680 Terminal Card

Software #2

96 SHEETS • 5 x 5 QUAD
10⅛ x 7⅞ • 53-110

NATIONAL BLANK BOOK COMPANY, INC.
Holyoke, Massachusetts 01040-Made in USA

6800 Software for the Terminal Card.

Book 2

Graphics Processor ROM

Graphics processor runs as a background program on the 6800 Display processor - functions are initiated by external commands through the Graphics Processor registers

176670 — Result Data Control & status register
- Bit #7 ready flag
- Bit #6 interrupt enable bit

176672 — Result Data (Read only)
16 Bits from Graphics Control

176674 — Command Control & Status Register
- Bit #7 command complete flag
- Bit #6 interrupt enable bit

176676 — Command Register (Write only)
16 bit data/control to graphics Control

24 March 79
ARB

Summary of Graphics Registers (internal)

The Y parameter Table 2 200 (1200_8)
if [X] = 2,200

Ø,X	PYH PYL	previous Y value
2,X	NYH NYL	new Y value
4,X	ΔYH ΔYL	the relative register ΔY
6,X	DYH DYL	the Y vector length
10,X	CPYH CPYL	temporary containing the running value of Y during a vector plot
12,X	CPYFH CPYFL	fractional part of the running value of Y during a vector plot
14,X	CPY1 CPY2 CPY3 CPY4	32 Bit updating value during a vector plot ie next CPY = current CPY ± CP
20,X	SDY	Sign of updating during vector plot

The X parameter Table 2 230 (1230_8)
if [X] = 2,230

0,X	PXH PXL	previous X value
2,X	NXH NXL	new X value
4,X	ΔXH ΔXL	the relative register ΔX
6,X	DXH DXL	the X vector length
10,X	CPXH CPXL	temporary containing the running value of X during Vector plot
12,X	CPXFH CPXFL	fractional part of the running value of X during vector plot
14,X	CPX1 CPX2 CPX3 CPX4	32 Bit updating value during a vector plot
20,X	SDX	Sign of updating value

24 March 79
ARD

Plotting Table at 2,260

if [X] = 2,260 (1260_8)

0,X	YV YV+1	Y position value
2,X	XV XV+1	X position value
4,X	YVAL YVAL+1	Horizontal line Base Address
6,X	XVAL XVAL+1	BYTE Offset in line
10,X	XOR	; Explicit Bit Location
11,X	CSR	Status Control
12,X	XYVAL XYVAL+1	Absolute Byte Location of Bit to Be written
14,X	TEMP1 TEMP2 TEMP3 TEMP4	temporaries

Summary of Graphics instructions

XY Commands

Bit	[15	14	13]	12	11	10	[9-0]
	0	0	0	AI	V	PL	[Absolute X value 0-1023]
	0	0	1	AI	V	PL	[ΔX Value -511 to +512] 2's complement
	0	1	0	AI	V	PL	[Absolute Y value 0-1023]
	0	1	1	AI	V	PL	[ΔY Value -511 to +512]

if AI (Autoincrement) = 1

1.) P_ = N_ move value of other parameter

2.) N_ = N_ + Δ next value of other parameter

if Absolute mode (Bit 13 = 0)

1.) P_ = N_ of this parameter

2.) N_ = Data for this parameter

if Relative mode (Bit 13 = 1)

1.) Δ_ = Data for this parameter

if PL = 1

1.) if Bit 13 = 1 (relative mode)
   a) PX = NX
   b) NX = NX + ΔX
   c) PY = NY
   d) NY = NY + ΔY

2.) Plot single point (See CSR instruction!)
   if and only if $\emptyset \leq XV \leq 639.$
   $\emptyset \leq YV \leq 255.$

finished — (will not do vector even if V=1)

if V = 1

1.) if Bit 13 = 1 (relative mode)
   a) PX = NX
   b) NX = NX + ΔX
   c) PY = NY
   d) NY = NY + ΔY

2.) Plot vector with endpoints (See CSR Instuction)
   PX, PY NX, NY
   if and only if points are bounded by
   $\emptyset \leq XV \leq 639.$
   $\emptyset \leq YV \leq 255.$

3.) if Bit 13 = ∅ (Absolute mode)
   PX = NX
   PY = NY

24 March 79
ARB

Summary of Graphics Instructions

Control instruction

Bits	[15	14	13]	12	11	10	9	8	[7-0 Bits]
	1	1	0	C	H	H	H	H	[Control Register]

if C = 1 graphics screen is cleared

HHHH is MSB 4 Bits of Address for the Read/write function

Control Register

Bits →	7	6	5	4	3	2	1	0

Bit 7 mode: 1 = delete point, 0 = Add point

Bit 0 intensity on = 1, off = 0

Bit 6 Enable screen control Bit
if = 1
then for
Bit 5 = 1 Turn Display on
Bit 5 = 0 Turn Display off

Bit 4 Restore character set if = 1

Bits 1-3 unused

Read/Write Data to 6800 MPU

Bits [15 14 13] 12 [11 — 0]

1 1 1 $\overline{R/W}$ [(or 12 Bits of Address]

Read (0) or Write (1) data at Address

HH HH [12 Bits] on MPU Card

data written is the last Byte used in the ASCII Select: Instruction

During Read - two Bytes are returned at 176172 from address specified and Address +1

24 March 79
ARA

Summary of Graphics Instructions

ASCII Instructions

Bits [15 14 13] 12 [11-8] [7 — 0]

1 ∅ 1 1 XXXX [ASCII Code]

ASCII Select (Write Data for Write function)
where XXXX are don't cares

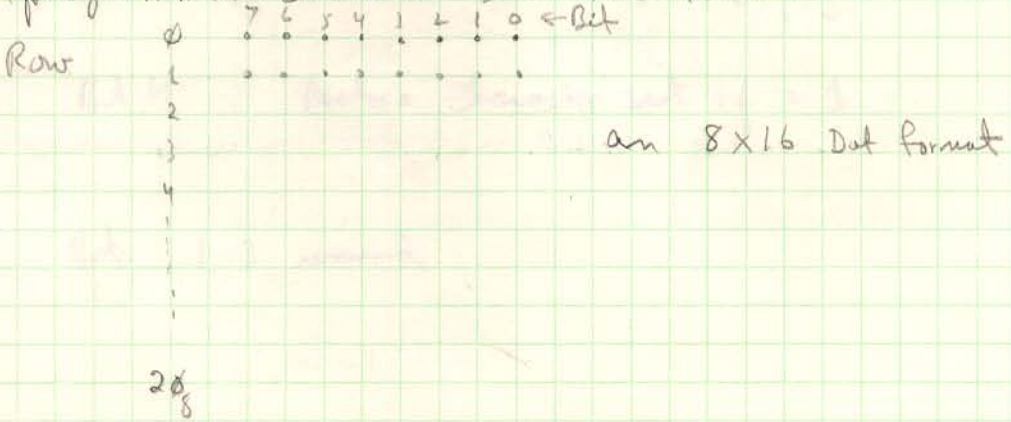
the ASCII code is any value from ∅ - 255
each value has a programmable character
associated with it — this instruction specifies
the character to be programed by the
Program ASCII Command

Program ASCII

1 ∅ 1 ∅ NNNN [7-∅ 8 Bit - stream]

reprograms ASCII Code row NNNN with
the 8 Bit stream

each programmable character has the format



an 8 x 16 Dot format

Bits [15 14 13] 12 11 14 9 8 [ASCII Code]

Plot ASCII character

MODE

1 ø ø [AIR AIC REV RD CD] [ASCII]

This command will plot the selected ASCII character at the position NX, NY

Normal characters result if Bits 12 - 8 = ø

RD = Raw Direction

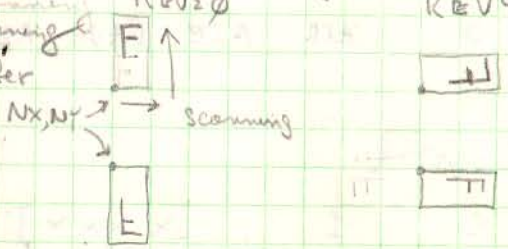
CD = Column Direction

RD, CD specify the direction of display scanning

Rev causes exchange of Raw/Column scanning

REV = ø REV = 1

eg RD = CD = ø character



RD = ø, CD = 1

RD = 1, CD = 1

RD = 1, CD = ø

Careful study shows that Mode [Rev RD CD] = (0,0,0) (1,0,1) (0,1,1) (1,1,0) form valid characters

AIR and AIC when set, update the NX, NY coordinates by the character width or height

ie MODE = [10000] gives

or Mode [0 1 1 1 ø] gives

24 March 79
ABD

Summary of All Graphic Instructions

XY Function

Bit [15	14	13]					
0	0	0	AI	V	PL	[XVALUE 0-1023]
0	0	1	AI	V	PL	[ΔX Value -511 to +512]
0	1	0	AI	V	PL	[YVALUE 0-1023]
0	1	1	AI	V	PL	[ΔY VAlue -511 to +512]

