B507,,221 025 005,DD 19 05,"CMP &0519,X"
B50A,,208 014,D0 0E,BNE 14 --> &B51A
B50C,+,165 043,A5 2B,LDA &2B
B50E,,221 026 005,DD 1A 05,"CMP &051A,X"
B511,,208 007,D0 07,BNE 7 --> &B51A
B513,",",165 044,A5 2C,LDA &2C
B515,,221 027 005,DD 1B 05,"CMP &051B,X"
B518,,240 031,F0 1F,BEQ 31 --> &B539
B51A,,138,8A,TXA
B51B,8,056,38,SEC
B51C,,233 015,E9 0F,SBC#&0F
B51E,,170,AA,TAX
B51F,&,134 038,86 26,STX &26
B521,,208 226,D0 E2,BNE -30 --> &B505
B523,,000,00,BRK
B524,!,033,21,EQUB &21
B525,C,067,43,xxx Invalid Code
B526,an',097 110 039,61 6E 27,"ADC (&276E,X)"
B529,t,116 032,74 20,"STZ &20,X"
B52B,mat,109 097 116,6D 61 74,ADC &7461
B52E,c,099,63,xxx Invalid Code
B52F,h,104,68,PLA
B530,,032 227,20 E3,xxx Invalid Code
B532,,000,00,BRK
B533,,032,20,EQUB &20
B534,No,078 111,4E 6F,xxx Invalid Code
B536,,032 227,20 E3,xxx Invalid Code
B538,,000,00,EQUB &00
B539,,189 025 005,BD 19 05,"LDA &0519,X"

B53C,*,133 042,85 2A,STA &2A
B53E,,189 026 005,BD 1A 05,"LDA &051A,X"
B541,+,133 043,85 2B,STA &2B
B543,,188 027 005,BC 1B 05,"LDY &051B,X"
B546,,192 005,C0 05,CPY#&05
B548,v,240 118,F0 76,BEQ 118 --> &B5C0
B54A,*,178 042,B2 2A,LDA (&2A)
B54C,},125 028 005,7D 1C 05,"ADC &051C,X"
B54F,*,146 042,92 2A,STA (&2A)
B551,7,133 055,85 37,STA &37
B553,,160 001,A0 01,LDY#&01
B555,*,177 042,B1 2A,"LDA (&2A),Y"
B557,},125 029 005,7D 1D 05,"ADC &051D,X"
B55A,*,145 042,91 2A,"STA (&2A),Y"
B55C,8,133 056,85 38,STA &38
B55E,,200,C8,INY
B55F,*,177 042,B1 2A,"LDA (&2A),Y"
B561,},125 030 005,7D 1E 05,"ADC &051E,X"
B564,*,145 042,91 2A,"STA (&2A),Y"
B566,9,133 057,85 39,STA &39
B568,,200,C8,INY
B569,*,177 042,B1 2A,"LDA (&2A),Y"
B56B,},125 031 005,7D 1F 05,"ADC &051F,X"
B56E,*,145 042,91 2A,"STA (&2A),Y"
B570,,168,A8,TAY
B571,7,165 055,A5 37,LDA &37
B573,8,056,38,SEC
B574,!,253 033 005,FD 21 05,"SBC &0521,X"
B577,7,133 055,85 37,STA &37
B579,8,165 056,A5 38,LDA &38
B57B,"""",253 034 005,FD 22 05,"SBC &0522,X"
B57E,7,004 055,04 37,TSB &37
B580,9,165 057,A5 39,LDA &39
B582,#,253 035 005,FD 23 05,"SBC &0523,X"
B585,7,004 055,04 37,TSB &37
B587,,152,98,TYA
B588,$,253 036 005,FD 24 05,"SBC &0524,X"
B58B,7,005 055,05 37,ORA &37
B58D,,240 015,F0 0F,BEQ 15 --> &B59E
B58F,,152,98,TYA
B590,],093 031 005,5D 1F 05,"EOR &051F,X"
B593,]$,093 036 005,5D 24 05,"EOR &0524,X"
B596,,016 004,10 04,BPL 4 --> &B59C

B598,,176 004,B0 04,BCS 4   --> &B59E
B59A,,128 018,80 12,BRA 18   --> &B5AE
B59C,,176 016,B0 10,BCS 16   --> &B5AE
B59E,&,188 038 005,BC 26 05,"LDY &0526,X"
B5A1,',189 039 005,BD 27 05,"LDA &0527,X"
B5A4,,132 011,84 0B,STY &0B
B5A6,,133 012,85 0C,STA &0C
B5A8,,032 198 155,20 C6 9B,JSR &9BC6
B5AB,L,076 011 144,4C 0B 90,JMP &900B
B5AE,,138,8A,TXA
B5AF,8,056,38,SEC
B5B0,,233 015,E9 0F,SBC#&0F
B5B2,&,133 038,85 26,STA &26
B5B4,,164 027,A4 1B,LDY &1B
B5B6,,132 010,84 0A,STY &0A
B5B8,,032 229 140,20 E5 8C,JSR &8CE5
B5BB,8,208 056,D0 38,BNE 56   --> &B5F5
B5BD,L,076 241 180,4C F1 B4,JMP &B4F1
B5C0,,032 199 177,20 C7 B1,JSR &B1C7
B5C3,,138,8A,TXA
B5C4,,024,18,CLC
B5C5,i,105 028,69 1C,ADC#&1C
B5C7,J,133 074,85 4A,STA &4A
B5C9,,169 005,A9 05,LDA#&05
B5CB,K,133 075,85 4B,STA &4B
B5CD,,032 141 166,20 8D A6,JSR &A68D
B5D0,*,165 042,A5 2A,LDA &2A
B5D2,7,133 055,85 37,STA &37
B5D4,+,165 043,A5 2B,LDA &2B
B5D6,8,133 056,85 38,STA &38
B5D8,i,032 105 179,20 69 B3,JSR &B369
B5DB,&,166 038,A6 26,LDX &26
B5DD,,138,8A,TXA
B5DE,,024,18,CLC
B5DF,i!,105 033,69 21,ADC#&21
B5E1,J,133 074,85 4A,STA &4A
B5E3,,032 143 156,20 8F 9C,JSR &9C8F
B5E6,,240 182,F0 B6,BEQ -74   --> &B59E
B5E8,,189 029 005,BD 1D 05,"LDA &051D,X"
B5EB,0,048 004,30 04,BMI 4   --> &B5F1
B5ED,,176 175,B0 AF,BCS -81   --> &B59E
B5EF,,128 189,80 BD,BRA -67   --> &B5AE
B5F1,,144 171,90 AB,BCC -85   --> &B59E

B5F3,,128 185,80 B9,BRA -71   --> &B5AE
B5F5,L,076 000 144,4C 00 90,JMP &9000
B5F8,,000,00,BRK
B5F9,"""",034,22,EQUB &22
B5FA,,227,E3,xxx Invalid Code
B5FB,va,032 118 097,20 76 61,JSR &6176
B5FE,ri,114 105,72 69,ADC (&69)
B600,abl,097 098 108,61 62 6C,"ADC (&6C62,X)"
B603,e,101,65,xxx Invalid Code
B604,,000,00,BRK
B605,#,035,23,EQUB &23
B606,T,084,54,xxx Invalid Code
B607,o,111,6F,xxx Invalid Code
B608,o,111,6F,xxx Invalid Code
B609,ma,032 109 097,20 6D 61,JSR &616D
B60C,ny,110 121 032,6E 79 20,ROR &2079
B60F,,227,E3,xxx Invalid Code
B610,s,115,73,xxx Invalid Code
B611,,000,00,BRK
B612,$,036,24,EQUB &24
B613,N,078,4E,xxx Invalid Code
B614,o,111,6F,xxx Invalid Code
B615,,032 184,20 B8,xxx Invalid Code
B617,,000,00,EQUB &00
B618,,032 174 152,20 AE 98,JSR &98AE
B61B,,240 219,F0 DB,BEQ -37   --> &B5F8
B61D,,176 217,B0 D9,BCS -39   --> &B5F8
B61F,C,032 067 188,20 43 BC,JSR &BC43
B622,,032 134 155,20 86 9B,JSR &9B86
B625,(,032 040 179,20 28 B3,JSR &B328
B628,,032 213 142,20 D5 8E,JSR &8ED5
B62B,,201 184,C9 B8,CMP#&B8
B62D,,208 226,D0 E2,BNE -30   --> &B611
B62F,&,164 038,A4 26,LDY &26
B631,,192 150,C0 96,CPY#&96
B633,,176 207,B0 CF,BCS -49   --> &B604
B635,,152,98,TYA
B636,i,105 015,69 0F,ADC#&0F
B638,&,133 038,85 26,STA &26
B63A,7,165 055,A5 37,LDA &37
B63C,(,153 040 005,99 28 05,"STA &0528,Y"
B63F,8,165 056,A5 38,LDA &38
B641,),153 041 005,99 29 05,"STA &0529,Y"

```
B644,9,165 057,A5 39,LDA &39
B646,*,153 042 005,99 2A 05,"STA &052A,Y"
B649,,201 005,C9 05,CMP#&05
B64B,T,240 084,F0 54,BEQ 84   --> &B6A1
B64D,,032 175 150,20 AF 96,JSR &96AF
B650,&,164 038,A4 26,LDY &26
B652,*,165 042,A5 2A,LDA &2A
B654,!,153 033 005,99 21 05,"STA &0521,Y"
B657,+,165 043,A5 2B,LDA &2B
B659,"""",153 034 005,99 22 05,"STA &0522,Y"
B65C,",",165 044,A5 2C,LDA &2C
B65E,#,153 035 005,99 23 05,"STA &0523,Y"
B661,-,165 045,A5 2D,LDA &2D
B663,$,153 036 005,99 24 05,"STA &0524,Y"
B666,,169 001,A9 01,LDA#&01
B668,,032 024 174,20 18 AE,JSR &AE18
B66B,,032 213 142,20 D5 8E,JSR &8ED5
B66E,,201 136,C9 88,CMP#&88
B670,,208 005,D0 05,BNE 5   --> &B677
B672,,032 175 150,20 AF 96,JSR &96AF
B675,,164 027,A4 1B,LDY &1B
B677,,132 010,84 0A,STY &0A
B679,&,164 038,A4 26,LDY &26
B67B,*,165 042,A5 2A,LDA &2A
B67D,,153 028 005,99 1C 05,"STA &051C,Y"
B680,+,165 043,A5 2B,LDA &2B
B682,,153 029 005,99 1D 05,"STA &051D,Y"
B685,",",165 044,A5 2C,LDA &2C
B687,,153 030 005,99 1E 05,"STA &051E,Y"
B68A,-,165 045,A5 2D,LDA &2D
B68C,,153 031 005,99 1F 05,"STA &051F,Y"
B68F,,032 207 155,20 CF 9B,JSR &9BCF
B692,&,164 038,A4 26,LDY &26
B694,,165 011,A5 0B,LDA &0B
B696,&,153 038 005,99 26 05,"STA &0526,Y"
B699,,165 012,A5 0C,LDA &0C
B69B,',153 039 005,99 27 05,"STA &0527,Y"
B69E,L,076 011 144,4C 0B 90,JMP &900B
B6A1,;,032 059 157,20 3B 9D,JSR &9D3B
B6A4,,032 221 150,20 DD 96,JSR &96DD
B6A7,&,165 038,A5 26,LDA &26
B6A9,,024,18,CLC
B6AA,¡!,105 033,69 21,ADC#&21
```

