System Monitor Commands

Apple II contains a powerful machine level monitor for use by the advanced programmer. To enter the monitor either press RESET button on keyboard or CALL-151 (Hex FF65) from Basic. Apple II will respond with an "*" (asterisk) prompt character on the TV display. This action will not kill current BASIC program which may be re-entered by a $C^{\rm C}$ (control C). NOTE: "adrs" is a four digit hexidecimal number and "data" is a two digit hexidecimal number. Remember to press "return" button at the end of each line.

<u>Command Format</u>	<pre>Example</pre>	<u>Description</u>
Examine Memory		
adrs	*CØF2	Examines (displays) single memory location of (adrs)
adrsl.adrs2	*1024.1048	Examines (displays) range of memory from (adrsl) thru (adrs2)
(return)	*(return)	Examines (displays) next 8 memory locations.
.adrs2	*.4096	Examines (displays) memory from current location through location (adrs2)
Change Memory		
adrs:data data data	*A256:EF 2Ø 43	Deposits data into memory starting at location (adrs).
:data data data	*:FØ A2 12	Deposits data into memory starting after (adrs) last used for deposits.
Move Memory		
adrsl <adrs2. adrs3M</adrs2. 	*100 <b010.b410m< td=""><td>Copy the data now in the memory range from (adrs2) to (adrs3) into memory locations starting at (adrs1).</td></b010.b410m<>	Copy the data now in the memory range from (adrs2) to (adrs3) into memory locations starting at (adrs1).
Verify Memory		
adsr1 <adrs2 adrs3V</adrs2 	*100 <b010.b410v< td=""><td>Verify that block of data in memory range from (adrs2) to (adrs3) exactly matches data block starting at memory location (adrs1) and displays differences if any.</td></b010.b410v<>	Verify that block of data in memory range from (adrs2) to (adrs3) exactly matches data block starting at memory location (adrs1) and displays differences if any.

Command Format	<u>Example</u>	<u>Description</u>
Cassette I/O		
adrsl.adrs2R	*300.4FFR	Reads cassette data into specified memory (adrs) range. Record length must be same as memory range or an error will occur.
adrs1.adrs2W	*800.9FFW	Writes onto cassette data from specified memory (adrs) range.
Display		
I	*I	Set inverse video mode. (Black characters on white background)
М	*N	Set normal video mode. (White characters on black background)
Dis-assembler		
adrsL	*C800L	Decodes 20 instructions starting at memory (adrs) into 6502 assembly nmenonic code.
L	*[Decodes next 20 instructions starting at current memory address.
Mini-assembler		
(Turn-on)	*F666G	Turns-on mini-assembler. Prompt character is now a "!" (exclamation point).
<pre>\$(monitor: command)</pre>	\$C800L	Executes any monitor command from miniassembler then returns control to miniassembler. Note that many monitor commands change current memory address reference so that it is good practice to retype desired address reference upon return to miniassembler.
adrs:(6502 MNEMONIC instruction)	!CØ10:STA 23FF	Assembles a mnemonic 6502 instruction into machine codes. If error, machine will refuse instruction, sound bell, and reprint line with up arrow under error.

Command Format	Example	Description
<u>communa i oi mat</u>	<u> </u>	<u>Besch (peron</u>
(space) (6502 mnemonic instruction)	! STA Ø1FF	Assembles instruction into next available memory location. (Note space between "f" and instruction)
(TURN-OFF)	! (Reset Button)	Exits mini-assembler and returns to system monitor.
Monitor Program Exe	ecution and Debuging	
adrsG	*300G	Runs machine level program starting at memory (adrs).
adrsT	*800T	Traces a program starting at memory location (adrs) and continues trace until hitting a breakpoint. Break occurs on instruction 00 (BRK), and returns control to system monitor. Opens 6502 status registers (see note 1)
asrdS	*CØ5ØS	Single steps through program beginning at memory location (adrs). Type a letter S for each additional step that you want displayed. Opens 6502 status registers (see Note 1).
(Control E)	*EC	Displays 6502 status registers and opens them for modification (see Note 1)
(Control Y)	*YC	Executes user specified machine

Executes user specified machine language subroutine starting at

memory location (3F8).

Note 1:

 $65 \mbox{\it 02}$ status registers are open if they are last line displayed on screen. To change them type ":" then "data" for each register.

Example: A = 3C X = FF $Y = \emptyset\emptyset$ P = 32 S = F2 *: FF Changes A register Changes A register only Changes A, X, and Y registers *:FF ØØ 33

To change S register, you must first retype data for A, X, Y and P.

Hexidecimal Arithmetic

datal+data2	*78+34	Performs hexidecimal plus data2.	sum of datal
datal-data2	*AE-34	Performs hexidecimal datal minus data2.	difference of

<u>Command Format</u>	<u>Example</u>	<u>Description</u>
Set Input/Output Por	<u>rts</u>	
(X) (Control P)	*5PC	Sets printer output to I/O slot number (X). (see Note 2 below)
(X) (Control K)	*2KC	Sets keyboard input to I/O slot number (X). (see Note 2 below)

Note 2:

Only slots 1 through 7 are addressable in this mode. Address Ø (Ex: MP^{C} or ØK $^{\text{C}}$) resets ports to internal video display and keyboard. These commands will not work unless Apple II interfaces are plugged into specificed I/O slot.

Multiple Commands

<u> </u>		
	*100L 400G AFFT	Multiple monitor commands may be given on same line if separated by a "space".
	*LLLL	Single letter commands may be repeated without spaces.

SPECIAL CONTROL AND EDITING CHARACTERS

"Control" characters are indicated by a super-scripted "C" such as G^C . They are obtained by holding down the CTRL key while typing the specified letter. Control characters are NOT displayed on the TV screen. B^C and C^C must be followed by a carriage return. Screen editing characters are indicated by a sub-scripted "E" such as D_C . They are obtained by pressing and releasing the ESC key then typing specified letter. Edit characters send information only to display screen and does not send data to memory. For example, U^C moves to cursor to right and copies text while A_E moves cursor to right but does not copy text.

<u>CHARACTER</u>	DESCRIPTION OF ACTION
RESET key	Immediately interrupts any program execution and resets computer. Also sets all text mode with scrolling window at maximum. Control is transferred to System Monitor and Apple prompts with a "*" (asterisk) and a bell. Hitting RESET key does NOT destroy existing BASIC or machine language program.
Control B	If in System Monitor (as indicated by a "*"), a control B and a carriage return will transfer control to BASIC, scratching (killing) any existing BASIC program and set HIMEM: to maximum installed user memory and LOMEM: to 2048.
Control C	If in BASIC, halts program and displays line number where stop occurred*. Program may be continued with a CON command. If in System Monitor, (as indicated by "*"), control C and a carriage return will enter BASIC without killing current program.
Control G	Sounds bell (beeps speaker)
Control H	Backspaces cursor and deletes any overwritten characters from computer but not from screen. Apply supplied keyboards have special key "4" on right side of keyboard that provides this functions without using control button.
Control J	Issues line feed only
Control V	Compliment to $H^{\mathbb{C}}$. Forward spaces cursor and copies over written characters. Apple keyboards have "+" key on right side which also performs this function.
Control X	Immediately deletes current line.

* If BASIC program is expecting keyboard input, you will have

to hit carriage return key after typing control C.

SPECIAL CONTROL AND EDITING CHARACTERS

(continued)

CHARACTER	DESCRIPTION OF ACTION
A _E	Move cursor to right
B _E	Move cursor to left
c_{E}	Move cursor down
D _E	Move cursor up
EE	Clear text from cursor to end of line
FE	Clear text from cursor to end of page
[@] E	Home cursor to top of page, clear text to end of page.

Special Controls and Features

<u>Hex</u>	BASIC Example	<u>Description</u>
Display Mo	de Controls	
C05Ø C051 C052 C053 C054 C055 C056 C057	10 POKE -16304,0 20 POKE -16303,0 30 POKE -16302,0 40 POKE -16301,0 50 POKE -16300,0 60 POKE -16299,0 70 POKE -16298,0 80 POKE -16297,0	Set color graphics mode Set text mode Clear mixed graphics Set mixed graphics (4 lines text) Clear display Page 2 (BASIC commands use Page 1 only) Set display to Page 2 (alternate) Clear HIRES graphics mode Set HIRES graphics mode
TEXT Mode	<u>Controls</u>	
0020	90 POKE 32,L1	Set left side of scrolling window to location specified by L1 in range of \emptyset to 39.
0021	100 POKE 33,W1	Set window width to amount specified by Wl. Ll+Wl<40. Wl>0
0022	110 POKE 34,11	Set window top to line specified by Tl in range of Ø to 23
0023	120 POKE 35,B1	Set window bottom to line specified by Bl in the range of \emptyset to 23. B1>T1
0024	130 CH=PEEK(36) 140 POKE 36,CH 150 TAB(CH+1)	Read/set cusor horizontal position in the range of Ø to 39. If using TAB, you must add "1" to cusor position read value; Ex. 14Ø and 15Ø perform identical function.
0025	160 CV=PEEK(37) 170 POKE 37,CV 180 VTAB(CV+1)	Similar to above. Read/set cusor vertical position in the range \emptyset to 23.
0032	190 POKE 50,127 200 POKE 50,255	Set inverse flag if 127 (Ex. 190) Set normal flag if 255(Ex. 200)
FC58	210 CALL -936	(@ _E) Home cusor, clear screen
FC42	220 CALL -958	(F _E) Clear from cusor to end of page

<u>Hex</u>	BASIC Example	<u>Description</u>
FC9C	23Ø CALL -868	(E _E) Clear from cusor to end of line
FC66	240 CALL -922	(J ^C) Line feed
FC7Ø	250 CALL -912	Scroll up text one line

Miscellaneous

CØ3Ø	360 X=PEEK(-16336) 365 POKE -16336,0	Toggle speaker
CØØØ	370 X=PEEK(-16384	Read keyboard; if X>127 then key was pressed.
CØ1Ø	380 POKE -16368,0	Clear keyboard strobe – always after reading keyboard.
CØ61	390 X=PEEK(16287)	Read PDL(Ø) push button switch. If X>127 then switch is "on".
CØ62	400 X=PEEK(-16286)	Read PDL(1) push button switch.
CØ63	410 X=PEEK(-16285	Read PDL(2) push button switch.
CØ58	420 POKE -16296,0	Clear Game I/O ANØ output
CØ59	430 POKE -16295,0	Set Game I/O ANØ output
CØ5A	440 POKE -16294,0	Clear Game I/O ANl output
CØ5B	450 POKE -16293,0	Set Game I/O AN1 output
CØ5C	460 POKE -16292,0	Clear Game I/O AN2 output
CØ5D	470 POKE -16291,0	Set Game I/O AN2 output
CØ5E	480 POKE -16290,0	Clear Game I/O AN3 output
CØ5F	490 POKE -16289,0	Set Game I/O AN3 output

```
*******
       APPLE II
    SYSTEM MONITOR
   COPYRIGHT 1977 BY
   APPLE COMPUTER, INC. *
   ALL RIGHTS RESERVED
     S. WOZNIAK
       A. BAUM
*******
      TITLE
                      "APPLE II SYSTEM MONITOR"
LOC0
       EPZ $00
EPZ $01
LOC1
WNDLFT EPZ $20
WNDWDTH EPZ
            $21
WNDTOP
        EPZ
            $22
WNDBTM
       EPZ
            $23
CH
        EPZ
            $24
CV
        EPZ
            $25
GBASL
        EPZ
            $26
GBASH
        EPZ
            $27
BASL
        EPZ
            $28
            $29
BASH
        EPZ
BAS2L
        EPZ $2A
BAS2H
        EPZ
            $2B
        EPZ
            $2C
LMNEM
        EPZ
             $2C
RTNL
        EPZ
            $2C
            $2D
V2
        EPZ
RMNEM
        EPZ
            $2D
RTNH
            $2D
MASK
        EPZ
             $2E
CHKSUM
       EPZ
            $2E
FORMAT
            $2E
        EPZ
LASTIN
        EPZ
            $2F
LENGTH
        EPZ
            $2F
SIGN
             $2F
COLOR
        EPZ
            $30
MODE
        EPZ
            $31
INVFLG
       EPZ
            $32
PROMPT
        EPZ
            $33
YSAV
        EPZ
             $34
YSAV1
        EPZ
            $35
CSWL
        EPZ
             $36
CSWH
        EPZ
            $37
KSWL
        EPZ
            $38
KSWH
        EPZ
             $39
PCL
            $3A
PCH
        EPZ
             $3B
XQT
        EPZ
             $3C
            $3C
A1L
       EPZ
A1H
        EPZ
            $3D
A2L
        EPZ
            $3E
A2H
        EPZ
             $3F
A3L
       EPZ
            $40
АЗН
       EPZ
            $41
A4L
        EPZ
            $42
A4H
       EPZ
            $43
A5L
        EPZ
            $44
A5H
       EPZ
```

```
XREG
                          EQU
                                 $46
                 YREG
                          EOU
                                 $47
                 STATUS
                          EQU
                                 $48
                 SPNT
                          EOU
                                 $49
                 RNDL
                          EQU
                                 $4E
                 RNDH
                          EOU
                                 $4F
                 ACL
                          EQU
                                 $50
                 ACH
                          EQU
                 XTNDL
                          EQU
                                 $52
                 XTNDH
                          EOU
                                 $53
                 AUXL
                          EQU
                                 $54
                 AUXH
                          EQU
                                 $55
                 PICK
                          EQU
                                 $95
                                 $0200
                 IN
                          EQU
                 USRADR
                          EOU
                                 $03F8
                 NMT
                                 $03FB
                          EOU
                 IROLOC
                          EOU
                                 $03FE
                 IOADR
                          EQU
                                 $C000
                 KBD
                          EQU
                                 $C000
                 KBDSTRB
                                 $C010
                          EOU
                 TAPEOUT
                          EOU
                                 $C020
                 SPKR
                          EQU
                                 $C030
                 TXTCLR
                          EQU
                                 $C050
                 TXTSET
                          EQU
                                 $C051
                 MIXCLR
                          EQU
                                 $C052
                 MIXSET
                          EOU
                                 $C053
                 LOWSCR
                          EOU
                                 $C054
                 HISCR
                          EQU
                                 $C055
                 LORES
                          EQU
                                 $C056
                 HIRES
                          EQU
                                 $C057
                 TAPEIN
                          EOU
                                 $C060
                 PADDL0
                          EOU
                                 $C064
                 PTRIG
                          EQU
                                 $C070
                 BASIC
                          EQU
                                 $E000
                 BASIC2
                          EQU
                                 $E003
                                           ROM START ADDRESS
                          ORG
                                 $F800
F800: 4A
                 PLOT
                          LSR
                                           Y-COORD/2
F801: 08
                          PHP
                                           SAVE LSB IN CARRY
F802: 20 47 F8
                          JSR
                                 GBASCALC
                                           CALC BASE ADR IN GBASL, H
F805: 28
                          PLP
                                           RESTORE LSB FROM CARRY
F806: A9 0F
                                 #$0F
                                           MASK $0F IF EVEN
                          LDA
F808: 90 02
                          BCC
                                 RTMASK
F80A: 69 E0
                          ADC
                                 #$E0
                                           MASK $F0 IF ODD
F80C: 85 2E
                 RTMASK
                          STA
                                 MASK
F80E: B1 26
                 PLOT1
                          LDA
                                 (GBASL), Y DATA
F810: 45 30
                          EOR
                                 COLOR
                                           EOR COLOR
F812: 25 2E
                          AND
                                 MASK
                                            AND MASK
F814: 51 26
                          EOR
                                 (GBASL),Y
                                              XOR DATA
                                               TO DATA
F816: 91 26
                                 (GBASL),Y
                          STA
F818: 60
                          RTS
F819: 20 00 F8 HLINE
                          JSR
                                 PLOT
                                           PLOT SQUARE
F81C: C4 2C
                 HLINE1
                          CPY
                                           DONE?
F81E: B0 11
                          BCS
                                 RTS1
                                            YES, RETURN
                                           NO, INCR INDEX (X-COORD)
F820: C8
                          INY
F821: 20 0E F8
                          JSR
                                 PLOT1
                                           PLOT NEXT SQUARE
F824: 90 F6
                          BCC
                                 HLINE1
                                           ALWAYS TAKEN
                                           NEXT Y-COORD
F826: 69 01
                 VLINEZ
                          ADC
                                 #$01
                                           SAVE ON STACK
F828: 48
                 VLINE
                          PHA
                                           PLOT SQUARE
F829: 20 00 F8
                          JSR
                                 PLOT
F82C: 68
                          PLA
F82D: C5 2D
                          CMP
                                 7/2
                                           DONE?
F82F: 90 F5
                          BCC
                                 VLINEZ
                                            NO, LOOP
F831: 60
                 RTS1
                          RTS
F832: A0 2F
                 CLRSCR
                          LDY
                                 #$2F
                                           MAX Y, FULL SCRN CLR
F834: D0 02
                          BNE
                                 CLRSC2
                                           ALWAYS TAKEN
F836: A0 27
                 CLRTOP
                          LDY
                                 #$27
                                           MAX Y, TOP SCREEN CLR
F838: 84 2D
                 CLRSC2
                          STY
                                 V2
                                           STORE AS BOTTOM COORD
                           FOR
                                VLINE CALLS
F83A: A0 27
                          LDY
                                 #$27
                                           RIGHTMOST X-COORD (COLUMN)
F83C: A9 00
                                           TOP COORD FOR VLINE CALLS
                 CLRSC3
                          LDA
                                 #$00
                                           CLEAR COLOR (BLACK)
F83E: 85 30
                          STA
                                 COLOR
F840: 20 28 F8
                          JSR
                                 VLINE
                                           DRAW VLINE
F843: 88
                          DEY
                                           NEXT LEFTMOST X-COORD
F844: 10 F6
                          BPL
                                 CLRSC3
                                           LOOP UNTIL DONE
F846: 60
                          RTS
                 GBASCALC PHA
                                           FOR INPUT 000DEFGH
F847: 48
F848: 4A
                          LSR
F849: 29 03
                          AND
                                 #$03
F84B: 09 04
                          ORA
                                 #$04
                                              GENERATE GBASH=000001FG
F84D: 85 27
                          STA
                                 GBASH
F84F: 68
                                           AND GBASL=HDEDE000
                          PLA
F850: 29 18
                          AND
                                 #$18
F852: 90 02
                          BCC
                                 GBCALC
F854: 69 7F
                          ADC
                                 #$7F
F856: 85 26
                 GBCALC
                          STA
                                 GBASL
```