ASCII Character Write

1	0	0	AIR	AIC	REV	RD	CD	[ASCII Code]

ASCII Control

1	0	1	1	[X XXX]	[ASCII Select]
1	0	1	0	N N N N	[8 Bit stream]

Control Register

1	1	0	C	H H H H	[8 Bit CSR]

Read/Write function

1	1	1	R/W	[Low 12 Bit Address]

Graphic Rom

340 004

255	255	#,255,255	Here Indicator
340	120	#INIT	Pointers to entry points
340	16	#DATA	
340	20	#SYST	
340	16	#PCHH	

340 016

71	RTS	Return for Non-Routines

340 020

SYST:	316	340	030	LDX #ID	Point at Text
	275	376	220	JSR PRINT	print ID
	71			RTS	return to caller

340 030

ID:	15	12	40	63	/CR LF Sp 3
	61	40	112	101	1 Sp J A
	116	40	40	61	N Sp Sp 1
	71	67	71	40	9 7 9 Sp
	55	40	107	122	- Sp G R
	101	120	110	111	A P H I
	103	40	122	117	C Sp R O
	115	40	15	12	M Sp CR LF
	52	0	0		* Nu Nu /

Here Is Button Handler

```
340 100
HEREIS:  163  0  217   COM  GRAPH      flip-flop graphic
         174  0  216   INC  UPDFLG     make display do an update
         226  54       LDA  A  #2C     clear Hereis IRQ Flag
         73            RTI             Return from interrupt
```

INITIALIZATION OF GRAPHIC PROCESSOR

```
340 120
INIT:    316  60  0    LDX  #,60,0     Set Base pointers
         337  272      STX  EVNSCN     for Display Memory
         316  130 0    LDX  #,130,0
         337  274      STX  ODDSCN

         316  377 177  LDX  #,377,177  Set for 128 lines/scan
         377  1   246  STX  GRPSTRT+4
         316  120  31   LDX  #,120,31   Set for Continuous graphic
                                        scan 'on'
         377  1   244  STX  GRSTRT+2

         316  377  7    LDX  #,377,7    Set for starting Text
         377  1   254  STX  GRPSTP+4
         316  120  0    LDX  #,120,0    Set for Display 'Off'
         377  1   252  STX  GRPSTP+2

         177  0   217  CLR  GRAPH      Initialize Graphics Off
```

25 March 78
ARD

```
                17              SEI             Hold all interrupts
340 166
                316 340 100     LDX  #, HEREIS  Place address of 'HereIs' Button
                377 17  37Ø     STX  IRQ13      handler at Vector Address 13

                2Ø6 7           LDA A #, 7      Set up interrupt control
                                                  for 'HereIs' Button
                227 56          STA A  $2E
                226 54          LDA A  $2C      clear pending interrupt

                316 Ø  Ø        LDX  #, Ø, Ø    Clear PIA Control Registers
                337 22          STX: 22         Graphic 'IN' PIA
                337 20          STX  2Ø         Set for input
                337 26          STX  26         Graphic OUT PIA (& 60Hz CLOCK)
                316 377 377     LDX  #, 377, 377
                337 24          STX  24         Set OUT Direction
340 220
                316 347 1Ø      LDX  #, GRPHIN  Load GRAPHIC IN/out
                377 17  356     STX  IRQ8       interrupt Vectors with
340 226
                316 347 1Ø      LDX  #, GRPHOUT Entry points
                377 17  36Ø     STX  IRQ9
```

```
316 55 4    LDX  #,55,4      Enable interrupt for input
337 22      STX  22

316 4 54    LDX  #,5,4       Disable Interrupt for output
                             & Reenable 60 Hz clock interrupt
337 26      STX  26

226 20      LDA A 20         Clear pending input IRQ Flag
226 55      LDA A 25         clear pending output IRQ Flag

16          CLI              enable interrupts
71          RTS              Return to caller
                             initialization complete
```

25 March 79
ASD

Calculation of X-coordinate Value

Enter routine w. H [X] = Table Address

```
341 040
NCXVAL:  177 000 115    CLR    CHK+1      X PLOT OK
         155 002        TST    2,X        check High order of X value
         053 011        BMI    NOPLOT     if negative don't plot
341 047
         275 341 030    JSR    BXADC      go Compute Bit position
         046 004        BNE    NOPLOT     if not=0, don't plot
         200 120        SUB A  #,120      XVAL+1 in range?
         045 003        BCS    GOOD,      if so - good
NOPLOT:  163 000 115    COM    CHK+1      inhibit plot
GOOD:    071            RTS

341 030
BXADC:   246 003        LDA A  3,X        get Low Byte of XV
         137            CLR B
         204 007        AND A  #,7        Mask low 3 Bits
341 035
         213 100        ADD A  #,TBLEL    Compute Address of Bit Pattern Entry  341 100
341 037                                                                        TBLE:
         311 341        ADC B  #,TBLEH
         247 015        STA A  15,X       Save in temporary
         347 014        STA B  14,X
         337 116        STX    SAVEX      Save in temporary
```

```
356 14      LDX   14,X      get address of Bit pattern
246 0       LDA A 0,X       Load Bit pattern
336 16      LDX   SAVEX     restore X register
247 10      STA A 8,X       Save Bit pattern in XOR
246 3       LDA A 3,X       get XV to compute Bit Address
346 2       LDA B 2,X
 14         CLC             \
126         RORB            |
106         RORA            |
127         ASRB            | ÷ 8
106         RORA            |
127         ASRB            |
106         RORA            /
247 7       STA A 7,X       Save result in XVAL
347 6       STA B 6,X
 71         RTS             Return to Caller
```

Bit Pattern table

```
TBLE: 200     200
      100     100
       40      40
       20      20
       10      10
        4       4
        2       2
        1       1
```

25 March 71
ARB

Calculation of Y-coordinate value

enter with [x] = Address of Table

```
341 120
NCYVAL: 177 0 114   CLR  CHK      Y PLOT OK
        346 0       LDA B 0,X     check value
        47  4       BEQ  CALC     must = 0
        163 0 114   Com  CHK      else inhibit plotting
        71          RTS           return to caller
CALC:   346 1       LDA B 1,X     get low order value
        14          CLC           } ÷ 2
        126         ROR B         }
        206 177     LDA A #177    } Compute <B,A> = 127. - YV/2
        20          SBA           }
        137         CLR B         }
        110         ASL A         } Compute 16. x <B,A>
        131         ROL B         }
        110         ASL A         }
        131         ROL B         }
        110         ASL A         }
        131         ROL B         }
        110         ASL A         }
        131         ROL B         }
```

EVEN

SAVE

```
247  5      STA A  5,X   } save 16*<B,A>
347  4      STA B  4,X   }
110         ASL A       }
131         ROL B       } compute 64. * <B,A>
110         ASL A       }
131         ROL B       }
253  5      ADD A  5,X   } compute 80. * <B,A>
351  4      ADC B  4,X   } (ie 80 Bytes per scan line!)
146  1      ROR  1,X     Check EVEN/ODD Scan
 44  10     BCC  EVEN    if even-branch
151  1      ROL  1,X     restor value
233  275    ADD A  ODD+1 } compute Absolute Address
331  274    ADC B  ODD   } for an ODD scan
 40  6      BRA  SAVE
EVEN: 151 1 ROL  1,X     restor value
233  273    ADD A  EVEN+1 } compute Absolute Address
231  272    ADC B  EVEN   } for an EVEN scan
SAVE: 247 5 STA A  5,X   } save value in YVAL
347  4      STA B  4,X   }
 71         RTS          Return to caller
```

25 March 79
ARD

Plot XY Point Routine

```
341 230
PLOTXY:  376  2  202   LDX   NY      }  load value
         377  2  260   STX   PTBLE   }
         376  2  232   LDX   NX      }  load value
         377  2  262   STX   PTBLE+2 }

341 244
VECTXY:  316  2  260   LDX   #,PTBLE      Point to Plot table
341 247
         275  341  0    JSR   NCXVAL       Compute XVAL, XOR
341 252
         275  341  120  JSR   NCYVAL       Compute YVAL

341 255
NXYPLOT: 336  114        LDX   CHH        OK to PLOT ?
         47   1          BEQ   CONT       if so - branch
         71              RTS              else don't plot

CONT:    316  2  260     LDX   #,PTBLE     Point at Plot Table
341 260
         275  341  310   JSR   XYBIS       go do point
         71              RTS              return to caller
```