B6AC,J,133 074,85 4A,STA &4A
B6AE,,169 005,A9 05,LDA#&05
B6B0,K,133 075,85 4B,STA &4B
B6B2,,032 025 165,20 19 A5,JSR &A519
B6B5,,032 216 165,20 D8 A5,JSR &A5D8
B6B8,,032 213 142,20 D5 8E,JSR &8ED5
B6BB,,201 136,C9 88,CMP#&88
B6BD,,208 008,D0 08,BNE 8   --> &B6C7
B6BF,;,032 059 157,20 3B 9D,JSR &9D3B
B6C2,,032 221 150,20 DD 96,JSR &96DD
B6C5,,164 027,A4 1B,LDY &1B
B6C7,,132 010,84 0A,STY &0A
B6C9,&,165 038,A5 26,LDA &26
B6CB,,024,18,CLC
B6CC,i,105 028,69 1C,ADC#&1C
B6CE,J,133 074,85 4A,STA &4A
B6D0,,169 005,A9 05,LDA#&05
B6D2,K,133 075,85 4B,STA &4B
B6D4,,032 025 165,20 19 A5,JSR &A519
B6D7,,128 182,80 B6,BRA -74   --> &B68F
B6D9,*,032 042 184,20 2A B8,JSR &B82A
B6DC,,032 166 155,20 A6 9B,JSR &9BA6
B6DF,%,164 037,A4 25,LDY &25
B6E1,,192 026,C0 1A,CPY#&1A
B6E3,,176 014,B0 0E,BCS 14   --> &B6F3
B6E5,,165 011,A5 0B,LDA &0B
B6E7,,153 204 005,99 CC 05,"STA &05CC,Y"
B6EA,,165 012,A5 0C,LDA &0C
B6EC,,153 230 005,99 E6 05,"STA &05E6,Y"
B6EF,%,230 037,E6 25,INC &25
B6F1,0,128 048,80 30,BRA 48   --> &B723
B6F3,,000,00,BRK
B6F4,%,037,25,EQUB &25
B6F5,T,084,54,xxx Invalid Code
B6F6,o,111,6F,xxx Invalid Code
B6F7,o,111,6F,xxx Invalid Code
B6F8,ma,032 109 097,20 6D 61,JSR &616D
B6FB,ny,110 121 032,6E 79 20,ROR &2079
B6FE,s,228 115,E4 73,CPX &73
B700,,000,00,BRK
B701,&,038,26,EQUB &26
B702,N,078,4E,xxx Invalid Code
B703,o,111,6F,xxx Invalid Code

B704,,032 228,20 E4,xxx Invalid Code
B706,,000,00,EQUB &00
B707,,032 166 155,20 A6 9B,JSR &9BA6
B70A,%,166 037,A6 25,LDX &25
B70C,,240 242,F0 F2,BEQ -14 --> &B700
B70E,%,198 037,C6 25,DEC &25
B710,,188 203 005,BC CB 05,"LDY &05CB,X"
B713,,189 229 005,BD E5 05,"LDA &05E5,X"
B716,,132 011,84 0B,STY &0B
B718,,133 012,85 0C,STA &0C
B71A,L,076 005 144,4C 05 90,JMP &9005
B71D,*,032 042 184,20 2A B8,JSR &B82A
B720,,032 166 155,20 A6 9B,JSR &9BA6
B723,,165 032,A5 20,LDA &20
B725,,240 003,F0 03,BEQ 3 --> &B72A
B727,K,032 075 156,20 4B 9C,JSR &9C4B
B72A,,160 004,A0 04,LDY#&04
B72C,,132 010,84 0A,STY &0A
B72E,=,164 061,A4 3D,LDY &3D
B730,>,165 062,A5 3E,LDA &3E
B732,,132 011,84 0B,STY &0B
B734,,133 012,85 0C,STA &0C
B736,L,076 011 144,4C 0B 90,JMP &900B
B739,,032 166 155,20 A6 9B,JSR &9BA6
B73C,,032 166 178,20 A6 B2,JSR &B2A6
B73F,,128 217,80 D9,BRA -39 --> &B71A
B741,,032 224 142,20 E0 8E,JSR &8EE0
B744,,201 135,C9 87,CMP#&87
B746,,240 241,F0 F1,BEQ -15 --> &B739
B748,,164 010,A4 0A,LDY &0A
B74A,,136,88,DEY
B74B,,032 188 155,20 BC 9B,JSR &9BBC
B74E,d,100 010,64 0A,STZ &0A
B750,,165 011,A5 0B,LDA &0B
B752,,133 022,85 16,STA &16
B754,,165 012,A5 0C,LDA &0C
B756,,133 023,85 17,STA &17
B758,L,076 174 143,4C AE 8F,JMP &8FAE
B75B,,032 224 142,20 E0 8E,JSR &8EE0
B75E,,201 133,C9 85,CMP#&85
B760,,240 223,F0 DF,BEQ -33 --> &B741
B762,,198 010,C6 0A,DEC &0A
B764,o,032 111 146,20 6F 92,JSR &926F

B767,,224 242,E0 F2,CPX#&F2
B769,,240 009,F0 09,BEQ 9   --> &B774
B76B,,200,C8,INY
B76C,,224 229,E0 E5,CPX#&E5
B76E,,240 004,F0 04,BEQ 4   --> &B774
B770,,224 228,E0 E4,CPX#&E4
B772,v,208 118,D0 76,BNE 118   --> &B7EA
B774,,218,DA,PHX
B775,+,165 043,A5 2B,LDA &2B
B777,",",005 044,05 2C,ORA &2C
B779,-,005 045,05 2D,ORA &2D
B77B,X,208 088,D0 58,BNE 88   --> &B7D5
B77D,*,198 042,C6 2A,DEC &2A
B77F,5,240 053,F0 35,BEQ 53   --> &B7B6
B781,0R,048 082,30 52,BMI 82   --> &B7D5
B783,,177 011,B1 0B,"LDA (&0B),Y"
B785,,201 013,C9 0D,CMP#&0D
B787,L,240 076,F0 4C,BEQ 76   --> &B7D5
B789,:,201 058,C9 3A,CMP#&3A
B78B,H,240 072,F0 48,BEQ 72   --> &B7D5
B78D,,201 139,C9 8B,CMP#&8B
B78F,D,240 068,F0 44,BEQ 68   --> &B7D5
B791,,200,C8,INY
B792,"""",201 034,C9 22,CMP#&22
B794,,208 004,D0 04,BNE 4   --> &B79A
B796,E+,069 043,45 2B,EOR &2B
B798,+,133 043,85 2B,STA &2B
B79A,+,166 043,A6 2B,LDX &2B
B79C,,208 229,D0 E5,BNE -27   --> &B783
B79E,),201 041,C9 29,CMP#&29
B7A0,,208 002,D0 02,BNE 2   --> &B7A4
B7A2,",",198 044,C6 2C,DEC &2C
B7A4,(,201 040,C9 28,CMP#&28
B7A6,,208 002,D0 02,BNE 2   --> &B7AA
B7A8,",",230 044,E6 2C,INC &2C
B7AA,",",201 044,C9 2C,CMP#&2C
B7AC,,208 213,D0 D5,BNE -43   --> &B783
B7AE,",",166 044,A6 2C,LDX &2C
B7B0,,208 209,D0 D1,BNE -47   --> &B783
B7B2,*,198 042,C6 2A,DEC &2A
B7B4,,208 205,D0 CD,BNE -51   --> &B783
B7B6,h,104,68,PLA
B7B7,,201 242,C9 F2,CMP#&F2

B7B9,H,240 072,F0 48,BEQ 72   --> &B803
B7BB,,132 010,84 0A,STY &0A
B7BD,,201 228,C9 E4,CMP#&E4
B7BF,,240 009,F0 09,BEQ 9   --> &B7CA
B7C1,*,032 042 184,20 2A B8,JSR &B82A
B7C4,,032 198 155,20 C6 9B,JSR &9BC6
B7C7,L#,076 035 183,4C 23 B7,JMP &B723
B7CA,*,032 042 184,20 2A B8,JSR &B82A
B7CD,,164 010,A4 0A,LDY &0A
B7CF,,032 029 184,20 1D B8,JSR &B81D
B7D2,L,076 220 182,4C DC B6,JMP &B6DC
B7D5,h,104,68,PLA
B7D6,,177 011,B1 0B,"LDA (&0B),Y"
B7D8,,200,C8,INY
B7D9,,201 139,C9 8B,CMP#&8B
B7DB,:,240 058,F0 3A,BEQ 58   --> &B817
B7DD,,201 013,C9 0D,CMP#&0D
B7DF,,208 245,D0 F5,BNE -11   --> &B7D6
B7E1,,000,00,BRK
B7E2,(,040,28,EQUB &28
B7E3,r,238 032 114,EE 20 72,INC &7220
B7E6,ang,097 110 103,61 6E 67,"ADC (&676E,X)"
B7E9,e,101,65,xxx Invalid Code
B7EA,,000,00,BRK
B7EB,',039,27,EQUB &27
B7EC,s,238 032 115,EE 20 73,INC &7320
B7EF,ynt,121 110 116,79 6E 74,"ADC &746E,Y"
B7F2,ax,097 120,61 78,xxx Invalid Code
B7F4,,000,00,BRK
B7F5,),041,29,EQUB &29
B7F6,N,078,4E,xxx Invalid Code
B7F7,o,111,6F,xxx Invalid Code
B7F8,su,032 115 117,20 73 75,JSR &7573
B7FB,c,099,63,xxx Invalid Code
B7FC,h,104,68,PLA
B7FD,li,032 108 105,20 6C 69,JSR &696C
B800,ne,110 101,6E 65,xxx Invalid Code
B802,,000,00,EQUB &00
B803,,132 027,84 1B,STY &1B
B805,,032 213 142,20 D5 8E,JSR &8ED5
B808,,201 242,C9 F2,CMP#&F2
B80A,,208 222,D0 DE,BNE -34   --> &B7EA
B80C,,032 025 176,20 19 B0,JSR &B019