ACC

EOU

\$45

F858:					ASL		
F859:	0A					A	
F85A:	05	26			ORA	GBASL	
F85C:	85	26			STA	GBASL	
F85E:	60				RTS		
F85F:	A5	30		NXTCOL	LDA	COLOR	INCREMENT COLOR BY 3
F861:					CLC		
F862:	69	03			ADC	#403	
F864:				SETCOL			SETS COLOR=17*A MOD 16
F866:						COLOR	DEID COHOR-I/ A MOD IO
		30					DOMIL HALD DAMES OF GOLOD FOLIAL
F868:					ASL		BOTH HALF BYTES OF COLOR EQUAL
F869:					ASL		
F86A:					ASL		
F86B:						A	
F86C:					ORA	COLOR	
F86E:	85	30			STA	COLOR	
F870:	60				RTS		
F871:	4A			SCRN	LSR	A	READ SCREEN Y-COORD/2
F872:	08				PHP		SAVE LSB (CARRY)
F873:	20	47	F8		JSR	GBASCALC	CALC BASE ADDRESS
F876:					LDA	(GBASL),Y	GET BYTE
F878:					PLP		RESTORE LSB FROM CARRY
							IF EVEN, USE LO H
F87B:						A	IF EVEN, OSE DO H
F87C:					LSR		
F87D:					LSR		SHIFT HIGH HALF BYTE DOWN
F87E:					LSR		
		0F				#\$0F	MASK 4-BITS
F881:					RTS		
F882:	Α6	3A		INSDS1	LDX	PCL	PRINT PCL,H
F884:					LDY		
F886:	20	96	FD		JSR	PRYX2	
F889:	20	48	F9		JSR	PRBLNK	FOLLOWED BY A BLANK
F88C:							GET OP CODE
F88E:				INSDS2			
F88F:						A	EVEN/ODD TEST
F890:						IEVEN	
F892:					ROR		BIT 1 TEST
F893:					BCS		XXXXXX11 INVALID OP
F895:					CMP		MMMMII INVINDID OI
F897:					DEO		OPCODE \$89 INVALID
F899:							MASK BITS
F89B:				IEVEN			
F89C:					TAX	A	LSB INTO CARRY FOR L/R TEST
			ПО.			mama w	CEE FORME INDEX DIFFE
F89D:					LDA	FMII,A	GET FORMAT INDEX BYTE R/L H-BYTE ON CARRY
E 0 7 0		79	F.8		JSR	SCRNZ	R/L H-BYTE ON CARRY
F8A0:							
F8A3:	D0	04			BNE	GETFMT	
F8A3: F8A5:	D0 A0	04 80		ERR	BNE LDY	#\$80	SUBSTITUTE \$80 FOR INVALID OPS
F8A3: F8A5: F8A7:	D0 A0 A9	04 80 00		ERR	BNE LDY LDA	#\$80	SUBSTITUTE \$80 FOR INVALID OPS SET PRINT FORMAT INDEX TO 0
F8A3: F8A5: F8A7: F8A9:	D0 A0 A9 AA	04 80 00		ERR GETFMT	BNE LDY LDA TAX	#\$80 #\$00	SET PRINT FORMAT INDEX TO 0
F8A3: F8A5: F8A7:	D0 A0 A9 AA	04 80 00		ERR GETFMT	BNE LDY LDA TAX LDA	#\$80 #\$00 FMT2,X	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE
F8A3: F8A5: F8A7: F8A9:	D0 A0 A9 AA BD	04 80 00 A6		ERR GETFMT	BNE LDY LDA TAX LDA	#\$80 #\$00 FMT2,X	SET PRINT FORMAT INDEX TO 0
F8A3: F8A5: F8A7: F8A9: F8AA:	D0 A0 A9 AA BD 85	04 80 00 A6 2E		ERR GETFMT	BNE LDY LDA TAX LDA STA	#\$80 #\$00 FMT2,X FORMAT	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD:	D0 A0 A9 AA BD 85	04 80 00 A6 2E		ERR GETFMT	BNE LDY LDA TAX LDA STA AND	#\$80 #\$00 FMT2,X FORMAT #\$03	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD:	D0 A0 A9 AA BD 85 29	04 80 00 A6 2E 03		ERR GETFMT	BNE LDY LDA TAX LDA STA AND	#\$80 #\$00 FMT2,X FORMAT #\$03	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8AF:	D0 A0 A9 AA BD 85 29	04 80 00 A6 2E 03		ERR GETFMT	BNE LDY LDA TAX LDA STA AND	#\$80 #\$00 FMT2,X FORMAT #\$03 (P=1 BYTE,	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8AF:	D0 A0 A9 AA BD 85 29 85 98	04 80 00 A6 2E 03	F9	ERR GETFMT	BNE LDY LDA TAX LDA STA AND	#\$80 #\$00 FMT2,X FORMAT #\$03 (P=1 BYTE,	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1-2 BYTE, 2-3 BYTE)
F8A3: F8A5: F8A7: F8A9: F8AA: F8AF: F8B1: F8B3:	D0 A0 A9 AA BD 85 29 85 98	04 80 00 A6 2E 03 2F	F9	ERR GETFMT	BNE LDY LDA TAX LDA STA AND	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,:	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE
F8A3: F8A5: F8A7: F8A9: F8AD: F8AF: F8B1: F8B3: F8B4: F8B6:	D0 A0 A9 AA BD 85 29 85 98 29 AA	04 80 00 A6 2E 03 2F	F9	ERR GETFMT	BNE LDY LDA TAX LDA STA AND STA TYA AND TAX	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,:	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1-2 BYTE, 2-3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8B1: F8B3: F8B4:	D0 A0 A9 AA BD 85 29 85 98 29 AA 98	04 80 00 A6 2E 03 2F 8F	F9	ERR GETFMT	BNE LDY LDA TAX LDA STA AND STA TYA AND TAX	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8BF: F8B6: F8B6: F8B6: F8B8:	D0 A0 A9 AA BD 85 29 85 98 29 AA 98 A0	04 80 00 A6 2E 03 2F 8F	F9	ERR GETFMT	BNE LDY LDA TAX LDA STA AND STA AND TYA AND TAX TYA LDY	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1-2 BYTE, 2-3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8B1: F8B3: F8B4: F8B6: F8B8: F8B8: F8B8:	D0 A0 A9 AA BD 85 29 85 98 29 AA 98 AO E0	04 80 00 A6 2E 03 2F 8F	F9	ERR GETFMT	BNE LDY LDA TAX LDA STA AND STA AND TYA AND TAX TYA LDY CPX	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1-2 BYTE, 2-3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8B1: F8B4: F8B4: F8B6: F8B8: F8B8: F8B8: F8B8:	D0 A0 A9 AA BD 85 29 85 98 29 AA 98 AO E0 F0	04 80 00 A6 2E 03 2F 8F	F9	ERR	BNE LDY LDA TAX LDA STA AND STA AND TAX TYA AND TAX LDY CPX BEQ	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1-2 BYTE, 2-3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT
F8A3: F8A5: F8A7: F8A9: F8AB: F8AF: F8B1: F8B4: F8B6: F8B7: F8B8: F8B8: F8BB: F8BB:	D0 A0 A9 AA BD 85 29 85 98 29 AA 98 AO E0 F0 4A	04 80 00 A6 2E 03 2F 8F	F9	ERR	BNE LDY LDA TAX LDA STA AND STA AND TAX TYA AND TAX TYA LDY CPX BEQ LSR	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8BAF: F8B1: F8B4: F8B6: F8B7: F8B8: F8BA: F8BA: F8BC: F8BC: F8BC:	D0 A0 A9 AA BD 85 29 AA 98 A0 E0 F0 4A 90	04 80 00 A6 2E 03 2F 8F 03 8A 0B	F9	ERR	BNE LDY LDA TAX LDA STA AND STA AND TAX TYA AND TAX TYA BEQ LSR BCC	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$8A MNNDX3 A	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1-2 BYTE, 2-3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT
F8A3: F8A5: F8A7: F8A9: F8AB: F8AF: F8B1: F8B4: F8B4: F8B6: F8B8: F8B8: F8B8: F8B8: F8BC: F8BE: F8BE:	D0 A0 A9 AA BD 85 29 AA 98 A0 E0 F0 4A 90 4A	04 80 00 A6 2E 03 2F 8F 03 8A 0B	F9	ERR GETFMT MNNDX1	BNE LDY LDA TAX LDA STA AND STA TYA AND TAX LDY CPX BEQ LSR BCC LSR	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8B1: F8B3: F8B4: F8B6: F8B6: F8B8: F8B8: F8B8: F8B8: F8BE: F8BE: F8BE:	D0 A9 AA BD 85 29 85 98 A0 E0 F0 4A 90 4A	04 80 00 A6 2E 03 2F 8F 03 8A 0B	F9	ERR GETFMT MNNDX1	BNE LDY LDA TAX LDA STA AND STA TYA AND TAX TYA LDY CPX BEQ LSR LSR	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX
F8A3: F8A5: F8A7: F8A9: F8AA: F8AF: F8B1: F8B3: F8B4: F8B6: F8B7: F8B8: F8B8: F8B8: F8BE: F8BE: F8BE: F8BE: F8BE:	D0 A9 AA BD 85 29 85 98 AA 98 AO F0 4A 90 4A 09	04 80 00 A6 2E 03 2F 8F 03 8A 0B	F9	ERR GETFMT MNNDX1	BNE LDY LDA STA AND STA AND TAX AND TAX CPX BEQ LSR BCC LSR LSR ORA	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX
F8A3: F8A5: F8A7: F8A9: F8AD: F8AF: F8B1: F8B1: F8B6: F8B6: F8B6: F8B6: F8B8: F8BC: F8BE:	D0 A9 AA BD 85 29 AA 98 AO E0 F0 4A 4A 09 88	04 80 00 A6 2E 03 2F 8F 03 8A 0B	F9	ERR GETFMT MNNDX1	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA BEQ LSR BCC LSR BCC LSR ORA DEY	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A #\$20	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX
F8A3: F8A5: F8A7: F8A9: F8AA: F8AF: F8B1: F8B3: F8B6: F8B7: F8B8: F8B8: F8BC: F8BC: F8BC: F8BC: F8C5: F8C5: F8C6:	D0 A9 AA BD 85 29 AA 98 AO E0 F0 4A 4A 09 88 D0	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08	F9	ERR GETFMT MNNDX1	BNE LDY LDA TAX LDA STA AND STA AND TAX TYA AND CPX BEQ LSR BCC LSR LSR ORA DEY BNE	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY10->00111XXX 4) XXXYYY10->00110XXX 4) XXXYYY10->00110XXX
F8A3: F8A5: F8A7: F8A9: F8AD: F8AF: F8B1: F8B4: F8B6: F8B7: F8B8: F8B8: F8BC: F8BE: F8BE: F8BE: F8C3: F8C3: F8C6: F8C6: F8C6: F8C6:	D0 A0 A9 AA BD 85 29 85 98 A0 E0 F0 4A 4A 09 88 D0 C8	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08	F9	ERR GETFMT MNNDX1 MNNDX2	BNE LDY LDA TAX LDA STA AND STA AND TAX TYA AND TAX TYA LDY CPX BEQ LSR BCC LSR CRA DEY BNE LNY	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A #\$20	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX
F8A3: F8A5: F8A7: F8A9: F8AB: F8AF: F8B1: F8B4: F8B6: F8B6: F8B7: F8B8: F8B8: F8BC: F8BE: F8BE: F8BE: F8C1: F8C2: F8C3: F8C3: F8C6:	D0 A0 A9 AA BD 85 29 85 98 A0 E0 F0 4A 4A 09 88 D0 C8 88	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08	F9	ERR GETFMT MNNDX1	BNE LDY LDA TAX LDA STA AND STA TYA AND TAX LDY CPX BEQ LSR BCC LSR LSR CRA DEY BNE INY DEY	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A #\$20	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY10->00111XXX 4) XXXYYY10->00110XXX 4) XXXYYY10->00110XXX
F8A3: F8A5: F8A7: F8A9: F8AA1: F8AD: F8B1: F8B3: F8B4: F8B6: F8B7: F8B8: F8BC: F8BE: F8BC: F8BE: F8C1: F8C2: F8C3: F8C3: F8C3: F8C5: F8C8:	D0 A0 A9 AA BD 85 29 AA 98 A0 E0 F0 4A 4A 09 88 D0 C8 88 D0	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08	F9	ERR GETFMT MNNDX1 MNNDX2	BNE LDY LDA STA AND STA AND TAX TYA AND CPX BEQ LSR BCC LSR LSR ORA DEY BNE	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A #\$20	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY10->00111XXX 4) XXXYYY10->00110XXX 4) XXXYYY10->00110XXX
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8AF: F8B1: F8B4: F8B6: F8B7: F8B8: F8BC: F8BE: F8BC: F8BE: F8C1: F8C3: F8C3: F8C3: F8C4: F8C4: F8C5: F8C6: F8C6: F8C7: F8C8: F8C8: F8C8: F8C8: F8C8: F8C8: F8C8: F8C8: F8C8: F8C8: F8C8:	D0 A0 A9 AA BD 85 29 AA 98 A0 E0 F0 4A 4A 09 88 BD C8 88 D0 60	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA	F9	ERR GETFMT MNNDX1 MNNDX2	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND TAX TYA BEQ LSR BCC RA DEY BNE RTS	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A A #\$20 MNNDX3	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->0000XXXXX
F8A3: F8A5: F8A7: F8A9: F8AA: F8AD: F8B1: F8B1: F8B4: F8B6: F8B7: F8B8: F8BC: F8BE: F8C1: F8C2: F8C3: F8C3: F8C4: F8C5: F8C6: F8C8: F8C9: F8CA: F8CC: F8CC:	D0 A0 A9 AA BD 85 29 85 89 A0 E0 F0 4A 4A 09 88 D0 60 FF	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA	F9	ERR GETFMT MNNDX1 MNNDX2 MNNDX3	BNE LDY LDA STA AND STA AND TAX TYA AND TAX TYA BCC CPX BEQ LSR BCC LSR BCC LSR DEY BNE INY DEY BNE INY DEY BNE RTS DFB	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$8A MNNDX3 A MNNDX3 A A #\$20 MNNDX2	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY10->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->00100XXX
F8A3: F8A5: F8A7: F8A9: F8AA1: F8AB1: F8B4: F8B6: F8B6: F8B7: F8BC1: F8BC1: F8C2: F8C3: F8C6: F8C6: F8C8: F8C9: F8C8: F8C9: F8C9: F8C9:	D0 A0 A9 AA BD 85 29 AA 98 A0 E0 F0 4A 4A 09 88 BD 60 FF 20	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA FF 82	F9	ERR GETFMT MNNDX1 MNNDX2	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND CPX BEQ LSR BCC LSR BCC LSR DEY BNE INY DEY BNE INY DEY BNE INY DEY BNE JSR	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$8A MNNDX3 A MNNDX3 A A #\$20 MNNDX2	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->000XXXXX
F8A3: F8A5: F8A7: F8A9: F8AB1: F8B4: F8B6: F8B6: F8B7: F8B8: F8B8: F8B6: F8B7: F8B8: F8B6: F8B6: F8B7: F8C1: F8C2: F8C3: F8C3: F8C5: F8C6: F8C6: F8C7: F8C8: F8C8: F8C8: F8C9: F8C8: F8C9: F8C8: F8C9: F8C8: F8C9: F8C8:	D0 A0 A9 AA BD 85 29 AA 98 A0 E0 F0 4A 4A 09 88 D0 C8 88 D0 C8 88 D0 44 44 44 45 46 46 46 46 46 46 46 46 46 46 46 46 46	04 80 00 A6 2E 03 2F 8F 03 8A 0B 20 FA F2 FF 82	FF FF F8	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP	BNE LDY LDA TAX LDA STA AND STA AND TAX TYA AND TAX TYA LDY CPX BEQ LSR BCC LSR BCC LSR LSR USR BCC LSR BCC LS	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A #\$20 MNNDX2 MNNDX2	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY10->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->00100XXX
F8A3: F8A5: F8A7: F8A9: F8AD: F8AF: F8B1: F8B4: F8B6: F8B7: F8B8: F8B8: F8BC: F8BC: F8C1: F8C2: F8C3: F8C6: F8C6: F8C6: F8C6: F8C6: F8C8: F8C8: F8C9: F8C8: F8C9: F8C8: F8C9: F8C9: F8C9: F8C9: F8C8: F8C9: F8C8: F8C9: F8C8: F8C9: F8C8: F8C9: F8C8: F8C9: F8C8:	D0 A0 A9 AA BD 85 98 29 AA 90 4A 40 90 4A 40 60 60 F0 60 60 F0 48 BD 60 60 60 60 60 60 60 60 60 60 60 60 60	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA F2 FF 82 3A	FF FF F8	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND CPX BEQ LSR BCQ	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE, LENGTH #\$8F #\$03 #\$8A MNNDX3 A A #\$20 MNNDX3 I SFF,\$FF,\$F	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->000XXXXX
F8A3: F8A5: F8A7: F8A9: F8AA1: F8AF: F8B1: F8B1: F8B6: F8B6: F8B6: F8B6: F8B6: F8B6: F8B6: F8C1: F8C2: F8C3: F8C3: F8C4: F8C9:	D0 A0 A9 AA BD 85 98 29 AA 90 4A 4A 09 60 FF 48 BD 60 FF 48 BD 60 FF 48 BD 60 FF 48 BD 60 60 60 60 60 60 60 60 60 60 60 60 60	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA F2 FF 82 3A DA	FF FF FD	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP	BNE LDY LDA STA AND STA AND TAX TYA AND TAX LDY CPX BEQ LSR BCQ BCQ LS	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE, LENGTH #\$8F #\$03 #\$8A MNNDX3 A MNNDX3 A #\$20 MNNDX2 MNNDX1 \$FF,\$FF,\$F: INSDS1 (PCL),Y PRBYTE	INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->0010XXX 5) XXXXXX000->000XXXXX F GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX
F8A3: F8A5: F8A7: F8A9: F8AA1: F8AB1: F8B1: F8B4: F8B6: F8B7: F8B6: F8B6: F8BC1: F8C1: F8C2: F8C3: F8C4: F8C5: F8C6: F8C9: F8C3: F8C4: F8C7: F8C8: F8C8: F8C9: F8C8: F8C9: F8C8: F8C9: F8C	D0 A0 A9 AA BD 529 85 98 A0 E0 F0 4A 09 88 D0 60 FF 20 48 BD 60 FF 20 48 A0 60 FF 40 60 60 60 60 60 60 60 60 60 60 60 60 60	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA F2 FF 82 3A 01	FF FF F8	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA BEQ LSR BCC LSR BC	#\$80 #\$00 FMT2,X FORMAT #\$03 (P=1 BYTE,: LENGTH #\$8F #\$8A MNNDX3 A A MNNDX3 A A MNNDX3 A (P\$10 MNNDX1 SFF,\$FF,\$F: INSDS1 (PCL),Y PRBYTE #\$01	SET PRINT FORMAT INDEX TO 0 INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->000XXXXX
F8A3: F8A5: F8A7: F8A9: F8AA1: F8AB1: F8B3: F8B4: F8B6: F8B7: F8B8: F8BC1: F8BC1: F8C2: F8C3: F8C4: F8C5: F8C6: F8C9: F8	D0 A0 A9 AA BD 529 AA 29 AA 29 AA 60 F0 4A 4A 09 88 BD 60 FF 20 48 BD 60 FF 20 48 BD 60 60 60 60 60 60 60 60 60 60 60 60 60	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA FF 82 3AA 01 4A	FF FF F8	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP PRNTBL	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND CPX BEQ LSR BCC LSR BC	#\$80 #\$00 FMT2,X FORMAT #\$03 (P=1 BYTE, LENGTH #\$8F #\$8A MNNDX3 A MNNDX3 A A #\$20 MNNDX2 MNNDX1 \$FF,\$FF,\$F: INSDS1 (PCL),Y PRBYTE #\$01 PRBL2	INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->0010XXX F GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX PRINT 2 BLANKS
F8A3: F8A5: F8A7: F8A9: F8AA1: F8AB1: F8B4: F8B6: F8B6: F8B6: F8B6: F8B6: F8B6: F8C1: F8C2: F8C3: F8C4: F8C6: F8C8: F8C9	D0 A0 A9 AA B5 98 85 98 A0 E0 F0 4A 4A 09 8B D0 C8 8B D0 6F F20 48 B1 20 C4	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA F2 FF 82 3A D1 4A 2F	FF FF F8	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP PRNTBL	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND CPX BEQ LSR BCC LSR BC	#\$80 #\$00 FMT2,X FORMAT #\$03 (P=1 BYTE, LENGTH #\$8F #\$8A MNNDX3 A MNNDX3 A A #\$20 MNNDX2 MNNDX1 \$FF,\$FF,\$F: INSDS1 (PCL),Y PRBYTE #\$01 PRBL2	INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->0010XXX 4) XXXXYY100->0010XXX F GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX PRINT 2 BLANKS PRINT INST (1-3 BYTES)
F8A3: F8A5: F8A7: F8AA9: F8AA1: F8AB1: F8BB1: F8BB1: F8BB4: F8B6: F8BC: F8BE: F8BC: F8BE: F8C1: F8C2: F8C3: F8C4: F8C4: F8C9:	D0 A0 A9 ABD 85 29 A8 A00 F0 4A	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA F2 3A DA 01 4A 2F	FF FF F8 FD F9	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP PRNTBL	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND CPX BEQ LSR BCQ LSR CPY LSR LDY LDY LSR LDY LSR LDY LSR LDY LSR LSR COPA LSR LDY LSR LOSR LOSR LOSR LOSR LOSR LOSR LOSR	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE,: LENGTH #\$8F #\$03 #\$8A MNNDX3 A A #\$20 MNNDX3 I SFF,\$FF,\$F! INSDS1 (PCL),Y PRBYTE #\$01 PRBL2 LENGTH	INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->0010XXX F GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX PRINT 2 BLANKS
F8A3: F8A5: F8A7: F8A9: F8AA1: F8AB1: F8B4: F8B6: F8B6: F8B6: F8B6: F8B6: F8B6: F8C1: F8C2: F8C3: F8C4: F8C6: F8C8: F8C9	D0 A0 A9 ABD 85 29 A8 A00 F0 4A	04 80 00 A6 2E 03 2F 8F 03 8A 0B 08 20 FA F2 3A DA 01 4A 2F	FF FF F8 FD F9	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP PRNTBL	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND CPX BEQ LSR BCQ LSR CPY LSR LDY LDY LSR LDY LSR LDY LSR LDY LSR LSR COPA LSR LDY LSR LOSR LOSR LOSR LOSR LOSR LOSR LOSR	#\$80 #\$00 FMT2,X FORMAT #\$03 (P=1 BYTE, LENGTH #\$8F #\$8A MNNDX3 A MNNDX3 A A #\$20 MNNDX2 MNNDX1 \$FF,\$FF,\$F: INSDS1 (PCL),Y PRBYTE #\$01 PRBL2	INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->0010XXX 4) XXXXYY100->0010XXX F GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX PRINT 2 BLANKS PRINT INST (1-3 BYTES)
F8A3: F8A5: F8A7: F8AA9: F8AA1: F8AB1: F8BB1: F8BB1: F8BB4: F8B6: F8BC: F8BE: F8BC: F8BE: F8C1: F8C2: F8C3: F8C4: F8C4: F8C9:	D0 A0 A9 ABD B5 29 A8 A0 F0 4A A0 BBD 60 FF 20 A2 20 CC8 90	04 80 00 A6 2E 03 2F 8F 03 8A 0B 20 FA F2 FF 82 3A DA 01 4A 2F F1	FF FF FD F9	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP PRNTBL	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND TAX CPX BEQ LSR BCQ LSR CPY INY BCC	#\$80 #\$00 FMT2,X FORMAT #\$03 P=1 BYTE, LENGTH #\$8F #\$03 #\$8A MNNDX3 A A #\$20 MNNDX3 I SFF,\$FF,\$F INSDS1 (PCL),Y PRBYTE #\$01 PRBL2 LENGTH PRNTOP	INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->0010XXX 4) XXXXYY100->0010XXX F GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX PRINT 2 BLANKS PRINT INST (1-3 BYTES)
F8A3: F8A5: F8A7: F8AA9: F8AA1: F8AB1: F8BA1: F8BB1: F8BB4: F8BB4: F8BB2: F8BE1: F8BE3: F8BE3: F8C1: F8C3: F8C4: F8C4: F8C9: F	D0 A0 A9 ABD 85 29 85 829 AA 98 A0 C8 88 D0 C8 88 D0 C8 88 D0 C8 A2 C4 C9 A2	04 80 00 A6 2E 03 2F 8F 03 8A 08 20 FA FF 82 3A DA 01 4A 2F F1 03	FF FF FD F9	ERR GETFMT MNNDX1 MNNDX2 MNNDX3 INSTDSP PRNTOP PRNTBL	BNE LDY LDA LDA STA AND STA AND TAX TYA AND TAX TYA AND TAX CPX BEQ LSR BCQ LSR CPY INY BCC	#\$80 #\$00 FMT2,X FORMAT #\$03 (P=1 BYTE,: LENGTH #\$8F #\$8A MNNDX3 A A MNNDX3 A A MNNDX3 (PCL),Y PRBYTE #\$01 PRBL2 LENGTH PRNTOP #\$03	INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING MASK FOR 2-BIT LENGTH 1=2 BYTE, 2=3 BYTE) OPCODE MASK FOR 1XXX1010 TEST SAVE IT OPCODE TO A AGAIN FORM INDEX INTO MNEMONIC TABLE 1) 1XXX1010->00101XXX 2) XXXYYY01->00111XXX 3) XXXYYY10->00110XXX 4) XXXYYY10->00100XXX 5) XXXXXX000->000XXXXX F GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX PRINT 2 BLANKS PRINT INST (1-3 BYTES) IN A 12 CHR FIELD

F8E7:	90 68	F2			BCC PLA	PRNTBL	RECOVER MNEMONIC INDEX FETCH 3-CHAR MNEMONIC (PACKED IN 2-BYTES) SHIFT 5 BITS OF CHARACTER INTO A (CLEARS CARRY) ADD "?" OFFSET OUTPUT A CHAR OF MNEM OUTPUT 3 BLANKS CNT FOR 6 FORMAT BITS IF X=3 THEN ADDR.
F8EA:	Α8				TAY		
F8EB:	В9	C0	F9		LDA	MNEML,Y	
F8EE:	85 Ba	2C	EΝ		STA	LMNEM	FETCH 3-CHAR MNEMONIC
F8F3:	85	2D	rA		STA	RMNEM , I	(FACKED IN 2-BILES)
F8F5:	Α9	00		PRMN1	LDA	#\$00	
F8F7:	Α0	05			LDY	#\$05	
F8FB:	26	2D		PRMN2	ROL.	T.MNEM	CHARACTER INTO A
F8FD:	2A	20			ROL	EL IIVEL I	(CLEARS CARRY)
F8FE:	88				DEY		
F8FF:	D0	F8			BNE	PRMN2	
F901:	20	ED BL	ED		TSR	#\$BF	ADD "?" OFFSET OUTPUT A CHAR OF MNEM
F906:	CA	טני	ГD		DEX	C001	OUTFUL A CHAR OF PINEP
F907:	D0	EC			BNE	PRMN1	
F909:	20	48	F9		JSR	PRBLNK	OUTPUT 3 BLANKS
F90C:	A4 A2	2F			TDX	LENGTH #¢06	CNT FOR 6 FORMAT BITS
F910:	E0	03		PRADR1	CPX	#\$03	CNI TOR O TORMET BITS
F912:	F0	1C			BEQ	PRADR5	IF X=3 THEN ADDR.
F914:	06	2E		PRADR2	ASL	FORMAT	
F916:	90	B3	FQ		PCC.	PRADR3	
F91B:	20	ED	FD		JSR	COUT	
F91E:	BD	В9	F9		LDA	CHAR2-1,X	
F921:	F0	03			BEQ	FRADRS FORMAT PRADR3 CHAR1-1,X COUT CHAR2-1,X PRADR3 COUT	
F923:	20	ED	FD	2 מת גמת	JSR	COUT	
F927:	D0	E7		PRADR3	BNE	PRADR1	
F929:	60				RTS		
F92A:	88			PRADR4	DEY		
F92B:	30	E7	מש	PRADR4 PRADR5	BMI	PRADR2	
F930:	A5	2E	гD	PRADR5	LDA	FORMAT	
F932.	('9	H: X			CMP	#\$E8	HANDLE REL ADR MODE SPECIAL (PRINT TARGET, NOT OFFSET)
F934:	B1	3A			LDA	(PCL),Y	SPECIAL (PRINT TARGET,
F936:	20	F2	FQ	DEI.ADD	BCC	PRADR4	NOT OFFSET)
F93B:	AA	50	ГЭ	KELIADK	TAX	PCADJ3	PCL, PCH+OFFSET+1 TO A, Y
F93C:	E8				INX		
F93D:	D0	01			BNE	PRNTYX	+1 TO Y,X
F93F:	C8						•
E010.	QΩ			DDMTVY	INY		
F940:	98	עם	ĒD	PRNTYX	TYA	DDRVTF	OUTDUT TARGET AND
F940:	98	עם	ĒD	PRNTYX	TYA	DDRVTF	OUTDUT TARGET AND
F940:	98	עם	ĒD	PRNTYX	TYA	DDRVTF	OUTDUT TARGET AND
F940:	98	עם	ĒD	PRNTYX	TYA	DDRVTF	OUTDUT TARGET AND
F940:	98	עם	ĒD	PRNTYX	TYA	DDRVTF	OUTDUT TARGET AND
F940: F941: F944: F945: F948: F94A: F94C: F94F:	98 20 8A 4C A2 A9 20 CA	DA DA 03 A0 ED	ĒD	PRNTYX	TYA JSR TXA JMP LDX LDA JSR DEX	PRBYTE PRBYTE #\$03 #\$A0 COUT	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK
F940: F941: F944: F945: F948: F94C: F94C: F950:	98 20 8A 4C A2 A9 20 CA D0	DA DA 03 A0 ED	ĒD	PRNTYX	TYA JSR TXA JMP LDX LDA JSR DEX BNE	DDRVTF	OUTPUT TARGET ADR
F940: F941: F944: F945: F948: F94A: F94C: F94F: F950: F952:	98 20 8A 4C A2 A9 20 CA D0 60	DA DA 03 A0 ED F8	ĒD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS	PRBYTE PRBYTE #\$03 #\$A0 COUT	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0
F940: F941: F944: F945: F948: F94C: F94C: F950:	98 20 8A 4C A2 A9 20 CA D0 60 38	DA DA 03 A0 ED F8	ĒD	PRNTYX	TYA JSR TXA JMP LDX LDA JSR DEX BNE	PRBYTE PRBYTE #\$03 #\$A0 COUT	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK
F940:: F941: F944:: F945: F948: F94A:: F94F:: F950:: F952: F953:: F956:: F956::	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4	DA DA 03 A0 ED F8 2F 3B	ĒD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE
F940: F941: F944: F948: F948: F94A: F94C: F950: F952: F953: F954: F956: F958:	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA	DA DA 03 A0 ED F8 2F 3B	ĒD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ PCADJ2	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX	PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN
F940:: F941: F944:: F945: F948: F94A:: F94F:: F950:: F952: F953:: F956:: F956::	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA 10	DA 03 A0 ED F8 2F 3B 01	ĒD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ PCADJ2	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE
F940: F941: F944: F945: F948: F94C: F950: F950: F953: F954: F956: F958: F958: F958: F958: F958:	98 20 8A 4C A2 A9 20 CA D0 38 A5 A4 AA 10 88 65	DA 03 A0 ED F8 2F 3B 01 3A	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ PCADJ2	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX BPL	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH)
F940: F944: F944: F945: F948: F94C: F950: F950: F953: F956: F958: F959: F959: F958: F958: F959: F958:	98 20 8A 4C A2 20 CA D0 60 38 A5 A4 AA 10 88 65 90	DA 03 A0 ED F8 2F 3B 01 3A	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ PCADJ2 PCADJ3	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC	PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A
F940: F941: F944: F945: F948: F948: F946: F950: F953: F956: F958: F958: F958: F958: F958: F958: F958: F958:	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA 65 90 C8	DA 03 A0 ED F8 2F 3B 01 3A	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4	TYA JSR TXA JMP LDX LDA JSR DEX BNE SEC LDA LDY TAX BPL ADC BCC INY	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH
F940: F944: F944: F945: F948: F94C: F950: F950: F953: F956: F958: F959: F959: F958: F958: F959: F958:	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA 65 90 C8	DA 03 A0 ED F8 2F 3B 01 3A	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ PCADJ2 PCADJ3	TYA JSR TXA JMP LDX LDA JSR DEX BNE SEC LDA LDY TAX BPL DEY ADC BCC INY RTS	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A
F940: F941: F944: F945: F948: F948: F946: F950: F953: F956: F958: F958: F958: F958: F958: F958: F958: F958:	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA 65 90 C8	DA 03 A0 ED F8 2F 3B 01 3A	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * *	TYA JSR TXA JMP LDX LDA JSR DEX BNE SEC LDA LDY TAX BPL DEY ADC BCC INY RTS	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES:	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE
F940: F941: F944: F945: F948: F948: F946: F950: F953: F956: F958: F958: F958: F958: F958: F958: F958: F958:	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA 65 90 C8	DA 03 A0 ED F8 2F 3B 01 3A	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * *	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC LINY RTS FMT1	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE
F940: F941: F944: F945: F948: F948: F946: F950: F953: F956: F958: F958: F958: F958: F958: F958: F958: F958:	98 20 8A 4C A2 A9 20 CA D0 60 88 65 90 C8 60	DA DA 03 A0 ED F8 2F 3B 01 3A 01	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * *	TYA JSR TXA JMP LDX LDA JSR BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC INT FMT1 IF Y:	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE
F940: F941: F944: F945: F948: F94C: F950: F950: F953: F954: F958: F958: F958: F958: F958: F958: F958: F958: F958: F958: F958:	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA A0 C8 65 90 C8	DA DA 03 A0 ED F8 2F 3B 01 3A 01	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * *	TYA JSR TXA JMP LDX LDA JSR BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC INT FMT1 IF Y:	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX)
F940: F941: F941: F945: F948: F948: F94C: F952: F952: F953: F956: F958: F958: F958: F958: F956: F961:	98 20 8A 4C A2 A9 20 CA D0 60 38 A5 A4 AA 10 88 65 90 C8 60	DA DA 03 A0 ED F8 2F 3B 01 3A 01 20 0D 04	FD FD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC INY RTS FMT1 IF Y: IF Y: DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXY0 INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D
F940: F941: F944: F948: F948: F948: F946: F950: F952: F958: F958: F958: F956: F956: F956: F966:	98 20 8A 4C A2 A9 20 CA 60 38 A5 A4 AA 10 88 65 90 C8 60	DA DA 03 A0 ED F8 2F 3B 01 3A 01 20 0D 04 22	FD FD FD 54	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX BPL ADC BCC INY RTS FMT1 IF Y:	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXY0 INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D
F940: F941: F941: F945: F948: F948: F94C: F952: F952: F953: F956: F958: F958: F958: F958: F956: F961:	98 20 8A 4C A2 20 CA D0 60 38 A5 A4 AA 10 865 90 C8 60	DA DA 03 A0 ED F8 2F 3B 01 3A 01 20 0D 04 22 33	FD FD FD 54	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC INY RTS FMT1 IF Y: IF Y: DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22
F940: F941: F941: F944: F948: F948: F948: F952: F953: F956: F958: F958: F958: F961: F966: F967: F967: F967: F967: F967:	98 20 8A 4C A9 20 CA D0 60 38 A5 4A AA 10 88 65 90 C8 60	DA DA 03 A0 ED F8 2F 3B 01 3A 01 20 0D 0D 422 33 04 04 04	FD FD 54 90 OD	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA LDA LDA SEC LDA LDY TAX BPL DEY ADC BCC LINY RTS FMT1 IF Y: IF Y: DFB DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9 \$54,\$33,\$0	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN LEFT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04
F940: F941: F944: F948: F948: F948: F952: F952: F953: F958: F958: F958: F958: F960: F961: F967:	98 20 8A 4C A9 20 CA D0 60 88 85 60 C8 60 60 60 60 60 60 60 60 60 60 60 60 60	DA DA 03 A0 ED F8 2F 3B 01 3A 01 20 0D 04 22 3 04 04 33	FD FD FD 54 90 0D 20	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR LDX LDA JSR BNE RTS SEC LDA LDY TAX ADC BCC INY RTS FMT1 IF Y: DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN LEFT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04
F940: F941: F944: F948: F948: F948: F950: F952: F953: F956: F958: F958: F956: F956: F966: F967: F967: F967: F976:	98 20 8A 4C A9 20 CA D0 60 88 65 60 03 80 03 54 80 00 54 0D	DA DA 03 A0 ED F8 2F 3B 01 3A 01 20 0D 04 22 33 04 33 80	FD FD FD 54 90 0D 20	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA JSR BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC INY ATTEM THE YEAR DFB DFB DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9 \$54,\$33,\$00 \$90,\$04,\$2	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04 0,\$54,\$33
F940: F941: F944: F948: F948: F948: F952: F952: F953: F958: F958: F958: F958: F960: F961: F967:	98 20 84 A2 A9 20 CA D00 38 A5 A4 AA A1 00 03 60 03 03 60 03 03 03 03 03 03 03 03 03 03 03 03 03	DA DA 03 A0 ED F8 2F 3B 01 3A 01 20 0D 04 22 233 04 04 33 80 04	FD FD FD 54 90 0D 20 04	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA LDA LDA SEC LDA LDY TAX BPL DEY ADC BCC LINY RTS FMT1 IF Y: IF Y: DFB DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9 \$54,\$33,\$0	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04 0,\$54,\$33
F940: F941: F941: F944: F948: F948: F948: F952: F953: F956: F958: F958: F958: F961: F966: F967: F967: F967: F977: F978: F978: F978:	98 20 4C A2 A9 20 CD 60 38 A5 A4 A10 88 65 90 80 03 40 00 54 00 00 00 00 00 00 00 00 00 00 00 00 00	DA DA 03 A0 ED F8 01 3A 01 20 0D 04 22 33 04 04 33 80 40 54 80	FD FD FD 54 90 0D 20 04 3B	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA JSR BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC INY ATTEM THE YEAR DFB DFB DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9 \$54,\$33,\$00 \$90,\$04,\$2	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04 0,\$54,\$33 4,\$90,\$04
F940: F941: F944: F948: F948: F948: F952: F953: F956: F958: F958: F958: F960: F961: F967: F967: F967: F977: F9778: F978: F978: F978: F978: F978: F978: F978:	98 20 4C A2 A9 20 CD 060 38 A5 A4 A10 88 65 90 C8 03 54 00 90 54 00 90 90 90 90 90 90 90 90 90 90 90 90	DA DA 03 A0 ED F8 01 3A 01 20 0D 04 22 33 80 04 33 80 90	FD FD FD 54 90 0D 20 04 3B	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JJMP LDX LDA LDX LDA JSR RTS SEC LDA LDY TAX BPL DEY ADC BCC LINY RTS FMT1 IF Y: IF Y: DFB DFB DFB DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9 \$54,\$33,\$0 \$90,\$04,\$2 \$0D,\$80,\$0 \$20,\$54,\$3	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04 0,\$54,\$33 4,\$90,\$04 B,\$0D,\$80
F940: F941: F941: F944: F948: F948: F948: F952: F953: F956: F958: F958: F958: F961: F966: F967: F967: F967: F977: F978: F978: F978:	98 20 84 A2 A9 20 CD0 60 38 A5 A4 AA 65 90 C8 60 03 54 80 90 90 90 90 90 90 90 90 90 90 90 90 90	DA DA 03 A0 ED F8 01 3A 01 20 0D 04 22 33 04 33 80 04 54 80 04 54 80 04 54 80 04 54 64 65 66 67 68 68 68 68 68 68 68 68	FD FD FD 544 90 0D 20 04 3B 00	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JMP LDX LDA JSR DEX BNE RTS SEC LDA LDY TAX BPL DEY ADC BCC INY RTS FMT1 IF Y: IF Y: DFB DFB DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9 \$54,\$33,\$0: \$90,\$04,\$2 \$0D,\$80,\$0	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04 0,\$54,\$33 4,\$90,\$04 B,\$0D,\$80
F940: F941: F944: F948: F948: F948: F946: F952: F953: F956: F958: F956: F956: F956: F966: F967: F966: F967: F967: F976: F9771: F978:	98 20 84 A2 A9 20 CA D60 38 A5 A4 AA 0 88 65 90 C8 60 0 38 54 80 90 90 90 90 90 90 90 90 90 90 90 90 90	DA DA 03 A0 ED F8 01 3A 01 20 00 04 22 33 04 04 04 54 80 944 0D	FD FD FD 544 90 0D 20 04 3B 00	PRNTYX PRNTAX PRNTX PRBLNK PRBL2 PRBL3 PCADJ PCADJ2 PCADJ3 PCADJ4 RTS2 * * * *	TYA JSR TXA JJMP LDX LDA LDX LDA JSR RTS SEC LDA LDY TAX BPL DEY ADC BCC LINY RTS FMT1 IF Y: IF Y: DFB DFB DFB DFB	PRBYTE PRBYTE #\$03 #\$A0 COUT PRBL2 LENGTH PCH PCADJ4 PCL RTS2 BYTES: =0 =1 \$04,\$20,\$5 \$80,\$04,\$9 \$54,\$33,\$0 \$90,\$04,\$2 \$0D,\$80,\$0 \$20,\$54,\$3	OUTPUT TARGET ADR OF BRANCH AND RETURN BLANK COUNT LOAD A SPACE OUTPUT A BLANK LOOP UNTIL COUNT=0 0=1-BYTE, 1=2-BYTE 2=3-BYTE TEST DISPLACEMENT SIGN (FOR REL BRANCH) EXTEND NEG BY DEC PCH PCL+LENGTH(OR DISPL)+1 TO A CARRY INTO Y (PCH) XXXXXXYO INSTRS THEN LEFT HALF BYTE THEN RIGHT HALF BYTE (X=INDEX) 4,\$30,\$0D 0,\$03,\$22 D,\$80,\$04 0,\$54,\$33 4,\$90,\$04 B,\$0D,\$80 0,\$22,\$44

```
F98A: 11 22 44
F98D: 33 0D
                        DFB
                             $11,$22,$44,$33,$0D
F98F: C8 44 A9
F992: 01 22
                        DFB
                              $C8,$44,$A9,$01,$22
F994: 44 33 0D
F997: 80 04
                        DFB
                              $44,$33,$0D,$80,$04
F999: 90 01 22
F99C: 44 33
                        DFB
                              $90,$01,$22,$44,$33
F99E: 0D 80 04
F9A1: 90
                        DFB
                              $0D,$80,$04,$90
F9A2: 26 31 87
F9A5: 9A
                        DFB
                              $26,$31,$87,$9A $ZZXXXY01 INSTR'S
               FMT2
F9A6: 00
                        DFB
                              $00
                                       ERR
F9A7: 21
                        DFB
                              $21
                                       TMM
F9A8: 81
                        DFB
                               $81
                                        Z-PAGE
F9A9: 82
                         DFB
                               $82
                                       ABS
F9AA: 00
                        DFB
                               $00
                                       IMPLIED
F9AB: 00
                        DFB
                               $00
                                       ACCUMULATOR
F9AC: 59
                        DFB
                              $59
                                       (ZPAG,X)
F9AD: 4D
                        DFB
                              $4D
                                        (ZPAG),Y
F9AE: 91
                        DFB
                               $91
                                       ZPAG,X
F9AF: 92
                        DFB
                               $92
                                       ABS,X
F9B0: 86
                        DFB
                               $86
                                       ABS,Y
F9B1: 4A
                        DFB
                              $4A
                                       (ABS)
F9B2: 85
                        DFB
                              $85
                                       ZPAG, Y
F9B3: 9D
                        DFB
                              $9D
                                       RELATIVE
F9B4: AC A9 AC
F9B7: A3 A8 A4
                CHAR1
                        ASC ",),#($"
F9BA: D9 00 D8
F9BD: A4 A4 00
                CHAR2
                        DFB $D9,$00,$D8,$A4,$A4,$00
                *CHAR2:
                        "Y",0,"X$$",0
                         MNEML
                                       IS OF FORM:
                         (A) XXXXX000
                          (B) XXXYY100
                         (C) 1XXX1010
                          (D) XXXYYY10
                          (E) XXXYYY01
                             (X=INDEX)
F9C0: 1C 8A 1C
F9C3: 23 5D 8B MNEMI
                        DFB $1C,$8A,$1C,$23,$5D,$
F9C6: 1B A1 9D
F9C9: 8A 1D 23
                        DFB
                              $1B,$A1,$9D,$8A,$1D,$23
F9CC: 9D 8B 1D
F9CF: A1 00 29
                        DFB
                              $9D,$8B,$1D,$A1,$00,$29
F9D2: 19 AE 69
F9D5: A8 19 23
                        DFB
                              $19,$AE,$69,$A8,$19,$23
F9D8: 24 53 1B
F9DB: 23 24 53
                        DFB
                              $24,$53,$1B,$23,$24,$53
F9DE: 19 A1
                        DFB
                              $19,$A1 (A) FORMAT ABOVE
F9E0: 00 1A 5B
                              $00,$1A,$5B,$5B,$A5,$69
F9E3: 5B A5 69
                        DFB
F9E6: 24 24
                        DFB
                              $24,$24 (B) FORMAT
F9E8: AE AE A8
F9EB: AD 29 00
                               $AE,$AE,$A8,$AD,$29,$00
                        DFB
F9EE: 7C 00
                              $7C,$00
                                        (C) FORMAT
                        DFB
F9F0: 15 9C 6D
F9F3: 9C A5 69
                        DFB
                              $15,$9C,$6D,$9C,$A5,$69
F9F6: 29 53
                        DFB
                              $29,$53 (D) FORMAT
F9F8: 84 13 34
F9FB: 11 A5 69
                               $84,$13,$34,$11,$A5,$69
F9FE: 23 A0
                        DFB
                              $23.$A0
                                         (E) FORMAT
FA00: D8 62 5A
FA03: 48 26 62 MNEMR
                        DFB
                              $D8,$62,$5A,$48,$26,$62
FA06: 94 88 54
FA09: 44 C8 54
                        DFB
                              $94,$88,$54,$44,$C8,$54
FA0C: 68 44 E8
FA0F: 94 00 B4
                        DFB
                              $68,$44,$E8,$94,$00,$B4
FA12: 08 84 74
FA15: B4 28 6E
                        DFB
                              $08,$84,$74,$B4,$28,$6E
FA18: 74 F4 CC
FA1B: 4A 72 F2
                         DFB
                              $74,$F4,$CC,$4A,$72,$F2
FA1E: A4 8A
                              $A4,$8A (A) FORMAT
                        DFB
FA20: 00 AA A2
FA23: A2 74 74
                        DFB
                              $00,$AA,$A2,$A2,$74,$74
FA26: 74 72
                        DFB
                              $74,$72
                                        (B) FORMAT
FA28: 44 68 B2
FA2B: 32 B2 00
                        DFB
                              $44,$68,$B2,$32,$B2,$00
                        DFB $22,$00 (C) FORMAT
FA2E: 22 00
FA30: 1A 1A 26
FA33: 26 72 72
                        DFB
                              $1A,$1A,$26,$26,$72,$72
FA36: 88 C8
                        DFB
                              $88,$C8
                                        (D) FORMAT
FA38: C4 CA 26
FA3B: 48 44 44
                        DFB
                               $C4,$CA,$26,$48,$44,$44
FA3E: A2 C8
                        DFB
                              $A2,$C8
                                        (E) FORMAT
```