341 31
XYBI

DELE

DDIT

XY BIT SET Routine

341 310

XYBIS:	246	5	LDA A 5,X	}	get YVAL
	346	4	LDA B 4,X	}	
	253	7	ADD A 7,X	}	Add XVAL
	351	6	ADC B 6,X	}	
	247	13	STA A 13,X	}	Save XYVAL
	347	12	STA B 12,X	}	
	246	10	LDA A 10,X		get XOR (Bit position)
	346	11	LDA B 11,X		get CSR (control register)
	126		ROR B		check intensity
	44	16	BCC END		if I = 0 end
	356	12	LDX 12,X		get XYVAL as Address
	131		ROL B		check for add/delete
	53	4	BMI DELETE		if bit set - delete
	252	0	ORA A 0,X		else ADD by oring Bit
	40	3	BRA DOIT		
DELETE	103		COM A		Bit to clear is now '0'
	244	0	AND A 0,X		mask with current Display
DOIT	247	0	STA A 0,X		move data back into display
	71		RTS		Return to caller

25 March 79
ARO

Auto Increment routine

Enter with [X] = Table Address

and C = 1 for P = N, & N = N + Δ
or C = ϕ for N = N + Δ

342 ϕ
LOAD

341 360

AI:	246	3	LDA A 3,X	}	get N - value
	346	2	LDA B 2,X	}	
	44	4	BCC INC		update P? - branch if not
	247	1	STA A 1,X	}	P = N
	347	ϕ	STA B ϕ,X	}	
INC:	253	5	ADD A 5,X	}	N + Δ
	351	4	ADC B 4,X	}	
	247	3	STA A 3,X	}	save new value
	347	2	STA B 2,X	}	
	71		RTS		

Δ:

PLUS

Load new Value Routine

Enter with [X] = Table Address

342 ØIØ

```
LOADV:  326  100   LDA B   OPR      check operation for Relative mode
                                      ie ΔX,Y
        305  40    BIT B   #40
        46   21    BNE     Δ        Branch if Δ mode
        246  3     LDA A   3,X   }
        247  1     STA A   1,X   }  move N to P
        246  2     LDA A   2,X   }
        247  Ø     STA A   Ø,X   }
        226  101   LDA A   OPR+1 }
        304  3     AND B   #3    }  get new value for N
        247  3     STA A   3,X   }
        347  2     STA B   2,X   }
        71         RTS            Return

Δ:      226  101   LDA A   OPR+1 }  get Δ value
        304  3     AND B   #3    }
        305  2     BIT B   #2     check for a negative Δ
        47   2     BEQ     PLUS   if plus branch
        312  374   ORA B   374    else make a 16 Bit negative number
PLUS:   247  5     STA A   5,X   }  save the new Δ
        347  4     STA B   4,X   }
        71         RTS            return
```

25 March 79
ARD

Absolute Vector length Calculation

Enter with [X] = Table Address

342 ø6ø

ABSD:	157	2ø	CLR	2ø, X	set sign = (+)
	246	3	LDA A	3, X	compute N_ − P_
	346	2	LDA B	2, X	
	240	1	SUB A	1, X	
	342	ø	SBC B	ø, X	
	52	1ø	BPL	PABSV	if positive - branch
	143	2ø	Com	2ø, X	set sign = (−)
	103		Com	A	compute 2's Complement
	123		Com	B	
	213	1	ADD A	#1	
	311	ø	ADC B	#ø	
PABSV:	247	7	STA A	7, X	same |Result|
	347	6	STA B	6, X	
	71		RTS		return to caller

342
VSE

Vector Set Constant Routine

Enter with [X] = Table Address
& A = step value

342 120
VSETC: 137 CLR B

347 14 STA B 14, X
247 15 STA A 15, X
347 16 STA B 16, X
347 17 STA B 17, X
} set step value size

347 13 STA B 13, X
306 200 LDA B #200
347 12 STA B 12, X
} set CP_F = ½

246 1 LDA A 1, X
346 0 LDA B 0, X
247 11 STA A 11, X
347 10 STA B 10, X
} move Step value (P) to CP_

71 RTS return to Caller

25 March 79
ARD

Set sign of increment value Routine

Enter with [X] = Address of Table

342 150

SETSGN:	155	24	TST 24, X	check sign
	47	26	BEQ SDONE	if plus - skip all
	143	17	COM 17, X	else compute 32 Bit 2's complement of CP1 – CP4
	143	16	COM 16, X	
	143	15	COM 15, X	
	143	14	COM 14, X	
	154	17	INC 17, X	
	46	12	BNE SDONE	
	154	16	INC 16, X	
	46	6	BNE SDONE	
	154	15	INC 15, X	
	46	2	BNE SDONE	
	154	14	INC 14, X	
SDONE:	71		RTS	Return to calle

DIVISION Routine

342 210

DIVISION:	206 20	LDA A	#,20	
	227 103	STA A	LPCNTR	Bit Counter Set up
	226 105	LDA A	DND+1	} get dividend
	326 104	LDA B	DND	} get dividend
	14	CLC		} do initial shift
	111	ROL A		} do initial shift
	131	ROL B		} do initial shift
LOOP:	220 107	SUB A	DSR+1	} subtract Division
	322 104	SBC B	DSR	} subtract Division
	44 4	BCC	RQT	if carry clear - good test
	233 107	ADD A	DSR+1	} else restor Dividend
	331 106	ADC B	DSR	} else restor Dividend
RQT	171 0 111	ROL	QT+1	} shift in computed bit
	171 0 110	ROL	QT	} shift in computed bit
	110	ASL A		} shift DND
	131	ROL B		} shift DND
	172 0 103	DEC	LPCNTR	16 Bits generated yet?
	46 357	BNE	LOOP	loop until finished
	163 0 111	COM	QT+1	} get True result
	163 0 110	COM	QT	} get True result
	71	RTS		Return to Caller

25 March 79
ARD

UPDATE VALUE Calculation

Enter with [X] = Address of Table

342 270
UPDATE:

```
246  13    LDA A  13,X
346  12    LDA B  12,X
253  17    ADD A  17,X
351  16    ADC B  16,X
247  13    STA A  13,X
347  12    STA B  12,X
246  11    LDA A  11,X
346  10    LDA B  10,X
251  15    ADC A  15,X
351  14    ADC B  14,X
247  11    STA A  11,X
347  10    STA B  10,X
356  10    LDX    10,X
71         RTS
```

32 Bit Add
ie CP, CPF = CP, CPF + CP1-CP4

place result in X (= CP)
return to caller

342 33
PLOTVC

VLOOP:

Plot Vector Handler

342 330

PLOTVC:	336	112	LDX	VCNTR	} points to plot = Δ + 1
	10		INX		
	337	112	STX	VCNTR	
	376	2 240	LDX	CPX	} Set up Plot table with values
	377	2 262	STX	XV	
	376	2 210	LDX	CPY	
	377	2 260	STX	YV	
	275	341 244	JSR	VECTXY	Plot first point
	40	53	BR	ENTER	Go to Entry point
VLOOP:	316	2 230	LDX	#, PX	Print at X values
	275	342 270	JSR	UPDATE	update X coordinate returned in [X]
	274	2 262	CPX	XV	has value changed?
	47	11	BEQ	SHIPX	if not - no need to calculate
	377	2 262	STX	XV	else update value
	316	2 260	LDX	#, PTABLE	} go compute
	275	341 0	JSR	NCXVAL	new XVAL

25 March 79
ARD

PLOTVC Continued

SKIPX:	316	2	200	LDX	#, PY	} go update Y Value
	275	342	270	JSR	UPDATE	
	274	2	260	CPX	YV	same value
	47	11		BEQ	SKIPY	also - no need to calculate
	377	2	260	STX	YV	else save new value
	316	2	260	LDX	#, PTABLE	} go compute YVAL
	275	341	120	JSR	NCYVAL	
SKIPY:	275	341	255	JSR	NXYPLOT	go plot point
ENTER:	336	112		LDX	VCNTR	} more points to do ?
	11			DEX		
	337	112		STX	VCNTR	
	46	316		BNE	VLOOP	loop until finished
	71			RTS		Return to caller

343 @
CLEAR

B:

A:

Clear Graphic Memory Routine

343 ØSØ

```
CLEAR:  117          CLR A
        306  60       LDA B #60        } Base Address of graphic memory

B:      327  270      STA B TEMP       }
A:      227  271      STA A TEMP+1     } Same Address
        336  270      LDX  TEMP        load INDEX with Address

        157   Ø       CLR  Ø, X        .
        157   1       CLR  1, X        .
        157   2       CLR  2, X        .
        157   3       CLR  3, X        .
        157   4       CLR  4, X        .
        157   5       CLR  5, X        .
        157   6       CLR  6, X        .
        157   7       CLR  7, X        .

        213  1Ø       ADD A #1Ø        update address
        44   35Ø      BCC A            if no carry go to A!
        311   Ø       ADC B #Ø         Add carry
        52   342      BPL B            if Address still < 200,000
                                       then go to B
        71            RTS              return to caller
```

Clear 8 Bytes (Quickly!)