B80F,,164 027,A4 1B,LDY &1B
B811,,032 029 184,20 1D B8,JSR &B81D
B814,L,076 002 144,4C 02 90,JMP &9002
B817,,132 010,84 0A,STY &0A
B819,L),076 041 156,4C 29 9C,JMP &9C29
B81C,,200,C8,INY
B81D,,177 011,B1 0B,"LDA (&0B),Y"
B81F,,201 013,C9 0D,CMP#&0D
B821,,240 004,F0 04,BEQ 4 --> &B827
B823,:,201 058,C9 3A,CMP#&3A
B825,,208 245,D0 F5,BNE -11 --> &B81C
B827,,132 010,84 0A,STY &0A
B829,`,096,60,RTS
B82A,,032 030 155,20 1E 9B,JSR &9B1E
B82D,,176 007,B0 07,BCS 7 --> &B836
B82F,o,032 111 146,20 6F 92,JSR &926F
B832,,169 128,A9 80,LDA#&80
B834,+,020 043,14 2B,TRB &2B
B836,,032 205 128,20 CD 80,JSR &80CD
B839,,144 185,90 B9,BCC -71 --> &B7F4
B83B,`,096,60,RTS
B83C,L,076 146 144,4C 92 90,JMP &9092
B83F,Li,076 105 155,4C 69 9B,JMP &9B69
B842,,132 010,84 0A,STY &0A
B844,L,076 002 144,4C 02 90,JMP &9002
B847,<,032 060 186,20 3C BA,JSR &BA3C
B84A,L,132 076,84 4C,STY &4C
B84C,u,032 117 146,20 75 92,JSR &9275
B84F,,032 229 140,20 E5 8C,JSR &8CE5
B852,,208 238,D0 EE,BNE -18 --> &B842
B854,L,165 076,A5 4C,LDA &4C
B856,H,072,48,PHA
B857,,032 174 152,20 AE 98,JSR &98AE
B85A,,240 227,F0 E3,BEQ -29 --> &B83F
B85C,u,032 117 146,20 75 92,JSR &9275
B85F,h,104,68,PLA
B860,L,133 076,85 4C,STA &4C
B862,,008,08,PHP
B863,&,032 038 188,20 26 BC,JSR &BC26
B866,L,164 076,A4 4C,LDY &4C
B868,,032 215 255,20 D7 FF,JSR &FFD7
B86B,',133 039,85 27,STA &27
B86D,(,040,28,PLP

B86E,,144 026,90 1A,BCC 26   --> &B88A
B870,',165 039,A5 27,LDA &27
B872,,208 200,D0 C8,BNE -56   --> &B83C
B874,,032 215 255,20 D7 FF,JSR &FFD7
B877,6,133 054,85 36,STA &36
B879,,170,AA,TAX
B87A,,240 009,F0 09,BEQ 9   --> &B885
B87C,,032 215 255,20 D7 FF,JSR &FFD7
B87F,,157 255 005,9D FF 05,"STA &05FF,X"
B882,,202,CA,DEX
B883,,208 247,D0 F7,BNE -9   --> &B87C
B885,,032 171 144,20 AB 90,JSR &90AB
B888,,128 197,80 C5,BRA -59   --> &B84F
B88A,',165 039,A5 27,LDA &27
B88C,,240 174,F0 AE,BEQ -82   --> &B83C
B88E,0,048 012,30 0C,BMI 12   --> &B89C
B890,,162 003,A2 03,LDX#&03
B892,,032 215 255,20 D7 FF,JSR &FFD7
B895,*,149 042,95 2A,"STA &2A,X"
B897,,202,CA,DEX
B898,,016 248,10 F8,BPL -8   --> &B892
B89A,,128 014,80 0E,BRA 14   --> &B8AA
B89C,,162 004,A2 04,LDX#&04
B89E,,032 215 255,20 D7 FF,JSR &FFD7
B8A1,l,157 108 004,9D 6C 04,"STA &046C,X"
B8A4,,202,CA,DEX
B8A5,,016 247,10 F7,BPL -9   --> &B89E
B8A7,9,032 057 165,20 39 A5,JSR &A539
B8AA,,032 006 189,20 06 BD,JSR &BD06
B8AD,8,032 056 179,20 38 B3,JSR &B338
B8B0,,128 157,80 9D,BRA -99   --> &B84F
B8B2,h,104,68,PLA
B8B3,h,104,68,PLA
B8B4,,128 142,80 8E,BRA -114   --> &B844
B8B6,,032 223 140,20 DF 8C,JSR &8CDF
B8B9,,240 140,F0 8C,BEQ -116   --> &B847
B8BB,,201 134,C9 86,CMP#&86
B8BD,,240 003,F0 03,BEQ 3   --> &B8C2
B8BF,,198 010,C6 0A,DEC &0A
B8C1,,024,18,CLC
B8C2,fL,102 076,66 4C,ROR &4C
B8C4,FL,070 076,46 4C,LSR &4C
B8C6,,169 255,A9 FF,LDA#&FF

B8C8,M,133 077,85 4D,STA &4D
B8CA,,032 153 146,20 99 92,JSR &9299
B8CD,,176 010,B0 0A,BCS 10 --> &B8D9
B8CF,,032 153 146,20 99 92,JSR &9299
B8D2,,144 251,90 FB,BCC -5 --> &B8CF
B8D4,,162 255,A2 FF,LDX#&FF
B8D6,M,134 077,86 4D,STX &4D
B8D8,,024,18,CLC
B8D9,,008,08,PHP
B8DA,L,006 076,06 4C,ASL &4C
B8DC,(,040,28,PLP
B8DD,fL,102 076,66 4C,ROR &4C
B8DF,",",201 044,C9 2C,CMP#&2C
B8E1,,240 231,F0 E7,BEQ -25 --> &B8CA
B8E3,;,201 059,C9 3B,CMP#&3B
B8E5,,240 227,F0 E3,BEQ -29 --> &B8CA
B8E7,,198 010,C6 0A,DEC &0A
B8E9,L,165 076,A5 4C,LDA &4C
B8EB,H,072,48,PHA
B8EC,M,165 077,A5 4D,LDA &4D
B8EE,H,072,48,PHA
B8EF,,032 174 152,20 AE 98,JSR &98AE
B8F2,,240 190,F0 BE,BEQ -66 --> &B8B2
B8F4,h,104,68,PLA
B8F5,M,133 077,85 4D,STA &4D
B8F7,h,104,68,PLA
B8F8,L,133 076,85 4C,STA &4C
B8FA,u,032 117 146,20 75 92,JSR &9275
B8FD,,008,08,PHP
B8FE,$L,036 076,24 4C,BIT &4C
B900,p,112 006,70 06,BVS 6 --> &B908
B902,M,165 077,A5 4D,LDA &4D
B904,,201 255,C9 FF,CMP#&FF
B906,,208 023,D0 17,BNE 23 --> &B91F
B908,$L,036 076,24 4C,BIT &4C
B90A,,016 005,10 05,BPL 5 --> &B911
B90C,?,169 063,A9 3F,LDA#&3F
B90E,,032 238 255,20 EE FF,JSR &FFEE
B911,p,032 112 186,20 70 BA,JSR &BA70
B914,6,132 054,84 36,STY &36
B916,L,006 076,06 4C,ASL &4C
B918,,024,18,CLC
B919,fL,102 076,66 4C,ROR &4C

B91B,$L,036 076,24 4C,BIT &4C
B91D,p,112 025,70 19,BVS 25 --> &B938
B91F,,133 027,85 1B,STA &1B
B921,d,100 025,64 19,STZ &19
B923,,169 006,A9 06,LDA#&06
B925,,133 026,85 1A,STA &1A
B927,,032 248 172,20 F8 AC,JSR &ACF8
B92A,,032 235 142,20 EB 8E,JSR &8EEB
B92D,,240 006,F0 06,BEQ 6 --> &B935
B92F,,201 013,C9 0D,CMP#&0D
B931,,208 247,D0 F7,BNE -9 --> &B92A
B933,,160 254,A0 FE,LDY#&FE
B935,,200,C8,INY
B936,M,132 077,84 4D,STY &4D
B938,(,040,28,PLP
B939,,176 011,B0 0B,BCS 11 --> &B946
B93B,C,032 067 188,20 43 BC,JSR &BC43
B93E,N,032 078 171,20 4E AB,JSR &AB4E
B941,+,032 043 179,20 2B B3,JSR &B32B
B944,,128 132,80 84,BRA -124 --> &B8CA
B946,d',100 039,64 27,STZ &27
B948,,032 174 144,20 AE 90,JSR &90AE
B94B,,128 247,80 F7,BRA -9 --> &B944
B94D,d=,100 061,64 3D,STZ &3D
B94F,,164 024,A4 18,LDY &18
B951,>,132 062,84 3E,STY &3E
B953,,032 224 142,20 E0 8E,JSR &8EE0
B956,,198 010,C6 0A,DEC &0A
B958,:,201 058,C9 3A,CMP#&3A
B95A,,240 011,F0 0B,BEQ 11 --> &B967
B95C,,201 013,C9 0D,CMP#&0D
B95E,,240 007,F0 07,BEQ 7 --> &B967
B960,,201 139,C9 8B,CMP#&8B
B962,,240 003,F0 03,BEQ 3 --> &B967
B964,*,032 042 184,20 2A B8,JSR &B82A
B967,,032 166 155,20 A6 9B,JSR &9BA6
B96A,=,165 061,A5 3D,LDA &3D
B96C,,133 028,85 1C,STA &1C
B96E,>,165 062,A5 3E,LDA &3E
B970,,133 029,85 1D,STA &1D
B972,L,076 005 144,4C 05 90,JMP &9005
B975,,032 229 140,20 E5 8C,JSR &8CE5
B978,,240 003,F0 03,BEQ 3 --> &B97D