FA40:	FF	FF	FF		DFB	SFF.SFF.SF	7
				STEP	JSR	TNSTDSP	F DISASSEMBLE ONE INST
FA46:		20	10	DILL	DT.A	INDIDDI	AT (PCT. H)
FA47:		20			CILIV	DTMI	AT (PCL,H) ADJUST TO USER STACK. SAVE
FA47:					PLA	KINL	ADJUST TO USER
						RTNH	
FA4A:							RTN ADR.
FA4C:	AZ DD	10	ED	VOTNIM	LDX	#\$U8	TNIE VEG ADEA
FA4E:	BD	10	FB	XQINIT	LDA	INITBL-1,X	INIT XEQ AREA
FA51:	95	3C				XQT,X	
FA53:					DEX		
FA54:					BNE	XQINIT	USER OPCODE BYTE
FA56:					LDA	(PCL,X)	USER OPCODE BYTE
FA58:	F0	42					SPECIAL IF BREAK
FA5A:	A4	2F			LDY	LENGTH	LEN FROM DISASSEMBLY
FA5C:	C9	20			CMP	#\$20	
FA5E:	F0	59			BEQ	XJSR	HANDLE JSR, RTS, JMP,
FA60:	C9	60			CMP	#\$60	HANDLE JSR, RTS, JMP, JMP (), RTI SPECIAL
FA62:	F0	45			BEQ		
FA64:	C9	4C			CMP	#\$4C	
FA66:	F0	5C			BEQ		
FA68:	C9	6C					
FA6A:					BEO	#\$6C XJMPAT	
FA6C:					CMP		
FA6E:					BEQ	YPTT	
FA70:					AND	#\$15	
FA72:					EUD	#¢1/	
FA74:					CMD	#\$14 #\$04	CODY HOLD INCH HO VEO ADEA
							COPY USER INST TO XEQ AREA
FA76:	F U	02		*****	BEQ	AQZ	WITH TRAILING NOPS
FA78:	BI	3A		XQI	LDA	(PCL),Y	CHANGE REL BRANCH
FA7A:		3C	00		STA	XQT,Y	DISP TO 4 FOR
FA7D:					DEI		JMP TO BRANCH OR
FA7E:					BPL	XQ1	NBRANCH FROM XEQ.
FA80:					JSR	RESTORE	RESTORE USER REG CONTENTS. XEQ USER OP FROM RAM
FA83:	4C	3C	00		JMP	XQT	XEQ USER OP FROM RAM
FA86:	85	45		IRQ	STA	ACC	(RETURN TO NBRANCH)
FA88:					PLA PHA		
FA89:	48				PHA		**IRQ HANDLER
FA8A:	0A				ASL	A	
FA8B:	0A				ASL	A	
FA8C:					ASL		
FA8D:		0.3			BMT	BREAK	TEST FOR BREAK
FA8F:					TMP	(TROLOC)	TEST FOR BREAK USER ROUTINE VECTOR IN RAM
					PLP	(1102200)	ODDIC ROOTING VEGTOR IN REE.
FA93:						C 7/1/1	SAVE REG'S ON BREAK
		40	PP		JON	DAVI	AAAAA NO G DAA AVAG
ENG6.					DT 7		TNCT LIDING DC
FA96:		27			PLA	DCI	INCLUDING PC
FA97:	85				STA	PCL	INCLUDING PC
FA97: FA99:	85 68				STA PLA		INCLUDING PC
FA97: FA99: FA9A:	85 68 85	3B			STA PLA STA	PCH	
FA97: FA99: FA9A: FA9C:	85 68 85 20	3B 82	F8	XBRK	STA PLA STA JSR	PCH INSDS1	PRINT USER PC.
FA97: FA99: FA9A: FA9C: FA9F:	85 68 85 20 20	3B 82 DA	F8 FA	XBRK	STA PLA STA JSR	PCH INSDS1	PRINT USER PC.
FA97: FA99: FA9A: FA9C: FA9F:	85 68 85 20 20 4C	3B 82 DA 65	F8 FA FF	XBRK	STA PLA STA JSR	PCH INSDS1	
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5:	85 68 85 20 20 4C 18	3B 82 DA 65	F8 FA FF	XBRK	STA PLA STA JSR JSR JMP CLC	PCH INSDS1	PRINT USER PC. AND REG'S GO TO MONITOR
FA97: FA99: FA9A: FA9C: FA9F:	85 68 85 20 20 4C 18	3B 82 DA 65	F8 FA FF	XBRK	STA PLA STA JSR JSR JMP	PCH INSDS1	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5:	85 68 85 20 20 4C 18 68	3B 82 DA 65	F8 FA FF	XBRK	STA PLA STA JSR JSR JCC JMP CLC PLA	PCH INSDS1	PRINT USER PC. AND REG'S GO TO MONITOR
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5: FAA6:	85 68 85 20 20 4C 18 68 85	3B 82 DA 65	F8 FA FF	XBRK	STA PLA STA JSR JSR JCC JMP CLC PLA	PCH INSDS1 RGDSP1 MON	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5: FAA6: FAA7:	85 68 85 20 20 4C 18 68 85 68	3B 82 DA 65	F8 FA FF	XBRK XRTI	STA PLA STA JSR JSR JMP CLC PLA STA	PCH INSDS1 RGDSP1 MON STATUS	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5: FAA6: FAA7: FAA9:	85 68 85 20 20 4C 18 68 85 68	3B 82 DA 65	F8 FA FF	XBRK XRTI	STA PLA STA JSR JSR JMP CLC PLA STA PLA	PCH INSDS1 RGDSP1 MON STATUS	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5: FAA6: FAA7: FAA9: FAAA:	85 68 85 20 20 4C 18 68 85 68 85	3B 82 DA 65	F8 FA FF	XBRK XRTI XRTS	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA	PCH INSDS1 RGDSP1 MON STATUS	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK
FA97: FA99: FA9C: FA9C: FAAC: FAAC: FAAC: FAAC: FAAC: FAAC:	85 88 85 20 4C 18 88 88 88 88 88 88 88 88	3B 82 DA 65 48 3A	F8 FA FF	XBRK XRTI XRTS PCINC2	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA	PCH INSDS1 RGDSP1 MON STATUS PCL	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA6: FAA7: FAA9: FAAA: FAAC: FAAC:	85 68 85 20 4C 18 68 85 68 85 68 85 A5	3B 82 DA 65 48 3A 3B 2F	F8 FA FF	XBRK XRTI XRTS PCINC2	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA LDA	PCH INSDS1 RGDSP1 MON STATUS PCL	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0)
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA6: FAA7: FAA9: FAAA: FAAC: FAAC: FAAF:	85 68 85 20 4C 18 68 85 68 85 68 85 20	3B 82 DA 65 48 3A 3B 2F 56	F8 FA FF	XBRK XRTI XRTS PCINC2	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA LDA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0)
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA6: FAA7: FAA9: FAAA: FAAC: FAAD: FAAB:	85 68 85 20 4C 18 68 85 68 85 68 85 20 85	3B 82 DA 65 48 3A 3B 2F 56	F8 FA FF	XBRK XRTI XRTS PCINC2	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA LDA JSR	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0)
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5: FAA7: FAA7: FAAA: FAAA: FAAC: FAAC: FAAB: FAAB: FAAB:	85 85 20 4C 18 85 68 85 68 85 20 85 18	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA LDA JSR STY CLC	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0)
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5: FAA6: FAA7: FAAC: FAAA: FAAC: FAAB: FAAB: FAB1: FAB4: FAB6: FAB6:	85 85 20 4C 18 85 68 85 68 85 20 4C 18 18 90	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA LDA JSR STY CLC BCC	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0)
FA97: FA99: FA9A: FA9C: FA9F: FAA5: FAA6: FAA7: FAA9: FAAC: FAAD: FAAB1: FAB1: FAB4: FAB6:	85 85 20 4C 18 68 85 68 85 20 4C 18 18 90 18	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA LDA JSR STY CLC CLC CLC	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN
FA97: FA99: FA9C: FA9C: FA9F: FAA5: FAA6: FAA7: FAA9: FAAA: FAAC: FAAD: FAAB1: FAB1: FAB4: FAB6: FAB9: FAB9: FAB9:	85 85 20 4C 18 85 68 85 68 85 20 4C 18 85 18 90 18	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC STA CLC STA CLC STA STA STA STA CLC STA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN
FA97: FA99: FA9C: FA9C: FA9F: FAA5: FAA6: FAA7: FAA9: FAAC: FAAC: FAAB1: FAB4: FAB6: FAB7: FAB7: FAB7: FAB9:	85 85 20 4C 18 85 68 85 68 85 20 4C 18 20 4C 18 20 4C 18 18 18 18 18 18 18 18 18 18 18 18 18	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JSR JSR CLC PLA STA PLA STA PLA STA STA STA CLC STY CLC BCC CLC JSR TAX	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA6: FAA7: FAA7: FAAC: FAAD: FAAB: FABA: FAB6: FAB7: FAB9: FAB9: FAB9: FAB9: FAB9: FAB9:	85 85 20 4C 18 85 68 85 A5 20 84 18 90 18 20 AA	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC BCC CLC BCC CLC TAX TYA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN
FA97: FA99: FA9A: FA9C: FA9F: FAA2: FAA5: FAA7: FAA7: FAAA: FAAC: FAAB: FAB1: FAB6: FAB7: FAB9:	85 85 20 4C 18 85 68 85 A5 20 84 18 90 18 20 AA 98	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC BCC CLC JSR CLC JSR TAX TYA PHA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR
FA97: FA99: FA9A: FA9C: FA9F: FAAC: FAAA: FAAA: FAAA: FAAC: FAAC: FAAB: FABA: FABA: FABB: FABB: FABB: FABB: FABB: FABC:	85 85 20 4C 18 85 85 85 85 20 84 18 90 18 20 AA 98 48 84	3B 82 DA 65 48 3A 3B 2F 56 3B	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA LDA JSR CLC BCC CLC JSR TAX PTYA PHA TXA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR
FA97: FA99: FA9A: FA9C: FA9F: FAA5: FAA6: FAA7: FAA0: FAAC: FAAD: FAB1: FAB1: FAB4: FAB6: FAB7: FAB9: FAB9: FAB0:	85 85 20 4C 18 85 68 85 68 85 20 84 18 90 18 20 AA 98 48 48	3B 82 DA 65 48 3B 2F 56 3B 14 54	F8 FA FF	XBRK XRTI XRTS PCINC2 PCINC3 XJSR	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA CLC STA CLC STA LDA JSR STY LDA JSR TYA PLA TYA PHA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR
FA97: FA99: FA9C: FA9C: FA9C: FAA5: FAA6: FAA7: FAA9: FAAA: FAAC: FAAB1: FAB1: FAB6: FAB7: FAB9: FAB7: FAB9: FAB7: FAB9: FAB7: FAB9: FAB7: FAB1: FAB7: FAB7: FAB7: FAB7: FAB7: FAB7: FAB7: FAB87:	85 85 20 4C 18 85 68 85 68 85 20 4C 18 85 20 4C 85 45 45 45 45 46 46 46 46 46 46 46 46 46 46 46 46 46	3B 82 DA 65 48 3B 2F 56 3B 14 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC STA CLC GUC GUC TAX TYA PHA TXA PHA LDY	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR
FA97: FA99: FA97: FA97: FA97: FAA2: FAA5: FAA7: FAA7: FAAA: FAAC: FAAB: FAAB: FABF: FABB:	85 85 20 4C 18 85 68 85 68 85 A5 20 42 48 48 48 48 48 48 48 48	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA LDA STA LDA JSR STY CLC BCC CLC CLC TAX TYA PHA TXA PHA LDY CLC	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR
FA97: FA99: FA97: FA97: FA97: FAA2: FAA5: FAA7: FAA7: FAAA: FAAC: FAAB1: FAB4: FAB6: FAB7: FAB8: FAB7:	85 85 20 4C 18 85 85 85 85 85 85 20 18 20 AA 98 48 48 A5 18 18 18 18 18 18 18 18 18 18 18 18 18	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA LDA JSR STY CLC BCC CLC JSR TAX TYA PHA TXA PHA TXA PHA LDY CLC LDA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAAA: FAAC: FAAB: FAAB: FABB: FABF: FABB: FABF: FABG: FABG: FABF: FABG:	85 68 20 20 4C 18 85 68 85 A5 20 42 AA 98 48 48 48 48 18 18 18 18 18 18 18 18 18 18 18 18 18	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC BCC CLC JSR TYA PHA TXA PHA LDY CLC LDA TAX	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA5: FAA6: FAA7: FAA9: FAAC: FAAB1: FAB1: FAB6: FAB7: FAB8: FAB6: FAB7: FAB7: FAB0: FAB7: FAB0: FAB7: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8:	85 68 20 20 4C 18 85 68 85 A5 20 84 18 20 AA 98 48 48 AB 18 48 AB 18 48 48 48 48 48 48 48 48 48 48 48 48 48	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC STA LDA JSR CLC BCC CLC JSR TAX PHA TXA PHA LDY CLC LDD TXA PHA LDY CLC LDA TAX DEY	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 (PCL), Y	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA9C: FA9C: FA9C: FAAC: FAAC: FAAC: FAAA: FAAC: FABC: FABC: FABC: FABC: FABC: FABC: FABC: FABC: FACC:	85 68 20 20 4C 68 85 68 85 68 85 20 20 18 20 48 48 48 48 48 48 48 48 48 48 48 48 48	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR	STA PLA STA JSR JMP CLC PLA STA LDA JSR TYA PLA TYA PHA LDY CLC LDA TYA PHA LDY CLC LDA TAX DEY LDA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL), Y	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA96: FAA6: FAA7: FAA7: FAAA: FAAC: FAAB: FAAB: FABH:	85 68 85 20 20 4C 88 85 68 85 68 85 20 20 18 20 AA 48 48 48 A0 18 B1 86	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA LDA STY CLC BCC CLC BCC CLC LDC LDC LDC LDC LDC LDC LDC LDC L	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL), Y PCH	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA6: FAA7: FAAA: FAAC: FAAB: FAAB: FAB6: FAB7: FAB6: FAB7: FAB6: FAB7: FAB6: FAB7: FAB6: FAB7: FAB7: FAB6: FAB7:	85 68 85 20 20 18 85 85 85 85 85 85 85 84 18 91 84 84 84 84 84 84 84 84 84 84 84 84 84	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54 02 3A 3B 3A 3B 3A	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA PLA STA LDA JSR STY CLC BCC CLC BCC LCC LC JSR TAX TYA PHA TXA PHA TXA PHA TXA PHA TXA CLC LDA TAX CLC LDA TAX STA STA CLC CLC SSTA STA STA STA STA STA STA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAAA: FAAC: FAAB: FABB: FABB: FABB: FABB: FABB: FABC: FABF: FABC: FAC: FAC: FAC: FAC: FAC: FAC: FAC: FA	85 68 20 20 18 68 85 85 85 85 85 85 85 85 85 85 85 85 85	3B 82 DA 65 48 3B 2F 53B 14 54 02 3A 3B 3A 3B 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA LDA JSR STY CLC BCC CLC JSR TYA PHA TXA PHA LDY CLC LDA TAX TYA DEY LDA STX STX BCS	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y PCH PCL XJMP	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAA7: FAA7: FAAB: FAAC: FAAF: FAB1: FAB6: FAB7: FAB7: FAB8: FAB7: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8:	85 68 85 20 4C 18 85 85 85 85 85 85 20 42 48 48 48 48 85 85 85 85 85 85 85 85 85 85 85 85 85	3B 82 DA 65 48 3B 2F 53B 14 54 02 3A 3B 3A 3B 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT NEWPCL	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA LDA JSR STY CLC BCC CLC JSR TYA PHA TXA PHA LDY CLC LDA TAX TYA DEY LDA STX STX BCS	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAAA: FAAC: FAAB: FABB: FABB: FABB: FABB: FABB: FABC: FABF: FABC: FAC: FAC: FAC: FAC: FAC: FAC: FAC: FA	85 68 85 20 4C 18 85 85 85 85 85 85 20 42 48 48 48 48 85 85 85 85 85 85 85 85 85 85 85 85 85	3B 82 DA 65 48 3B 2F 53B 14 54 02 3A 3B 3A 3B 54	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA LDA JSR STY CLC BCC CLC JSR TYA PHA TXA PHA LDY CLC LDA TAX TYA DEY LDA STX STX BCS	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y PCH PCL XJMP	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAA7: FAA7: FAAB: FAAC: FAAF: FAB1: FAB6: FAB7: FAB7: FAB8: FAB7: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8: FAB7: FAB8:	85 68 85 20 24 18 68 85 86 85 85 82 84 84 84 84 84 84 84 84 84 84 84 84 84	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54 02 3A 3B 3A 3B 2D	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC BCC CLC JSR TAX TYA PHA TXA PHA LDY CLC LDA TXA DEY LDA STX ELDA STX CLC CLC LDA TXA DEY LDA STX LDA STX CLC CLC LDA TXA DEY LDA STX STA LDA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL), Y PCH PCL XJMP RTNH	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAAA: FAAC: FAAB: FABA: FABF: FABF: FABF: FABG: FABF: FABF: FABG: FABF:	85 85 20 24 C 18 85 8 68 5 85 20 18 85 85 85 85 20 18 85 85 85 85 85 85 85 85 85 85 85 85 85	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54 02 3A 3A 3B 3A 52 2C	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT NEWPCL RTNJMP	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA CLC BCC CLC BCC CLC LDA TXA PHA TXA PHA TXA PHA TXA PHA TXA CLC LDA TAX TYA CLC LDA TAX TYA PHA TXA DEY CLC LDA TAX DEY LDA STX STA BCS LDA PHA PHA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y RTNH RTNL	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE LOAD PC FOR JMP, (JMP) SIMULATE.
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAAA: FAAC: FAAB: FABA: FABF: FABF: FABF: FABG: FABF: FABF: FABG: FABF:	85 85 20 24 C 18 85 8 68 5 85 20 18 85 85 85 85 20 18 85 85 85 85 85 85 85 85 85 85 85 85 85	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54 02 3A 3A 3B 3A 52 2C	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMP XJMPAT NEWPCL RTNJMP	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA CLC BCC CLC BCC CLC LDA TXA PHA TXA PHA TXA PHA TXA PHA TXA CLC LDA TAX TYA CLC LDA TAX TYA PHA TXA DEY CLC LDA TAX DEY LDA STX STA BCS LDA PHA PHA	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y RTNH RTNL	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE
FA97: FA99: FA97: FA97: FA97: FAA2: FAA7: FAA7: FAAA: FAAC: FAAB: FABA: FABF: FABF: FABF: FABG: FABF: FABF: FABG: FABF:	85 85 20 4C 18 85 85 85 85 85 85 85 85 85 85 85 85 85	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54 02 3A 3A 3A 3A 3A 3A 3A 2D 2C 8E	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMPAT NEWPCL RTNJMP REGDSP	STA PLA STA JSR JMP CLC PLA STA PLA STA PLA STA PLA STA PLA STA CLC BCC CLC BCC CLC TXA TYA PHA TXA PHA TXA PHA TXA PHA TXA PHA TXA TXA PHA TXA TXA TXA PHA TXA TXA TXA TXA TXA TXA TXA TXA TXA TX	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y RTNH RTNL	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE LOAD PC FOR JMP, (JMP) SIMULATE.
FA97: FA99: FA99: FA9A: FA9C: FAA7: FAA7: FAAA: FAAC: FAAB: FAAB: FABB: FABF: FABB: FABF: FABG: FABF:	85 85 20 4C 18 85 85 85 85 85 85 82 4 81 82 84 84 84 84 84 84 84 84 84 84 84 84 84	3B 82 DA 65 48 3A 3B 2F 56 3B 14 54 02 3A 3A 3B 3A 52 2D 54 02 3A 3B 54 02 3A 3A 54 54 54 3A 54 54 3A 3A 3A 3A 54 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A	F8 FA FF F9	XBRK XRTI XRTS PCINC2 PCINC3 XJSR XJMP XJMPAT NEWPCL RTNJMP REGDSP	STA PLA STA JSR JSR JMP CLC PLA STA PLA STA PLA STA PLA STA CLC BCC CLC BCC CLC TAX TYA PHA TXA PHA LDY CLC LDA TAX DEY LDA STX STA BCS LDA PHA BCS LDA PHA JSR	PCH INSDS1 RGDSP1 MON STATUS PCL PCH LENGTH PCADJ3 PCH NEWPCL PCADJ2 #\$02 (PCL),Y (PCL),Y (PCL),Y CROUT #ACC	PRINT USER PC. AND REG'S GO TO MONITOR SIMULATE RTI BY EXPECTING STATUS FROM STACK, THEN RTS RTS SIMULATION EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0) UPDATE PC BY LEN UPDATE PC AND PUSH ONTO STACH FOR JSR SIMULATE LOAD PC FOR JMP, (JMP) SIMULATE.