25 March 79
ARB

Calc routine (or use AI:)

Enter with [X] = Table Address

343 114

```
CALC:   246  3      LDA A  3,X
        346  2      LDA B  2,X
        247  1      STA A  1,X
        347  0      STA B  0,X
        253  5      ADD A  5,X
        351  4      ADC B  4,X
        247  3      STA A  3,X
        347  2      STA B  2,X
        71          RTS
```

move N_ to P_

move N_ + Δ to N_

return to caller

343 14
XY FNC

XOP:

NYAI:

YOP:

Main XY FUNCTION Scan Routine

343 146

XY FNCT:	226	102	LDA A	GCSR	Use current status information during plotting
	267	2 271	STA A	CSR	
	226	100	LDA A	OPR	Check for a Y operation
	205	100	BIT A	#100	
	46	25	BNE	YOP	Branch if a Y operation
XOP:	205	20	BIT A	#20	Check for Auto increment
	47	7	BEQ	NYAI	if Not - branch
	316	2 200	LDX	#PY	Go update PY and Auto increment
	15		SEC		
	275	341 360	JSR	AI	
NYAI:	316	2 230	LDX	#PX	go load new value
	275	342 10	JSR	LOADV	
	40	26	BR	NG	
YOP:	205	20	BIT A	#20	check for Auto increment
	47	7	BEQ	NXAI	if not - branch
	316	2 236	LDX	#PX	go update PX and Auto increment NX
	15		SEC		
	275	341 360	JSR	AI	

26 March 79
ABB

```
NXAI:   316  2  200    LDX  #PY      } go load new value
        275  342  10    JSR  LOADV    }
        40   1         BR   N6
        0              ---

N6:     226  100       LDAA OPR      get operation
        205  14        BIT A # 14    check if PL or V set
        46   1         BNE  PORV     yes - branch
        71             RTS           else no Plot/Vector - finished

PORV:   205  40        BIT. A  #40,  Relative mode ?
        47   21        BEQ  VECHK    if not - branch ahead

        316  2  230    LDX  #PX      } else
        275  343  114   JSR  CALC    } vector update PX = NX
                                          NX = NX + Δ
        316  2  232    LDX  #PY      } vector update PY = NY
        275  343  114   JSR  CALC    }      NY = NY + Δ
        40   1         BR   .+3
        0              ---

        206  100       LDA A OPR     get operation
VECHK:  205  10        BIT  A #10    is a vector called for ?
        46   4         BNE  VECTOR   if so - branch
        275  341  230   JSR  PLOTXY   else plot point
        71             RTS           finished
```

```
VECTOR: 316  2  230    LDX  #PX      } go compute |DX|
                                       and its sign
        275 342 60     JSR  ABSD     }

        316  2  200    LDX  #PY      } go compute |DY|
                                       and its sign
        275 342 60     JSR  ABSD     }

        40   2         BRA  .+4

343 310
CHH1:   376  2  236    LDX  DX       } Does DX = DY ?
        274  2  206    CPX  DY       }

        46   60        BNE  CHH2,    if not-skip ahead

        337  112       STX  VCNTR    same as count

        316  2 230     LDX  #PX      }
        206  1         LDA A #1      } set update count = 1
                                       in X direction
        275 342 120    JSR  VSETC    }

        275 342 150    JSR  SETSGN   go set proper sign

        316  2  200    LDX  #PY      }
        206  1         LDA A #1      } set update count = 1
                                       in X direction
        275 342 120    JSR  VSETC    }

        275 342 150    JSR  SETSGN   go set proper sign

        275 342 330    JSR  PLOTVC   go plot Vector
```

26 March 79
ARO

343 353

```
VECTDN:  226 100      LDA  OPR    }
         205  40      BIT  A #40  } check if operation is Relative
         46   14      BNE  DONE     if so - skip load update
         376 2 202    LDX  NY     }
         377 2 200    STX  PY     } update PY
         376 2 232    LDX  NX     }
         377 2 230    STX  PX     } update PX
DONE:    71           RTS
```

344 ØØØ

```
CHK2:    376 2 236    LDX  DX       is DX = Ø ?
         46  37       BNE  CHK3     if not skip ahead
         376 2 206    LDX  DY     }
         337  112     STX  VCNTR  } Same DY as counter

         316 2 230    LDX  #PX    }
         117          CLR  A      } set update value = Ø
                                  } in X direction
         275 342 120  JSR  VSETC  }
         316 2 200    LDX  #PY    }
         206  1       LDA  A #1   } set update value = 1
                                  } in Y direction
         275 342 120  JSR  VSETC  }
         275 342 150  JSR  SET SGN  set proper sign
         275 342 330  JSR  PLOTVC   go plot vector
         176 343 353  JMP  VECTDN   go finish up
```

344 ϕ44

```
CHK3: 376  2 206   LDX  DY      is DY = ϕ ?
       46  43      BNE  CHK4    if not - branch

      376  2 236   LDX  DX      } make count = DX
      337  112     STX  VCNTR   }

      316  2  230  LDX  #PX     }
      206  1       LDA  A #1    } set up data = 1
      275  342  120 JSR: VSETC  }   in x direction
      275  342  150 JSR  SETSGN  set proper sign

      316  2   200  LDX  #PY     }
      117           CLR  A       } set up data = ϕ
      275  342  120 JSR  VSETC   }   in Y direction

      275  342  330 JSR  PLOTVC  go plot vector
      176  343  353 JMP  VECTDN  go finish up
```

26 March 79
ASD

```
344 114
CHK4:   266  2  237    LDA A  DX+1
        366  2  236    LDA B  DX
        260  2  207    SUB A  DY+1
        362  2  206    SBC B  DY
        45   72        BCS    CHK5      if DY>DX branch

        376  2  236    LDX    DX        else DX is longer
        337  112       STX    VCNTR     save DX as Count
        337  106       STX    DSR       and as the divisor
        376  2  246    LDX    DY        move DY to Dividend
        337  104       STX    DND
        275  342  210  JSR    DIVIDE    go compute a 16 bit fraction

        316  2  230    LDX    #PX       set update = 1
        206  1         LDA A  #1        in X direction
        275  342  120  JSR    VSETC
        275  342  150  JSR    SETSGN    set proper sign
```

Determine which is Longer

344 224
CHK5:

```
316 2 200      LDX #PY        }
117            CLR A          } setup Yslop to ∅
275 342 120    JSR VSETC      }
336 110        LDX QT         } load in fractional setup
377 2 216      STX CP3Y       }
316 2 200      LDX #PY        } set the proper sign
275 342 150    JSR SETSGN     }

275 342 330    JSR ; PLOTVC,    go plot vector
176 343 353    JMP VECTDN       go finish up

344 224
CHHS: 376 2 206   LDX DY        DY is larger
337 112           STX VCNTR     same as count
337 106           STX DSR       and as divisor
376 2 236         LDX DX        }
337 104           STX DND       } same DX as Dividend
275 342 210       JSR DIVIDE      go compute 16 bit fraction
```

26 March 79
ASD

```
316 2 230      LDX  #PX      } set X size to zero
117            CLR  A        }
275 342 120    JSR  VSETC
336 120        LDX  QT       } load fractional step
377 2 246      STX  CP3X     }
316 2 230      LDX  #PX      } go set proper sign
275 342 150    JSR  SETSGN   }

316 2 200      LDX  #PY      } load Y step = 1
206 1          LDA  A #1     }
275 342 120    JSR  VSETC    }
275 342 150    JSR  SETSGN   set proper sign

275 342 330    JSR  PLOTVC   go plot Vector
176 343 353    JMP  VECTDN   go finish up
```

Character Writing Handler

344 310

CWRITE:	226 100	LDA A ORR		get code
	26	TAB		save A
	204 4	AND A #4		check reverse bit
	46 71	BNE	REV	if reverse mode – branch
	316 2 262	LDX	#XV	} set up Row to be along X
	337 134	STX	ROWADD	
	376 2 232	LDX	NX	} some value of Row position
	337 136	STX	ROWRST	
	316 2 260	LDX	#YV	} setup Col to be along Y
	337 142	STX	COLADD	
	376 2 202	LDX	NY	} some column value
	337 144	STX	COLST	
	27	TBA		get code
	204 2	AND A #2		check Row direction
	46 5	BNE	AA0	if reversed – branch
	316 0 1	LDX	#1	else indicate (+1)
	40 3	BRA	AA1	skip
AA0:	316 377 377	LDX	#377,377	make (-1)
AA1:	337 140	STX	ROWUPD	save updating value