B97A,L,076 000 144,4C 00 90,JMP &9000
B97D,,032 174 152,20 AE 98,JSR &98AE
B980,,240 243,F0 F3,BEQ -13 --> &B975
B982,,176 011,B0 0B,BCS 11 --> &B98F
B984,,032 172 185,20 AC B9,JSR &B9AC
B987,C,032 067 188,20 43 BC,JSR &BC43
B98A,(,032 040 179,20 28 B3,JSR &B328
B98D,,128 014,80 0E,BRA 14 --> &B99D
B98F,,032 172 185,20 AC B9,JSR &B9AC
B992,&,032 038 188,20 26 BC,JSR &BC26
B995,,032 248 172,20 F8 AC,JSR &ACF8
B998,',133 039,85 27,STA &27
B99A,,032 171 144,20 AB 90,JSR &90AB
B99D,,024,18,CLC
B99E,,165 027,A5 1B,LDA &1B
B9A0,e,101 025,65 19,ADC &19
B9A2,,133 028,85 1C,STA &1C
B9A4,,165 026,A5 1A,LDA &1A
B9A6,i,105 000,69 00,ADC#&00
B9A8,,133 029,85 1D,STA &1D
B9AA,,128 201,80 C9,BRA -55 --> &B975
B9AC,u,032 117 146,20 75 92,JSR &9275
B9AF,,165 028,A5 1C,LDA &1C
B9B1,,133 025,85 19,STA &19
B9B3,,165 029,A5 1D,LDA &1D
B9B5,,133 026,85 1A,STA &1A
B9B7,d,100 027,64 1B,STZ &1B
B9B9,,032 235 142,20 EB 8E,JSR &8EEB
B9BC,X,240 088,F0 58,BEQ 88 --> &BA16
B9BE,,201 220,C9 DC,CMP#&DC
B9C0,T,240 084,F0 54,BEQ 84 --> &BA16
B9C2,,201 013,C9 0D,CMP#&0D
B9C4,,240 009,F0 09,BEQ 9 --> &B9CF
B9C6,,032 235 142,20 EB 8E,JSR &8EEB
B9C9,K,240 075,F0 4B,BEQ 75 --> &BA16
B9CB,,201 013,C9 0D,CMP#&0D
B9CD,,208 247,D0 F7,BNE -9 --> &B9C6
B9CF,,164 027,A4 1B,LDY &1B
B9D1,,177 025,B1 19,"LDA (&19),Y"
B9D3,0,048 028,30 1C,BMI 28 --> &B9F1
B9D5,,200,C8,INY
B9D6,,200,C8,INY
B9D7,,177 025,B1 19,"LDA (&19),Y"

B9D9,,170,AA,TAX
B9DA,,200,C8,INY
B9DB,,177 025,B1 19,"LDA (&19),Y"
B9DD,,201 032,C9 20,CMP#&20
B9DF,,240 249,F0 F9,BEQ -7 --> &B9DA
B9E1,,201 220,C9 DC,CMP#&DC
B9E3,.,240 046,F0 2E,BEQ 46 --> &BA13
B9E5,,138,8A,TXA
B9E6,,024,18,CLC
B9E7,e,101 025,65 19,ADC &19
B9E9,,133 025,85 19,STA &19
B9EB,,144 226,90 E2,BCC -30 --> &B9CF
B9ED,,230 026,E6 1A,INC &1A
B9EF,,128 222,80 DE,BRA -34 --> &B9CF
B9F1,,000,00,BRK
B9F2,*,042,2A,EQUB &2A
B9F3,O,079,4F,xxx Invalid Code
B9F4,ut,117 116,75 74,"ADC &74,X"
B9F6,of,032 111 102,20 6F 66,JSR &666F
B9F9,,032 220,20 DC,xxx Invalid Code
B9FB,,000,00,BRK
B9FC,+,043,2B,EQUB &2B
B9FD,No,078 111 032,4E 6F 20,LSR &206F
BA00,,245,F5,xxx Invalid Code
BA01,,000,00,BRK
BA02,-,045,2D,EQUB &2D
BA03,#,141 035,8D 23,xxx Invalid Code
BA05,,000,00,BRK
BA06,",",044,2C,EQUB &2C
BA07,To,084 111,54 6F,xxx Invalid Code
BA09,o,111,6F,xxx Invalid Code
BA0A,ma,032 109 097,20 6D 61,JSR &616D
BA0D,ny,110 121 032,6E 79 20,ROR &2079
BA10,s,245 115,F5 73,"SBC &73,X"
BA12,,000,00,EQUB &00
BA13,,200,C8,INY
BA14,,132 027,84 1B,STY &1B
BA16,`,096,60,RTS
BA17,/,032 047 157,20 2F 9D,JSR &9D2F
BA1A,,032 145 155,20 91 9B,JSR &9B91
BA1D,,032 188 150,20 BC 96,JSR &96BC
BA20,$,166 036,A6 24,LDX &24
BA22,,240 215,F0 D7,BEQ -41 --> &B9FB

BA24,*,165 042,A5 2A,LDA &2A
BA26,+,005 043,05 2B,ORA &2B
BA28,",",005 044,05 2C,ORA &2C
BA2A,-,005 045,05 2D,ORA &2D
BA2C,,240 005,F0 05,BEQ 5 --> &BA33
BA2E,$,198 036,C6 24,DEC &24
BA30,L,076 005 144,4C 05 90,JMP &9005
BA33,,188 255 004,BC FF 04,"LDY &04FF,X"
BA36,,189 019 005,BD 13 05,"LDA &0513,X"
BA39,L2,076 050 183,4C 32 B7,JMP &B732
BA3C,,198 010,C6 0A,DEC &0A
BA3E,,165 010,A5 0A,LDA &0A
BA40,,133 027,85 1B,STA &1B
BA42,,165 011,A5 0B,LDA &0B
BA44,,133 025,85 19,STA &19
BA46,,165 012,A5 0C,LDA &0C
BA48,,133 026,85 1A,STA &1A
BA4A,,032 213 142,20 D5 8E,JSR &8ED5
BA4D,#,201 035,C9 23,CMP#&23
BA4F,,208 176,D0 B0,BNE -80 --> &BA01
BA51,,032 180 150,20 B4 96,JSR &96B4
BA54,*,164 042,A4 2A,LDY &2A
BA56,,152,98,TYA
BA57,`,096,60,RTS
BA58,$,166 036,A6 24,LDX &24
BA5A,,224 020,E0 14,CPX#&14
BA5C,,176 167,B0 A7,BCS -89 --> &BA05
BA5E,,032 188 155,20 BC 9B,JSR &9BBC
BA61,,165 011,A5 0B,LDA &0B
BA63,,157 000 005,9D 00 05,"STA &0500,X"
BA66,,165 012,A5 0C,LDA &0C
BA68,,157 020 005,9D 14 05,"STA &0514,X"
BA6B,$,230 036,E6 24,INC &24
BA6D,L,076 011 144,4C 0B 90,JMP &900B
BA70,,169 006,A9 06,LDA#&06
BA72,,128 002,80 02,BRA 2 --> &BA76
BA74,,169 007,A9 07,LDA#&07
BA76,d7,100 055,64 37,STZ &37
BA78,8,133 056,85 38,STA &38
BA7A,,169 238,A9 EE,LDA#&EE
BA7C,9,133 057,85 39,STA &39
BA7E,,169 032,A9 20,LDA#&20
BA80,:,133 058,85 3A,STA &3A