FADE:						#ACC/256	
FAE0:					STA		
FAE2:					LDX	#\$FB	
FAE4:	Α9	A0		RDSP1	LDA	#\$A0	
					JSR	COUT	
FAE9:					LDA	RTBL-\$FB,X	
FAEC:					JSR		
FAEF:					LDA	#\$BD	
FAF1:					JSR	COUT ACC+5,X	
FAF4:					LDA	ACC+5,X	
FAF6:			FD			PRBYTE	
FAF9:					INX		
FAFA:						RDSP1	
FAFC:	60				RTS		
				BRANCH	CLC		BRANCH TAKEN,
FAFE:						#\$01	ADD LEN+2 TO PC
FB00:	В1	3A				(PCL),Y	
FB02:	20	56	F9			PCADJ3	
FB05:	85	3A			STA	PCL	
FB07:	98				TYA		
FB08:	38				SEC		
FB09:					BCS	PCINC2	
FB0B:	20	4A	FF	NBRNCH	JSR	SAVE	NORMAL RETURN AFTER
FB0E:	38				SEC		XEQ USER OF
FB0F:	B0	9E			BCS	PCINC3	GO UPDATE PC
FB11:	EΑ			INITBL	NOP		
FB12:	EΑ				NOP		DUMMY FILL FOR
FB13:	4C	0B	FB		JMP	NBRNCH	XEQ AREA
FB16:	4C	FD	FA		JMP	BRANCH	
FB19:	C1			RTBL	DFB	\$C1	
FB1A:	D8				DFB	\$D8	
FB1B:	D9				DFB	\$D9	
FB1C:	D0					\$D0	
FB1D:	D3				DFB	\$D3	
FB1E:	AD	70	C0	PREAD	LDA	PTRIG	TRIGGER PADDLES
FB21:							INIT COUNT
FB23:					NOP		COMPENSATE FOR 1ST COUNT
FB24:					NOP		
			C0			PADDIO.X	COUNT Y-REG EVERY
FB28:	10	04	-	1112122	BPI.	RTS2D	12 USEC
FB2A:					INY	111020	12 0020
FB2B:						DREAD2	EXIT AT 255 MAX
FB2D:					DEY	FREADZ	EATT AT 255 MAX
FB2E:				ртерп	DTC		
FB2F:				INIT	T.DA	#\$00	CLR STATUS FOR DEBUG
	כת	00		TIATI		# 500	
FB31.	25	1 Q			STA	STATIC	SOETWADE
FB31:					DIA	DIAIUD	SOFTWARE
FB33:	AD	56	C0		LDA	LORES	DOT I WAKE
FB33: FB36:	AD AD	56 54	C0		LDA LDA	LORES LOWSCR	INIT VIDEO MODE
FB33: FB36:	AD AD	56 54	C0		LDA LDA	LORES LOWSCR	INIT VIDEO MODE
FB33: FB36: FB39: FB3C:	AD AD AD A9	56 54 51 00	C0	SETTXT	LDA LDA LDA LDA	LORES LOWSCR TXTSET #\$00	DOT I WAKE
FB33: FB36: FB39: FB3C: FB3E:	AD AD AD A9 F0	56 54 51 00 0B	C0 C0	SETTXT	LDA LDA LDA LDA LDA BEQ	LORES LOWSCR TXTSET #\$00 SETWND	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW
FB33: FB36: FB39: FB3C: FB3E: FB40:	AD AD AD A9 F0 AD	56 54 51 00 0B 50	C0 C0	SETTXT SETGR	LDA LDA LDA LDA BEQ LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE
FB33: FB36: FB39: FB3C: FB3E: FB40: FB43:	AD AD AD A9 F0 AD AD	56 54 51 00 0B 50 53	C0 C0 C0	SETTXT SETGR	LDA LDA LDA LDA BEQ LDA LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS
FB33: FB36: FB39: FB3C: FB3E: FB40: FB43: FB46:	AD AD AD A9 F0 AD AD 20	56 54 51 00 0B 50 53 36	C0 C0 C0 C0 C0 F8	SETTXT SETGR	LDA LDA LDA LDA BEQ LDA LDA LDA JSR	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE
FB33: FB36: FB39: FB3C: FB3E: FB40: FB43: FB46: FB49:	AD AD A9 F0 AD AD 20 A9	56 54 51 00 0B 50 53 36 14	C0 C0 C0 C0 C0 F8	SETTXT	LDA LDA LDA LDA LDA LDA LDA LDA JSR LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW
FB33: FB36: FB39: FB3C: FB3E: FB40: FB43: FB46: FB49: FB4B:	AD AD A9 F0 AD AD 20 A9 85	56 54 51 00 0B 50 53 36 14 22	C0 C0 C0 C0 C0 F8	SETTXT	LDA LDA LDA LDA BEQ LDA LDA JSR LDA STA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW
FB33: FB36: FB39: FB3C: FB3E: FB40: FB43: FB46: FB49: FB4B: FB4D:	AD AD A9 F0 AD AD AD AD AD AD A9	56 54 51 00 0B 50 53 36 14 22	C0 C0 C0 C0 C0 F8	SETTXT	LDA LDA LDA LDA BEQ LDA LDA JSR LDA STA LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG,
FB33: FB36: FB39: FB3C: FB40: FB40: FB46: FB49: FB4B: FB4D: FB4F:	AD AD A9 F0 AD AD 20 A9 85 A9	56 54 51 00 0B 50 53 36 14 22 00 20	C0 C0 C0 C0 C0 F8	SETTXT	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW
FB33: FB36: FB39: FB3C: FB40: FB44: FB46: FB49: FB4B: FB4D: FB4F: FB51:	AD AD A9 F0 AD 20 A9 85 A9	56 54 51 00 0B 50 53 36 14 22 00 20 28	C0 C0 C0 C0 C0 F8	SETTXT	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG,
FB33: FB36: FB39: FB3C: FB40: FB446: FB49: FB4B: FB4B: FB4B: FB4F: FB51: FB53:	AD AD A9 F0 AD AD 20 A9 85 A9 85 A9	56 54 51 00 0B 50 53 36 14 22 00 20 28 21	C0 C0 C0 C0 C0 F8	SETTXT	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG,
FB33: FB36: FB39: FB3C: FB40: FB44: FB46: FB4B: FB4D: FB4F: FB51: FB53: FB55:	AD AD A9 F0 AD A9 85 A9 85 A9	56 54 51 00 08 50 53 36 14 22 00 20 28 21 18	C0 C0 C0 C0 C0 F8	SETTXT	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24
FB33: FB36: FB39: FB3C: FB40: FB40: FB49: FB4B: FB4D: FB4F: FB51: FB55: FB55:	AD AD AD F0 AD AD AD A9 85 A9 85 A9 85	56 54 51 00 0B 50 53 36 14 22 00 20 28 21 18 23	C0 C0 C0 C0 C0 F8	SETTXT	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG,
FB33: FB36: FB39: FB3C: FB40: FB40: FB49: FB4B: FB4D: FB4F: FB51: FB55: FB55: FB57: FB59:	AD A9 F0 AD A9 85 A9 85 A9	56 54 51 00 0B 50 53 36 14 22 00 28 21 18 23 17	C0 C0 C0 C0 F8	SETTXT SETGR SETWND	LDA LDA LDA LDA LDA LDA LDA LDA LDA JSR LDA STA LDA STA LDA STA LDA STA LDA STA LDA STA LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23
FB33: FB36: FB39: FB3C: FB40: FB40: FB49: FB4B: FB4D: FB51: FB51: FB55: FB57: FB59: FB59:	AD AD A9 F0 AD A9 85 A9 85 A9 85 A9 85 A9	56 54 51 00 0B 50 53 36 14 22 00 28 21 18 23 17 25	C0 C0 C0 C0 C0 F8	SETTXT SETGR SETWND	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24
FB33: FB36: FB37: FB3C: FB40: FB40: FB49: FB49: FB49: FB4F: FB51: FB55: FB57: FB57: FB59: FB59: FB59:	AD AD AD F0 AD AD AD 85 A9 85 A9 85 A9 4C	56 54 51 00 0B 50 53 36 14 22 00 20 28 21 18 23 17 25 22	C0 C0 C0 C0 C0 F8	SETTXT SETGR SETWND	LDA LDA LDA LDA LDA BEQ LDA LDA LDA LDA LDA STA LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG
FB33: FB36: FB37: FB30: FB40: FB49: FB49: FB49: FB4F: FB51: FB55: FB57: FB59: FB59: FB59: FB59: FB59: FB59: FB59:	AD AD F0 AD AD 85 A9 85 A9 85 4C 20	56 54 51 00 0B 50 53 36 14 22 00 20 28 21 18 23 17 25 22 A4	CO CO CO CO F8 FC FB	SETTXT SETGR SETWND TABV MULPM	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX
FB33: FB36: FB39: FB3C: FB40: FB40: FB49: FB4P: FB4F: FB51: FB55: FB57: FB59: FB59: FB59: FB59: FB59: FB59: FB59: FB59: FB59:	AD AD F0 AD AD 85 A9 85 A9 85 4C 20 A0	56 54 51 00 0B 50 53 36 14 22 00 20 28 21 18 23 17 25 22 A4 10	CO CO CO CO F8 FC FB	SETTXT SETGR SETWND TABV MULPM MUL	LDA LDA LDA LDA LDA LDA LDA LDA LDA STA LDA ST	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS
FB33: FB36: FB39: FB3C: FB40: FB40: FB48: FB48: FB4B: FB51: FB55: FB57: FB57: FB59: FB59: FB50: FB50: FB63: FB63:	AD AD F0 AD AD 85 A9 85 A9 85 A9 4C 20 A0 A5	56 54 51 00 0B 50 53 36 14 22 00 20 28 21 18 23 17 25 22 A4 10 50	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL	LDA LDA LDA LDA LDA LDA LDA LDA STA LDA LDA STA LDA STA LDA LDA LDA STA LDA LDA LDA LDA LDA LDA LDA LDA LDA LD	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND
FB33: FB36: FB39: FB3C: FB40: FB44: FB49: FB4B: FB4F: FB51: FB55: FB57: FB59:	AD AD AD F0 AD AD 85 A9 A9 85 A9 85 A9 85 A9 85 A9 85 A9 85 A9 85 A9 85 A9 85 A9 A9 A9 A9 A9 A9 A9 A9 A9 A9 A9 A9 A9	56 54 51 00 08 50 53 36 14 22 00 20 28 21 18 23 17 25 22 A4 10 50	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$10 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND
FB33: FB36: FB39: FB3C: FB40: FB44: FB49: FB4B: FB4D: FB51: FB57: FB57: FB57: FB50: FB60:	AD A	56 54 51 00 08 50 53 36 14 22 00 20 28 21 18 23 17 25 22 A4 10 50	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$10 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY,
FB33: FB36: FB37: FB30: FB40: FB40: FB49: FB49: FB49: FB49: FB51: FB55: FB57: FB57: FB59:	AD AD AD FO AD	56 54 51 00 0B 50 53 36 14 22 00 28 21 18 23 17 25 22 A4 10 50	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND
FB33: FB36: FB37: FB30: FB40: FB49: FB49: FB49: FB4F: FB51: FB55: FB57: FB59:	AD AD AD 20 AD 85 AD 85 AD 85 AD 4C AD AD AD AD 85 AD 85 AD 85 AD	56 54 51 00 08 50 53 36 14 22 00 28 21 18 23 17 25 22 A4 10 50 50 50 50 50 50 50 50 50 50 50 50 50	CO CO CO CO F8 FC FB	SETTXT SETGR SETWND TABV MULPM MUL MUL2	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD.
FB33: FB36: FB39: FB3C: FB40: FB44: FB48: FB48: FB4B: FB51: FB55: FB57: FB59: FB59: FB56: FB67: FB66: FB66: FB67: FB68: FB68: FB68: FB68: FB68:	AD AD AD 20 AD 85 AD 85 AD 85 AD 4C AD AD AD 85 AD 85 AD 85 AD 85 AD 85 AD	56 54 51 00 0B 50 53 36 14 22 00 28 21 18 23 25 22 A4 10 50 0C FE 54	CO CO CO CO F8 FC FB	SETTXT SETGR SETWND TABV MULPM MUL	LDA LDA LDA LDA LDA LDA LDA LDA LDA STA LDA LDA LDA LDA LDA LDA LDA LDA LDA LD	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX)
FB33: FB36: FB39: FB3C: FB40: FB440: FB449: FB4B: FB4B: FB51: FB55: FB57: FB59: FB59: FB59: FB56: FB66: FB66: FB66: FB66: FB66: FB68	AD AD F0 AD AD 85 A9 85 A9 85 AC 20 A0 A5	56 54 51 00 0B 50 53 36 14 22 00 28 21 18 23 17 25 22 A4 10 50 0C FE 54 56	CO CO CO CO F8 FC FB	SETTXT SETGR SETWND TABV MULPM MUL MUL2	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$14 WNDTOP #\$18 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB39: FB3C: FB40: FB440: FB449: FB4B: FB4D: FB51: FB55: FB57: FB59: FB59: FB59: FB59: FB60: FB66: FB66: FB67: FB68	AD AD F0 AD 85 A9 A5	56 54 51 00 0B 50 53 36 14 22 00 28 21 18 23 17 25 22 A4 10 50 0C FE 54 56	CO CO CO CO F8 FC FB	SETTXT SETGR SETWND TABV MULPM MUL MUL2	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB39: FB3C: FB3E: FB49: FB49: FB49: FB49: FB51: FB55: FB57: FB57: FB59: FB59: FB59: FB60: FB63: FB66: FB66: FB68:	AD AD AD 85 A9 B5	56 54 51 00 08 50 53 36 14 22 00 02 28 21 18 23 17 25 22 A4 10 50 50 50 50 50 50 50 50 50 50 50 50 50	CO CO CO CO F8 FC FB	SETTXT SETGR SETWND TABV MULPM MUL MUL2	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB37: FB30: FB40: FB49: FB49: FB49: FB49: FB49: FB51: FB55: FB57: FB59: FB59: FB59: FB59: FB59: FB60: FB63: FB63: FB64: FB64: FB64: FB66: FB67: FB68:	AD AD AD FO AD 85	56 54 51 00 08 50 53 614 22 00 28 21 18 23 17 25 22 A4 10 50 50 60 60 60 60 60 60 60 60 60 60 60 60 60	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MULPM MUL MUL2	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$914 WNDDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$110 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X XTNDL+2,X	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB39: FB3C: FB40: FB46: FB49: FB4B: FB4B: FB51: FB55: FB57: FB59: FB59: FB58: FB66: FB67: FB68: FB66: FB68:	AD AD A9 F0 AD 20 85 A9 85 A9 85 A9 85 A9 85 A9 85 AD A5 AD	56 54 51 00 08 50 53 314 22 20 28 21 18 23 17 25 22 A4 10 50 50 65 65 75 76 76 76 76 76 76 76 76 76 76 76 76 76	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL MUL2 MUL3	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X XTNDL+2,X MUL3 #\$03	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB39: FB3C: FB40: FB449: FB449: FB449: FB48: FB51: FB55: FB57: FB57: FB58: FB57: FB58: FB66: FB66: FB68: FB6	AD AD AD F0 AD	56 54 51 00 0B 53 36 14 22 00 20 28 21 12 25 22 A4 10 50 0C FE 54 56 54 F7 03	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MULPM MUL MUL2	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X XTNDL+2,X #\$03 \$76	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB37: FB36: FB40: FB44: FB46: FB48: FB4F: FB51: FB57: FB57: FB58: FB57: FB58: FB66: FB68:	AD AD AD F0 AD AD 85 AD	56 54 51 00 08 50 53 36 14 22 00 28 21 18 23 17 25 22 A4 10 50 60 60 60 60 60 60 60 60 60 60 60 60 60	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL MUL2 MUL3	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X XTNDL+2,X MUL3 #\$03	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB37: FB36: FB40: FB44: FB48: FB48: FB49: FB49: FB51: FB53: FB57: FB57: FB58: FB58: FB57: FB66: FB67: FB68: FB68: FB68: FB67: FB68: FB67: FB68: FB67: FB68: FB67: FB68: FB67: FB68: FB67: FB68: FB67: FB68: FB68: FB67: FB68:	AD AD AP 85 AP	56 54 51 00 0B 53 36 14 22 00 20 22 23 21 25 22 24 10 50 50 50 60 60 60 60 60 60 60 60 60 60 60 60 60	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL MUL2 MUL3	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X XTNDL+2,X #\$03 \$76	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB37: FB40: FB49: FB49: FB49: FB49: FB51: FB57: FB57: FB57: FB58: FB59: FB60: FB66: FB66: FB67: FB68: FB67: FB78:	AD AD AP AD AP AD AD AP AD AP AD AP AD AP AD AP	56 54 51 00 0B 53 36 14 22 00 20 22 23 21 25 22 24 10 50 50 50 60 60 60 60 60 60 60 60 60 60 60 60 60	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL MUL2 MUL3	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X AUXL+2,X XTNDL+2,X XTNDL+2,X MUL3 #\$03 \$76 \$550	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB39: FB36: FB40: FB449: FB4B: FB4B: FB51: FB57: FB57: FB59: FB58: FB57: FB66: FB67: FB67: FB68: FB78: FB71: FB78: FB77: FB78:	AD AD AD 20 AD 85 AP 85	56 54 51 00 50 53 36 14 22 20 20 28 21 18 23 7 25 22 A4 10 50 6 54 F7 6 7 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL MUL2 MUL3	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X XTNDL+2,X XTNDL+2,X MUL3 #\$03 \$76 \$50 MUL5	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB39: FB37: FB40: FB449: FB4B: FB4B: FB4B: FB51: FB57: FB59: FB57: FB59: FB57: FB59: FB66: FB67: FB68: FB67: FB68: FB67: FB68:	AD AD AD 20 AD 85 AP 85 AP 85 AP 85 AP 65	56 54 51 00 50 53 36 14 22 20 20 28 21 18 23 7 25 22 A4 10 50 6 54 F7 6 7 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL MUL2 MUL3	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X XTNDL+2,X XTNDL+2,X MUL3 #\$03 \$76 \$50 MUL5	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD
FB33: FB36: FB39: FB36: FB40: FB449: FB4B: FB4B: FB51: FB57: FB57: FB59: FB58: FB57: FB66: FB67: FB67: FB68: FB78: FB71: FB78: FB77: FB78:	AD AD AD 20 AD 85 AP 85 AP 85 AP 85 AP 65	56 54 51 00 50 53 36 14 22 20 20 28 21 18 23 7 25 22 A4 10 50 6 54 F7 6 7 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7	CO CO CO CO F8	SETTXT SETGR SETWND TABV MULPM MUL MUL2 MUL3	LDA	LORES LOWSCR TXTSET #\$00 SETWND TXTCLR MIXSET CLRTOP #\$14 WNDDTOP #\$00 WNDLFT #\$28 WNDWDTH #\$18 WNDBTM #\$17 CV VTAB MD1 #\$10 ACL A MUL4 #\$FE XTNDL+2,X XTNDL+2,X XTNDL+2,X MUL3 #\$03 \$76 \$50 MUL5	INIT VIDEO MODE SET FOR TEXT MODE FULL SCREEN WINDOW SET FOR GRAPHICS MODE LOWER 4 LINES AS TEXT WINDOW SET FOR 40 COL WINDOW TOP IN A-REG, BTTM AT LINE 24 VTAB TO ROW 23 VTABS TO ROW IN A-REG ABS VAL OF AC AUX INDEX FOR 16 BITS ACX * AUX + XTND TO AC, XTND IF NO CARRY, NO PARTIAL PROD. ADD MPLCND (AUX) TO PARTIAL PROD

FB81:	20	70.4	DD.	DTMDM	TOD	MD1	ADO VAL OF AC ALLY
EDO 4							ABS VAL OF AC, AUX.
FB84:				DIV	TDX	#\$10	INDEX FOR 16 BITS
FB86:				DIV2			
FB88:						ACH	,
FB8A:					ROL	XTNDL	XTND/AUX
FB8C:	26	53			ROL	XTNDH	TO AC.
FB8E:	38				SEC		
FB8F:	Α5	52			LDA	XTNDL	
FB91:	E5	54			SBC	AUXL	MOD TO XTND.
FB93:	AA				TAX		
FB94:	Α5	53			LDA	XTNDH	
FB96:						AUXH	
FB98:					BCC		
FB9A:						XTNDL	
FB9C:						XTNDH	
FB9E:					INC		
				DTII		ACL	
FBA0:				DIV3	DEY	DILLO	
FBA1:					BNE	DIV2	
FBA3:					RTS		
FBA4:				MD1	LDY	#\$00 SIGN	ABS VAL OF AC, AUX
FBA6:	84	2F			STY	SIGN	WITH RESULT SIGN
FBA8:	A2	54			LDX	#AUXL	IN LSB OF SIGN.
FBAA:	20	AF	FB		JSR	MD3	
FBAD:	A2	50			LDX	#ACL	
FBAF:							X SPECIFIES AC OR AUX
FBB1:				1100	DDI	MDRTS	A DIBETTIBO NE OK NON
		UD				MDKIS	
FBB3:					SEC		
FBB4:					TYA		
FBB5:	F5	00			SBC	LOC0,X	COMPL SPECIFIED REG
FBB7:	95	00			STA	LOC0,X	IF NEG.
FBB9:	98				TYA		
FBBA:	F5	01			SBC	LOC1,X	
FBBC:						LOC1,X	
FBBE:						SIGN	
				MDDMG		SIGN	
FBC0:				MDRTS	RTS		
FBC1:				BASCALC			CALC BASE ADR IN BASL,H
FBC2:	4A				LSR	A	FOR GIVEN LINE NO
FBC3:	29	03			AND	#\$03	0<=LINE NO.<=\$17 ARG=000ABCDE, GENERATE
FBC5:	09	04			ORA	#\$04	ARG=000ABCDE, GENERATE
FBC7:	85	29				BASH	BASH=000001CD
FBC9:					PLA		AND
FBCA:						#\$18	BASL=EABAB000
							BASI-EABABOOO
FBCC:						BSCLC2	
FBCE:						#\$7F	
FBD0:	85	28		BSCLC2	STA	BASL	
FBD2:	0A				ASL		
FBD3:	0A				ASL		
FBD4:	05	28			ORA	BASL	
FBD6:						BASL	
FBD8:					RTS	21102	
FBD9:				ספדד1		#07	BELL CHAR? (CNTRL-G)
				БЕППТ			BELL CHAR: (CNIKL-G)
					BNE	RTS2B	NO DEMITEN
FBDB:							NO, RETURN
FBDD:	Α9	40			LDA	#\$40	NO, RETURN DELAY .01 SECONDS
FBDD: FBDF:	A9 20	40 A8	FC		JSR	WAIT	
FBDD: FBDF: FBE2:	A9 20 A0	40 A8 C0			JSR		
FBDD: FBDF: FBE2:	A9 20 A0	40 A8 C0		BELL2	JSR LDY	WAIT	
FBDD: FBDF: FBE2:	A9 20 A0 A9	40 A8 C0 0C		BELL2	JSR LDY LDA	WAIT #\$C0	DELAY .01 SECONDS
FBDD: FBDF: FBE2: FBE4: FBE6:	A9 20 A0 A9 20	40 A8 C0 0C A8	FC	BELL2	JSR LDY LDA JSR	WAIT #\$C0 #\$OC WAIT	DELAY .01 SECONDS TOGGLE SPEAKER AT
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9:	A9 20 A0 A9 20 AD	40 A8 C0 0C A8 30	FC	BELL2	JSR LDY LDA JSR LDA	WAIT #\$C0 #\$0C	DELAY .01 SECONDS TOGGLE SPEAKER AT
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC:	A9 20 A0 A9 20 AD 88	40 A8 C0 0C A8 30	FC	BELL2	JSR LDY LDA JSR LDA DEY	WAIT #\$C0 #\$OC WAIT SPKR	DELAY .01 SECONDS TOGGLE SPEAKER AT
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBED:	A9 20 A0 A9 20 AD 88 D0	40 A8 C0 0C A8 30	FC		JSR LDY LDA JSR LDA DEY BNE	WAIT #\$C0 #\$OC WAIT	DELAY .01 SECONDS TOGGLE SPEAKER AT
FBDD: FBDF: FBE2: FBE4: FBE6: FBEC: FBEC: FBED: FBEF:	A9 20 A0 A9 20 AD 88 D0 60	40 A8 C0 0C A8 30 F5	FC	RTS2B	JSR LDY LDA JSR LDA DEY BNE RTS	WAIT #\$C0 #\$OC WAIT SPKR BELL2	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC.
FBDD: FBDF: FBE2: FBE4: FBE6: FBEC: FBEC: FBED: FBEF:	A9 20 A0 A9 20 AD 88 D0 60 A4	40 A8 C0 0C A8 30 F5	FC	RTS2B	JSR LDY LDA JSR LDA DEY BNE RTS LDY	WAIT #\$C0 #\$0C WAIT SPKR BELL2	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBED: FBEF: FBF0: FBF2:	A9 20 A0 A9 20 AD 88 D0 60 A4 91	40 A8 C0 0C A8 30 F5	FC C0	RTS2B STOADV	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA	WAIT #\$C0 #\$OC WAIT SPKR BELL2 CH (BASL),Y	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBEC: FBED: FBEF: FBF0: FBF2:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6	40 A8 C0 0C A8 30 F5 24 28 24	FC C0	RTS2B	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBED: FBEF: FBF0: FBF2:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6	40 A8 C0 0C A8 30 F5 24 28 24	FC C0	RTS2B STOADV	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA	WAIT #\$C0 #\$OC WAIT SPKR BELL2 CH (BASL),Y CH	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT)
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBEC: FBED: FBEF: FBF0: FBF2:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5	40 A8 C0 0C A8 30 F5 24 28 24 24	FC C0	RTS2B STOADV	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA	WAIT #\$C0 #\$OC WAIT SPKR BELL2 CH (BASL),Y CH	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBEC: FBED: FBFF: FBF6: FBF6:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5 C5	40 A8 C0 0C A8 30 F5 24 28 24 24 21	FC C0	RTS2B STOADV	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP	WAIT #\$C0 #\$OC WAIT SPKR BELL2 CH (BASL),Y CH	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT)
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBED: FBEF0: FBF1: FBF4: FBF6: FBF8:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0	40 A8 C0 0C A8 30 F5 24 24 24 21 66	FC C0	RTS2B STOADV	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH?
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBED: FBEF: FBF0: FBF4: FBF6: FBF8: FBF8: FBF8: FBF8:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0 60	40 A8 C0 0C A8 30 F5 24 24 24 21 66	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO, RETURN
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBED: FBF2: FBF6: FBF6: FBF6: FBF8: FBFA: FBFA: FBFA: FBFA:	A9 20 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0 60 C9	40 A8 C0 0C A8 30 F5 24 28 24 24 21 66	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO, RETURN CONTROL CHAR?
FBDD: FBDF: FBE2: FBE4: FBE6: FBEC: FBED: FBF0: FBF6: FBF4: FBF6: FBFA: FBFA: FBFC: FBFC: FBFF:	A9 20 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0 C9 B0	40 A8 C0 OC A8 30 F5 24 24 24 21 66 A0 EF	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT.
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBED: FBF0: FBF6: FBF4: FBF4: FBFA: FBFC: FBFC: FBFF: FBFF: FBFF:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 B0 A8	40 A8 C0 OC A8 30 F5 24 24 24 21 66 A0 EF	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY INC LDA CMP BCS RTS CMP BCS TAY	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV	DELAY .01 SECONDS TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO, RETURN CONTROL CHAR? NO, OUTPUT IT. INVERSE VIDEO?
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBEF: FBF6: FBF6: FBF6: FBFA: FBFA: FBFF: FBFD: FBFF: FBFD: FBFF: FBFC:	A9 20 A9 20 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 B0 A8	40 A8 C0 OC A8 30 F5 24 22 21 66 A0 EF	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS RTS CMP BCS TAY BPL	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT.
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBEF: FBF0: FBF4: FBF6: FBF4: FBFA: FBFG: FBFD: FBFF: FBFF: FBFF: FBFF: FBFF:	A9 20 A0 A9 20 88 D0 60 A4 91 E6 A5 C5 B0 C9 B0 A8 10 C9	40 A8 C0 OC A8 30 F5 24 28 24 21 66 A0 EF EC 8D	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS RTS CMP BCS RTS CMP BCS RTS CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO, RETURN CONTROL CHAR? NO, OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR?
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBEF: FBF6: FBF6: FBF6: FBFA: FBFA: FBFF: FBFD: FBFF: FBFD: FBFF: FBFC:	A9 20 A0 A9 20 88 D0 60 A4 91 E6 A5 C5 B0 C9 B0 A8 10 C9	40 A8 C0 OC A8 30 F5 24 28 24 21 66 A0 EF EC 8D	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS RTS CMP BCS RTS CMP BCS RTS CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT.
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBEF: FBF0: FBF4: FBF6: FBF4: FBFA: FBFG: FBFD: FBFF: FBFF: FBFF: FBFF: FBFF:	A9 20 A0 A9 20 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 B0 A8 10 C9 F0	40 A8 C0 OC A8 30 F5 24 24 24 21 66 A0 EF EC 8D 5A	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS TAY BCS TAY BPL CMP BEQ	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO, RETURN CONTROL CHAR? NO, OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR?
FBDD: FBDF: FBE2: FBE4: FBE6: FBED: FBEC: FBFD: FBF7: FBF6: FBF6: FBF6: FBF6: FBFF: FBFF: FBFF: FBFF: FC01: FC02: FC04:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 B0 A8 10 C9 F0 C9	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CCH WNDWDTH CR #\$A0 STOADV \$TOADV #\$8D CR	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR?
FBDD: FBDF: FBE2: FBE4: FBE6: FBEC: FBED: FBFF0: FBF6: FBF4: FBFA: FBFA: FBFC: FBFF: FC01: FC02: FC04: FC06: FC08:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 B0 A8 10 C9 F0 F0	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 5A	FC CO	RTS2B STOADV ADVANCE RTS3	JSR LDY LDA JSR LDA DEY BNE RTS LDY INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BCS CMP BCS CMP BEQ CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV STOADV #\$8D CR #\$8A	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO, RETURN CONTROL CHAR? NO, OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT.
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBE7: FBF6: FBF6: FBF6: FBFA: FBFC: FBFF: FC01: FC02: FC04: FC06: FC06: FC08: FC06:	A9 20 A0 A9 20 AD 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 B0 A8 10 C9 F0 C9 F0 C9	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 5A 88	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP BEQ CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$8B CR #\$88	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO, RETURN CONTROL CHAR? NO, OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H)
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBE7: FBF0: FBF7: FBF6: FBF8: FBFA: FBFA: FBFC: FC01: FC06: FC08: FC08: FC0C: FC06:	A9 20 A0 A9 20 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 F0 C9 F0 C9	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 5A 88 C9	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP BEQ CMP BEQ CMP BNE	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$8A LF #\$88 BELL1	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL.
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBFD: FBF7: FBF6: FBF6: FBF7: FBF7: FBF0: FBF7: FC01: FC02: FC04: FC06: FC08: FC08: FC08: FC00:	A9 20 A0 A9 20 88 D0 60 A4 91 E6 A5 C5 B0 60 C9 F0 C9 F0 C9 C9 C9	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 8A C9 24	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS RTS CMP BCS CMP BEQ CMP	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$8A LF #\$88 BELL1 CH	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL. DECREMENT CURSOR H INDEX
FBDD: FBDF: FBE2: FBE4: FBE6: FBE9: FBEC: FBFF: FBF7: FBF6: FBF8: FBF6: FBF6: FBF6: FBFC: FBF0: FFFC0: FC02: FC04: FC06: FC08: FC07:	A9 20 A0 A9 20 88 D0 60 60 60 C9 B0 A8 10 C9 F0 C9 D0 C9	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 8C 9 24 ES EC 8D 5A 8C 8C 8C 8C 8C 8C 8C 8C 8C 8C 8C 8C 8C	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP BEQ CMP BNE CMP BEQ CMP BNE CMP BEQ CMP BNE	WAIT #\$C0 #\$OC WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$8A LF #\$88 BELL1 CH RTS3	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL. DECREMENT CURSOR H INDEX IF POS, OK. ELSE MOVE UP
FBDD: FBDF: FBE2: FBE4: FBE6: FBEC: FBEF0: FBFF0: FBFF2: FBFA: FBFC: FBFC: FBFC: FC01: FC04: FC06: FC08: FC06: FC06: FC06: FC06: FC16: FC11:	A9 20 A0 A9 20 88 060 A4 91 E6 A5 C5 B0 C9 F0 C9 F0 C9 F0 C9 A5 C5 A8	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 8A 8C 9 24 ES ES ES ES ES ES ES ES ES ES ES ES ES	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ LDA	WAIT #\$C0 #\$OC WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$8A LF #\$88 BELL1 CH RTS3 WNDWDTH	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL. DECREMENT CURSOR H INDEX
FBDD: FBDF: FBE4: FBE4: FBE6: FBEC: FBED: FBFF0: FBF6: FBF6: FBF6: FBF7: FC01: FC04: FC06: FC08: FC08: FC08: FC08: FC06: FC06: FC10: FC11:	A9 20 A0 A9 20 88 060 A4 91 E6 A5 C5 B0 C9 F0 C9 F0 C9 F0 C9 F0 C6 A5 S5	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 88 C9 24 24 21 21 21 21 21 21 21 21 21 21 21 21 21	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP BEQ CMP BEQ CMP BEQ CMP BNE BEQ CMP BNE BEQ CMP BNE BEQ CMP STA	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$8A LF #\$88 BELL1 CH RTS3 WNDWDTH CH	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL. DECREMENT CURSOR H INDEX IF POS, OK. ELSE MOVE UP SET CH TO WNDWDTH-1
FBDD: FBDF: FBE2: FBE4: FBE6: FBEC: FBEF0: FBFF0: FBFF2: FBFA: FBFC: FBFC: FBFC: FC01: FC04: FC06: FC08: FC06: FC06: FC06: FC06: FC16: FC11:	A9 20 A0 A9 20 88 060 A4 91 E6 A5 C5 B0 C9 F0 C9 F0 C9 F0 C9 F0 C6 A5 S5	40 A8 C0 OC A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 88 C9 24 24 21 21 21 21 21 21 21 21 21 21 21 21 21	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP BEQ CMP BEQ CMP BEQ CMP BNE BEQ CMP BNE BEQ CMP BNE BEQ CMP STA	WAIT #\$C0 #\$OC WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$8A LF #\$88 BELL1 CH RTS3 WNDWDTH	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL. DECREMENT CURSOR H INDEX IF POS, OK. ELSE MOVE UP
FBDD: FBDF: FBE4: FBE4: FBE6: FBEC: FBED: FBFF0: FBF6: FBF6: FBF6: FBF7: FC01: FC04: FC06: FC08: FC08: FC08: FC08: FC06: FC06: FC10: FC11:	A9 20 A0 A9 20 AB B0 60 A4 91 E6 A5 C5 B0 60 C9 B0 C9 F0 C9 D0 C9 F0 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9	40 A8 C0 0C A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 8A 8C 92 4 24 24 24 24 24 24 24 24 24 24 24 24	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP BEQ CMP BNE DEC BPL LDA STA DEC	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$88 ELL1 CH RTS3 WNDWDTH CH CH	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL. DECREMENT CURSOR H INDEX IF POS, OK. ELSE MOVE UP SET CH TO WNDWDTH-1
FBDD: FBDF: FBE2: FBE4: FBE6: FBE7: FBE7: FBF0: FBF6: FBF8: FBF6: FBF7: FC01: FC02: FC06: FC08: FC08: FC08: FC08: FC08: FC08: FC10: FC12: FC14: FC16: FC18:	A9 20 A0 A9 20 A0 60 A4 91 E6 A5 C5 B0 C9 B0 C9 F0 C9 D0 C9 D0 C6 10 A5 C5 C5 A5 C5 A5 C5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5	40 A8 C0 0C A8 30 F5 24 24 21 66 A0 EF EC 8D 5A 88 C9 24 24 22 24 22 24 24 24 24 26 26 26 26 26 26 26 26 26 26 26 26 26	FC CO	RTS2B STOADV ADVANCE RTS3 VIDOUT	JSR LDY LDA JSR LDA DEY BNE RTS LDY STA INC LDA CMP BCS RTS CMP BCS TAY BPL CMP BEQ CMP BOEC LDA	WAIT #\$C0 #\$0C WAIT SPKR BELL2 CH (BASL),Y CH CH WNDWDTH CR #\$A0 STOADV #\$8D CR #\$88 ELL1 CH RTS3 WNDWDTH CH CH	TOGGLE SPEAKER AT 1 KHZ FOR .1 SEC. CURSOR H INDEX TO Y-REG STORE CHAR IN LINE INCREMENT CURSOR H INDEX (MOVE RIGHT) BEYOND WINDOW WIDTH? YES CR TO NEXT LINE NO,RETURN CONTROL CHAR? NO,OUTPUT IT. INVERSE VIDEO? YES, OUTPUT IT. CR? YES. LINE FEED? IF SO, DO IT. BACK SPACE? (CNTRL-H) NO, CHECK FOR BELL. DECREMENT CURSOR H INDEX IF POS, OK. ELSE MOVE UP SET CH TO WNDWDTH-1 (RIGHTMOST SCREEN POS)