26 March 79
ABO

```
        27              TBA             get code
        204  1          AND A  #1       Colum reversed ?
        46   5          BNE    AA2      branch if yes
        316  0  1       LDX    #1       else update + (+1)
        40   3          BRA    AA3      skip
AA2:    316  377 377    LDX    # 377, 377   make update (-1)
AA3:    337  146        STX    COLUPD   Save update value
        176  345  336    JMP    CXWRT    Jump to remainder of Routine

REV:    316  2  262     LDX    #XV      }
        337  142        STX    COLADD   }  Colum along X
        376  2  232     LDX    NX       }
        337  144        STX    COLRST   }  colum Reset value
        316  2  260     LDX    #YV      }
        337  134        STX    ROWADD   }  Row along Y
        376  2  202     LDX    NY       }
        337  136        STX    RWRST    }  Row Reset value
```

```
        27          TDA             get code
        204 2       AND A #2        reversed Direction ?
        46  5       BNE AA4         if so - branch
        316 0 1     LDX #1          else update = (+1)
        40  3       BRA AA5         branch
AA4:    316 377 377 LDX #377,377    update = (-1)
AA5:    337 146     STX COLUPD      done updating
        176 345 300 JMP REV2        go finish routine
```

28 March 77
AD

ASCII Select Routine

345 64

SELECT	226	100	LDA A	OPR	get Operation
	205	20	BIT A	#20	check Bit
	47	24	BEQ	RPLCDE	if replacing — branch
	226	101	LDA A	OPR+1	else specifying word
	227	132	STA A	BYTE	Save value for Write function
	137		CLR B		
	110		ASL A		ASCII code * 20₈
	131		ROL B		
	110		ASL A		
	131		ROL B		Computing Address of Character in RAM
	110		ASL A		
	131		ROL B		
	110		ASL A		
	131		ROL B		
	312	20	ORA B	#20	
	227	127	STA A	RAD+1	} save char Address
	327	126	STA B	RAD	
	71		RTS		

RPLC

```
RPLCDE:  204  17     AND A  #17        Mask line select
         326  127    LDA B  RAD+1      get Low byte Addr
         304  360    AND B  #360       mask out line sel
         33          ABA               mask in new select
         227  127    STA A  RAD+1      Save Address
         336  126    LDX    RAD        get Complete Address
         226  101    LDA A  OPR+1      get data
         247  0      STA A  0,X        store data in character
         71          RTS               Return
```

27 March 79
ARB

Read/Write function

345 140

RDWRT:	226	101	LDA A OPR+1	} get Operation
	326	100	LDA B OPR	}
	304	17	AND B #17	Mask High Bits
	332	130	ORA B HADD	OR in High Bits
	227	125	STA A RDADD+1	} some address
	327	124	STA B RDADD	}
	336	124	LDX RDADD	get address
	226	100	LDA A OPR	ge code
	204	20	AND A #20	a write?
	47	4	BEQ RDONLY	if not - branch
	226	132	LDA A BYTE	} else write Byte
	247	0	STA A 0,X	} of Data
RDONLY	356	0	LDX 0,X	} Read data and
	337	24	STX 24	} Send to CPU
	71		RTS	Return to Caller

345 200

```
CONTRL:  226 100      LDA A  OPR      get operation
         205  20      BIT A  #20      clear bit set
          47   5      BEQ    NCLR     branch if not
         275 343  50  JSR    CLEAR    go clear display memory
         226 100      LDA A  OPR      get operation
NCLR:    204  17      AND A  #17      Most high bits
         110          ASL A           shift 4 places
         110          ASL A
         110          ASL A
         110          ASL A
         227 130      STA A  HADD     Some bits for Address of R/W function
         226 101      LDA A  OPR11    } Some new Control
         227 102      STA A  GCSR     }
         205 100      BIT A  #100     check for Display control bit
          47  10      BEQ    OFF      Branch for off
         206 377      LDA A  #377     } setup for graphics
         227 217      STA A  GRAPH    }
         174 000 216  INC    UPDFLG
          71          RTS
OFF:     177 000 217  CLR    GRAPH    } setup for graphic off
         174 000 216  INC    UPDFLG   }
          71          RTS
```

28 March 78
ARD

345 300

```
REV2:   27          TBA          }   test for Row direction
        204 1       AND A #1     }
        46  5       BNE  AA6         if set for (-) branch
        316 0 1     LDX  # 1         else set to (+1)
        40  3       BRA  AA7
AA6     316 377 377 LDX  # 377,377   set for (-1)
AA7     337 140     STX  ROWDPO      save update
        40  7       BR   CXWRT       Branch to Character write routine
```

Character Writing routine

345 330

```
CXWRT:  316 10 20   LDX  # 10, 20    } set up loop counters
        337 120     STX  BT/LT       }

        226 101     LDA A OPR+1      get ASCII code
        137         CLR  B
        110         ASL  A           compute 16. x ASCII
        131         ROL  B           to get Offset from Base Address
        110         ASL  A
        131         ROL  B
```

```
110          ASL A
131          ROL B
110          ASL A
131          ROL B
212 17       ORA A #17      } fiel in remainder of Addr
312 20       ORA B #20      }
227 123      STA A CADR+1   } save the computed Addr
327 122      STA B CADR     }

226 102      LDA A GCSR     } set current status in
267 2 271    STA A CSR      } Plot Table

336 142      LDX COLADD     } set up parameter specifies
326 144      LDA B COLRST   } as column to
347 0        STA B 0,X      } its start values
226 145      LDA A COLRST+1 }
247 1        STA A 1,X      }

336 122      LDX CADR       get character Addr
```

28 March 1979
ARB

```
LINLOP:  246  0      LDA A 0,X      get row of dots from the
                                    the specified character
         227  235    STA A CWORD    and save
         336  134    LDX  ROWADD
         326  136    LDA B ROWRST   } set up parameter
         347  0      STA B 0,X      } specified as Row
         226  137    LDA A ROWRST+1 } to its start values
         247  1      STA A 1,X

DTLOP:   171  0  235  ROL  CWORD    is Bit set in Row
         44   3      BCC  SKIPLT    if not - skip plot
         275  341  244 JSR  VECTXY  go plot the point

SKIPLT   336  134    LDX  ROWADD
         246  1      LDA A 1,X      } update Row position
         233  141    ADD A ROWUPD+1 } according to specification
         247  1      STA A 1,X
         346  0      LDA B 0,X
         331  140    ADC B ROWUPD
         347  0      STA B 0,X

         172  0  120 DEC  BT       Row finished ?
         46   345    BNE  DOTLOP    Loop until row done
         206  10     LDA A #10      } reset row dot count
         227  120    STA A CT       }
```

```
336  142      LDX  COLADD
246  1        LDA  A  1,X
233  147      ADD  A  COLUPD+1
247  1        STA  A  1,X
346  ϕ        LDA  B  ϕ,X
331  146      ADC  B  COLUPD
347  ϕ        STA  B  ϕ,X
```

update Column Position according to specification

```
336  122      LDX  CADR
11            DEX
337  122      STX  CADR
```

update Character Row Address

```
172  ϕ 121    DEC  LT
46   273      BNE  LINLOP
```

how rows to do?
Incremental all rows checked

28 March 1979
ARD

```
        226 100     LDA A ORR   } update along X direction?
        204 20      AND A #20   }
        47  14      BEQ   AA8     if not - skip
        376 2 232   LDX   NX    }
        377 2 230   STX   PX    }  Px = Nx
        376 2 262   LDX   XV    }  Nx = Nx + Xupdate
        377 2 232   STX   NX    }
AA8     226 100     LDA A ORR   } update along Y direction
        204 10      AND A #10   }
        47  14      BEQ   AA9     if not - skip
        376 2 202   LDX   NY    }
        377 2 200   STX   PY    }  Py = Ny
        376 2 260   LDX   YV    }  Ny = Ny + Yupdate
        377 2 202   STX   NY    }
AA9     71          RTS           finished!
```

347
RETURN

GRPH I

347 000

```
RETURN: 17          SEI          Hold interrupts
        206 55      LDA A #55    } enable interrupt to graphics
        227 22      STA A 22     }   Processor
        226 20      LDA A 20     set ready flag
        73          RTI          Return from previous interrupt

GRPH IN: 206 4      LDA A #4     } disable further interrupts
         227 22     STA A: 22    }   to Display processor
         16         CLI          Enable all other interrupts

        206 0       LDA A #0     }
        306 347     LDA B #347   } return address → Return
        66          PSH A        }
        67          PSH B        }

        226 21      LDA A 21     } get Graphic Processor Instruction
        326 20      LDA B 20     }
        227 25      STA A 25     } place at output
        327 24      STA B 24     }
        227 101     STA A OPR+1  } done for processor
        327 100     STA B OPR    }
```