BA82,,160 255,A0 FF,LDY#&FF
BA84,;,132 059,84 3B,STY &3B
BA86,,200,C8,INY
BA87,7,162 055,A2 37,LDX#&37
BA89,,152,98,TYA
BA8A,,032 241 255,20 F1 FF,JSR &FFF1
BA8D,,144 006,90 06,BCC 6   --> &BA95
BA8F,L},076 125 155,4C 7D 9B,JMP &9B7D
BA92,,032 231 255,20 E7 FF,JSR &FFE7
BA95,d,100 030,64 1E,STZ &1E
BA97,`,096,60,RTS
BA98,,032 205 128,20 CD 80,JSR &80CD
BA9B,M,144 077,90 4D,BCC 77   --> &BAEA
BA9D,=,165 061,A5 3D,LDA &3D
BA9F,7,133 055,85 37,STA &37
BAA1,,133 018,85 12,STA &12
BAA3,>,165 062,A5 3E,LDA &3E
BAA5,8,133 056,85 38,STA &38
BAA7,,133 019,85 13,STA &13
BAA9,,160 003,A0 03,LDY#&03
BAAB,7,177 055,B1 37,"LDA (&37),Y"
BAAD,,024,18,CLC
BAAE,e7,101 055,65 37,ADC &37
BAB0,7,133 055,85 37,STA &37
BAB2,,144 002,90 02,BCC 2   --> &BAB6
BAB4,8,230 056,E6 38,INC &38
BAB6,,160 000,A0 00,LDY#&00
BAB8,7,177 055,B1 37,"LDA (&37),Y"
BABA,,145 018,91 12,"STA (&12),Y"
BABC,,201 013,C9 0D,CMP#&0D
BABE,,208 019,D0 13,BNE 19   --> &BAD3
BAC0,,200,C8,INY
BAC1,,208 004,D0 04,BNE 4   --> &BAC7
BAC3,8,230 056,E6 38,INC &38
BAC5,,230 019,E6 13,INC &13
BAC7,7,177 055,B1 37,"LDA (&37),Y"
BAC9,,145 018,91 12,"STA (&12),Y"
BACB,0,048 015,30 0F,BMI 15   --> &BADC
BACD,,032 223 186,20 DF BA,JSR &BADF
BAD0,,032 223 186,20 DF BA,JSR &BADF
BAD3,,200,C8,INY
BAD4,,208 226,D0 E2,BNE -30   --> &BAB8
BAD6,8,230 056,E6 38,INC &38

```
BAD8,,230 019,E6 13,INC &13
BADA,,128 220,80 DC,BRA -36   --> &BAB8
BADC,L,076 005 190,4C 05 BE,JMP &BE05
BADF,,200,C8,INY
BAE0,,208 004,D0 04,BNE 4   --> &BAE6
BAE2,,230 019,E6 13,INC &13
BAE4,8,230 056,E6 38,INC &38
BAE6,7,177 055,B1 37,"LDA (&37),Y"
BAE8,,145 018,91 12,"STA (&12),Y"
BAEA,`,096,60,RTS
BAEB,,162 255,A2 FF,LDX#&FF
BAED,(,134 040,86 28,STX &28
BAEF,<,134 060,86 3C,STX &3C
BAF1,,032 207 187,20 CF BB,JSR &BBCF
BAF4,,165 011,A5 0B,LDA &0B
BAF6,7,133 055,85 37,STA &37
BAF8,,165 012,A5 0C,LDA &0C
BAFA,8,133 056,85 38,STA &38
BAFC,d;,100 059,64 3B,STZ &3B
BAFE,d,100 010,64 0A,STZ &0A
BB00,,032 178 141,20 B2 8D,JSR &8DB2
BB03,,032 030 155,20 1E 9B,JSR &9B1E
BB06,,144 226,90 E2,BCC -30   --> &BAEA
BB08,,165 031,A5 1F,LDA &1F
BB0A,,240 009,F0 09,BEQ 9   --> &BB15
BB0C,,185 000 007,B9 00 07,"LDA &0700,Y"
BB0F,,200,C8,INY
BB10,,201 032,C9 20,CMP#&20
BB12,,240 248,F0 F8,BEQ -8   --> &BB0C
BB14,,136,88,DEY
BB15,;,132 059,84 3B,STY &3B
BB17,,032 152 186,20 98 BA,JSR &BA98
BB1A,,160 007,A0 07,LDY#&07
BB1C,<,132 060,84 3C,STY &3C
BB1E,,160 000,A0 00,LDY#&00
BB20,,169 013,A9 0D,LDA#&0D
BB22,;,210 059,D2 3B,CMP (&3B)
BB24,,240 196,F0 C4,BEQ -60   --> &BAEA
BB26,,200,C8,INY
BB27,;,209 059,D1 3B,"CMP (&3B),Y"
BB29,,208 251,D0 FB,BNE -5   --> &BB26
BB2B,,169 032,A9 20,LDA#&20
BB2D,,136,88,DEY
```

```
BB2E,,240 004,F0 04,BEQ 4   --> &BB34
BB30,;,209 059,D1 3B,"CMP (&3B),Y"
BB32,,240 249,F0 F9,BEQ -7   --> &BB2D
BB34,,200,C8,INY
BB35,,169 013,A9 0D,LDA#&0D
BB37,;,145 059,91 3B,"STA (&3B),Y"
BB39,,200,C8,INY
BB3A,,200,C8,INY
BB3B,,200,C8,INY
BB3C,,200,C8,INY
BB3D,?,132 063,84 3F,STY &3F
BB3F,,165 018,A5 12,LDA &12
BB41,9,133 057,85 39,STA &39
BB43,,165 019,A5 13,LDA &13
BB45,:,133 058,85 3A,STA &3A
BB47,,032 004 190,20 04 BE,JSR &BE04
BB4A,7,133 055,85 37,STA &37
BB4C,,165 019,A5 13,LDA &13
BB4E,8,133 056,85 38,STA &38
BB50,,136,88,DEY
BB51,,165 006,A5 06,LDA &06
BB53,,197 018,C5 12,CMP &12
BB55,,165 007,A5 07,LDA &07
BB57,,229 019,E5 13,SBC &13
BB59,,176 016,B0 10,BCS 16   --> &BB6B
BB5B,,032 229 189,20 E5 BD,JSR &BDE5
BB5E,,032 172 187,20 AC BB,JSR &BBAC
BB61,,000,00,BRK
BB62,,000,00,EQUB &00
BB63,,134 032,86 20,STX &20
BB65,s,115,73,xxx Invalid Code
BB66,pa,112 097,70 61,BVS 97   --> &BBC9
BB68,c,099,63,xxx Invalid Code
BB69,e,101,65,xxx Invalid Code
BB6A,,000,00,EQUB &00
BB6B,9,177 057,B1 39,"LDA (&39),Y"
BB6D,7,145 055,91 37,"STA (&37),Y"
BB6F,,152,98,TYA
BB70,,208 004,D0 04,BNE 4   --> &BB76
BB72,:,198 058,C6 3A,DEC &3A
BB74,8,198 056,C6 38,DEC &38
BB76,,136,88,DEY
BB77,,152,98,TYA
```

```
BB78,e9,101 057,65 39,ADC &39
BB7A,:,166 058,A6 3A,LDX &3A
BB7C,,144 001,90 01,BCC 1   --> &BB7F
BB7E,,232,E8,INX
BB7F,=,197 061,C5 3D,CMP &3D
BB81,,138,8A,TXA
BB82,>,229 062,E5 3E,SBC &3E
BB84,,176 229,B0 E5,BCS -27   --> &BB6B
BB86,,160 001,A0 01,LDY#&01
BB88,+,165 043,A5 2B,LDA &2B
BB8A,=,145 061,91 3D,"STA (&3D),Y"
BB8C,,200,C8,INY
BB8D,*,165 042,A5 2A,LDA &2A
BB8F,=,145 061,91 3D,"STA (&3D),Y"
BB91,,200,C8,INY
BB92,?,165 063,A5 3F,LDA &3F
BB94,=,145 061,91 3D,"STA (&3D),Y"
BB96,8,056,38,SEC
BB97,,152,98,TYA
BB98,e=,101 061,65 3D,ADC &3D
BB9A,=,133 061,85 3D,STA &3D
BB9C,,144 002,90 02,BCC 2   --> &BBA0
BB9E,>,230 062,E6 3E,INC &3E
BBA0,,160 255,A0 FF,LDY#&FF
BBA2,,200,C8,INY
BBA3,;,177 059,B1 3B,"LDA (&3B),Y"
BBA5,=,145 061,91 3D,"STA (&3D),Y"
BBA7,,201 013,C9 0D,CMP#&0D
BBA9,,208 247,D0 F7,BNE -9   --> &BBA2
BBAB,`,096,60,RTS
BBAC,,165 018,A5 12,LDA &12
BBAE,,133 000,85 00,STA &00
BBB0,,133 002,85 02,STA &02
BBB2,,165 019,A5 13,LDA &13
BBB4,,133 001,85 01,STA &01
BBB6,,133 003,85 03,STA &03
BBB8,,032 207 187,20 CF BB,JSR &BBCF
BBBB,,162 016,A2 10,LDX#&10
BBBD,,189 019 191,BD 13 BF,"LDA &BF13,X"
BBC0,,157 239 007,9D EF 07,"STA &07EF,X"
BBC3,,202,CA,DEX
BBC4,,208 247,D0 F7,BNE -9   --> &BBBD
BBC6,,162 128,A2 80,LDX#&80
```

BBC8,,158 127 004,9E 7F 04,"STZ &047F,X"
BBCB,,202,CA,DEX
BBCC,,208 250,D0 FA,BNE -6 --> &BBC8
BBCE,`,096,60,RTS
BBCF,,165 024,A5 18,LDA &18
BBD1,,133 029,85 1D,STA &1D
BBD3,,165 006,A5 06,LDA &06
BBD5,,133 004,85 04,STA &04
BBD7,,165 007,A5 07,LDA &07
BBD9,,133 005,85 05,STA &05
BBDB,,169 128,A9 80,LDA#&80
BBDD,,020 031,14 1F,TRB &1F
BBDF,d$,100 036,64 24,STZ &24
BBE1,d&,100 038,64 26,STZ &26
BBE3,d%,100 037,64 25,STZ &25
BBE5,d,100 028,64 1C,STZ &1C
BBE7,`,096,60,RTS
BBE8,,165 004,A5 04,LDA &04
BBEA,,024,18,CLC
BBEB,J,133 074,85 4A,STA &4A
BBED,i,105 005,69 05,ADC#&05
BBEF,,133 004,85 04,STA &04
BBF1,,165 005,A5 05,LDA &05
BBF3,K,133 075,85 4B,STA &4B
BBF5,i,105 000,69 00,ADC#&00
BBF7,,133 005,85 05,STA &05
BBF9,`,096,60,RTS
BBFA,,165 004,A5 04,LDA &04
BBFC,8,056,38,SEC
BBFD,,233 005,E9 05,SBC#&05
BBFF,,032 030 189,20 1E BD,JSR &BD1E
BC02,0,165 048,A5 30,LDA &30
BC04,,146 004,92 04,STA (&04)
BC06,,160 001,A0 01,LDY#&01
BC08,.,165 046,A5 2E,LDA &2E
BC0A,E1,069 049,45 31,EOR &31
BC0C,),041 128,29 80,AND#&80
BC0E,E1,069 049,45 31,EOR &31
BC10,,145 004,91 04,"STA (&04),Y"
BC12,,200,C8,INY
BC13,2,165 050,A5 32,LDA &32
BC15,,145 004,91 04,"STA (&04),Y"
BC17,,200,C8,INY

BC18,3,165 051,A5 33,LDA &33
BC1A,,145 004,91 04,"STA (&04),Y"
BC1C,,200,C8,INY
BC1D,4,165 052,A5 34,LDA &34
BC1F,,145 004,91 04,"STA (&04),Y"
BC21,`,096,60,RTS
BC22,-,240 045,F0 2D,BEQ 45 --> &BC51
BC24,0,048 212,30 D4,BMI -44 --> &BBFA
BC26,,165 004,A5 04,LDA &04
BC28,8,056,38,SEC
BC29,,233 004,E9 04,SBC#&04
BC2B,,032 030 189,20 1E BD,JSR &BD1E
BC2E,,160 003,A0 03,LDY#&03
BC30,-,165 045,A5 2D,LDA &2D
BC32,,145 004,91 04,"STA (&04),Y"
BC34,,136,88,DEY
BC35,",",165 044,A5 2C,LDA &2C
BC37,,145 004,91 04,"STA (&04),Y"
BC39,,136,88,DEY
BC3A,+,165 043,A5 2B,LDA &2B
BC3C,,145 004,91 04,"STA (&04),Y"
BC3E,*,165 042,A5 2A,LDA &2A
BC40,,146 004,92 04,STA (&04)
BC42,`,096,60,RTS
BC43,z,122,7A,PLY
BC44,,250,FA,PLX
BC45,*,165 042,A5 2A,LDA &2A
BC47,H,072,48,PHA
BC48,+,165 043,A5 2B,LDA &2B
BC4A,H,072,48,PHA
BC4B,",",165 044,A5 2C,LDA &2C
BC4D,H,072,48,PHA
BC4E,,218,DA,PHX
BC4F,Z,090,5A,PHY
BC50,`,096,60,RTS
BC51,,024,18,CLC
BC52,,165 004,A5 04,LDA &04
BC54,6,229 054,E5 36,SBC &36
BC56,,032 030 189,20 1E BD,JSR &BD1E
BC59,6,164 054,A4 36,LDY &36
BC5B,,240 008,F0 08,BEQ 8 --> &BC65
BC5D,,185 255 005,B9 FF 05,"LDA &05FF,Y"
BC60,,145 004,91 04,"STA (&04),Y"