FC1E:					BCS	RTS4	IF TOP LINE THEN RETURN
FC20:					DEC		DEC CURSOR V-INDEX
FC22:				VTAB	LDA		GET CURSOR V-INDEX
FC24:	20	C1	FB	VTABZ			GENERATE BASE ADR
FC27:	65	20			ADC	WNDLFT	ADD WINDOW LEFT INDEX
FC29:	85	28			STA	BASL	TO BASL
FC2B:	60			RTS4	RTS		
FC2C:	49	C0		ESC1	EOR	#\$C0	ESC?
FC2E:	F0	28			BEQ	HOME	IF SO, DO HOME AND CLEAR
FC30:	69	FD			ADC	#\$FD	ESC-A OR B CHECK
FC32:	90	C0				ADVANCE	A, ADVANCE
FC34:						BS	B, BACKSPACE
FC36:							ESC-C OR D CHECK
FC38:						LF	C, DOWN
FC3A:						UP	D, GO UP
FC3C:							ESC-E OR F CHECK
FC3E:							E, CLEAR TO END OF LINE
					DNE	CLREOL RTS4	•
FC40:							NOT F, RETURN
FC42:				CLREOP			CURSOR H TO Y INDEX
FC44:				~~ ~~~	LDA	CV	CURSOR V TO A-REGISTER
FC46:				CLEOP1	PHA		SAVE CURRENT LINE ON STK
FC47:					JSR	VTABZ CLEOLZ	CALC BASE ADDRESS
FC4A:			FC				CLEAR TO EOL, SET CARRY
FC4D:	A0	00			LDY	#\$00	CLEAR FROM H INDEX=0 FOR REST
FC4F:	68				PLA		INCREMENT CURRENT LINE
FC50:	69	00			ADC	#\$00	(CARRY IS SET)
FC52:	C5	23			CMP	WNDBTM	DONE TO BOTTOM OF WINDOW?
FC54:	90	F0			BCC	CLEOP1	NO, KEEP CLEARING LINES
FC56:						VTAB	YES, TAB TO CURRENT LINE
FC58:				HOME			INIT CURSOR V
FC5A:				попь	STA		AND H-INDICES
FC5C:						#\$00	AND II INDICED
					STY		MILITAL CLEAR MO END OF DAGE
FC5E:							THEN CLEAR TO END OF PAGE
FC60:						CLEOP1	
FC62:				CR		#\$00	CURSOR TO LEFT OF INDEX
FC64:					STA	CH	(RET CURSOR H=0)
FC66:				LF	INC		INCR CURSOR V(DOWN 1 LINE)
FC68:	Α5	25			LDA	CV	
FC6A:	C5	23			CMP	WNDBTM	OFF SCREEN?
FC6C:	90	В6			BCC	VTABZ	NO, SET BASE ADDR
FC6E:	C6	25			DEC	CV	DECR CURSOR V (BACK TO BOTTOM)
FC70:	Α5	22		SCROLL	LDA	WNDTOP	START AT TOP OF SCRL WNDW
FC72:	48				PHA		
FC73:			FC			VTABZ	GENERATE BASE ADR
H'('''/6 •					T.D.Z	RAST.	COPY RASI. H
FC76:					LDA		COPY BASL,H
FC78:	85	2A			STA	BAS2L	COPY BASL,H TO BAS2L,H
FC78: FC7A:	85 A5	2A 29			STA LDA	BAS2L BASH	
FC78: FC7A: FC7C:	85 A5 85	2A 29 2B			STA LDA STA	BAS2L BASH BAS2H	TO BAS2L,H
FC78: FC7A: FC7C: FC7E:	85 A5 85 A4	2A 29 2B			STA LDA STA LDY	BAS2L BASH BAS2H	TO BAS2L,H INIT Y TO RIGHTMOST INDEX
FC78: FC7A: FC7C: FC7E: FC80:	85 85 85 A4 88	2A 29 2B			STA LDA STA LDY DEY	BAS2L BASH BAS2H	TO BAS2L,H
FC78: FC7A: FC7C: FC7E: FC80: FC81:	85 85 85 A4 88 68	2A 29 2B 21			STA LDA STA LDY DEY PLA	BAS2L BASH BAS2H WNDWDTH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82:	85 85 85 A4 88 68	2A 29 2B 21			STA LDA STA LDY DEY PLA ADC	BAS2L BASH BAS2H	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER
FC78: FC7A: FC7C: FC7E: FC80: FC81:	85 85 85 A4 88 68	2A 29 2B 21			STA LDA STA LDY DEY PLA ADC	BAS2L BASH BAS2H WNDWDTH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82:	85 85 84 88 68 69 C5	2A 29 2B 21 01 23			STA LDA STA LDY DEY PLA ADC CMP	BAS2L BASH BAS2H WNDWDTH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84:	85 85 84 88 68 69 C5 B0	2A 29 2B 21 01 23 0D			STA LDA STA LDY DEY PLA ADC CMP	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE?
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86:	85 85 84 88 68 69 C5 B0 48	2A 29 2B 21 01 23 0D			STA LDA STA LDY DEY PLA ADC CMP BCS PHA	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE?
FC78: FC7A: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89:	85 85 84 88 68 69 C5 B0 48 20	2A 29 2B 21 01 23 0D	FC		STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR)
FC78: FC7A: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89:	85 85 84 88 68 69 C5 B0 48 20	2A 29 2B 21 01 23 0D	FC		STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR)
FC78: FC7A: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89:	85 85 84 88 69 C5 B0 48 20 B1	2A 29 2B 21 01 23 0D 24 28 2A	FC	SCRL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC8C: FC8E:	85 A5 85 A4 88 69 C5 B0 48 20 B1 88	2A 29 2B 21 01 23 0D 24 28 2A	FC	SCRL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86: FC86: FC88: FC89: FC8C: FC8C:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10	2A 29 2B 21 01 23 0D 24 28 2A F9	FC	SCRL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC8C: FC8C: FC90: FC91:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10 30	2A 29 2B 21 01 23 0D 24 28 2A F9 E1	FC	SCRL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BMI	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN)
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC8C: FC8C: FC90: FC91:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10 30	2A 29 2B 21 01 23 0D 24 28 2A F9 E1	FC	SCRL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BMI	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN)
FC78: FC7A: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC89: FC89: FC89: FC90: FC91: FC93: FC97:	85 85 A4 88 69 C5 B0 48 20 B1 91 88 10 30 A0 20	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E	FC	SCRL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPLI BMI LDY JSR	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86: FC86: FC89: FC90: FC91: FC97: FC93: FC93: FC97: FC97:	85 85 A4 88 69 C5 B0 48 20 B1 91 88 10 30 A0 B0 B0	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86	FC	SCRL2	STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BML LDY JSR BLD LDY BML LDY JSR BCS	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET
FC78: FC7A: FC7C: FC80: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC8C: FC90: FC91: FC97: FC97: FC97: FC97:	85 85 A4 88 69 C5 B0 48 20 B1 91 88 10 30 A0 20 B0 A4	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24	FC	SCRL2 SCRL3 CLREOL	STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BMI LDY JSR BDL BMI LDY JSR BCS LDY	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLECLZ VTAB CH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE
FC78: FC7A: FC7C: FC80: FC81: FC81: FC82: FC86: FC88: FC89: FC89: FC91: FC91: FC91: FC97: FC97: FC9A: FC9A: FC9A:	85 85 A4 88 69 C5 B0 48 20 B1 91 88 10 20 B0 A0 20 B0 A4 A9	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0	FC	SCRL2 SCRL3 CLREOL CLEOLZ	STA LDA STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA BPL BMI LDY JSR LDY LDA LDY LDA LDY LDA	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX
FC78: FC7A: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC8C: FC90: FC91: FC91: FC97: FC97: FC97: FC9A: FC9C:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10 30 A0 20 B0 A4 A9 91	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BMI LDY JSR BCS LDY LDA STA STA	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7A: FC7C: FC80: FC81: FC82: FC84: FC86: FC86: FC89: FC89: FC97: FC97: FC97: FC97: FC9A: FC9C:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10 30 A0 20 B0 A4 A9 91 C8	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY LDA STA DEY LDY JSR BCS LDY JSR LDY	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX
FC78: FC7A: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC89: FC89: FC90: FC97: FC97: FC97: FC9A: FC96: FC97: FC97: FC97: FC97: FC97: FC97: FC97: FC97: FC97:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10 30 A0 20 B0 A4 A9 91 C8 C4	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPH BMI LDY JSR BCS LDY LDA STA INY CPY	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y WNDWDTH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7A: FC7C: FC80: FC81: FC82: FC84: FC86: FC86: FC89: FC89: FC97: FC97: FC97: FC97: FC9A: FC9C:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10 30 A0 20 B0 A4 A9 91 C8 C4	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPH BMI LDY JSR BCS LDY LDA STA INY CPY	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7A: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC89: FC89: FC90: FC97: FC97: FC97: FC9A: FC96: FC97: FC97: FC97: FC97: FC97: FC97: FC97: FC97: FC97:	85 85 84 88 69 C5 B0 48 20 B1 91 88 10 30 A0 20 B0 A4 A9 91 C8 C4 90	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28 21 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOLZ	STA LDA STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA LDA STA BPL BMI LDY JSR LDY LDA STA LDY LDA STA LDY LDA STA LDY LDA STA LOY CPY BCC RTS	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y WNDWDTH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7A: FC7C: FC80: FC81: FC82: FC84: FC86: FC86: FC89: FC90: FC97: FC97: FC97: FC97: FC97: FC97: FC97: FC98: FC97: FC98:	85 85 84 88 69 C5 B0 48 20 B1 88 10 30 A0 B0 A4 A9 91 C8 C8 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28 27 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOLZ	STA LDA STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA LDA STA BPL BMI LDY JSR LDY LDA STA LDY LDA STA LDY LDA STA LDY LDA STA LOY CPY BCC RTS	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y WNDWDTH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC89: FC90: FC91: FC97: FC97: FC9A: FC97: FC9A: FC97: FC9A: FC97: FC9A: FC9C: FC9A: FC9C: FC9A: FC9C: FC9A: FC9C: FC9A: FC9A: FC9C: FC9A: FC9C: FC9A: FC9A:	85 85 84 88 69 C5 B0 48 20 B1 88 10 30 A0 B0 A4 A9 91 C8 C8 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28 27 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOLZ	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BMI LDY JSR BCS LDY LDA STA TOPY BCS LDY LDA STA DEY BPL BMI LDY JSR BCS LDY LDA STA STA STA STA STA STA STA STA STA ST	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y WNDWDTH	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7A: FC7C: FC80: FC81: FC82: FC84: FC86: FC86: FC89: FC90: FC97: FC97: FC97: FC97: FC97: FC97: FC97: FC98: FC97: FC98:	85 85 84 88 69 C5 B0 88 10 30 A0 20 B1 A9 91 C4 A9 60 38 48	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 86 24 20 28 21 F9 E1 F9 E1 F9 E1 F9 E1 F9 E1 F9 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT	STA LDA STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA LDA STA BPL BMI LDY JSR LDY LDA STA LDY LDA STA LDY LDA STA LDY LDA STA LOY CPY BCC RTS	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLECLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC87: FC97:	85 85 84 88 69 69 60 81 91 88 10 30 80 80 80 80 80 80 80 80 80 8	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28 21 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BMI LDY JSR BCS LDY LDA STA INY CPY BCC RTS SEC PHA SBC	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y WNDWDTH CLEOL2 #\$1	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE'
FC78: FC7C: FC7C: FC80: FC80: FC81: FC82: FC84: FC86: FC89: FC87: FC97:	85 A5 88 88 69 69 88 10 30 A20 B1 88 10 30 A20 B1 A9 C8 C8 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9	2A 29 2B 21 01 23 0D 24 2A F9 E1 00 9E 86 24 A0 28 21 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL BMI LDY JSR BCS LDY LDA STA INY CPY BCC RTS SEC PHA SBC	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y WNDWDTH CLEOL2 #\$1	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH)
FC78: FC7A: FC7C: FC7E: FC80: FC81: FC82: FC84: FC86: FC89: FC89: FC97: FC97: FC97: FC9A: FC97: FC9A: FCA2: FCA2: FCA3: FCA5: FCA7: FCA6: FCA7: FCA6: FCA7: FCA6: FCA7: FCA6: FCA7: FCA6: FCA7: FCA7: FCA7: FCA7: FCA7: FCA8: FCA7: FCA8: FCA7:	85 85 84 88 69 C5 B0 48 20 81 91 30 A0 20 B0 A4 A9 C8 C4 90 60 38 E9 C6 60 60 60 60 60 60 60 60 60 60 60 60 60	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 24 A0 28 21 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDA STA LDA STA LDY BMI LDY LDA STA LDY LDA STA LDY STA LDY STA LDY STA LDY LDA STA LNY CPY BCC RTS SEC PHA SBC BNE PLA	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$10 WAIT3	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH)
FC78: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC87: FC91: FC91: FC91: FC91: FC92: FC97:	85 85 84 88 69 C5 B0 48 20 B1 91 810 30 A0 A0 B0 A4 A9 C8 C9 C9 C9 C9 C9 C9 C9 C9 C9 C9	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 28 24 A0 28 21 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDA STA LDA STA LDY BMI LDY JSR BCS LDY LDA STA INY CPY CPY RTS SEC PHA SBC SBC	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$1 WAIT3	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH)
FC78: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC87: FC97:	85 85 84 88 69 C5 B0 88 10 30 A0 20 A1 A9 91 C8 48 E9 D0 38 E9 D0 B0 B1 B1 B1 B1 B1 B1 B1 B1 B1 B1	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 86 24 A0 28 21 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDA STA LDY STA LDY JSR LDY JSR LDY JSR BCS ENT STA LDY STA LDY STA LDY LDY LDA STA INY CPY RCS STA STA INY CPY RCS STA STA STA STA STA STA STA STA STA ST	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$10 WAIT3	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH)
FC78: FC7A: FC7C: FC7C: FC80: FC80: FC81: FC82: FC84: FC86: FC89: FC89: FC97: FC97: FC97: FC9A: FC97: FC9A: FCA2: FCA3: FCA5: FCA7: FCA6: FCA7: FCA7: FCA7: FCA7: FCA8: FCA7: FCA8: FCA9:	85 85 86 86 86 86 86 86 86 81 81 81 81 81 81 81 81 81 81 81 81 81	2A 29 2B 21 01 23 0D 24 28 2A F9 E1 00 9E 864 2A 28 21 F9	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDA STA LDY DEY PLA ADC CMP BCS PHA JSR LDA STA DEY BPL LDA STA LDY LDA STA LDY BCS LDY BCC RTS STA LDY BCC RTS SEC PHA SBC BNE PLA SBC BNE PLA SBC RTS	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$40 (BASL),Y WNDWDTH CLEOL2 #\$11 WAIT3	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH) 1.0204 USEC (13+27/2*A+5/2*A*A)
FC78: FC7A: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC89: FC87: FC97: FC97: FC97: FC9A: FC97: FC9A: FCA7: FCA7: FCA7: FCA8: FCA7: FCA8: FCA9:	85 85 868 69 C5 B0 48 10 30 20 B1 48 40 20 60 60 60 60 60 60 60 60 60 60 60 60 60	2A 29 2B 21 01 23 0D 24 2A F9 E1 00 9E 824 A0 28 21 F9 01 FC 01 F6 42	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDA STA LDA STA LDA STA LDY BPL BMI LDY JSR LDA STA LDY STA LDY STA LDY STA LDY LDA STA LNY CPY BCC RTS SEC PHA SBC BNE PLA SBC BNE PLA SBC BNE PLA SBC BNE RTS INC	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$101 WAIT3 #\$01 WAIT2 A4L	INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH) 1.0204 USEC (13+27/2*A+5/2*A*A) INCR 2-BYTE A4
FC78: FC7A: FC7C: FC7C: FC80: FC80: FC81: FC82: FC84: FC86: FC89: FC89: FC97: FC97: FC97: FC9A: FC97: FC9A: FCA2: FCA3: FCA5: FCA7: FCA6: FCA7: FCA7: FCA8: FCA7: FCA8: FCA9:	85 85 86 86 86 86 86 86 86 86 86 86	2A 29 2B 21 01 23 0D 24 28 2A P9 E1 09 E86 24 A0 28 21 F9 01 F6 42 02	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDA STA LDA STA LDY BPL BMI LDY LDA STA LOY	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$101 WAIT3 #\$01 WAIT3 #\$01 WAIT2 A4L NXTA1	TO BAS2L,H INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH) 1.0204 USEC (13+27/2*A+5/2*A*A)
FC78: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC87: FC97:	85 85 86 86 86 86 86 86 86 86 86 86 86 86 86	2A 29 2B 21 01 23 0D 24 2A 2A F9 E1 00 28 24 A0 28 21 F9 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDA STA LDA STA LDA STA LDY BMI LDY JSR BCS LDY LDA STA INY CPY BCC RTS SEC PHA SBC BNE RTS SBC BNE RTS INC	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLECLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$1 WAIT3 #\$01 WAIT3 #\$01 WAIT3 #\$4 NXTA1 A4H	INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH) 1.0204 USEC (13+27/2*A+5/2*A*A) INCR 2-BYTE A4 AND A1
FC78: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC87: FC91: FC91: FC91: FC91: FC92: FC97: FC94: FC97: FC97: FC97: FC97: FC98: FC97: FC98: FC97: FC98: FC97: FC98:	85 85 85 86 86 86 86 81 81 81 81 81 81 81 81 81 81 81 81 81	2A 29 2B 21 01 23 0D 24 28 2A F9 86 24 20 28 21 F9 01 FC 01 F6 42 43 3C	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3 NXTA4	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDA STA LDY BMI LDY JSR BCS LDY LDA STA INY CPY EXTA INY CPY EXTA INY CPY EXTA SEC PHA SBC EXTS SEC PHA SBC EXTS SEC EXTS EXTS EXTS EXTS EXTS EXTS EXTS EXTS	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLECLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLECLZ #\$1 WAIT3 #\$01 WAIT3 #\$1 WAIT2 A4L NXTA1 A4H A1L	INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH) 1.0204 USEC (13+27/2*A+5/2*A*A) INCR 2-BYTE A4
FC78: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC86: FC89: FC87: FC97: FC97: FC97: FC9A: FC97: FC9A: FCA2: FCA3: FCA5: FCA7: FCA6: FCA7: FCA8: FCA7: FCA8: FCA7: FCA8: FCA9: FCA8: FCA9: FCA8: FCA9: FCA8: FCA9: FCA8:	85 85 86 86 86 86 86 86 81 81 81 81 81 81 81 81 81 81 81 81 81	2A 29 2B 21 01 23 0D 24 28 2A 29 86 24 A28 21 F9 01 FC 01 F6 42 33 35 35 35 35 36 36 36 36 36 36 36 36 36 36 36 36 36	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3 NXTA4	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDY JSR LDA STA LDY JSR LDY JSR BCS LDY JSR BCS LDY JSR BCS LDY JSR BCS LDY LDA STA INY CPY BCC PHA SBC BNE RTS SBC BNE RTS INC BNE RTS INC BNE CMP	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$01 WAIT3 #\$01 WAIT3 #\$01 WAIT2 A4L NXTA1 A4H A1L A2L	INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH) 1.0204 USEC (13+27/2*A+5/2*A*A) INCR 2-BYTE A4 AND A1 INCR 2-BYTE A1.
FC78: FC7C: FC7C: FC80: FC81: FC82: FC84: FC86: FC88: FC89: FC87: FC91: FC91: FC91: FC91: FC92: FC97: FC94: FC97: FC97: FC97: FC97: FC98: FC97: FC98: FC97: FC98: FC97: FC98:	85 85 86 86 86 86 86 86 81 81 81 81 81 81 81 81 81 81 81 81 81	2A 29 2B 21 01 23 0D 24 28 2A 29 86 24 A28 21 F9 01 FC 01 F6 42 33 35 35 35 35 36 36 36 36 36 36 36 36 36 36 36 36 36	FC	SCRL2 SCRL3 CLREOL CLEOLZ CLEOL2 WAIT WAIT2 WAIT3 NXTA4	STA LDA STA LDA STA LDY PLA ADC CMP BCS PHA JSR LDA STA LDY JSR LDA STA LDY JSR LDY JSR BCS LDY JSR BCS LDY JSR BCS LDY JSR BCS LDY LDA STA INY CPY BCC PHA SBC BNE RTS SBC BNE RTS INC BNE RTS INC BNE CMP	BAS2L BASH BAS2H WNDWDTH #\$01 WNDBTM SCRL3 VTABZ (BASL),Y (BAS2L),Y SCRL2 SCRL1 #\$00 CLEOLZ VTAB CH #\$A0 (BASL),Y WNDWDTH CLEOL2 #\$01 WAIT3 #\$01 WAIT3 #\$01 WAIT2 A4L NXTA1 A4H A1L A2L	INIT Y TO RIGHTMOST INDEX OF SCROLLING WINDOW INCR LINE NUMBER DONE? YES, FINISH FORM BASL,H (BASE ADDR) MOVE A CHR UP ON LINE NEXT CHAR OF LINE NEXT LINE (ALWAYS TAKEN) CLEAR BOTTOM LINE GET BASE ADDR FOR BOTTOM LINE CARRY IS SET CURSOR H INDEX STORE BLANKS FROM 'HERE' TO END OF LINES (WNDWDTH) 1.0204 USEC (13+27/2*A+5/2*A*A) INCR 2-BYTE A4 AND A1