28 March 79
ARD

```
        304 340     AND B  #340     Mask all except Codetype
        53  3       BMI    NXYFCT   an XY Function ?
        176 343 140 JMP    XY FNCT  if so - go do it

NXYFNCT: 301 200    CMP B  #200     else check for Character write
        46  3       BNE    NWCHR    if not - branch
        176 344 316 JMP    CWRITE   else do character write

NWCHR:  301 240     CMP B  #240     check for Select
        46  3       BNE    NSLCT    if not - branch
        176 345 64  JMP    SELECT   else do it

NSLCT:  301 300     CMP B  #300     check for Control Register
        46  3       BNE    NCNTRL   if not - branch
        176 345 200 JMP    CONTROL  else do it

NCNTRL: 176 345 140 JMP    READ     go do Read/Write
```

Software Changes – Installed 30 March 79

Control: after Loading CSR

```
        205  20        BIT A  #20        check reader character set bit
        47   5         BEQ    A1
        275  354  210  JSR    RAMSET      Y, restore Character set

A0:     226  101       LDA A  OPR+1       get control again
A1:     205  100       BIT A  #100        screen set position?
        47   16        BEQ    A3          if not skip to end

        205  40        BIT A  #40         screen on
        47   4         BEQ    OFF         if not – go turn off
        206  377       LDA A  #,377       else set flag for on
        40   1         BRA    A2

OFF:    117            CLR A              set flag for off
A2:     227  217       STA A  GRAPH       store flag
        174  0  216     INC    UPDFLG      make display see change
A3:     71             RTS                Return to caller
```

2 March 79
BBB

Software Changes - Read/write Installed 30 March 79

```
RD/WT:  326  100    LDA B  OPR
        27          TB A
        304  17     AND B  #17
        332  130    ORA B  HADD
        327  124    STA B  RDADD
        326  101    LDA B  OPR+1
        327  125    STA B  RDADD+1
        336  124    LDX    RDADD
        204  20     AND A  #.20
        47   4      BEQ    RDONLY
        226  132    LDA A  BYTE
        247  0      STA A  0,X

RDONLY: 246  1      LDA A  1,X
        227  24     STA A  24
        346  0      LDA B  0,X
        327  25     STA B  25
        71          RTS
```

PRINTER ROM

This version has handler for only the Serial UART Interface

330 0

255	255	'Here' Indicator
330	100	#INIT
330	20	DATA
330	21	SYST
331	55	PCHH

330 20
DATA: 71 RTS

330 21
SYST 316 330 30 LDX H, ID
275 376 220 JSR PRINT
71 RTS

330 30
ID 15 12 40 61 C_R L_F S_P 1
65 40 115 101 5 S_P M A
122 40 40 61 R S_P S_P 1
71 67 71 40 9 7 9 S_P
55 40 120 122 - S_P S P R
111 116 124 105 I N T E
122 40 122 117 R S_P R O
115 15 12 52 M C_R L_F *
0 N_U

31 March 79
AD

```
330 100
INIT:   17          SEI
        316 00      LDX #0,0        } get to DDR's
        337 42      STX 42          }
        316 140 0   LDX #140,0      } set Directions
        337 40      STX 40          }
        316 4 4     LDX #4,4        } get to Data Registers
        337 42      STX 42          }

        316 330 260 LDX #PARKEY     } set interrupt vector for Paper Key
        377 17 372  STX IRQ14       }
        206 7       LDA A #7        } set interrupt control for Paper Key
        227 57      STA A 57        }
        226 55      LDA A 55        clear any pending interrupt

        206 122     LDA A #122      } set printer off!
        227 266     STA A PRNTON    }

        316 330 360 LDX #,PRNTER    } load Printer interrupt vector
        377 17 346  STX IRQ4        }

        316 330 210 LDX #PRNTOF     } go say 'REMOTE OFF'
        275 330 170 JSR MOV         }
        16          CLI             enable interrupts
        71          RTS
```

300 17
MOV:

300 210
PRNTOF

300 224
PRNTON

```
300 170
MOV:    337 362      STX  $F2          Same 'from' Address
        316 40 42    LDX  # 40 42      } set 'TO' Address
        337 360      STX  $F0          }
        206 20       LDA A #20         } set number of Bytes
        227 364      STA A $F4         }
        275 373 320  JSR  FILL         go move data
        71           RTS               return

300 210
PROMOF  40 122 105 115   : /␣REMOTE␣␣␣OFF␣␣␣␣/
        117 124 105 40
        40 117 106 106
        40 40 40 40

300 224
PROMON  40 122 105 115   /␣REMOTE␣␣␣␣ON␣␣␣␣␣/
        117 124 105 40
        40 117 116 40
        40 40 40 40
```

31 March 79
ABD

Paper Advance Key Handler

330 260

PPRKEY:	275	330 270	JSR PAPER	go check key
			RTI	Return from interrupt

330 270

PAPER:	226	57	LDA A 57	is IRQ flag set?
	53	1	BMI SKP	if so - skip
	71		RTS	else button not depressed
SKP:	226	55	LDA A 55	clear IRQ Flag
	226	266	LDA A PRNTON	Flip-flop on/off
	103		Com A	
	227	266	STA A PRNTON	
	201	255	CMP A #255	are we on?
	46	7	BNE P0	if not - go set off
	316	330 230	LDX #, PRNTON	point to on
	206	57	LDA A #57	Control to enable Remote interrupts
	40	5	BRA P1	
P0:	316	330 216	LDX #, PRNTOF	point to off
	206	4	LDA A #4	disable interrupt on Remote
P1:	227	43	STA A 43	set control
	226	41	LDA A 41	clear pending interrupt

330 36
PRINTER

```
206  0        LDA A  #0      }
227  40       STA A  40      }  strobe DAV Flip-Flop
206  40       LDA A  #40     }  to insure next character
227  40       STA A  40      }  will be seen

275  330 120  JSR   NOT      go indicate on/off
71            RTS
```

330 360

```
PRINTER  206  53   LDA A  #53    }  get to DDR
         227  43   STA A  43    }
         137       CLR B        }
         327  41   STA B  41    }  make port for input
         206  57   LDA A  #57   }
         227  43   STA A  43    }  get back to Data Register
         327  40   STA B  40       Enable Reading of Data
         226  41   LDA A  41       Read Data & clear IRQ flag
         306  40   LDA B  #40   }
         327  40   STA B  40    }  disable Read - ready
                                    UART for next character

176  374  240  JMP  IN           go process character as
                                  if from keyboard
```

3 March 79
RDD

```
331 30
PRNDY:  226 223   LDA A  DVAL     get 60ths of a second counter
        204 20    AND A  #20      check Bit for flash rate control
        46  4     BNE    SHIP     if bit set - skip
        206 377   LDA A  #377     else set character = ■
        40  2     BRA    DOP

SHIP:   206 40    LDA A  #40      set character = space
DOP:    221 267   CMP A  PR+1     current character the same?
        47  5     BEQ    PRCHH    if so skip ahead
        267 40 56 STA A  40 56    else reset character
        227 267   STA A  PR+1

331 55
PRCHH   306 255   LDA B  #255     } is Remote ON?
        321 266   CMP B  PRNTA    }
        46  37    BNE    FIN      if not - we are finished
        275 330 270 JSR  PAPER    first check if paper flag has
                                  been depressed
        275 354 260 JSR  TMCH     second check time while
                                  waiting for Remote
```

FIN!

FDNE

```
226   40      LDA A  40        } is terminal ready
53    333     BMI    PRNDY     } if not enter wait loop
204   20      AND A  #20       } is UART ready for character
47    327     BEQ    PRNDY     } if not enter wait loop

206   53      LDA A  #53       } get to DDR
227   43      STA A  43        }
206   377     LDA A  #377      } set I/O for Output
227   41      STA A  41        }
206   57      LDA A  #57       } get back to Data Register
227   43      STA A  43        }

60            TSX              } retrieve character from stack
246   12      LDA A  12,X      }
227   41      STA A  41        send ASCII character

FIN!  206   40    LDA A  #40   } make sure character = space
      221   267   CMP A  PR+1  } when leaving routine
      47    5     BEQ    FDNE
      267   40 56 STA A  40 56
      227   267   STA A  PR+1
FDNE! 71          RTS
```