BC62,,136,88,DEY
BC63,,208 248,D0 F8,BNE -8 --> &BC5D
BC65,6,165 054,A5 36,LDA &36
BC67,,146 004,92 04,STA (&04)
BC69,`,096,60,RTS
BC6A,9,165 057,A5 39,LDA &39
BC6C,,201 128,C9 80,CMP#&80
BC6E,%,240 037,F0 25,BEQ 37 --> &BC95
BC70,8,144 056,90 38,BCC 56 --> &BCAA
BC72,,178 004,B2 04,LDA (&04)
BC74,,170,AA,TAX
BC75,,240 022,F0 16,BEQ 22 --> &BC8D
BC77,7,178 055,B2 37,LDA (&37)
BC79,,233 001,E9 01,SBC#&01
BC7B,9,133 057,85 39,STA &39
BC7D,,160 001,A0 01,LDY#&01
BC7F,7,177 055,B1 37,"LDA (&37),Y"
BC81,,233 000,E9 00,SBC#&00
BC83,:,133 058,85 3A,STA &3A
BC85,,177 004,B1 04,"LDA (&04),Y"
BC87,9,145 057,91 39,"STA (&39),Y"
BC89,,200,C8,INY
BC8A,,202,CA,DEX
BC8B,,208 248,D0 F8,BNE -8 --> &BC85
BC8D,,178 004,B2 04,LDA (&04)
BC8F,,160 003,A0 03,LDY#&03
BC91,7,145 055,91 37,"STA (&37),Y"
BC93,L,128 076,80 4C,BRA 76 --> &BCE1
BC95,,178 004,B2 04,LDA (&04)
BC97,,170,AA,TAX
BC98,,240 012,F0 0C,BEQ 12 --> &BCA6
BC9A,,160 001,A0 01,LDY#&01
BC9C,,177 004,B1 04,"LDA (&04),Y"
BC9E,,136,88,DEY
BC9F,7,145 055,91 37,"STA (&37),Y"
BCA1,,200,C8,INY
BCA2,,200,C8,INY
BCA3,,202,CA,DEX
BCA4,,208 246,D0 F6,BNE -10 --> &BC9C
BCA6,,169 013,A9 0D,LDA#&0D
BCA8,,208 231,D0 E7,BNE -25 --> &BC91
BCAA,,178 004,B2 04,LDA (&04)
BCAC,7,146 055,92 37,STA (&37)

```
BCAE,,160 004,A0 04,LDY#&04
BCB0,9,165 057,A5 39,LDA &39
BCB2,,240 026,F0 1A,BEQ 26   --> &BCCE
BCB4,,160 001,A0 01,LDY#&01
BCB6,,177 004,B1 04,"LDA (&04),Y"
BCB8,7,145 055,91 37,"STA (&37),Y"
BCBA,,200,C8,INY
BCBB,,177 004,B1 04,"LDA (&04),Y"
BCBD,7,145 055,91 37,"STA (&37),Y"
BCBF,,200,C8,INY
BCC0,,177 004,B1 04,"LDA (&04),Y"
BCC2,7,145 055,91 37,"STA (&37),Y"
BCC4,,200,C8,INY
BCC5,9,196 057,C4 39,CPY &39
BCC7,,176 005,B0 05,BCS 5   --> &BCCE
BCC9,,177 004,B1 04,"LDA (&04),Y"
BCCB,7,145 055,91 37,"STA (&37),Y"
BCCD,,200,C8,INY
BCCE,,152,98,TYA
BCCF,,024,18,CLC
BCD0,+,128 043,80 2B,BRA 43   --> &BCFD
BCD2,,178 004,B2 04,LDA (&04)
BCD4,6,133 054,85 36,STA &36
BCD6,,240 011,F0 0B,BEQ 11   --> &BCE3
BCD8,,168,A8,TAY
BCD9,,177 004,B1 04,"LDA (&04),Y"
BCDB,,153 255 005,99 FF 05,"STA &05FF,Y"
BCDE,,136,88,DEY
BCDF,,208 248,D0 F8,BNE -8   --> &BCD9
BCE1,,178 004,B2 04,LDA (&04)
BCE3,8,056,38,SEC
BCE4,,128 023,80 17,BRA 23   --> &BCFD
BCE6,,160 003,A0 03,LDY#&03
BCE8,,177 004,B1 04,"LDA (&04),Y"
BCEA,-,133 045,85 2D,STA &2D
BCEC,,136,88,DEY
BCED,,177 004,B1 04,"LDA (&04),Y"
BCEF,",",133 044,85 2C,STA &2C
BCF1,,136,88,DEY
BCF2,,177 004,B1 04,"LDA (&04),Y"
BCF4,+,133 043,85 2B,STA &2B
BCF6,,178 004,B2 04,LDA (&04)
BCF8,*,133 042,85 2A,STA &2A
```

```
BCFA,,024,18,CLC
BCFB,,169 004,A9 04,LDA#&04
BCFD,e,101 004,65 04,ADC &04
BCFF,,133 004,85 04,STA &04
BD01,,144 002,90 02,BCC 2   --> &BD05
BD03,,230 005,E6 05,INC &05
BD05,`,096,60,RTS
BD06,7,162 055,A2 37,LDX#&37
BD08,,160 003,A0 03,LDY#&03
BD0A,,177 004,B1 04,"LDA (&04),Y"
BD0C,,149 003,95 03,"STA &03,X"
BD0E,,136,88,DEY
BD0F,,177 004,B1 04,"LDA (&04),Y"
BD11,,149 002,95 02,"STA &02,X"
BD13,,136,88,DEY
BD14,,177 004,B1 04,"LDA (&04),Y"
BD16,,149 001,95 01,"STA &01,X"
BD18,,178 004,B2 04,LDA (&04)
BD1A,,149 000,95 00,"STA &00,X"
BD1C,,128 220,80 DC,BRA -36   --> &BCFA
BD1E,,133 004,85 04,STA &04
BD20,,176 002,B0 02,BCS 2   --> &BD24
BD22,,198 005,C6 05,DEC &05
BD24,,164 005,A4 05,LDY &05
BD26,,196 003,C4 03,CPY &03
BD28,,144 010,90 0A,BCC 10   --> &BD34
BD2A,,208 004,D0 04,BNE 4   --> &BD30
BD2C,,197 002,C5 02,CMP &02
BD2E,,144 004,90 04,BCC 4   --> &BD34
BD30,`,096,60,RTS
BD31,,032 172 187,20 AC BB,JSR &BBAC
BD34,L,076 161 144,4C A1 90,JMP &90A1
BD37,7,133 055,85 37,STA &37
BD39,,201 128,C9 80,CMP#&80
BD3B,W,144 087,90 57,BCC 87   --> &BD94
BD3D,V,169 086,A9 56,LDA#&56
BD3F,8,133 056,85 38,STA &38
BD41,,169 132,A9 84,LDA#&84
BD43,9,133 057,85 39,STA &39
BD45,Z,090,5A,PHY
BD46,,160 000,A0 00,LDY#&00
BD48,,200,C8,INY
BD49,8,177 056,B1 38,"LDA (&38),Y"
```

BD4B,,016 251,10 FB,BPL -5   --> &BD48
BD4D,7,197 055,C5 37,CMP &37
BD4F,,240 013,F0 0D,BEQ 13   --> &BD5E
BD51,,200,C8,INY
BD52,,152,98,TYA
BD53,8,056,38,SEC
BD54,e8,101 056,65 38,ADC &38
BD56,8,133 056,85 38,STA &38
BD58,,144 236,90 EC,BCC -20   --> &BD46
BD5A,9,230 057,E6 39,INC &39
BD5C,,128 232,80 E8,BRA -24   --> &BD46
BD5E,,160 000,A0 00,LDY#&00
BD60,8,177 056,B1 38,"LDA (&38),Y"
BD62,0,048 006,30 06,BMI 6   --> &BD6A
BD64,,032 148 189,20 94 BD,JSR &BD94
BD67,,200,C8,INY
BD68,,208 246,D0 F6,BNE -10   --> &BD60
BD6A,z,122,7A,PLY
BD6B,`,096,60,RTS
BD6C,H,072,48,PHA
BD6D,J,074,4A,LSR A
BD6E,J,074,4A,LSR A
BD6F,J,074,4A,LSR A
BD70,J,074,4A,LSR A
BD71,w,032 119 189,20 77 BD,JSR &BD77
BD74,h,104,68,PLA
BD75,),041 015,29 0F,AND#&0F
BD77,,201 010,C9 0A,CMP#&0A
BD79,,144 002,90 02,BCC 2   --> &BD7D
BD7B,i,105 006,69 06,ADC#&06
BD7D,i0,105 048,69 30,ADC#&30
BD7F,H,072,48,PHA
BD80,#,165 035,A5 23,LDA &23
BD82,,197 030,C5 1E,CMP &1E
BD84,,176 003,B0 03,BCS 3   --> &BD89
BD86,,032 146 186,20 92 BA,JSR &BA92
BD89,h,104,68,PLA
BD8A,,230 030,E6 1E,INC &1E
BD8C,l,108 014 002,6C 0E 02,JMP (&020E)
BD8F,l,032 108 189,20 6C BD,JSR &BD6C
BD92,,169 032,A9 20,LDA#&20
BD94,$,036 031,24 1F,BIT &1F
BD96,0,048 010,30 0A,BMI 10   --> &BDA2