FCC0:	F5	3 12			SBC	NOH.	
FCC2:					INC		(CARRY SET IF &qt=)
FCC4:						RTS4B	(Chaci bhi ii age-)
FCC4:					INC		
						AIH	
FCC8:	60			RTS4B	RTS		
FCC9:	ΑU	4B		HEADR	TDA	#\$4B	WRITE A*256 'LONG I'
FCCB:	20	DB	FC		JSR	ZERDLY	WRITE A*256 'LONG 1' HALF CYCLES (650 USEC EACH)
FCCE:	D0	F9			BNE	HEADR	(650 USEC EACH)
FCD0:	69	FΕ			ADC	#\$FE	
FCD2:	BU	F.2			BCS	HEADR	THEN A 'SHORT O'
FCD4:	Α0	21			LDY	#\$21	(400 USEC) WRITE TWO HALF CYCLES
FCD6:	20	DB	FC	WRBIT	JSR	ZERDLY	WRITE TWO HALF CYCLES
FCD9:					TNY		OF 250 USEC ('0')
FCDA:	C8				INY		OR 500 USEC ('0')
FCDB:				ZERDLY			
FCDC:						ZERDLY	
FCDE:					BCC	WRTAPE	Y IS COUNT FOR
FCE0:							TIMING LOOP
				ONEDLY		пүзд	TIMING EGGI
FCE3:						ONEDLY	
				WRTAPE	TDX	TAPEOUT	
FCE8:	Α0	2C			LDY	#\$2C	
FCEA:					DEX		
FCEB:					RTS		
FCEC:	Α2	08		RDBYTE	LDX	#\$08	8 BITS TO READ
FCEE:	48			RDBYT2	PHA		READ TWO TRANSITIONS
FCEF:	20	FA	FC		JSR	RD2BIT	(FIND EDGE)
FCF2:	68				PLA		
FCF3:					ROL		NEXT BIT
FCF4:					I.DV		COUNT FOR SAMPLES
						πγэд	COUNT FOR DAMFILED
FCF6:					DEX	DDDVmo	
FCF7:						RDBYT2	
FCF9:	60				RTS		
FCFA:	20	FD	FC	RD2BIT	JSR	RDBIT	
FCFD:	88			RDBIT	DEY		DECR Y UNTIL
FCFE:	AD	60	C0		LDA	TAPEIN	TAPE TRANSITION
FD01:	45	2F			EOR	LASTIN	
FD03:						RDBIT	
FD05:						LASTIN	
FD07:						LASTIN	
FD09:							SET CARRY ON Y
FD09:					RTS	#200	SEI CARRI ON I
						OTT	
				RDKEY			
FD0E:	В1	28			LDA		SET SCREEN TO FLASH
	В1	28			LDA PHA	(BASL),Y	SET SCREEN TO FLASH
FD0E:	B1 48	28			LDA	(BASL),Y	SET SCREEN TO FLASH
FD0E: FD10:	B1 48 29	28 3F			LDA PHA	(BASL),Y #\$3F	SET SCREEN TO FLASH
FD0E: FD10: FD11:	B1 48 29 09	28 3F 40			LDA PHA AND ORA	(BASL),Y #\$3F	SET SCREEN TO FLASH
FD0E: FD10: FD11: FD13:	B1 48 29 09	28 3F 40 28			LDA PHA AND ORA STA PLA	(BASL),Y #\$3F #\$40 (BASL),Y	
FD0E: FD10: FD11: FD13: FD15: FD17:	B1 48 29 09 91 68	28 3F 40 28			LDA PHA AND ORA STA PLA	(BASL),Y #\$3F #\$40 (BASL),Y	
FD0E: FD10: FD11: FD13: FD15: FD17: FD18:	B1 48 29 09 91 68 6C	28 3F 40 28	00		LDA PHA AND ORA STA PLA JMP	(BASL),Y #\$3F #\$40 (BASL),Y	SET SCREEN TO FLASH GO TO USER KEY-IN
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B:	B1 48 29 09 91 68 6C E6	28 3F 40 28 38 4E	00		LDA PHA AND ORA STA PLA JMP INC	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL	GO TO USER KEY-IN
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D:	B1 48 29 09 91 68 6C E6 D0	3F 40 28 38 4E 02	00		LDA PHA AND ORA STA PLA JMP INC BNE	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2	
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD1F:	B1 48 29 09 91 68 6C E6 D0 E6	3F 40 28 38 4E 02 4F	00	KEYIN	LDA PHA AND ORA STA PLA JMP INC BNE INC	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH	GO TO USER KEY-IN INCR RND NUMBER
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD1F: FD21:	B1 48 29 09 91 68 6C E6 D0 E6 2C	3F 40 28 38 4E 02 4F 00	00	KEYIN KEYIN2	LDA PHA AND ORA STA PLA JMP INC BNE INC BIT	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN?
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD1F: FD21: FD24:	B1 48 29 09 91 68 6C E6 D0 E6 2C	3F 40 28 38 4E 02 4F 00 F5	00	KEYIN KEYIN2	LDA PHA AND ORA STA PLA JMP INC BNE INC BIT BPL	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KBD KEYIN	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD1F: FD21: FD24: FD26:	B1 48 29 91 68 6C E6 D0 E6 2C 10 91	3F 40 28 38 4E 02 4F 00 F5 28	00	KEYIN KEYIN2	LDA PHA AND ORA STA PLA JMP INC BNE INC BIT BPL STA	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD1F: FD21: FD24: FD26: FD28:	B1 48 29 09 91 68 6C E6 D0 E6 2C 10 91 AD	3F 40 28 38 4E 02 4F 00 F5 28 00	00 C0	KEYIN KEYIN2	LDA PHA AND ORA STA PLA JMP INC BNE INC BIT BPL STA LDA	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD1F: FD21: FD24: FD26: FD28: FD28:	B1 48 29 91 68 6C E6 D0 E6 2C 10 91 AD	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10	00 C0	KEYIN KEYIN2	LDA PHA AND ORA STA PLA JMP INC BNE INC BIT BPL STA LDA BIT	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD21: FD24: FD24: FD26: FD28: FD28: FD28:	B1 48 29 09 91 68 6C E6 D0 E6 2C 10 91 AD 2C 60	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10	00 C0 C0	KEYIN KEYIN2	LDA PHA AND ORA STA JMP INC BNE INC BIT BPL STA LDA BIT RTS	#\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD29:	B1 48 29 09 91 68 6C E6 D0 E6 2C 10 91 AD 2C 60 20	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10	00 C0 C0 C0 FD	KEYIN KEYIN2 ESC	LDA PHA AND ORA STA JMP INC BNE INC BIT BPL STA LDA BIT RTS JSR	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD24: FD24: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD29:	B1 48 29 09 91 68 6C E6 2C 10 91 AD 2C 60 20 20	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C	CO CO CO FD FC	KEYIN KEYIN2 ESC	LDA PHA AND ORA PLA JMP INC BNE INC BSTA BIT BPL STA LDA BIT RTS JSR JSR	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC.
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD24: FD24: FD28: FD28: FD28: FD28: FD28: FD28: FD29:	B1 48 29 09 91 68 6C E6 2C 10 91 AD 2C 60 20 20	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C	CO CO CO FD FC FD	KEYIN KEYIN2 ESC	LDA PHA AND ORA PLA JMP INC BNE INC BSTA BIT BPL STA LDA BIT RTS JSR JSR	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC.
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD21: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD29: FD32: FD35: FD35: FD35: FD35:	B1 48 29 991 68 6C E6 D0 E6 2C 10 91 AD 20 20 20 C9	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B	CO CO FD FC FD	KEYIN KEYIN2 ESC	LDA PHA AND ORA STA PLA JMP INC BNE INC STA BIT BPL STA LDA BIT RTS JSR JSR JSR CMP	#\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC?
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD24: FD24: FD28: FD28: FD28: FD28: FD28: FD28: FD29:	B1 48 29 991 68 6C E6 D0 E6 2C 10 91 AD 20 20 20 C9	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B	CO CO FD FC FD	KEYIN KEYIN2 ESC	LDA PHA AND ORA STA PLA JMP INC BNE INC STA BIT BPL STA LDA BIT RTS JSR JSR JSR CMP	#\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC.
FD0E: FD10: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD21: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD29: FD32: FD35: FD35: FD35: FD35:	B1 48 29 991 68 6C E6 D0 E6 2C 10 91 AD 20 20 20 C9 F0	3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3	CO CO FD FC FD	KEYIN KEYIN2 ESC	LDA PHA AND ORA STA PLA JMP INC BNE INC STA BIT BPL STA LDA BIT RTS JSR JSR JSR CMP	#\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC?
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD1D: FD1D: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD35: FD35: FD35: FD35: FD36: FD38:	B1 48 29 91 68 6C E6 2C 10 91 AD 2C 60 20 20 C9 F0 60	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3	CO CO CO FD FC FD	KEYIN KEYIN2 ESC RDCHAR	LDA PHA AND ORA PLA JMP INC BNE STA BIT BPL STA LDA BIT RTS JSR JSR JSR JSR JSR PER STA LORP BEQ RTS	#\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC?
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD1D: FD1D: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD35: FD35: FD35: FD35: FD36: FD38:	B1 48 29 91 68 6C E6 D0 E6 2C 10 91 AD 20 20 20 C9 F0 60 A5	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3	CO CO CO FD FC FD	KEYIN KEYIN2 ESC RDCHAR	LDA PHA AND ORA PLA JMP INC BNE STA BIT BPL STA LDA BIT RTS JSR JSR JSR JSR JSR PER STA LORP BEQ RTS	#\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC?
FD0E: FD10: FD11: FD13: FD17: FD18: FD18: FD1B: FD1D: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD28: FD29: FD35: FD35: FD35: FD35: FD36: FD36: FD30:	B1 48 29 991 68 6C E6 D0 E6 2C 10 91 AD 2C 60 20 C9 F0 60 A5 48	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3	CO CO CO FD FC FFD	KEYIN KEYIN2 ESC RDCHAR	LDA PHA AND ORA PLA JMP INC BNE STA BIT BPL STA LDA BIT JSR	#\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC?
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD19: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD37:	B1 48 29 91 68 6C E6 D0 E6 2C 10 20 20 C9 F0 60 A5 48	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3 32 FF	CO CO CO FD FC FFD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BNE INC BIT BPL STA LDA BIT RTS JSR JSR CMP BEQ RTS LDA PHA LDA	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN
FD0E: FD10: FD11: FD15: FD17: FD18: FD18: FD19: FD21: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD32: FD35: FD35: FD35: FD35: FD35: FD36: FD37:	B1 48 29 991 68 6C E6 2C 10 91 AD 2C 60 20 20 20 48 A9 85	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3 32 FF 32	CO CO CO FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA STA PLA JMP INC BIT BPL STA LDA PLS JSR JSR JSR JSR JSR LDA PHA EDA PHA STA	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN RDKEY ESC1 RDKEY #\$9B ESC INVFLG	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE
FD0E: FD10: FD11: FD11: FD15: FD17: FD18: FD1B: FD1D: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD35: FD35: FD35: FD36: FD37:	B1 48 29 991 68 6C E6 D0 E6 2C 10 91 AD 2C 60 20 20 20 48 A9 85 BD	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3 32 00	00 C0 C0 FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA STA PLA JMP INC BIT BPL STA LDA BIT RTS JSR JSR JSR CMP BEQ RTS LDA PHA STA LDA LDA	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN RDKEY ESC1 RDKEY #\$9B ESC INVFLG #\$FF INVFLG IN,X	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN
FD0E: FD11: FD11: FD15: FD17: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD28: FD29: FD35: FD35: FD35: FD36: FD37:	B1 48 29 91 68 6C E6 D0 20 20 20 C9 F0 63 A5 A9 85 BD 20	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3 32 FF 32 00 ED	00 C0 C0 FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BNE BIT BPL STA LDA JSR JSR JSR JSR JSR LDA PHA LDA PHA LDA LDA PHA LDA LDA LDA PHA LDA LDA LDA JSR	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN RDKEY ESC1 RDKEY #\$9B ESC INVFLG	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD32: FD35: FD37: FD37: FD37: FD37: FD44: FD47: FD44: FD47: FD44:	B1 48 29 91 68 6C E6 2C 10 91 AD 2C 60 20 20 48 A9 85 BD 20 88 88 88 88 88 88 88 88 88 88 88 88 88	28 3F 40 28 38 4E 00 F5 28 00 10 0C 2C 0C 9B 32 00 ED	CO CO CO FD FC FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA LDA BIT JSR JSR JSR JSR JSR LDA PHA LDA LDA LDA LDA LDA LDA LDA LDA LDA LD	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN RDE KEYIN (BASL),Y KBD KEYIN (BASL),Y KBD KEYIN RDKEY ESC1 RDKEY #\$9B ESC INVFLG #\$FF INVFLG IN,X COUT	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD32: FD35: FD37: FD37: FD37: FD37: FD44: FD47: FD44: FD47: FD44:	B1 48 29 91 68 6C E6 2C 10 91 AD 2C 60 20 20 48 A9 85 BD 20 88 88 88 88 88 88 88 88 88 88 88 88 88	28 3F 40 28 38 4E 00 F5 28 00 10 0C 2C 0C 9B 32 00 ED	CO CO CO FD FC FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA STA PLA JMP INC BNE INC BIT BPL STA LDA BIT JSR JSR JSR JSR JSR LDA PHA LDA STA	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG #\$FF INVFLG IN,X COUT	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE
FD0E: FD10: FD11: FD11: FD15: FD17: FD18: FD18: FD19: FD21: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD35: FD35: FD35: FD37: FD37: FD37: FD44: FD47: FD44: FD47: FD48: FD48: FD48: FD49:	B1 48 29 91 68 6C E6 D0 E6 2C 10 20 20 C9 F0 60 A5 BD 85 BD	28 3F 40 28 38 4E 02 4F 00 00 2C 0C 9B F3 32 FF 32 00 ED 32 00 ED	000 C0 C0 C0 FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA STA PLA JMP INC BIT BPL STA BIT RTS JSR CMP BEQ RTS LDA STA LDA	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD1B: FD1D: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD37: FD37: FD37: FD37: FD37: FD37: FD40: FD40: FD40: FD44: FD47: FD48:	B1 48 29 91 66 E6 D0 E6 20 20 20 20 E6 48 BD 28 BD 29 85 BD C9	28 3F 40 28 38 4E 02 4F 00 00 2C 0C 9B F3 32 FF 32 00 ED 32 88	000 C0 C0 C0 FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA STA PLA JMP INC BIT BPL STA LDA BIT RTS JSR JSR JSR LDA PHA STA LDA PHA STA LDA PHA STA LDA STA LDA CMP	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN (BASL),Y KBD TINUFLG TINU	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD37: FD37: FD37: FD37: FD37: FD37: FD37: FD40: FD42: FD45:	B1 48 29 91 66C E6 D0 E6 20 20 20 C9 F0 68 5 BD 68 5 BD 67 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0	28 3F 40 28 38 4E 02 4F 00 00 2C 0C 9B F3 32 FF 32 00 ED 32 88 1D	000 C0 C0 FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA LDA JSR	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN5 KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X #\$88 BCKSPC	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD35: FD35: FD37: FD37: FD37: FD37: FD42: FD43: FD44: FD45: FD45: FD45: FD45: FD46: FD47: FD47: FD48: FD48: FD49:	B1 48 29 09 168 6C E6 D0 E6 2C 20 20 20 60 A5 48 BD 68 85 BD F0 C9	28 3F 40 28 38 4E 02 4F 00 5 28 00 10 0C 2CC 9B F3 32 F5 20 00 ED 32 98 1D 98	CO CO CO FD FC FD CFD 02	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA LDA JSR JSR JSR JSR JSR JSR LDA PHA LDA PHA LDA LDA PHA LDA LDA PHA LDA CMP BEQ CMP	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN2 KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X ENCEYPE #\$88 BCKSPC #\$98	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS
FD0E: FD11: FD11: FD13: FD15: FD17: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD37: FD37: FD37: FD37: FD37: FD37: FD37: FD40: FD42: FD45:	B1 48 29 09 168 6C E6 D0 E6 2C 20 20 20 60 A5 48 BD 68 85 BD F0 C9	28 3F 40 28 38 4E 02 4F 00 5 28 00 10 0C 2CC 9B F3 32 F5 20 00 ED 32 98 1D 98	CO CO CO FD FC FD CFD 02	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA LDA JSR JSR JSR JSR JSR JSR LDA PHA LDA PHA LDA LDA PHA LDA LDA PHA LDA CMP BEQ CMP	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN5 KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X #\$88 BCKSPC	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD35: FD35: FD37: FD37: FD37: FD37: FD42: FD43: FD44: FD45: FD45: FD45: FD45: FD46: FD47: FD47: FD48: FD48: FD49:	B1 48 29 09 168 6C E66 D0 20 20 20 20 20 48 A9 85 BD C9 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0	28 3F 40 28 38 4E 02 4F 00 F5 28 00 00 C 2C 0C 9B F3 32 FF 32 00 ED 32 00 8B 10 00 8B 10 10 10 10 10 10 10 10 10 10 10 10 10	CO CO CO FD FC FD CFD 02	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA LDA BIT RTS JSR JSR JSR JSR LDA PHA LDA STA LDA PHA LDA STA STA STA STA STA STA STA STA STA ST	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN2 KBD KEYIN (BASL),Y KBD KEYIN (BASL),Y KBD KEYIN INVELG IN,X INVELG IN,	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD38: FD38: FD38: FD37: FD38: FD39: FD40: FD41: FD41: FD42: FD42: FD44: FD45: FD46: FD46: FD47: FD48: FD48: FD49: FD59:	B1 48 29 09 168 6C E66 D0 E6C 20 20 20 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6	28 3F 40 28 38 4E 02 4F 00 F5 28 00 10 0C 2C 0C 9B F3 32 FF 32 00 ED 32 00 88 1D 8 1D 1 1D 1 1D 1 1D 1 1D 1 1D 1 1D 1 1 1D 1	000 C0 C0 FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA STA PLA JMP INC BIT BPL STA BIT RTS JSR CMP BEQ RTS LDA STA LDA CMP BEQ CMP BEQ CMP	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN2 KBD KEYIN (BASL),Y KBD KEYIN (BASL),Y KBD KEYIN INVELG IN,X INVELG IN,	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS BS, CTRL-X
FD0E: FD10: FD11: FD11: FD15: FD17: FD18: FD18: FD19: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD35: FD35: FD37: FD37: FD37: FD37: FD47: FD47: FD47: FD48: FD49:	B1 48 29 91 68 6C E6 20 20 20 20 60 A5 48 A9 5 BD 68 BD 68 BD 68 BD 69 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0	28 3F 40 28 38 4E 00 F5 28 00 0 C 2C 0C BF 3 3 2 F7 3 2 00 ED 3 2 00 8 1 00 00 8 1 00 00 00 00 00 00 00 00 00 00 00 00 0	000 C0 C0 FD FC FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA BIT BPL STA BIT RTS JSR JSR JSR JSR JSR JSR JSR LDA PHA LDA STA LDA PHA LDA STA LDA PHA STA LDA PHA STA LDA STA LDA PHA STA LDA STA LDA PHA STA LDA CMP BEQ CMP BEQ CMP BEQ CMP BEQ CPX BCC	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X #\$88 BCKSPC #\$98 CANCEL #\$F8	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS BS, CTRL-X
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD18: FD19: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD28: FD37: FD37: FD37: FD37: FD37: FD37: FD40: FD40: FD40: FD40: FD40: FD48: FD58: FD58: FD58: FD58: FD58:	B1 48 29 91 68 6C E6 D0 20 20 20 C7 60 A5 48 BD 68 85 BC9 F0 F0 F0 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6 E6	28 340 28 38 4E 00 F5 28 00 10 0C 2C 0C 9B 3 3 2 F7 2 00 ED 3 0 8 0 10 10 10 10 10 10 10 10 10 10 10 10 1	CO CO CO FD FC FD O2 FD CO FF FD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA BIT BPL STA BIT RTS JSR JSR JSR JSR JSR JSR JSR LDA PHA LDA STA LDA PHA LDA STA LDA PHA STA LDA PHA STA LDA STA LDA PHA STA LDA STA LDA PHA STA LDA CMP BEQ CMP BEQ CMP BEQ CMP BEQ CPX BCC	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN2 RNDH KBD KEYIN (BASL),Y KBD TINUTUR #\$9B ESC INVFLG IN,X COUT INVFLG IN,X #\$88 BCKSPC #\$98 CANCEL #\$F8 NOTCR1 BELL	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS BS, CTRL-X MARGIN?
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD18: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD37: FD37: FD37: FD37: FD37: FD37: FD37: FD40: FD50:	B1 48 29 91 68 6C E6 D0 20 20 20 20 20 48 85 BD 68 85 BD F0 F0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0	28 3F 40 28 38 4E 02 F5 28 00 10 0C 2C 0C 9B F3 32 F7 32 00 ED 32 00 81 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 98 00 81 00 80 00 80 00 80 00 80 00 80 00 80 00 80 00 80 00 0	CO CO CO FD FC FD O2 FFD	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BIT BPL STA LDA BIT RTS JSR JSR JSR JSR LDA PHA LDA PHA LDA PHA LDA PHA LDA PHA LDA STA LDA PHA LDA STA LDA PHA LDA STA LD	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KBDSTRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X COUT INVFLG IN,X ESCANCEL #\$F8 NOTCR1 BELL	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS BS, CTRL-X MARGIN? YES, SOUND BELL
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD18: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD35: FD36: FD37: FD37: FD37: FD37: FD38: FD37: FD38: FD37: FD38: FD48: FD48: FD58:	B1 48 29 09 168 6C E66 D0 20 20 20 20 48 85 BD C9 F0 85 BD F0 91 E0 90 E0 E0 90 E0 E0 E0 90 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0	28 3F 40 28 38 4E 02 4F 00 F5 28 00 00 C2C 00 C9B F3 32 FF 32 00 81 00 81 00 81 00 81 00 81 81 81 81 81 81 81 81 81 81 81 81 81	CO CO CO FD FC FD CO FF FF	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA PLA JMP INC BNE STA PLA JMP INC BIT BPL STA LDA RTS JSR JSR JSR JSR LDA PHA LDA STA STA STA STA LDA STA STA STA STA STA STA STA STA STA ST	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN (BASL),Y KBD TRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X #\$88 BCKSPC #\$98 CANCEL #\$F8 NOTCR1 BELL NXTCHAR	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS BS, CTRL-X MARGIN? YES, SOUND BELL ADVANCE INPUT INDEX
FD0E: FD11: FD11: FD13: FD17: FD18: FD18: FD18: FD24: FD24: FD26: FD28: FD28: FD28: FD28: FD35: FD36: FD37: FD37: FD37: FD37: FD38: FD37: FD38: FD37: FD38: FD48: FD48: FD58:	B1 48 29 91 68 6C E6 20 20 20 20 20 60 A5 48 5 BD F0 91 E0 91 E0 91 E0 91 E0 91 E0 91 E0 91 E0 91 E0 91 E0 91 E0 91 E0 91 E0 91 91 91 91 91 91 91 91 91 91 91 91 91	28 3F 40 28 4E 00 00 00 20 00 00 00 00 00 00	00 CO CO CO FD FC FD O2 FF	KEYIN KEYIN2 ESC RDCHAR NOTCR	LDA PHA AND ORA STA PLA JMP INC BIT BPL STA BIT RTS JSR CMP BEQ RTS LDA STA LDA LDA STA LDA STA LDA STA LDA LDA STA LDA STA LDA LDA STA LDA LDA STA LDA LDA STA LDA LDA LDA STA LDA LDA STA LDA LDA LDA STA LDA LDA LDA LDA LDA LDA LDA LDA LDA LD	(BASL),Y #\$3F #\$40 (BASL),Y (KSWL) RNDL KEYIN2 RNDH KBD KEYIN2 RNDH KBD KEYIN (BASL),Y KBD KEYIN (BASL),Y KBD TRB RDKEY ESC1 RDKEY #\$9B ESC INVFLG IN,X COUT INVFLG IN,X #\$88 BCKSPC #\$98 CANCEL #\$F8 NOTCR1 BELL NXTCHAR	GO TO USER KEY-IN INCR RND NUMBER KEY DOWN? LOOP REPLACE FLASHING SCREEN GET KEYCODE CLR KEY STROBE GET KEYCODE HANDLE ESC FUNC. READ KEY ESC? YES, DON'T RETURN ECHO USER LINE NON INVERSE CHECK FOR EDIT KEYS BS, CTRL-X MARGIN? YES, SOUND BELL