31 March 79
ARD

```
.IIF    NDF,.M68        .NLIST  ;CROSS-ASSEMBLER NOT LISTED
        ;DURING CHECKOUT DEFINE .M68 TO ENABLE LISTING
.IIF    DF,.M68         .LIST   SRC,SEQ,COM,MD,MC
        ;
        ;
        .TITLE  M6800 CROSS-ASSEMBLER
        ;
.IIF    DF,.M68 .SBTTL  CROSS-ASSEMBLER INTRODUCTION
        ;
.ENABLE ABS
        ;
        ;*************************************************************
        ;*                                                           *
        ;*      MACRO PACKAGE FOR THE MOTOROLA 6800 MICROPROCESSOR   *
        ;*              TO RUN UNDER MACRO 11.                       *
        ;*                                                           *
        ;*      BY ALAN R. BALDWIN                                   *
        ;*                                                           *
        ;*              V03 - OCTOBER 1980                           *
        ;*                                                           *
        ;*************************************************************
        ;
        ;
        ;THE FOLLOWING DIFFERENCES EXIST BETWEEN THIS CROSS-ASSEMBLER
        ;AND MOTOROLA'S M6800 ASSEMBLER
        ;       LABELS MUST TERMINATE WITH A :
        ;       COMMENTS START WITH A ;
        ;       IMMEDIATE MODE IS DENOTED BY A SEPERATE ARGUMENT - #
        ;
        ;
        ;DEFINITION OF ASSEMBLER DIRECTIVES WITH DIFFERENCES
        ;       END - USE       .END    END OF PROGRAM
        ;       EQU - USE  =            EQUATE SYMBOL
        ;       FCB                     FORM SINGLE-BYTE CONSTANT
        ;                               NO MORE THAN 10 ARGUMENTS
        ;       FCC   <ASCII CHRS>      FORM CONSTANT CHARACTERS
        ;       FDB                     FORM DOUBLE-BYTE CONSTANT
        ;                               NO MORE THAN 10 ARGUMENTS
        ;       MON - NOT IMPLEMENTED   RETURN TO MONITOR CONSOLE
        ;       NAM - USE       .SBTTL  PROGRAM NAME
        ;       OPT - NOT IMPLEMENTED   OPTION
        ;       ORG                     ORIGIN
        ;       PAGE - USE      .PAGE   ADVANCE LISTING TO TOP OF PAGE
        ;       RMB                     RESERVE MEMORY BYTES
        ;       SPC - NOT IMPLEMENTED   SPACE N LINES
        ;
        ;
        ;PROCESSOR CONDITION CODE REVIEW
        ;       0 - CARRY BIT (C)
        ;       1 - OVERFLOW BIT (V)
        ;       2 - ZERO BIT (Z)
        ;       3 - NEGATIVE BIT (N)
        ;       4 - INTERRUPT MASK BIT (I)
        ;       5 - HALF CARRY BIT (H)
        ;       6 - ALWAYS 1
        ;       7 - ALWAYS 1
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  SINGLE BYTE 'INHERENT' INSTRUCTIONS
        ;
.MACRO  AINST   H,I
 .MACRO H
 .NLIST SRC
 .BYTE  I
 .LIST  SRC
 .ENDM
.ENDM   AINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
AINST   NOP,    1       ;DO NOTHING
AINST   TAP,    6       ;A TO CC'S
AINST   TPA,    7       ;CC'S TO A
AINST   INX,    10      ;INCREMENT INDEX REGISTER
AINST   DEX,    11      ;DECREMENT INDEX REGISTER
AINST   CLV,    12      ;CLEAR V BIT
```

```
AINST   SEV,    13      ;SET C BIT
AINST   CLC,    14      ;CLEAR C BIT
AINST   SEC,    15      ;SET C BIT
AINST   CLI,    16      ;CLEAR I BIT
AINST   SEI,    17      ;SET I BIT
AINST   SBA,    20      ;ACCA=ACCA-ACCB
AINST   CBA,    21      ;COMPARE ACCA & ACCB
AINST   TAB,    26      ;ACCB=ACCA
AINST   TBA,    27      ;ACCA=ACCB
AINST   DAA,    31      ;DECIMAL ADJUST
AINST   ABA,    33      ;ACCA=ACCA+ACCB
AINST   TSX,    60      ;X=SP+1
AINST   INS,    61      ;SP=SP+1
AINST   PULA,   62      ;PULL A FROM STACK
AINST   PULB,   63      ;PULL B FROM STACK
AINST   DES,    64      ;SP=SP-1
AINST   TXS,    65      ;SP=X-1
AINST   PSHA,   66      ;PUSH A ONTO STACK
AINST   PSHB,   67      ;PUSH B ONTO STACK
AINST   RTS,    71      ;RETURN FROM SUBROUTINE
AINST   RTI,    73      ;RETURN FROM INTERRUPT
AINST   WAI,    76      ;WAIT FOR INTERRUPT
AINST   SWI,    77      ;SOFTWARE INTERRUPT
AINST   NEGA,   100     ;NEGATE A
AINST   COMA,   103     ;COMPLEMENT A
AINST   LSRA,   104     ;LOGICAL SHIFT RIGHT A
AINST   RORA,   106     ;ROTATE RIGHT A
AINST   ASRA,   107     ;ARITHMETIC SHIFT RIGHT A
AINST   ASLA,   110     ;ARITHMETIC SHIFT LEFT A
AINST   ROLA,   111     ;ROTATE LEFT A
AINST   DECA,   112     ;DECREMENT A
AINST   INCA,   114     ;INCREMENT A
AINST   TSTA,   115     ;TEST A
AINST   CLRA,   117     ;CLEAR A
AINST   NEGB,   120     ;NEGATE B
AINST   COMB,   123     ;COMPLEMENT B
AINST   LSRB,   124     ;LOGICAL SHIFT RIGHT B
AINST   RORB,   126     ;ROTATE RIGHT B
AINST   ASRB,   127     ;ARITHMETIC SHIFT RIGHT B
AINST   ASLB,   130     ;ARITHMETIC SHIFT LEFT B
AINST   ROLB,   131     ;ROTATE LEFT B
AINST   DECB,   132     ;DECREMENT B
AINST   INCB,   134     ;INCREMENT B
AINST   TSTB,   135     ;TEST B
AINST   CLRB,   137     ;CLEAR B
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  PUSH AND PULL OPCODES
        ;
.MACRO  PKRNL   I,J
 .IIF   IDN     <J>,A   ...A=0  ;PSH/PUL A
 .IIF   IDN     <J>,B   ...A=1  ;PSH/PUL B
 .BYTE  I+...A          ;INVALID ARGUMENT
.ENDM
        ;
.MACRO  PINST   H,I
 .MACRO H       J
 .NLIST SRC
 PKRNL  I,J
 .LIST  SRC
 .ENDM
.ENDM   PINST
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
PINST   PUL,    62      ;PUL BYTE FROM STACK S=S+1
PINST   PSH,    66      ;PUSH BYTE ONTO STACK S=S-1
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  RELATIVE BRANCH INSTRUCTIONS
        ;
.MACRO  BKRNL   I,J
 ...A=J-.-2
 .IIF   LT,...A+200     .ERROR  ;BRANCH OUT OF RANGE
 .IIF   GE,...A-200     .ERROR  ;BRANCH OUT OF RANGE
 .BYTE  I,...A
.ENDM
```

```
        ;
.MACRO  BINST   H,I
 .MACRO H       J
 .NLIST SRC
 BKRNL  I,J
 .LIST  SRC
 .ENDM
.ENDM   BINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPFERATION
BINST   BRA,    40      ;BRANCH ALWAYS
BINST   BHI,    42      ;BRANCH IF (C=0) AND (Z=0)
BINST   BLS,    43      ;BRANCH IF (C=1) OR (Z=1)
BINST   BCC,    44      ;BRANCH IF (C=0)
BINST   BCS,    45      ;BRANCH IF (C=1)
BINST   BNE,    46      ;BRANCH IF (Z=0)
BINST   BEQ,    47      ;BRANCH IF (Z=1)
BINST   BVC,    50      ;BRANCH IF (V=0)
BINST   BVS,    51      ;BRANCH IF (V=1)
BINST   BPL,    52      ;BRANCH IF (N=0)
BINST   BMI,    53      ;BRANCH IF (N=1)
BINST   BGE,    54      ;BRANCH IF (<N XOR V>=0)
BINST   BLT,    55      ;BRANCH IF (<N XOR V>=1)
BINST   BGT,    56      ;BRANCH IF (Z=0) AND (<N XOR V>=0)
BINST   BLE,    57      ;BRANCH IF (Z=1) OR (<N XOR V>=1)
BINST   BSR,    215     ;BRANCH TO SUBROUTINE
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  INSTRUCTIONS HAVING ONLY ACCX,INDEXED,AND EXTENDED MODES
        ;
        ;
.MACRO  CKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGUMENTS - THEN INDEXED
  .IIF  DIF     <K>,X   .ERROR  ;INDEX BAD