BD98,,201 013,C9 0D,CMP#&0D
BD9A,,208 227,D0 E3,BNE -29 --> &BD7F
BD9C,,032 238 255,20 EE FF,JSR &FFEE
BD9F,L,076 149 186,4C 95 BA,JMP &BA95
BDA2,,146 002,92 02,STA (&02)
BDA4,,230 002,E6 02,INC &02
BDA6,,208 029,D0 1D,BNE 29 --> &BDC5
BDA8,,230 003,E6 03,INC &03
BDAA,H,072,48,PHA
BDAB,,165 003,A5 03,LDA &03
BDAD,E,069 007,45 07,EOR &07
BDAF,,240 128,F0 80,BEQ 128 --> &BE31
BDB1,h,104,68,PLA
BDB2,`,096,60,RTS
BDB3,,024,18,CLC
BDB4,%,037 031,25 1F,AND &1F
BDB6,,240 013,F0 0D,BEQ 13 --> &BDC5
BDB8,,138,8A,TXA
BDB9,0,048 010,30 0A,BMI 10 --> &BDC5
BDBB,*,042,2A,ROL A
BDBC,,170,AA,TAX
BDBD,,240 006,F0 06,BEQ 6 --> &BDC5
BDBF,,032 146 189,20 92 BD,JSR &BD92
BDC2,,202,CA,DEX
BDC3,,208 250,D0 FA,BNE -6 --> &BDBF
BDC5,`,096,60,RTS
BDC6,*,165 042,A5 2A,LDA &2A
BDC8,,149 000,95 00,"STA &00,X"
BDCA,+,165 043,A5 2B,LDA &2B
BDCC,,149 001,95 01,"STA &01,X"
BDCE,",",165 044,A5 2C,LDA &2C
BDD0,,149 002,95 02,"STA &02,X"
BDD2,-,165 045,A5 2D,LDA &2D
BDD4,,149 003,95 03,"STA &03,X"
BDD6,`,096,60,RTS
BDD7,A,032 065 190,20 41 BE,JSR &BE41
BDDA,d=,100 061,64 3D,STZ &3D
BDDC,,160 000,A0 00,LDY#&00
BDDE,,169 255,A9 FF,LDA#&FF
BDE0,7,162 055,A2 37,LDX#&37
BDE2,,032 221 255,20 DD FF,JSR &FFDD
BDE5,,165 024,A5 18,LDA &18
BDE7,,133 019,85 13,STA &13

```
BDE9,d,100 018,64 12,STZ &12
BDEB,,160 001,A0 01,LDY#&01
BDED,,178 018,B2 12,LDA (&12)
BDEF,,201 013,C9 0D,CMP#&0D
BDF1,,208 030,D0 1E,BNE 30   --> &BE11
BDF3,,177 018,B1 12,"LDA (&12),Y"
BDF5,0,048 012,30 0C,BMI 12   --> &BE03
BDF7,,160 003,A0 03,LDY#&03
BDF9,,177 018,B1 12,"LDA (&12),Y"
BDFB,,240 020,F0 14,BEQ 20   --> &BE11
BDFD,,024,18,CLC
BDFE,,032 006 190,20 06 BE,JSR &BE06
BE01,,128 234,80 EA,BRA -22   --> &BDED
BE03,,200,C8,INY
BE04,,024,18,CLC
BE05,,152,98,TYA
BE06,e,101 018,65 12,ADC &12
BE08,,133 018,85 12,STA &12
BE0A,,144 002,90 02,BCC 2   --> &BE0E
BE0C,,230 019,E6 13,INC &13
BE0E,,160 001,A0 01,LDY#&01
BE10,`,096,60,RTS
BE11,,032 207 190,20 CF BE,JSR &BECF
BE14,Ba,013 066 097,0D 42 61,ORA &6142
BE17,d,100 032,64 20,STZ &20
BE19,pr,112 114,70 72,BVS 114   --> &BE8D
BE1B,o,111,6F,xxx Invalid Code
BE1C,g,103,67,xxx Invalid Code
BE1D,ra,114 097,72 61,ADC (&61)
BE1F,m,109 013 234,6D 0D EA,ADC &EA0D
BE22,L,076 134 143,4C 86 8F,JMP &8F86
BE25,d7,100 055,64 37,STZ &37
BE27,,169 006,A9 06,LDA#&06
BE29,8,133 056,85 38,STA &38
BE2B,6,164 054,A4 36,LDY &36
BE2D,,169 013,A9 0D,LDA#&0D
BE2F,,153 000 006,99 00 06,"STA &0600,Y"
BE32,`,096,60,RTS
BE33,L,076 146 144,4C 92 90,JMP &9092
BE36,/,032 047 157,20 2F 9D,JSR &9D2F
BE39,,208 248,D0 F8,BNE -8   --> &BE33
BE3B,%,032 037 190,20 25 BE,JSR &BE25
BE3E,L,076 145 155,4C 91 9B,JMP &9B91
```

BE41,6,032 054 190,20 36 BE,JSR &BE36
BE44,,136,88,DEY
BE45,9,132 057,84 39,STY &39
BE47,,165 024,A5 18,LDA &18
BE49,:,133 058,85 3A,STA &3A
BE4B,,169 130,A9 82,LDA#&82
BE4D,,032 244 255,20 F4 FF,JSR &FFF4
BE50,;,134 059,86 3B,STX &3B
BE52,<,132 060,84 3C,STY &3C
BE54,`,096,60,RTS
BE55,,032 229 189,20 E5 BD,JSR &BDE5
BE58,A,032 065 190,20 41 BE,JSR &BE41
BE5B,?,134 063,86 3F,STX &3F
BE5D,@,132 064,84 40,STY &40
BE5F,C,134 067,86 43,STX &43
BE61,D,132 068,84 44,STY &44
BE63,G,134 071,86 47,STX &47
BE65,H,132 072,84 48,STY &48
BE67,dA,100 065,64 41,STZ &41
BE69,,166 018,A6 12,LDX &12
BE6B,E,134 069,86 45,STX &45
BE6D,,166 019,A6 13,LDX &13
BE6F,F,134 070,86 46,STX &46
BE71,+,162 043,A2 2B,LDX#&2B
BE73,=,134 061,86 3D,STX &3D
BE75,,162 128,A2 80,LDX#&80
BE77,>,134 062,86 3E,STX &3E
BE79,,166 024,A6 18,LDX &18
BE7B,B,134 066,86 42,STX &42
BE7D,,169 000,A9 00,LDA#&00
BE7F,,168,A8,TAY
BE80,7,162 055,A2 37,LDX#&37
BE82,,032 221 255,20 DD FF,JSR &FFDD
BE85,$,128 036,80 24,BRA 36 --> &BEAB
BE87,6,032 054 190,20 36 BE,JSR &BE36
BE8A,,162 000,A2 00,LDX#&00
BE8C,,160 006,A0 06,LDY#&06
BE8E,,032 247 255,20 F7 FF,JSR &FFF7
BE91,,128 024,80 18,BRA 24 --> &BEAB
BE93,,169 003,A9 03,LDA#&03
BE95,,128 002,80 02,BRA 2 --> &BE99
BE97,,169 001,A9 01,LDA#&01
BE99,H,072,48,PHA

BE9A,>,032 062 186,20 3E BA,JSR &BA3E
BE9D,Z,090,5A,PHY
BE9E,R,032 082 155,20 52 9B,JSR &9B52
BEA1,,032 188 150,20 BC 96,JSR &96BC
BEA4,z,122,7A,PLY
BEA5,*,162 042,A2 2A,LDX#&2A
BEA7,h,104,68,PLA
BEA8,,032 218 255,20 DA FF,JSR &FFDA
BEAB,L,076 005 144,4C 05 90,JMP &9005
BEAE,>,032 062 186,20 3E BA,JSR &BA3E
BEB1,,032 150 155,20 96 9B,JSR &9B96
BEB4,*,164 042,A4 2A,LDY &2A
BEB6,,169 000,A9 00,LDA#&00
BEB8,,032 206 255,20 CE FF,JSR &FFCE
BEBB,,128 238,80 EE,BRA -18 --> &BEAB
BEBD,>,032 062 186,20 3E BA,JSR &BA3E
BEC0,H,072,48,PHA
BEC1,,032 172 150,20 AC 96,JSR &96AC
BEC4,,032 150 155,20 96 9B,JSR &9B96
BEC7,z,122,7A,PLY
BEC8,*,165 042,A5 2A,LDA &2A
BECA,,032 212 255,20 D4 FF,JSR &FFD4
BECD,,128 220,80 DC,BRA -36 --> &BEAB
BECF,h,104,68,PLA
BED0,7,133 055,85 37,STA &37
BED2,h,104,68,PLA
BED3,8,133 056,85 38,STA &38
BED5,,128 003,80 03,BRA 3 --> &BEDA
BED7,,032 227 255,20 E3 FF,JSR &FFE3
BEDA,,032 169 141,20 A9 8D,JSR &8DA9
BEDD,,016 248,10 F8,BPL -8 --> &BED7
BEDF,l7,108 055 000,6C 37 00,JMP (&0037)
BEE2,,169 005,A9 05,LDA#&05
BEE4,,218,DA,PHX
BEE5,*,162 042,A2 2A,LDX#&2A
BEE7,,160 000,A0 00,LDY#&00
BEE9,,032 241 255,20 F1 FF,JSR &FFF1
BEEC,,250,FA,PLX
BEED,.,165 046,A5 2E,LDA &2E
BEEF,*,230 042,E6 2A,INC &2A
BEF1,,208 010,D0 0A,BNE 10 --> &BEFD
BEF3,+,230 043,E6 2B,INC &2B
BEF5,,208 006,D0 06,BNE 6 --> &BEFD