FD6A: A) 8E	FD	GETLNZ	JSR	CROUT PROMPT COUT	OUTPUT CR
	5 33 n = n	תב	GETLN	LDA	PROMPT	OUTPUT PROMPT CHAR
FD6C: 20				T'DX	#\$01	INIT INPUT INDEX
FD71: 82			BCKSPC	TXA	11401	WILL BACKSPACE TO 0
FD72: F	F3			BEQ	GETLNZ	
FD74: C				DEX		
			NXTCHAR	JSR	RDCHAR	
FD78: C				CMP	#PICK	USE SCREEN CHAR FOR CTRL-U
FD7A: DO				T.D.V	(BASL),Y	FOR CIRL-U
rb/c. b.				DDA	(DASH),I	
FD7E: C	9 E0		CAPTST	CMP	#\$E0	
FD80: 9	02					CONVERT TO CAPS
FD82: 2					#\$DF	
						ADD TO INPUT BUF
FD87: C:					#\$8D NOTCR	
FD8B: 20	9C	FC		JSR	CLREOL	CLR TO EOL IF CR
FD8E: A	9 8D		CROUT	LDA	#\$8D	
FD90: D) 5B			BNE	COUT	
			PRA1	LDY	A1H	PRINT CR,A1 IN HEX
FD94: A	3 3 C	מש	מעעמ	LDX	CDOUT	
FD99: 2	0 40	F9	PRYX2	JSR	PRNTYX	
FD9C: A				LDY	#\$00	
FD9E: A	AD			LDA	#\$00 #\$AD COUT	PRINT '-'
FDA0: 40				JMP	COUT	
			XAM8			CET TO EINICH AT
FDA5: 0:				STA		SET TO FINISH AT MOD 8=7
FDA9: A				LDA		FIGE 6-7
FDAB: 8	5 3F			STA	A2H	
FDAD: A	5 3C		MODSCHK	LDA	A1L	
FDAF: 2				AND		
FDB1: D	0 03	- III	XAM	BNE	DATAOUT	
			DATAOUT			
FDB8: 2			DITITIOUT	JSR	COUT	OUTPUT BLANK
FDBB: B				LDA	(A1L),Y	
FDBD: 2				JSR	PRBYTE	OUTPUT BYTE IN HEX
FDC0: 2					NXTA1	
FDC3: 90			RTS4C	BCC	MODSCHK	CHECK IF TIME TO, PRINT ADDR
FDC5: 60			XAMPM	T.SR	Α	DETERMINE IF MON
FDC7: 9				BCC	XAM	MODE IS XAM
FDC9: 42	A			LSR	A	ADD, OR SUB
FDCA: 42				LSR		
FDCB: A				LDA		
FDCD: 90				BCC		SUB: FORM 2'S COMPLEMENT
FDD1: 6			ADD	ADC	A1L	bob. Tolar 2 b com benent
FDD3: 48	3			PHA		
FDD4: A				T TO 7		
FDD6: 20	רוים כ			LDA	#\$BD	
		FD		JSR	#\$BD COUT	PRINT '=', THEN RESULT
FDD9: 68	3			JSR PLA	COUT	,
FDD9: 68	3			JSR PLA PHA	COUT	PRINT BYTE AS 2 HEX
FDD9: 68 FDDA: 48 FDDB: 42 FDDC: 42	3 A A			JSR PLA	COUT	,
FDD9: 68 FDDA: 48 FDDB: 48 FDDC: 48 FDDD: 48	3 3 A A A			JSR PLA PHA LSR LSR LSR	COUT A A A	PRINT BYTE AS 2 HEX
FDD9: 66 FDDA: 41 FDDB: 42 FDDC: 42 FDDD: 42 FDDE: 42	3 A A A A		PRBYTE	JSR PLA PHA LSR LSR LSR LSR	A A A A	PRINT BYTE AS 2 HEX
FDD9: 66 FDDA: 44 FDDB: 47 FDDC: 47 FDDD: 42 FDDE: 42 FDDF: 20	3 A A A A O E5		PRBYTE	JSR PLA PHA LSR LSR LSR LSR JSR	COUT A A A	PRINT BYTE AS 2 HEX
FDD9: 68 FDDA: 48 FDDB: 48 FDDC: 48 FDDD: 48 FDDE: 48 FDDF: 20 FDE2: 68	3 3 A A A A O E5	FD	PRBYTE	JSR PLA PHA LSR LSR LSR LSR JSR PLA	COUT A A A PRHEXZ	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG
FDD9: 66 FDDA: 44 FDDB: 47 FDDC: 47 FDDD: 42 FDDE: 42 FDDF: 20	3 3 4 4 4 4 0 E5 3 9 OF	FD	PRBYTE PRHEX	JSR PLA PHA LSR LSR LSR LSR LSR LSR ASR AND	COUT A A A PRHEXZ	PRINT BYTE AS 2 HEX
FDD9: 66 FDDA: 41 FDDB: 42 FDDC: 42 FDDD: 42 FDDF: 20 FDE2: 66 FDE3: 23	3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD	PRBYTE PRHEX PRHEXZ	JSR PLA PHA LSR LSR LSR LSR LSR LSR ASR AND	COUT A A A PRHEXZ #\$0F #\$B0	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG
FDD9: 66 FDDA: 44 FDDB: 47 FDDC: 47 FDDD: 47 FDDE: 47 FDDF: 26 FDE2: 66 FDE3: 27 FDE5: 07 FDE7: C7 FDE9: 96	3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD	PRBYTE PRHEX PRHEXZ	JSR PLA PHA LSR LSR LSR LSR LSR LSR AND ORA CMP BCC	A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDD: 44 FDDF: 26 FDE2: 66 FDE3: 22 FDE5: 03 FDE7: C5 FDE9: 96 FDEB: 66	3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD	PRBYTE PRHEX PRHEXZ	JSR PLA PHA LSR LSR LSR LSR LSR CSR LSR CMP BCC ADC	A A A A PRHEXZ #\$9F #\$B0 #\$BA COUT #\$06	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDD: 44 FDDF: 26 FDE2: 66 FDE3: 22 FDE5: 09 FDE7: C3 FDE9: 96 FDE9: 66 FDE9: 66	3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT	JSR PLA PHA LSR LSR LSR LSR LSR CMP BCC ADC JMP	A A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL)	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDD: 44 FDDF: 26 FDE2: 66 FDE3: 22 FDE5: 03 FDE7: C5 FDE9: 96 FDEB: 66	3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT	JSR PLA PHA LSR LSR LSR LSR PLA AND ORA CMP BCC ADC JMP CMP	A A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDE: 46 FDDE: 66 FDE2: 66 FDE3: 25 FDE5: 06 FDE7: C5 FDE9: 96 FDEB: 66 FDED: 66 FDF0: C5 FDF0: C5 FDF1: 26 FDF2: 96 FDF4: 25	33	FD 00	PRHEX PRHEXZ COUT COUT1	JSR PLA PHA LSR LSR LSR LSR JSR PLA AND ORA CMP BCC ADC JMP BCC AND	A A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDE: 44 FDDE: 66 FDE2: 66 FDE3: 29 FDE5: 09 FDE7: C5 FDE9: 96 FDE9: 66 FDE0: 66 FDE0: 66 FDE0: 68 FDE7: 99 FDE8: 69 FDE8: 69 FDE8: 69 FDE9: 99 FDE	33 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRHEX PRHEXZ COUT COUT1	JSR PLA PHA LSR LSR LSR LSR JSR PLA AND ORA CMP BCC ADC JMP CMP BCC ADC JMP CMP STY	A A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDF: 26 FDE2: 66 FDE3: 22 FDE5: 02 FDE7: 02 FDE9: 96 FDE9: 66 FDE0: 66 FDE0: 66 FDE0: 68 FDE0: 84 FDE0: 84 FDE0: 44	33 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR PLA AND ORA CMP BCC ADC JMP CMP BCC ADC JMP CMP BCC ADC JMP CMP BCC ADC JMP CMP BCC ADC ADC ADC ADC ADC ADC ADC ADC ADC A	COUT A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG
FDD9: 66 FDDA: 44 FDDD: 44 FDDD: 44 FDDF: 26 FDE2: 66 FDE3: 22 FDE5: 03 FDE7: 96 FDE7: 96 FDE9: 96 FDE	33 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR PLA AND ORA CMP BCC ADC JMP CMP BCC ADC JMP STY PHA JSR	COUT A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG OUTPUT A-REG AS ASCII
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDF: 26 FDE2: 66 FDE3: 22 FDE5: 02 FDE7: 02 FDE9: 96 FDE9: 66 FDE0: 66 FDE0: 66 FDE0: 68 FDE0: 84 FDE0: 84 FDE0: 44	33 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR PLA AND ORA CMP BCC ADC ADC ADC AND STY BCC AND STY JSR PHA JSR PLA	COUT A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDD9: 44 FDDF: 20 FDE2: 66 FDE3: 22 FDE5: 02 FDE7: C2 FDE9: 66 FDE9: 66 FDE9: 66 FDE9: 66 FDE9: 20 FDF4: 22 FDF4: 22 FDF6: 84 FDF9: 26 FDF7: 66	3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR JSR PLA AND ORA CMP BCC ADC JMP BCC ADC JMP BCC AND STY PHA JSR PLA LDY RTS	A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1 VIDOUT YSAV1	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG OUTPUT A-REG AS ASCII RESTORE A-REG
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDE: 44 FDDE: 66 FDE3: 25 FDE5: 05 FDE7: C5 FDE9: 66 FDE0: 66 FDE0: 66 FDE0: 66 FDE0: 66 FDF0: 67 FDF1: 66	33 34 44 44 45 46 47 47 47 47 47 47 47 47 47 47 47 47 47	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR JSR PLA AND ORA CMP BCC ADC JMP BCC ADC JMP BCC AND STY PHA JSR PLA LDY RTS DEC	A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1 VIDOUT YSAV1 YSAV	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG OUTPUT A-REG AS ASCII RESTORE A-REG AND Y-REG
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDF: 26 FDE2: 66 FDE3: 25 FDE5: 05 FDE7: 66 FDE7: 84 FDF6: 84 FDF7: 66 FDF7: 67 FDF	33 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR PLA AND ORA CMP BCC JMP BCC AND STY PHA JSR PLA LDY RTS DEC BEQ	AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG OUTPUT A-REG AS ASCII RESTORE A-REG AND Y-REG THEN RETURN
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDF: 26 FDE2: 66 FDE3: 22 FDE5: 03 FDE7: 66 FDE7: 66 FDE8: 63 FDE9: 94 FDE	3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR PLA AND ORA CMP BCC ADC ADC ADC AND STY BCC AND STY FLA LDY RTS DEC BEQ DEX	A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1 VIDOUT YSAV1 YSAV XAM8	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV Y-REG OUTPUT A-REG AS ASCII RESTORE A-REG AND Y-REG THEN RETURN
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDF: 26 FDE2: 66 FDE3: 25 FDE5: 05 FDE7: 66 FDE7: 84 FDF6: 84 FDF7: 66 FDF7: 67 FDF	3 3 4 4 3 5 6 3 4 4 3 5 6 6 3 4 4 5 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR JSR PLA AND ORA CMP BCC ADC JMP BCC AND STY PHA JSR PLA LDY RTS DEC DEC BEQ DEX BNE	A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1 VIDOUT YSAV1 YSAV1 XAM8 SETMDZ	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG OUTPUT A-REG AS ASCII RESTORE A-REG AND Y-REG THEN RETURN
FDD9: 66 FDDA: 41 FDDB: 42 FDDD: 42 FDDD: 42 FDDE: 66 FDE2: 66 FDE7: C2 FDE9: 96 FDE7: C3 FDE9: 96 FDE0: 66 FDF0: C3 FDF4: 22 FDF6: 88 FDF6: 88 FDF6: 66 FDF0: 66 FDF0: 66 FDF0: 66 FDF0: 66 FDF0: 66 FDF0: 67 FDF7: 66 FDF7: 67 FDF	3 3 3 4 4 3 5 3 4 4 3 5 5 3 4 5 5 4 5 6 5 4 6 6 6 6 6 6 6 6 6 6 6 6	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR JSR PLA AND ORA CMP BCC ADC JMP BCC AND STY PHA JSR PLA LDY RTS DEC BEQ DEX BNE CMP	A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1 VIDOUT YSAV1 YSAV1 XAM8 SETMDZ	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG OUTPUT A-REG AS ASCII RESTORE A-REG AND Y-REG THEN RETURN BLANK TO MON AFTER BLANK
FDD9: 66 FDDA: 44 FDDB: 44 FDDC: 44 FDDF: 26 FDE2: 66 FDE3: 25 FDE5: 66 FDE7: C5 FDE9: 96 FDE7: C5 FDE9: 96 FDE7: C6 FDF0: C6 FD0:	3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	FD 00	PRBYTE PRHEX PRHEXZ COUT COUT1 COUTZ	JSR PLA PHA LSR LSR LSR LSR JSR PLA AND ORA CMP BCC ADC JMP BCC AND STY PHA JSR PLA LDY RTS DEC BEQ DEX DEC BNE CMP	A A A A A PRHEXZ #\$0F #\$B0 #\$BA COUT #\$06 (CSWL) #\$A0 COUTZ INVFLG YSAV1 VIDOUT YSAV1 YSAV XAM8 SETMDZ #\$BA XAMPM MODE	PRINT BYTE AS 2 HEX DIGITS, DESTROYS A-REG PRINT HEX DIG IN A-REG LSB'S VECTOR TO USER OUTPUT ROUTINE DON'T OUTPUT CTRL'S INVERSE MASK WITH INVERSE FLAG SAV Y-REG SAV A-REG OUTPUT A-REG AS ASCII RESTORE A-REG AND Y-REG THEN RETURN BLANK TO MON AFTER BLANK DATA STORE MODE?

```
FE0F: 91 40
                         STA
                                (A3L),Y
                                           STORE AS LOW BYTE AS (A3)
FE11: E6 40
                         INC
                                A3L
FE13: D0 02
                          BNE
                                RTS5
                                           TNCR A3, RETURN
FE15: E6 41
                          INC
                                АЗН
FE17: 60
                RTS5
                          RTS
                                           SAVE CONVERTED ':', '+',
FE18: A4 34
                SETMODE
                         LDY
                                YSAV
FE1A: B9 FF 01
                               IN-1,Y
                                             '-', '.' AS MODE.
                          LDA
FE1D: 85 31
                SETMDZ
                               MODE
                         STA
FE1F: 60
                          RTS
FE20: A2 01
                _{
m LT}
                          LDX
                                #$01
FE22: B5 3E
                          LDA
                                A2L,X
                                           COPY A2 (2 BYTES) TO
                LT2
FE24: 95 42
                         STA
                               A4L,X
                                             A4 AND A5
FE26: 95 44
                          STA
                               A5L,X
FE28: CA
                         DEX
FE29: 10 F7
                          BPL
                               LT2
FE2B: 60
                          RTS
                                           MOVE (A1 TO A2) TO
FE2C: B1 3C
                MOVE
                         LDA
                                (A1L),Y
FE2E: 91 42
                         STA
                                (A4L),Y
                                             (A4)
FE30: 20 B4 FC
                          JSR
                               NXTA4
FE33: 90 F7
                          BCC
                                MOVE
FE35: 60
                          RTS
FE36: B1 3C
                VFY
                                (A1L),Y
                                           VERIFY (A1 TO A2) WITH
                          LDA
FE38: D1 42
                          CMP
                                (A4L),Y
                                             (A4)
FE3A: F0 1C
                                VFYOK
                         BEO
FE3C: 20 92 FD
                          JSR
                                PRA1
FE3F: B1 3C
                          LDA
                                (A1L),Y
FE41: 20 DA FD
                          JSR
                                PRBYTE
FE44: A9 A0
                          LDA
                                #$A0
FE46: 20 ED FD
                                COUT
                          JSR
FE49: A9 A8
                          LDA
                                #$A8
FE4B: 20 ED FD
                          JSR
                                COUT
FE4E: B1 42
                          LDA
                                (A4L),Y
FE50: 20 DA FD
                          JSR
                                PRBYTE
FE53: A9 A9
                         LDA
                                #$A9
FE55: 20 ED FD
                                COUT
                          JSR
FE58: 20 B4 FC VFYOK
                          JSR
                               NXTA4
FE5B: 90 D9
                          BCC
                                VFY
FE5D: 60
                          RTS
FE5E: 20 75 FE LIST
                          JSR
                               A1PC
                                           MOVE A1 (2 BYTES) TO
                                             PC IF SPEC'D AND
FE61: A9 14
                         LDA
                                #$14
FE63: 48
                LIST2
                                             DISEMBLE 20 INSTRS
                          PHA
FE64: 20 D0 F8
                          JSR
                                TNSTDSP
FE67: 20 53 F9
                          JSR
                                PCADJ
                                           ADJUST PC EACH INSTR
FE6A: 85 3A
                          STA
                                PCL
FE6C: 84 3B
                          STY
                                PCH
FE6E: 68
                          PLA
FE6F: 38
                          SEC
FE70: E9 01
                         SBC
                                #$01
                                           NEXT OF 20 INSTRS
FE72: D0 EF
                          BNE
                               LIST2
FE74: 60
                          RTS
                                           IF USER SPEC'D ADR
FE75: 8A
                A1PC
                         TXA
                               A1 PCRTS
                                             COPY FROM A1 TO PC
FE76: F0 07
                          BEO
FE78: B5 3C
                A1PCLP
                         LDA
                               A1L,X
FE7A: 95 3A
                          STA
                                PCL,X
FE7C: CA
                          DEX
FE7D: 10 F9
                               A1PCLP
                         BPL
FE7F: 60
                A1PCRTS
                         RTS
FE80: A0 3F
                                #$3F
                                           SET FOR INVERSE VID
                SETINV
                         LDY
FE82: D0 02
                          BNE
                                SETIFLG
                                            VIA COUT1
FE84: A0 FF
                SETNORM
                         LDY
                                           SET FOR NORMAL VID
                                #$FF
FE86: 84 32
                SETIFLG
                         STY
                                INVFLG
FE88: 60
                          RTS
FE89: A9 00
                SETKBD
                                           SIMULATE PORT #0 INPUT
                         LDA
                                #$00
FE8B: 85 3E
                INPORT
                          STA
                                A2L
                                             SPECIFIED (KEYIN ROUTINE)
FE8D: A2 38
                INPRT
                          LDX
                                #KSWL
FE8F: A0 1B
                          LDY
                                #KEYIN
FE91: D0 08
                          BNE
                                IOPRT
                SETVID
                                           SIMULATE PORT #0 OUTPUT
FE93: A9 00
                         LDA
                                #$00
FE95: 85 3E
                OUTPORT
                         STA
                                A2L
                                             SPECIFIED (COUT1 ROUTINE)
FE97: A2 36
                OUTPRT
                          LDX
                                #CSWL
FE99: A0 F0
                          LDY
                                #COUT1
                                           SET RAM IN/OUT VECTORS
FE9B: A5 3E
                IOPRT
                          LDA
                                A2L
FE9D: 29 OF
                                #$0F
                         AND
FE9F: F0 06
                                TOPRT1
                         BEO
FEA1: 09 C0
                         ORA
                                #IOADR/256
FEA3: A0 00
                          LDY
                                #$00
FEA5: F0 02
                                IOPRT2
                          BEQ
FEA7: A9 FD
                IOPRT1
                         LDA
                                #COUT1/256
                TOPRT2
FEA9: 94 00
                         STY
                                LOCO.X
FEAB: 95 01
                          STA
                                LOC1,X
FEAD: 60
                          RTS
FEAE: EA
                          NOP
FEAF: EA
                         NOP
FEBO: 4C 00 E0 XBASIC
                          JMP
                                           TO BASIC WITH SCRATCH
                                BASIC
FEB3: 4C 03 E0
                BASCONT
                         JMP
                                BASIC2
                                           CONTINUE BASIC
```

FEB6:	~ ~			ao.	TOD	3.100	ADD MO DO TH ODHOUD
							ADR TO PC IF SPEC'D
FEB9:					JSR		RESTORE META REGS
FEBC:					JMP		GO TO USER SUBR
				REGZ	JMP	REGDSP	TO REG DISPLAY
FEC2:	C6	34		TRACE	DEC	YSAV	
FEC4:	20	75	FE	TRACE STEPZ	JSR	A1PC	ADR TO PC IF SPEC'D
FEC7:	4C	43	FA		JMP	STEP	TAKE ONE STEP
FECA:	4C	F8	03	USR	JMP	USRADR	TO USR SUBR AT USRADR
FECD:	Α9	40		WRITE	LDA	#\$40	
FECF:							WRITE 10-SEC HEADER
FED2:					LDY	#627	WRITE TO DEC HEIDER
				LID 1			
FED4:					LDX		
FED6:					EOR	(A1L,X)	
FED8:	48				PHA		
FED9:	A1	3C			LDA	(A1L,X) WRBYTE	
FEDB:	20	ED	FE		JSR	WRBYTE	
FEDE:	20	ВΑ	FC			NXTA1	
FEE1:					LDY		
FEE3:					PLA	11422	
FEE4:					BCC	WD 1	
FEE6:					LDY		
FEE8:						WRBYTE	
FEEB:	F0	4D			BEQ	BELL	
FEED:	A2	10		WRBYTE	LDX	#\$10	
FEEF:	0A			WRBYT2			
FEF0:	20	D6			JSR	WRBIT	
FEF3:						WRBYT2	
FEF5:						WINDIIZ	
			222	CDMON7	RTS	DT 1	HANDLE A CR AS BLANK
			FE	CRMON			
FEF9:					PLA PLA		THEN POP STACK
FEFA:	68				PLA		AND RTN TO MON
FEFB:	D0	6C				MONZ	
FEFD:	20	FΑ	FC	READ	JSR	RD2BIT	FIND TAPEIN EDGE
FF00:	Α9	16			LDA	#\$16	
FF02:					TSR	HEADR	DELAY 3.5 SECONDS
FF05:					CTA		INIT CHKSUM=\$FF
					DIA	CHASUM	INII CHRSOM-SFF
FF07:					JSK	KDZBII	FIND TAPEIN EDGE LOOK FOR SYNC BIT (SHORT 0) LOOP UNTIL FOUND SKIP SECOND SYNC H-CYCLE
FF0A:				RD2	LDY	#\$24	LOOK FOR SYNC BIT
FF0C:	20	FD	FC		JSR	RDBIT	(SHORT 0)
FF0F:	B0	F9			BCS	RD2	LOOP UNTIL FOUND
FF11:	20	FD	FC		JSR	RDBIT	SKIP SECOND SYNC H-CYCLE
FF14:	A0	3B			LDY	#\$3B	INDEX FOR 0/1 TEST
FF16:				RD3	JISR		READ A BYTE
FF19:				100	CTA	(A1L,X)	STORE AT (A1)
							SIORE AI (AI)
FF1B:						CHKSUM	HIDDAME DIBBITUG GUIGUM
FF1D:							UPDATE RUNNING CHKSUM
FF1F:	20	BA	FC		JSR	NXTA1	INC A1, COMPARE TO A2
FF22:	A0	35			LDY	#\$35	COMPENSATE 0/1 INDEX
FF24:	90	F0			BCC	KD3	LOOP UNTIL DONE
FF26:	20	EC	FC		JSR	RDBYTE	READ CHKSUM BYTE
FF29:	C5	2E			CMP	CHKSUM	
FF2B:					BEO	BELL	GOOD, SOUND BELL AND RETURN
	10			PRERR	LDA	#\$C5	GOOD, GOOND DEED THE RETORN
	7\0						
FF2D:		C5		PKEKK			DDIAM HEDDH MILEN DELL
FF2D: FF2F:	20	C5 ED	FD	PREKK	JSR		PRINT "ERR", THEN BELL
FF2D: FF2F: FF32:	20 A9	C5 ED D2	FD		LDA	#\$D2	PRINT "ERR", THEN BELL
FF2D: FF2F:	20 A9	C5 ED D2	FD			#\$D2	PRINT "ERR", THEN BELL
FF2D: FF2F: FF32:	20 A9 20	C5 ED D2 ED	FD FD		LDA	#\$D2 COUT	PRINT "ERR", THEN BELL
FF2D: FF2F: FF32: FF34: FF37:	20 A9 20 20	C5 ED D2 ED ED	FD FD FD		LDA JSR JSR	#\$D2 COUT COUT	PRINT "ERR", THEN BELL OUTPUT BELL AND RETURN
FF2D: FF2F: FF32: FF34: FF37:	20 A9 20 20 A9	C5 ED D2 ED ED 87	FD FD FD	BELL	LDA JSR JSR	#\$D2 COUT COUT #\$87	
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C:	20 A9 20 20 A9 4C	C5 ED D2 ED ED 87 ED	FD FD FD	BELL	LDA JSR JSR LDA JMP	#\$D2 COUT COUT #\$87 COUT	
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF3F:	20 A9 20 20 A9 4C A5	C5 ED D2 ED ED 87 ED 48	FD FD FD	BELL	LDA JSR JSR LDA JMP LDA	#\$D2 COUT COUT #\$87 COUT	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS
FF2D: FF2F: FF32: FF34: FF37: FF3C: FF3F: FF41:	20 A9 20 20 A9 4C A5 48	C5 ED D2 ED ED 87 ED 48	FD FD FD	BELL	LDA JSR JSR LDA JMP LDA PHA	#\$D2 COUT COUT #\$87 COUT STATUS	OUTPUT BELL AND RETURN
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF3F: FF41: FF42:	20 A9 20 20 A9 4C A5 48 A5	C5 ED D2 ED 87 ED 48	FD FD FD	BELL RESTORE	LDA JSR JSR LDA JMP LDA PHA LDA	#\$D2 COUT COUT #\$87 COUT STATUS	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF3F: FF41: FF42: FF44:	20 A9 20 20 A9 4C A5 48 A5	C5 ED D2 ED 87 ED 48	FD FD FD	BELL RESTORE	LDA JSR JSR LDA JMP LDA PHA LDA LDA	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS
FF2D: FF32: FF32: FF34: FF37: FF3A: FF3C: FF3F: FF41: FF42: FF44: FF46:	20 A9 20 A9 4C A5 48 A5 A6 A4	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY	#\$D2 COUT COUT #\$87 COUT STATUS	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF41: FF44: FF44: FF46: FF48:	20 A9 20 A9 4C A5 48 A5 A6 A4	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE	LDA JSR JSR LDA JMP LDA PHA LDA LDA LDX LDY PLP	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS
FF2D: FF32: FF32: FF34: FF37: FF3A: FF3C: FF3F: FF41: FF42: FF44: FF46:	20 A9 20 A9 4C A5 48 A5 A6 A4	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF41: FF44: FF44: FF46: FF48:	20 A9 20 A9 4C A5 48 A5 A6 A4 28	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE RESTR1	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF3F: FF41: FF44: FF44: FF46: FF48: FF49:	20 A9 20 A9 4C A5 48 A5 A6 A4 28 60 85	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF34: FF37: FF3AC: FF3F: FF41: FF44: FF46: FF48: FF48: FF48: FF48:	20 A9 20 20 A9 4C A5 48 A5 A6 A4 28 60 85 86	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS STA STX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF41: FF42: FF44: FF46: FF48: FF49: FF49: FF48: FF48: FF48:	20 A9 20 A9 4C A5 48 A5 A6 A4 28 60 85 86	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS STA STX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF34: FF37: FF3C: FF3C: FF3F: FF41: FF44: FF44: FF48: FF48: FF48: FF48: FF48: FF48: FF48: FF48: FF48:	20 20 20 40 45 48 A5 A6 A4 28 60 85 86 84	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS STA STY PHP	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF34: FF37: FF3C: FF3F: FF41: FF42: FF44: FF46: FF48: FF48: FF48: FF48: FF48: FF48: FF48: FF48: FF48:	20 20 20 40 45 48 A5 A6 A4 28 60 85 86 84 08 68	C5 ED D2 ED 87 ED 48 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	JSR JSR LDA JMP LDA LDA LDA LDA LDX LDY PLP RTS STA STY PHP PLA	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF34: FF37: FF3AC: FF3F: FF41: FF42: FF44: FF46: FF48:	20 20 20 42 45 48 A5 A6 A4 28 60 85 86 88 88 85	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	LDA JSR JSR JSR LDA PHA LDA LDA LDY PLP RTS STA STX STY PHP PLA STA	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF37: FF37: FF3C: FF3F: FF41: FF44: FF46: FF48: FF48: FF48: FF48: FF50: FF50: FF55: FF51:	20 A9 20 A9 4C A5 48 A5 A6 A4 28 60 85 86 88 85 BA	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	LDA JSR JSR LDA LDA PHA LDA LDA LDX LDY RTS STA STX STY PHP PLA STA TSX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF37: FF3A: FF3C: FF3F: FF41: FF44: FF46: FF48:	20 A9 20 A9 4C A5 48 A5 A6 A4 28 60 85 86 85 BA 86	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS STA STX STY PHP PLA STA TSX STX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF37: FF37: FF3C: FF3F: FF41: FF44: FF46: FF48: FF48: FF48: FF48: FF50: FF50: FF55: FF51:	20 A9 20 A9 4C A5 48 A5 A6 A4 28 60 85 86 85 BA 86	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE	LDA JSR JSR LDA LDA PHA LDA LDA LDX LDY RTS STA STX STY PHP PLA STA TSX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF37: FF3A: FF3C: FF3F: FF41: FF44: FF46: FF48:	20 20 20 4C A5 48 A5 A6 A4 28 60 85 86 88 85 BA 86 D8	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48	FD FD FD	BELL RESTORE RESTR1 SAVE	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS STA STX STY PHP PLA STA TSX STX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF34: FF37: FF37: FF3C: FF3F: FF41: FF44: FF46: FF48: FF48: FF48: FF50: FF51: FF55: FF55: FF55: FF55: FF55:	20 20 20 4C A5 48 A5 A6 A4 28 60 85 86 88 85 BA 86 D8	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47	FD FD FD	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA JMP LDA PHA LDA LDY PLP RTS STA STX STY PHP PLA STA TSX CLD RTS	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE
FF2D: FF2F: FF32: FF34: FF37: FF3C: FF3F: FF41: FF42: FF44: FF46: FF48: FF48: FF45: FF50: FF51: FF55:	20 A9 20 20 A9 4C A5 48 A5 A6 A4 28 60 85 86 88 85 BA 86 D8 60 20	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 48	FD FD FD FD	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR JSR LDA PHA LDA LDA LDY PLP RTS STA STX STY PHP PLA STA TSX STX CLD JSR	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS
FF2D: FF2F: FF32: FF34: FF37: FF3AC: FF3F: FF41: FF44: FF44: FF48: FF48: FF50: FF55:	20 A9 20 20 A9 4C A5 48 A5 A6 A4 28 60 85 86 88 BA 86 D8 60 20 20 20 20 20 20 20 20 20 20 20 20 20	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F	FD FD FD FD	BELL RESTORE RESTR1 SAVE SAVI	LDA JSR JSR JSR LDA PHA LDA LDA LDY PLP RTS STA STX STY STY PHP PLA STA TSX STX CLD JSR JSR	#\$D2 COUT COUT #\$87 COUT \$\$TATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN
FF2D: FF2F: FF32: FF37: FF3A: FF3C: FF3F: FF44: FF46: FF48: FF48: FF48: FF50: FF55:	20 A9 20 A9 4C A5 48 A5 A6 A4 28 60 85 86 88 BA 86 D8 60 20 20 20 20 20 20 20 20 20 20 20 20 20	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F 93	FD FD FD FFE FB FE	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA PHA LDA LDA LDX LDY PLP RTS STA STX STY PHP A STA TSX STX CLD RTS JSR JSR	#\$D2 COUT COUT #\$87 COUT \$\$\$7 COUT \$\$\$7 COUT \$\$TATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT SETVID	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS
FF2D: FF2F: FF32: FF34: FF37: FF3A: FF3C: FF3F: FF44: FF46: FF48: FF48: FF48: FF50: FF55: FF57: FF55: FF57: FF58: FF59: FF59: FF56: FF56: FF56: FF56: FF56:	20 A9 20 A9 4C A5 48 A5 A6 A4 28 60 85 86 88 BA BA BA BA BA BA BA BA BA BA BA BA BA	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F 93 89	FD FD FD FD FE FE FE FE	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA JMP LDA PHA LDA LDX LDY PLP RTS STA STX STY PHP PLA STA TSX STX CLD RTS JSR JSR JSR	#\$D2 COUT COUT #\$87 COUT \$\$\$7 COUT \$\$\$7 COUT \$\$TATUS ACC XREG YREG ACC XREG YREG STATUS STATUS SPNT SETNORM INIT SETVID SETKBD	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN AS I/O DEV'S
FF2D: FF2F: FF32: FF34: FF37: FF37: FF37: FF41: FF44: FF46: FF48: FF48: FF48: FF50: FF55: FF57: FF57: FF58: FF57: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF56:	20 A9 20 A9 4C A5 A6 A6 85 86 88 BA 86 BB BB BB BB BB BB BB BB BB BB BB BB BB	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F 93 89	FD FD FD FD	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA JMP LDA PHA LDA LDY PLP RTS STA STX PHP PLA STX CLD RTS JSR JSR JSR CLD	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT SETVID SETKBD	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN
FF2D: FF2F: FF32: FF34: FF37: FF37: FF37: FF41: FF42: FF44: FF46: FF48: FF50: FF51: FF55: FF55: FF55: FF55: FF55: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF66:	20 A9 20 A9 4C A5 A6 A4 28 60 85 86 BA 60 20 20 20 D8 20	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F 93 89	FD FD FD FD FE FE FB FE FE FF	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA JMP LDA PHA LDA LDY PLP RTS STA STX PHP PLA STX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT SETVID SETKBD BELL	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN AS I/O DEV'S MUST SET HEX MODE!
FF2D: FF2F: FF32: FF34: FF37: FF37: FF37: FF41: FF42: FF44: FF46: FF48: FF50: FF51: FF55: FF55: FF55: FF55: FF55: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF66:	20 A9 20 A9 4C A5 A6 A4 28 60 85 86 BA 60 20 20 20 D8 20	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F 93 89	FD FD FD FD FE FE FB FE FE FF	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA JMP LDA PHA LDA LDY PLP RTS STA STX PHP PLA STX	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT SETVID SETKBD BELL	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN AS I/O DEV'S
FF2D: FF2F: FF32: FF34: FF37: FF37: FF37: FF41: FF42: FF44: FF46: FF48: FF50: FF51: FF55: FF55: FF55: FF55: FF55: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF56: FF66:	20 A9 20 A9 4C A5 A6 A4 28 60 85 86 BA 60 20 20 20 D8 20 A9	C5 ED D2 ED 87 ED 48 45 46 47 48 49 84 2F 93 89 3A	FD FD FD FD FE FE FE FE FE	BELL RESTORE RESTR1 SAVE SAV1 RESET MON MONZ	LDA JSR JSR LDA JMP LDA PHA LDA LDY PLP RTS STA STX STY PHP PLA STA TSX STX CLD JSR JSR JSR LDA	#\$D2 COUT COUT #\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT SETVID SETKBD BELL	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN AS I/O DEV'S MUST SET HEX MODE!
FF2D: FF2F: FF32: FF34: FF37: FF37: FF37: FF41: FF42: FF44: FF46: FF48: FF48: FF45: FF55: FF55: FF55: FF57: FF58: FF57: FF58: FF57: FF58: FF57: FF58: FF57: FF58: FF57: FF58: FF59: FF57: FF58: FF57: FF58: FF57: FF58: FF57: FF58: FF57: FF58: FF59: FF57: FF58: FF59: FF57: FF58: FF59: FF57: FF58: FF59: FF58: FF59: FF65: FF66: FF66: FF66:	20 A9 20 A9 4C A5 A6 A4 28 60 85 BA 60 20 20 20 20 20 85 85 86 85 86 86 86 86 86 86 86 86 86 86 86 86 86	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F 93 89 3AA 33	FD FD FD FE FE FE FE FE FF	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA PHA LDA LDA LDY PLP RTS STA STX STX STX STX STX STX STX STX CLD JSR JSR JSR JSR JSR LDA STA	#\$D2 COUT COUT #\$87 COUT \$\$87 COUT \$\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT SETVID SETKBD BELL #\$AA PROMPT	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN AS I/O DEV'S MUST SET HEX MODE!
FF2D: FF2F: FF32: FF37: FF3A: FF3C: FF3F: FF44: FF44: FF46: FF48: FF48: FF50: FF55: FF55: FF55: FF55: FF55: FF56: FF56: FF56: FF66: FF66: FF66: FF66: FF66:	20 A9 20 A9 4C A5 A6 A4 28 60 85 BA 60 20 20 20 20 20 85 85 86 85 86 86 86 86 86 86 86 86 86 86 86 86 86	C5 ED D2 ED 87 ED 48 45 46 47 45 46 47 48 49 84 2F 93 89 3AA 33	FD FD FD FE FE FE FE FE FF	BELL RESTORE RESTR1 SAVE SAV1	LDA JSR JSR LDA PHA LDA LDA LDY PLP RTS STA STX STX STX STX STX STX STX STX CLD JSR JSR JSR JSR JSR LDA STA	#\$D2 COUT COUT #\$87 COUT \$\$87 COUT \$\$87 COUT STATUS ACC XREG YREG ACC XREG YREG STATUS SPNT SETNORM INIT SETVID SETKBD BELL #\$AA PROMPT	OUTPUT BELL AND RETURN RESTORE 6502 REG CONTENTS USED BY DEBUG SOFTWARE SAVE 6502 REG CONTENTS SET SCREEN MODE AND INIT KBD/SCREEN AS I/O DEV'S MUST SET HEX MODE! '*' PROMPT FOR MON

```
FF70: 20 C7 FF
                          JSR
                                ZMODE
                                            CLEAR MON MODE, SCAN IDX
FF73: 20 A7 FF NXTITM
                                            GET ITEM, NON-HEX
                          JSR
                                GETNUM
FF76: 84 34
                          STY
                                YSAV
                                              CHAR IN A-REG
FF78: A0 17
                          LDY
                                 #$17
                                              X-REG=0 IF NO HEX INPUT
                 CHRSRCH
FF7A: 88
                          DEY
FF7B: 30 E8
                          BMI
                                MON
                                            NOT FOUND, GO TO MON
FF7D: D9 CC FF
                          CMP
                                CHRTBL.Y
                                            FIND CMND CHAR IN TEL
FF80: D0 F8
                                CHRSRCH
                          BNE
FF82: 20 BE FF
                          JSR
                                TOSUB
                                            FOUND, CALL CORRESPONDING
FF85: A4 34
                          LDY
                                YSAV
                                              SUBROUTINE
FF87: 4C 73 FF
                          JMP
                                NXTITM
FF8A: A2 03
                DIG
                          LDX
                                #$03
FF8C: 0A
                          ASL
                                Α
FF8D: 0A
                          ASL
                                Α
                                            GOT HEX DIG,
FF8E: 0A
                          ASL
                                Α
                                              SHIFT INTO A2
FF8F: 0A
                          ASL
                                Α
FF90: 0A
                NXTBIT
                          ASL
                                Α
FF91: 26 3E
                          ROL
                                A2L
FF93: 26 3F
                          ROL
                                A2H
FF95: CA
                          DEX
                                            LEAVE X=$FF IF DIG
FF96: 10 F8
                          BPL
                                NXTBIT
                NXTBAS
FF98: A5 31
                          LDA
                                            IF MODE IS ZERO
FF9A: D0 06
                          BNE
                                NXTBS2
FF9C: B5 3F
                                A2H.X
                                             THEN COPY A2 TO
                          LDA
FF9E: 95 3D
                          STA
                                A1H,X
                                             A1 AND A3
FFA0: 95 41
                          STA
                                A3H,X
FFA2: E8
                NXTBS2
                          INX
FFA3: F0 F3
                          BEQ
                                NXTBAS
FFA5: D0 06
                          BNE
                                NXTCHR
                GETNUM
                                            CLEAR A2
FFA7: A2 00
                          LDX
                                #$00
FFA9: 86 3E
                          STX
                                A2L
FFAB: 86 3F
                          STX
                                A2H
FFAD: B9 00 02 NXTCHR
                          LDA
                                IN,Y
                                            GET CHAR
FFB0: C8
                          INY
FFB1: 49 B0
                                #$B0
                          EOR
FFB3: C9 0A
                          CMP
                                 #$0A
FFB5: 90 D3
                          BCC
                                DIG
                                            IF HEX DIG, THEN
FFB7: 69 88
                          ADC
                                 #$88
FFB9: C9 FA
                          CMP
                                #$FA
FFBB: B0 CD
                          BCS
                                DIG
FFBD: 60
                          RTS
FFBE: A9 FE
                TOSUB
                          LDA
                                #GO/256
                                            PUSH HIGH-ORDER
FFC0: 48
                          PHA
                                              SUBR ADR ON STK
                                            PUSH LOW-ORDER
FFC1: B9 E3 FF
                          LDA
                                SUBTBL,Y
FFC4: 48
                          PHA
                                              SUBR ADR ON STK
FFC5: A5 31
                          LDA
                                MODE
                                            CLR MODE, OLD MODE
FFC7: A0 00
                ZMODE
                          LDY
                                #$00
FFC9: 84 31
                          STY
                                MODE
                                              TO A-REG
FFCB: 60
                                             GO TO SUBR VIA RTS
                          RTS
FFCC: BC
                CHRTBL
                                            F("CTRL-C")
                                 $BC
                          DFB
                                            F("CTRL-Y")
FFCD: B2
                          DFB
                                 SB2
FFCE: BE
                          DFB
                                 SBE
                                            F("CTRL-E")
FFCF: ED
                          DFB
                                 $ED
                                            F("T")
                                            F("V")
FFD0: EF
                          DFB
                                 $EF
                                            F("CTRL-K")
                          DFB
                                 $C4
FFD1: C4
                          DFB
                                SEC
                                            F("S")
FFD2: EC
                                            F("CTRL-P")
FFD3: A9
                          DFB
                                 $A9
FFD4: BB
                          DFB
                                 $BB
                                            F("CTRL-B")
                          DFB
                                            F("-")
FFD5: A6
                                 $A6
                                            F("+")
FFD6: A4
                          DFB
                                 $A4
                          DFB
                                $06
                                            F("M") (F=EX-OR $B0+$89)
FFD7: 06
                                            F("&lt")
FFD8: 95
                          DFB
                                $95
FFD9: 07
                          DFB
                                 $07
                                            F("N")
FFDA: 02
                          DFB
                                 $02
                                            F("I")
FFDB: 05
                          DFB
                                            F("L")
FFDC: F0
                          DFB
                                 $F0
                                            F("W")
                                            F("G")
FFDD: 00
                          DFB
                                $00
FFDE: EB
                          DFB
                                 ŚEB
                                            F("R")
FFDF: 93
                          DFB
                                 $93
                                            F(":")
                          DFB
                                 $A7
                                            F(".")
FFE0: A7
FFE1: C6
                          DFB
                                 $C6
                                            F("CR")
                                            F(BLANK)
FFE2: 99
                          DFB
                                $99
FFE3: B2
                SUBTBL
                          DFB
                                BASCONT-1
FFE4: C9
                          DFB
                                USR-1
FFE5: BE
                          DFB
                                REGZ-1
FFE6: C1
                          DFB
                                TRACE-1
FFE7: 35
                          DFB
                                VFY-1
FFE8: 8C
                          DFB
                                INPRT-1
FFE9: C3
                          DFB
                                STEPZ-1
FFEA: 96
                          DFB
                                OUTPRT-1
FFEB: AF
                          DFB
                                XBASIC-1
FFEC: 17
                          DFB
                                SETMODE-1
                          DFB
                                SETMODE-1
FFED: 17
FFEE: 2B
                          DFB
                                MOVE - 1
FFEF: 1F
                          DFB
                                LT-1
```

FFF0:	83		DFB	SETNORM-1		
FFF1:	7F		DFB	SETINV-1		
FFF2:	5D		DFB	LIST-1		
FFF3:	CC		DFB	WRITE-1		
FFF4:	B5		DFB	GO-1		
FFF5:	FC		DFB	READ-1		
FFF6:	17		DFB	SETMODE-1		
FFF7:	17		DFB	SETMODE-1		
FFF8:	F5		DFB	CRMON-1		
FFF9:	03		DFB	BLANK-1		
FFFA:	FB		DFB	NMI	/ IMN	ECTOR
FFFB:	03		DFB	NMI/256		
FFFC:	59		DFB	RESET	RESET	VECTOR
FFFD:	FF		DFB	RESET/256		
FFFE:	86		DFB	IRQ	IRQ V	ECTOR
FFFF:	FA		DFB	IRQ/256		
		XQTNZ	EQU	\$3C		

```
******
                     APPLE-II
                  MINI-ASSEMBLER
               * COPYRIGHT 1977 BY
               \star APPLE COMPUTER INC.
               * ALL RIGHTS RESERVED *
                    S. WOZNIAK
                     A. BAUM
               ******
                TITLE "APPLE-II MINI-ASSEMBLER"
               FORMAT EQU $2E
               LENGTH
                             $2F
                        EOU
               MODE
                        EQU
                             $31
               PROMPT
                        EQU
                             $33
               YSAV
                        EQU
                              $34
               L
                        EQU
                             $35
               PCL
                        EOU
                              $3A
               PCH
                        EQU
                             $3B
               A1H
                        EQU
                             $3D
               A2L
                        EQU
                              $3E
                        EQU
                             $3F
               A2H
               A4L
                        EQU
                              $42
               A4H
                        EOU
                              $43
               FMT
                        EQU
                              $44
               IN
                        EQU
                              $200
               INSDS2
                        EQU
               INSTDSP
                        EQU
                              $F8D0
                        EQU
                              $F94A
               PRBL2
               PCADJ
                        EOU
                              $F953
               CHAR1
                        EQU
                              $F9B4
               CHAR2
                        EQU
               MNEML
                        EQU
                              $F9C0
               MNEMR
                        EQU
                              $FA00
               CURSUP
                              $FC1A
                        EOU
               GETLNZ
                        EQU
                              $FD67
               COUT
                        EQU
                              $FDED
               BL1
                        EQU
                              $FE00
               A1PCLP
                        EQU
                              $FE78
               BELL
                              $FF3A
                        EOU
               GETNUM
                        EQU
                              $FFA7
               TOSUB
                        EQU
                              $FFBE
               ZMODE
                        EQU
                              $FFC7
               CHRTBL
                        EQU
                        ORG
                              $F500
F500: E9 81
                                        IS FMT COMPATIBLE
               REL
                        SBC
                              #$81
F502: 4A
                        LSR
                                        WITH RELATIVE MODE?
F503: D0 14
                        BNE
                              ERR3
                                        NO.
F505: A4 3F
                        LDY
                             A2H
F507: A6 3E
                        LDX
                              A2L
                                        DOUBLE DECREMENT
F509: D0 01
                        BNE
                              REL2
F50B: 88
                        DEY
F50C: CA
               REL2
                        DEX
F50D: 8A
                        TXA
F50E: 18
                        CLC
F50F: E5 3A
                                        FORM ADDR-PC-2
                        SBC
F511: 85 3E
                        STA
                             A2L
F513: 10 01
                        BPL
                              REL3
F515: C8
                        INY
F516: 98
               REL3
                        TYA
```

F517:	r.	20			SBC	DCU	
F517:				ERR3			ERROR IF >1-BYTE BRANCH
F51B:				FINDOP			Entroit II /I EIIE Etaiton
				FNDOP2			MOVE INST TO (PC)
F520:							MOVE INST TO (FC)
F520:					DEY	(PCL),Y	
F523:						FNDOP2	
F525:			EC			CURSUP	
F528:							RESTORE CURSOR
F52B:							TYPE FORMATTED LINE
					TOD		UPDATE PC
F52E: F531:						PCH	OFDAIL FC
F531:						PCL	
F535:							CET NEVT IINE
						NXTLINE	GET NEXT LINE
F538:	20	BE	rr	FAKEMON3	JDK	TOSUB	GO TO DELIM HANDLER RESTORE Y-INDEX
				FAKEMON		GETNUM YSAV	READ PARAM
F540:							SAVE Y-INDEX
F542:				ENVENONO		#\$17	INIT DELIMITER INDEX
F544:				FAKEMON2		DECEME	CHECK NEXT DELIM ERR IF UNRECOGNIZED DELIM
F545:							
F547:			rr		CMP	FAKEMON2	COMPARE WITH DELIM TABLE
F54A:							
F54C:							MATCH, IS IT CR?
F54E:							NO, HANDLE IT IN MONITOR
F550:					LDA		
F552:						#\$0	
F554:					DEC		
F556:						BL1	HANDLE CR OUTSIDE MONITOR
F559:						NXTLINE	
F55C:				TRYNEXT			GET TRIAL OPCODE
F55E:			F8			INSDS2	GET FMT+LENGTH FOR OPCODE
F561:					TAX		
F562:							GET LOWER MNEMONIC BYTE
F565:	C5	42			CMP		MATCH?
F567:	D0	13			BNE	NEXTOP	NO, TRY NEXT OPCODE.
F569:	BD	C0	F9			MNEML,X	GET UPPER MNEMONIC BYTE
F56C:	C5	43			CMP	A4H	MATCH?
F56E:	D0	0C			BNE	NEXTOP	NO, TRY NEXT OPCODE
F570:	Α5	44				FMT	
F572:	A4	2E			LDY	FORMAT	GET TRIAL FORMAT
F574:	C0	9D				#\$9D	TRIAL FORMAT RELATIVE?
F576:	F0	88			BEQ	REL	YES.
F578:	C5	2E		NREL	CMP	FORMAT	SAME FORMAT?
F57A:	F0	9F			BEQ	FINDOP	YES.
F57C:	C6	3D		NEXTOP	DEC	A1H	NO, TRY NEXT OPCODE
F57E:	D0	DC			BNE	TRYNEXT	
F580:	E6	44				FMT	NO MORE, TRY WITH LEN=2
F582:	C6	35			DEC	L	WAS L=2 ALREADY?
F584:	F0	D6			BEQ	TRYNEXT	NO.
F586:	A4	34		ERR	LDY	YSAV	YES, UNRECOGNIZED INST.
F588:	98			ERR2	TYA		
F589:	AA				TAX		
F58A:		4A	F9		JSR	PRBL2	<u> </u>
F58D:	Α9	DE			LDA	11.455	PRINT ^ UNDER LAST READ
F58F:	20	מיז				#\$DE	PRINT ^ UNDER LAST READ CHAR TO INDICATE ERROR
F592:	2.0	உப	FD		JSR	#\$DE COUT	
F595:		3A	FF	RESETZ	JSR JSR	COUT	CHAR TO INDICATE ERROR
	A9	3A	FF	RESETZ	JSR JSR	COUT	CHAR TO INDICATE ERROR
F597:	Α9	3A A1	FF	RESETZ NXTLINE	JSR JSR LDA	COUT	CHAR TO INDICATE ERROR POSITION.
	A9 85	3A A1 33	FF	RESETZ NXTLINE	JSR JSR LDA STA	COUT BELL #\$A1 PROMPT	CHAR TO INDICATE ERROR POSITION.
F597:	A9 85 20	3A A1 33 67	FF FD	RESETZ NXTLINE	JSR JSR LDA STA JSR	COUT BELL #\$A1 PROMPT GETLNZ	CHAR TO INDICATE ERROR POSITION.
F597: F599:	A9 85 20 20	3A A1 33 67 C7	FF FD FF	RESETZ NXTLINE	JSR JSR LDA STA JSR JSR	COUT BELL #\$A1 PROMPT GETLNZ ZMODE	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE.
F597: F599: F59C:	A9 85 20 20 AD	3A A1 33 67 C7	FF FD FF	RESETZ NXTLINE	JSR JSR LDA STA JSR JSR LDA	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF
F597: F599: F59C: F59F:	A9 85 20 20 AD C9	3A A1 33 67 C7 00 A0	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR JSR LDA CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR
F597: F599: F59C: F59F: F5A2:	A9 85 20 20 AD C9 F0	3A A1 33 67 C7 00 A0 13	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR JSR LDA CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK?
F597: F599: F59C: F59F: F5A2: F5A4:	A9 85 20 20 AD C9 F0 C8	3A A1 33 67 C7 00 A0 13	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR JSR LDA CMP BEQ INY	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK?
F597: F599: F59C: F59F: F5A2: F5A4: F5A6:	A9 85 20 20 AD C9 F0 C8 C9	3A A1 33 67 C7 00 A0 13	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR JSR LDA CMP BEQ INY CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7:	A9 85 20 20 AD C9 F0 C8 C9 F0	3A A1 33 67 C7 00 A0 13	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR JSR LDA CMP BEQ INY CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1?
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9:	A9 85 20 20 AD C9 F0 C8 C9 F0 88	3A A1 33 67 C7 00 A0 13 A4 92	FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR LDA CMP BEQ INY CMP BEQ DEY	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB:	A9 85 20 20 AD C9 F0 C8 C9 F0 88	3A A1 33 67 C7 00 A0 13 A4 92	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR LDA CMP BEQ INY CMP BEQ DEY	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AC: F5AF:	A9 85 20 20 AD C9 F0 C8 C9 F0 88 20 C9	3A A1 33 67 C7 00 A0 13 A4 92 A7	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR LDA CMP BEQ CMP BEQ DEY JSR CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AC: F5AF:	A9 85 20 20 AD C9 F0 88 20 C9 D0	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR LDA CMP BEQ CMP BEQ DEY JSR CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR?
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5AB: F5AC: F5AF:	A9 85 20 20 AD C9 F0 C8 C9 F0 88 20 C9 D0 8A	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5	FF FD FF 02	RESETZ NXTLINE	JSR JSR LDA STA JSR LDA CMP BEQ INY CMP BEQ INY CMP BEQ CMP BEQ DEY JSR CMP BNE TXA	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR?
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AB: F5AC: F5AF: F5AF:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5	FF FD FF 02	RESETZ NXTLINE	JSR JSR JSR STA JSR JSR LDA LDA END	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR.
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AB: F5AB: F5AB: F5AB: F5AB:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20	A1 33 67 C7 00 A0 13 A4 92 A7 93 D5	FF FF FF	RESETZ NXTLINE	JSR JSR JSR STA JSR JSR LDA CMP BEQ CMP BEQ DEY JSR CMP BEQ TXA BEQ JSR TXA BEQ JSR	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON.
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AC: F5AF: F5B1: F5B1: F5B3: F5B4:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 A9	A1 33 67 C7 00 A0 13 A4 92 A7 93 D5	FF FD FF 02 FF	RESETZ NXTLINE ERR4	JSR JSR JSR STA JSR JSR LDA CMP BEQ CMP BEQ DEY JSR CMP BEQ TXA BEQ JSR TXA BEQ JSR	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH.
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AB: F5B1: F5B1: F5B3: F5B4: F5B6: F5B6: F5B6: F5B8:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 A9 85	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 03 3D	FF FD FF 02 FF	RESETZ NXTLINE ERR4	JSR JSR JSR LDA STA JSR JSR LDA CMP BEQ INY CMP BEQ EY SSR CMP BEQ TXA BEQ TXA BEQ JSR CMP BNE TXA BEQ JSR CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3 A1H	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH.
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AB: F5B1: F5B1: F5B3: F5B4: F5B6: F5B6: F5B6: F5B8:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 A9 85 20	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 03 3D 34	FF FD FF 02 FF FE FE	RESETZ NXTLINE ERR4 SPACE NXTMN	JSR JSR JSR JSR JSR JSR JSR LDA CMP BEQ INY CMP BEQ TXA BEQ JSR CMP BNE TXA BEQ JSR LDA JSR LDA JSR	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 AlpCLP #\$3 Alh GETNSP	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC
F597: F599: F59C: F59F: F5A2: F5A4: F5A6: F5A7: F5AB: F5AB: F5AC: F5B1: F5B1: F5B3: F5B4: F5B3: F5B4: F5B3: F5B4: F5B3:	A9 85 20 AD C9 F0 88 20 C9 D0 8A F0 20 A9 85 20 0A	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 03 3D 34	FF FD FF 02 FF FE FE	RESETZ NXTLINE ERR4 SPACE NXTMN NXTM	JSR JSR JSR JSR JSR JSR JSR JSR LDA BEQ INY CMP BEQ INY CMP BEQ JSR CMP BNE JSR CMP BNE JSR ASL JSR ASL SBC	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3 A1H GETNSP A #\$BE	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC
F597: F599: F590: F597: F5A2: F5A4: F5A6: F5A7: F5A8: F5A8: F5AF: F5B1: F5B1: F5B3: F5B4: F5B8: F5B9: F5B9: F5B9: F5BB: F5BB:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 A9 85 20 0A E9	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 03 3D 34 BE	FF FF FF FE	RESETZ NXTLINE ERR4 SPACE NXTMN NXTM	JSR JSR JSR JSR JSR JSR JSR JSR LDA BEQ INY CMP BEQ INY CMP BEQ JSR CMP BNE JSR CMP BNE JSR ASL JSR ASL SBC	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3 A1H GETNSP A #\$BE	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC GET FIRST MNEM CHAR.
F597: F599: F597: F59F: F522: F5A4: F5A6: F5A7: F5A8: F5AF: F5AF: F5B1: F5B1: F5B3:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 A9 85 20 0A E9 C9	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 03 3D 34 BE C2	FF FD FF 02 FF	RESETZ NXTLINE ERR4 SPACE NXTMN NXTM	JSR JSR JSR LDA STA JSR JSR LDA CMP BEQ INY CMP BEQ JSR CMP BNE TXA BEQ JSR LDA STA JSR LDA STA JSR ASL SBC CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3 A1H GETNSP A #\$BE #\$C2	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC GET FIRST MNEM CHAR. SUBTRACT OFFSET
F597: F599: F599: F59F: F5A2: F5A4: F5A6: F5A7: F5A9: F5AB: F5AB: F5B1: F5B1: F5B3: F5B6: F5B9: F5B9: F5BB: F5B0:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 A9 85 20 OA P0 985 20 20 20 20 20 20 20 20 20 20 20 20 20	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 03 3D 34 BE C2 C1	FF FD FF 02 FF	RESETZ NXTLINE ERR4 SPACE NXTMN NXTM	JSR JSR JSR LDA STA JSR JSR LDA CMP BEQ INY CMP BEQ JSR CMP BNE TXA BEQ JSR LDA STA JSR LDA STA JSR ASL SBC CMP	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3 A1H GETNSP A #\$BE #\$C2 ERR2	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC GET FIRST MNEM CHAR. SUBTRACT OFFSET LEGAL CHAR?
F597: F599: F599: F597: F522: F5A4: F5A6: F5A7: F5A9: F5AB: F5B1: F5B1: F5B6: F5B6: F5B8:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 0A E9 90 0A	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 803 33D 34 BE C2 C1	FF FD FF 02 FF	RESETZ NXTLINE ERR4 SPACE NXTMN NXTM	JSR JSR JSR JSR JSR JSR JSR LDA CMP BEQ INY CMP BEQ TXA JSR CMP BNE TXA BEQ ASL STA ASL SBC CMP ASL	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3 A1H GETNSP A #\$BE #\$C2 ERR2 A	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC GET FIRST MNEM CHAR. SUBTRACT OFFSET LEGAL CHAR? NO.
F597: F599: F599: F599: F597: F5A2: F5A4: F5A6: F5A7: F5A8: F5AB: F5B1: F5B1: F5B8: F5B8: F5B9: F5B9: F5B0:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 0A 9 5 20 0A 9 6 9 0A 0A 0A 0A 0A 0A 0A 0A 0A 0A 0A 0A 0A	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 80 33D 34 BE C2 C1	FF FD FF 02 FF FE	RESETZ NXTLINE ERR4 SPACE NXTMN NXTM	JSR JSR JSR JSR JSR JSR JSR LDA EMP BEQ INY CMP BEQ INY CMP BEQ JSR CMP BNE TXA ASL STA ASL STA ASL ASL	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 ERR2 A1PCLP #\$3 A1H GETNSP A #\$BE #\$C2 ERR2 A	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC GET FIRST MNEM CHAR. SUBTRACT OFFSET LEGAL CHAR? NO.
F597: F599: F599: F597: F522: F5A4: F5A6: F5A7: F5A8: F5A8: F5AF: F5B1: F5B3:	A9 85 20 20 AD C9 F0 88 20 C9 D0 8A F0 20 20 0A E9 90 0A 20	3A A1 33 67 C7 00 A0 13 A4 92 A7 93 D5 D2 78 03 3D 34 BE C2 C1	FF FD FF 02 FF	RESETZ NXTLINE ERR4 SPACE NXTMN NXTM	JSR JSR JSR JSR JSR JSR LDA CMP BEQ INY CMP BEQ DEY JSR CMP BNE TXA BEQ JSR LDA STA JSR ASL SBC CMP BCC ASL ASL LDX	COUT BELL #\$A1 PROMPT GETLNZ ZMODE IN #\$A0 SPACE #\$A4 FAKEMON GETNUM #\$93 ERR2 AlPCLP #\$3 AlH GETNSP A #\$BE #\$C2 ERR2 A A #\$4	CHAR TO INDICATE ERROR POSITION. '!' INITIALIZE PROMPT GET LINE. INIT SCREEN STUFF GET CHAR ASCII BLANK? YES ASCII '\$' IN COL 1? YES, SIMULATE MONITOR NO, BACKUP A CHAR GET A NUMBER ':' TERMINATOR? NO, ERR. NO ADR PRECEDING COLON. MOVE ADR TO PCL, PCH. COUNT OF CHARS IN MNEMONIC GET FIRST MNEM CHAR. SUBTRACT OFFSET LEGAL CHAR? NO.

```
F5CC: 26 42
                         ROL
                                A4L
F5CE: 26 43
                          ROL
                                A4H
F5D0: CA
                          DEX
F5D1: 10 F8
                         BPL
                                NXTM2
F5D3: C6 3D
                                           DONE WITH 3 CHARS?
                          DEC
                                A1H
F5D5: F0 F4
                                NXTM2
                                           YES, BUT DO 1 MORE SHIFT
                          BEO
F5D7: 10 E4
                          BPL
                                NXTMN
                                           NO
F5D9: A2 05
                FORM1
                          LDX
                                #$5
                                           5 CHARS IN ADDR MODE
F5DB: 20 34 F6
                                GETNSP
                                           GET FIRST CHAR OF ADDR
                FORM2
                         JSR
F5DE: 84 34
                                YSAV
                          STY
F5E0: DD B4 F9
                          CMP
                                CHAR1.X
                                           FIRST CHAR MATCH PATTERN?
F5E3: D0 13
                          BNE
                                FORM3
F5E5: 20 34 F6
                         JSR
                                GETNSP
                                           YES, GET SECOND CHAR
F5E8: DD BA F9
                          CMP
                                           MATCHES SECOND HALF?
                                CHAR2.X
F5EB: F0 0D
                          BEQ
                                FORM5
                                           YES.
F5ED: BD BA F9
                         LDA
                                CHAR2.X
                                           NO, IS SECOND HALF ZERO?
F5F0: F0 07
                          BEO
                                FORM4
                                           YES.
F5F2: C9 A4
                          CMP
                                #$A4
                                           NO, SECOND HALF OPTIONAL?
F5F4: F0 03
                          BEQ
                                FORM4
                                           YES.
F5F6: A4 34
                          LDY
                                YSAV
F5F8: 18
                FORM3
                          CLC
                                           CLEAR BIT-NO MATCH
F5F9: 88
                FORM4
                         DEY
                                           BACK UP 1 CHAR
F5FA: 26 44
                FORM5
                          ROL
                                FMT
                                           FORM FORMAT BYTE
F5FC: E0 03
                          CPX
                                #$3
                                           TIME TO CHECK FOR ADDR.
F5FE: D0 0D
                          BNE
                                FORM7
                                           NO
F600: 20 A7 FF
                                GETNUM
                         JSR
F603: A5 3F
                         LDA
                                A2H
F605: F0 01
                                           HIGH-ORDER BYTE ZERO
                         BEO
                                FORM6
F607: E8
                          INX
                                           NO, INCR FOR 2-BYTE
F608: 86 35
                FORM6
                          STX
                                           STORE LENGTH
F60A: A2 03
                                           RELOAD FORMAT INDEX
                          LDX
F60C: 88
                         DEY
                                           BACKUP A CHAR
                                           SAVE INDEX
F60D: 86 3D
                FORM7
                          STX
                                A1H
F60F: CA
                          DEX
                                           DONE WITH FORMAT CHECK?
F610: 10 C9
                          BPL
                                FORM2
                                           NO.
F612: A5 44
                          LDA
                                FMT
                                           YES, PUT LENGTH
F614: 0A
                         ASL
                                            IN LOW BITS
                                Α
F615: 0A
                         ASL
                                Α
F616: 05 35
                          ORA
                                т.
F618: C9 20
                          CMP
                                #$20
F61A: B0 06
                          BCS
                                FORM8
                                           ADD "$" IF NONZERO LENGTH
F61C: A6 35
                                           AND DON'T ALREADY HAVE IT
                          LDX
F61E: F0 02
                                FORM8
                          BEO
                                #$80
F620: 09 80
                          ORA
F622: 85 44
                FORM8
                         STA
                                FMT
F624: 84 34
                          STY
                                YSAV
F626: B9 00 02
                                IN,Y
                                           GET NEXT NONBLANK
                          LDA
F629: C9 BB
                          CMP
                                #$BB
                                           '' START OF COMMENT?
F62B: F0 04
                                FORM9
                                           YES
                          BEO
F62D: C9 8D
                                           CARRIAGE RETURN?
                          CMP
                                #$8D
F62F: D0 80
                          BNE
                                ERR4
                                           NO, ERR.
F631: 4C 5C F5 FORM9
                          JMP
                                TRYNEXT
F634: B9 00 02 GETNSP
                         LDA
                                IN,Y
F637: C8
                          INY
F638 · C9 A0
                          CMP
                                #$A0
                                           GET NEXT NON BLANK CHAR
                          BEO
F63A: F0 F8
                                GETNSP
F63C: 60
                          RTS
                                $F666
                          ORG
F666: 4C 92 F5 MINIASM JMP
                                RESETZ
```