BEF7,",",230 044,E6 2C,INC &2C
BEF9,,208 002,D0 02,BNE 2 --> &BEFD
BEFB,-,230 045,E6 2D,INC &2D
BEFD,`,096,60,RTS
BEFE,,169 013,A9 0D,LDA#&0D
BF00,,164 024,A4 18,LDY &18
BF02,,132 019,84 13,STY &13
BF04,d,100 018,64 12,STZ &12
BF06,d,100 032,64 20,STZ &20
BF08,,146 018,92 12,STA (&12)
BF0A,,169 255,A9 FF,LDA#&FF
BF0C,,160 001,A0 01,LDY#&01
BF0E,,145 018,91 12,"STA (&12),Y"
BF10,,200,C8,INY
BF11,,132 018,84 12,STY &12
BF13,`,096,60,RTS
BF14,,184,B8,CLV
BF15,,167,A7,xxx Invalid Code
BF16,,238 165 166,EE A5 A6,INC &A6A5
BF19,,166 141,A6 8D,LDX &8D
BF1B,,166 202,A6 CA,LDX &CA
BF1D,A,172 065 165,AC 41 A5,LDY &A541
BF20,J,025 165 074,19 A5 4A,"ORA &4AA5,Y"
BF23,.,046 129 201,2E 81 C9,ROL &C981
BF26,,016 000,10 00,BPL 0 --> &BF28
BF28,,000,00,BRK
BF29,o,111,6F,xxx Invalid Code
BF2A,w,021 119,15 77,"ORA &77,X"
BF2C,z,122,7A,PLY
BF2D,a I,097 129 073,61 81 49,"ADC (&4981,X)"
BF30,,015,0F,xxx Invalid Code
BF31,,218,DA,PHX
BF32,,162 128,A2 80,LDX#&80
BF34,"""",034,22,xxx Invalid Code
BF35,n,249 131 110,F9 83 6E,"SBC &6E83,Y"
BF38,{,123,7B,xxx Invalid Code
BF39,5,014 250 053,0E FA 35,ASL &35FA
BF3C,,018 134,12 86,ORA (&86)
BF3E,e.,101 046,65 2E,ADC &2E
BF40,,224 211,E0 D3,CPX#&D3
BF42,,127,7F,xxx Invalid Code
BF43,^[,094 091 216,5E 5B D8,"LSR &D85B,X"
BF46,,170,AA,TAX

BF47,,130,82,xxx Invalid Code
BF48,- T,045 248 084,2D F8 54,AND &54F8
BF4B,X,088,58,CLI
BF4C,1,128 049,80 31,BRA 49   --> &BF7F
BF4E,r,114 023,72 17,ADC (&17)
BF50,,248,F8,SED
BF51,,128 011,80 0B,BRA 11   --> &BF5E
BF53,,215,D7,xxx Invalid Code
BF54,P),080 041,50 29,BVC 41   --> &BF7F
BF56,| |,124 210 124,7C D2 7C,"JMP (&7CD2,X)"
BF59,,134 005,86 05,STX &05
BF5B,,128 021,80 15,BRA 21   --> &BF72
BF5D,R,082 182,52 B6,EOR (&B6)
BF5F,6|,054 124,36 7C,"ROL &7C,X"
BF61,6,153 152 054,99 98 36,"STA &3698,Y"
BF64,,004 128,04 80,TSB &80
BF66,@,064,40,RTI
BF67,,000,00,BRK
BF68,,001 016,01 10,"ORA (&10,X)"
BF6A,,127,7F,xxx Invalid Code
BF6B,*,042,2A,ROL A
BF6C,,170,AA,TAX
BF6D,,170,AA,TAX
BF6E,,227,E3,xxx Invalid Code
BF6F,,127,7F,xxx Invalid Code
BF70,,255,FF,xxx Invalid Code
BF71,,255,FF,xxx Invalid Code
BF72,,255,FF,xxx Invalid Code
BF73,,255,FF,xxx Invalid Code
BF74,z,122,7A,PLY
BF75,,195,C3,xxx Invalid Code
BF76,,030 024 190,1E 18 BE,"ASL &BE18,X"
BF79,s,115,73,xxx Invalid Code
BF7A,aqU,097 113 085,61 71 55,"ADC (&5571,X)"
BF7D,-{,045 123 140,2D 7B 8C,AND &8C7B
BF80,,155,9B,xxx Invalid Code
BF81,,145 136,91 88,"STA (&88),Y"
BF83,w,119,77,xxx Invalid Code
BF84,+,043,2B,xxx Invalid Code
BF85,,164 196,A4 C4,LDY &C4
BF87,S,083,53,xxx Invalid Code
BF88,|L,124 076 204,7C 4C CC,"JMP (&CC4C,X)"
BF8B,,202,CA,DEX

BF8C,,183,B7,xxx Invalid Code
BF8D,~,126 170 170,7E AA AA,"ROR &AAAA,X"
BF90,,170,AA,TAX
BF91,,166 129,A6 81,LDX &81
BF93,,000,00,BRK
BF94,,000,00,BRK
BF95,,000,00,BRK
BF96,,000,00,BRK
BF97,},125 163 242,7D A3 F2,"ADC &F2A3,X"
BF9A,,239,EF,xxx Invalid Code
BF9B,D,068,44,xxx Invalid Code
BF9C,~,126 031 001,7E 1F 01,"ROR &011F,X"
BF9F,M,161 077,A1 4D,"LDA (&4D,X)"
BFA1,,127,7F,xxx Invalid Code
BFA2,am,097 109 244,61 6D F4,"ADC (&F46D,X)"
BFA5,?,063,3F,xxx Invalid Code
BFA6,~\,126 092 145,7E 5C 91,"ROR &915C,X"
BFA9,#,035,23,xxx Invalid Code
BFAA,~v,172 126 118,AC 7E 76,LDY &767E
BFAD,,184,B8,CLV
BFAE,},141 026 125,8D 1A 7D,STA &7D1A
BFB1,>,029 062 171,1D 3E AB,"ORA &AB3E,X"
BFB4,",",044 129 009,2C 81 09,BIT &0981
BFB7,A,065 129,41 81,"EOR (&81,X)"
BFB9,,210 128,D2 80,CMP (&80)
BFBB,t,116 223,74 DF,"STZ &DF,X"
BFBD,,189 032 128,BD 20 80,"LDA &8020,X"
BFC0,,131,83,xxx Invalid Code
BFC1,,139,8B,xxx Invalid Code
BFC2,,031,1F,xxx Invalid Code
BFC3,,181 127,B5 7F,"LDA &7F,X"
BFC5,,130,82,xxx Invalid Code
BFC6,Y,089 173 171,59 AD AB,"EOR &ABAD,Y"
BFC9,m,128 109,80 6D,BRA 109 --> &C038
BFCB,c,099,63,xxx Invalid Code
BFCC,8,056,38,SEC
BFCD,",}",044 125 017,2C 7D 11,BIT &117D
BFD0,,212,D4,xxx Invalid Code
BFD1,,177 209,B1 D1,"LDA (&D1),Y"
BFD3,yh,121 104 188,79 68 BC,"ADC &BC68,Y"
BFD6,O,079,4F,xxx Invalid Code
BFD7,Yu,089 117 005,59 75 05,"EOR &0575,Y"
BFDA,", 9",044 158 057,2C 9E 39,BIT &399E

BFDD,{,123,7B,xxx Invalid Code
BFDE,,008,08,PHP
BFDF,,136,88,DEY
BFE0,;,059,3B,xxx Invalid Code
BFE1,l,166 108,A6 6C,LDX &6C
BFE3,1,049 207,31 CF,"AND (&CF,Y)"
BFE5,,209 140,D1 8C,"CMP (&8C),Y"
BFE7,}*,125 042 170,7D 2A AA,"ADC &AA2A,X"
BFEA,,170,AA,TAX
BFEB,,137 127,89 7F,BIT#&7F
BFED,,255,FF,xxx Invalid Code
BFEE,,255,FF,xxx Invalid Code
BFEF,,255,FF,xxx Invalid Code
BFF0,,232,E8,INX
BFF1,,129 000,81 00,"STA (&00,X)"
BFF3,,000,00,BRK
BFF4,,000,00,BRK
BFF5,,000,00,BRK
BFF6,,129 000,81 00,"STA (&00,X)"
BFF8,,000,00,BRK
BFF9,,000,00,BRK
BFFA,,000,00,BRK
BFFB,Ro,082 111,52 6F,EOR (&6F)
BFFD,g,103,67,xxx Invalid Code
BFFE,er,101 114,65 72,ADC &72

8-Bit Software

The BBC and Master Computer Public Domain Library

BASIC IV ROM Routines

Page Last Altered: undefined

REFERENCE DISC IMAGES (SSD)

Submitted by Steve Fewell

Here are a few discs that I have made available. The discs are in .SSD format and work with BeebEM (and probably other BBC emulators that support .SSD format).

Disc 1 contains several maths functions that I have written myself.
The transcendental functions on this disc (i.e. EXP/LOG) are evaluated using power-series expansions rather than continued-fraction expansions (as in BASIC). The programs are in BASIC not assembly language.
This disc provides a different perspective of these functions and an alternative implementation than the one used by BBC BASIC 4.
The disk contains the following BASIC programs:
COS - Cosine function
EXP - Exponential function
LN - Natural Logorithmn function
POWER - Evaluation of powers (using LN & EXP)
REAL-FL - Converts from real to Floating (scientific - #.####E##) format
SIN - Sine function
TAN - Tangent function
TRIG & TRIG2 - Graphs showing the trigonometry functions Sine, Cosine, Tangent, Arc Sine, Arc Cosine & Arc Tangent

Disc 2 contains several analysis programs that I have written myself.
These programs have helped me in analysing the BASIC ROM.
The disk contains the following BASIC programs:
FLOAT - Converts between packed Float variables and Real (base 10) values

FLOATH - Converts between packed Float variables and Real (base 10) values,
the variable's exponent (1 byte) and mantissa (4 bytes) are entered in Hexadecimal
SEARCHM - Searches memory between &8000 to &BFFF (The BASIC ROM) for the
assembly language that you type in. I.e. you can type assembly language(i.e.*LDA#ASC(">"):JSR &FFEE*,
or *EQUS"BASIC"*) and the program will return the locations which contain that informaton.
I used this program extensively when searching for the BASIC4 routine entry points
for a lot of the routines mentioned in THE ADVANCED BASIC ROM USER GUIDE book by Colin Pharo,
where I was only armed with the BASIC2 version of the routine's code.
DISASS - A quick and dirty 6502 disassembler that I used to disassemble the BASIC ROM.
ASSEMNE - A program to extract 6502 OPCODE Mnemonics from the BASIC ROM assembly routine's table.

Another piece of software that I have used extensively is FingerPrint by Dabs Press
This software enables a machine code routine to be monitored instruction by instruction
showing the real-time contents of the registers and flags after each step.
This software is not available from 8bs, as it is copyrighted; however, it may be
available elsewhere on the Internet.