  .IIF  LT,J    .ERROR  ;NEGATIVE OFFSET
  .BYTE I+40,J          ;OFFSET OUT OF RANGE
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARGUMENT - A, B, OR EXTENDED MODE
  ...A=60
  .IIF  IDN     <J>,A   ,...A=0         ;ACCA MODE
  .IIF  IDN     <J>,B   ,...A=20        ;ACCB MODE
  .IIF  NE,...A-60      .BYTE   I+...A
  .IIF  EQ,...A-60      .BYTE   I+...A,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  CINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 CKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   CINST
        ;
        ;

    ;MNEMONIC  OPCODE   ;OPERATION
        ;
CINST   NEG,    100     ;NEGATE
CINST   COM,    103     ;COMPLEMENT
CINST   LSR,    104     ;LOGICAL SHIFT RIGHT
CINST   ROR,    106     ;ROTATE RIGHT
CINST   ASR,    107     ;ARITHMETIC SHIFT RIGHT
CINST   ASL,    110     ;ARITHMETIC SHIFT LEFT
CINST   ROL,    111     ;ROTATE LEFT
CINST   DEC,    112     ;DECREMENT
CINST   INC,    114     ;INCREMENT
CINST   TST,    115     ;TEST
CINST   CLR,    117     ;CLEAR
        ;
        ;
```

```
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  JUMP AND JSR INSTRUCTIONS
        ;
.MACRO  DKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGUMENTS - INDEXED MODE
  .IIF  DIF     <K>,X   .ERROR  ;BAD INDEX
  .IIF  LT,J            .ERROR  ;NEGATIVE OFFSET
  .BYTE I+40,J          ;OFFSET TOO LARGE
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARGUMENT - EXTENDED MODE
  .BYTE I+60,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  DINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 DKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   DINST
        ;
        ;
     ;MNEMONIC  OPCODE  ;OPERATION
DINST   JMP,    116     ;JUMP
DINST   JSR,    215     ;JUMP TO SUBROUTINE
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ALL ACCX INSTRUCTIONS
        ;
.MACRO  EKRNL   I,J,K,L
 ...A=-1
 .IIF   IDN     <J>,A   ,...A=0         ;ACCA MODE
 .IIF   IDN     <J>,B   ,...A=100       ;ACCB MODE
  .IF   GE,...A                 ;ACCX MODES
   .IF  NB,<L>                  ;THREE ARGS - IMMEDIATE/INDEXED
    .IF IDN     <K>,#   ;CHECK IMMEDIATE
     .IIF       EQ      <I>-207 ,.ERROR ;STA #
     .BYTE      I+...A,L
     .MEXIT
    .ENDC
    .IF IDN     <L>,X   ;CHECK INDEXED
     .IIF       LT,K    .ERROR  ;NEGATIVE OFFSET
     .BYTE      I+...A+40,K     ;OFFSET TOO LARGE
     .MEXIT
    .ENDC
    .ERROR      ;BAD INSTRUCTION
    .MEXIT
   .ENDC
   .IF  NB,<K>                  ;TWO ARGS - DIRECT/EXTENDED
    .IIF        EQ,K&177400     .BYTE   I+...A+20,K
    .IIF        NE,K&177400     .BYTE   I+...A+60,K&177400/400,K&377
    .MEXIT
   .ENDC
  .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  EINST   H,I
 .MACRO H       J,K,L
 .NLIST SRC
 EKRNL  I,J,K,L
 .LIST  SRC
 .ENDM
.ENDM   EINST
        ;
        ;
     ;MNEMONIC  OPCODE  ;OPERATION
EINST   SUB,    200     ;SUBTRACT
EINST   CMP,    201     ;COMPARE
EINST   SBC,    202     ;SUBTRACT WITH CARRY
```

```
EINST   AND,    204     ;LOGICAL AND
EINST   BIT,    205     ;BIT TEST
EINST   LDA,    206     ;LOAD ACCUMULATOR
EINST   STA,    207     ;STORE ACCUMULATOR
EINST   EOR,    210     ;EXCLUSIVE OR
EINST   ADC,    211     ;ADD WITH CARRY
EINST   ORA,    212     ;LOGICAL OR
EINST   ADD,    213     ;ADD
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ALL SHORT FORM ACCX INSTRUCTIONS
        ;
.MACRO  SKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGS - IMMEDIATE/INDEXED
  .IF   IDN     <J>,#   ;CHECK IMMEDIATE
   .IIF EQ      <I>-207 ,.ERROR ;STAA #
   .IIF EQ      <I>-307 ,.ERROR ;STAB #
   .BYTE        I,K
   .MEXIT
  .ENDC
  .IF   IDN     <K>,X   ;CHECK INDEXED
   .IIF LT,J    .ERROR  ;NEGATIVE OFFSET
   .BYTE        I+40,J          ;OFFSET TOO LARGE
   .MEXIT
  .ENDC
  .ERROR        ;BAD INSTRUCTION
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARG - DIRECT/EXTENDED
  .IIF  EQ,J&177400     .BYTE   I+20,J
  .IIF  NE,J&177400     .BYTE   I+60,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  SINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 SKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   SINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
        ;
SINST   SUBA,   200     ;SUBTRACT
SINST   SUBB,   300
SINST   CMPA,   201     ;COMPARE
SINST   CPMB,   301
SINST   SBCA,   202     ;SUBTRACT WITH CARRY
SINST   SBCB,   302
SINST   ANDA,   204     ;LOGICAL AND
SINST   ANDB,   304
SINST   BITA,   205     ;BIT TEST
SINST   BITB,   305
SINST   LDAA,   206     ;LOAD ACCUMULATOR
SINST   LDAB,   306
SINST   STAA,   207     ;STORE ACCUMULATOR
SINST   STAB,   307
SINST   EORA,   210     ;EXCLUSIVE OR
SINST   EORB,   310
SINST   ADCA,   211     ;ADD WITH CARRY
SINST   ADCB,   311
SINST   ORAA,   212     ;LOGICAL OR
SINST   ORAB,   312
SINST   ADDA,   213     ;ADD
SINST   ADDB,   313
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  STACK AND INDEX REGISTER INSTRUCTIONS
        ;
.MACRO  FKRNL   I,J,K,L
```

```
 .IF    B,<K>           ;ONE ARG - DIRECT/EXTENDED MODE
  .IF   NB,<J>
   .IIF NE,J&177400     .BYTE   I+60,J&177400/400,J&377
   .IIF EQ,J&177400     .BYTE   I+20,J
   .MEXIT
  .ENDC
  .ERROR        ;BAD INSTRUCTION
  .MEXIT
 .ENDC
 .IF    IDN     <J>,#   ;IMMEDIATE MODE
  .IIF  EQ      <I>-217 ,.ERROR ;STS #
  .IIF  EQ      <I>-317 ,.ERROR ;STX #
  .IIF  NB,<L>  .BYTE   I,K,L
  .IIF  B,<L>   .BYTE   I,K&177400/400,K&377
  .MEXIT
 .ENDC
 .IF    B,<L>
  .IF   NB,<K>
   .IF  IDN     <K>,X           ;INDEXED
    .IIF        LT,J    .ERROR  ;NEGATIVE OFFSET
    .BYTE       I+40,J
    .MEXIT
   .ENDC
  .ENDC
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  FINST   H,I
 .MACRO H       J,K,L
 .NLIST SRC
 FKRNL  I,J,K,L
 .LIST  SRC
 .ENDM
.ENDM   FINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
        ;
FINST   CPX,    214     ;COMPARE TO INDEX
FINST   LDS,    216     ;LOAD STACK REGISTER
FINST   LDX,    316     ;LOAD INDEX REGISTER
FINST   STS,    217     ;STORE STACK REGISTER
FINST   STX,    317     ;STORE INDEX REGISTER
        ;
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ASSEMBLER DIRECTIVES
        ;
.MACRO  FCB     A,B,C,D,E,F,G,H,I,J
.NLIST  SRC
 .IIF   NB,<A>  .BYTE   A
 .IIF   NB,<B>  .BYTE   B
 .IIF   NB,<C>  .BYTE   C
 .IIF   NB,<D>  .BYTE   D
 .IIF   NB,<E>  .BYTE   E
 .IIF   NB,<F>  .BYTE   F
 .IIF   NB,<G>  .BYTE   G
 .IIF   NB,<H>  .BYTE   H
 .IIF   NB,<I>  .BYTE   I
 .IIF   NB,<J>  .BYTE   J
.LIST   SRC
.ENDM   FCB
        ;
.MACRO  FCC     H
.NLIST  SRC
.ASCII  /H/
.LIST   SRC
.ENDM   FCC
        ;
.MACRO  FDB     A,B,C,D,E,F,G,H,I,J
.NLIST  SRC
 .IIF   NB,<A>  .BYTE   A&177400/400,A&377
 .IIF   NB,<B>  .BYTE   B&177400/400,B&377
 .IIF   NB,<C>  .BYTE   C&177400/400,C&377
```

```
 .IIF   NB,<D>  .BYTE   D&177400/400,D&377
 .IIF   NB,<E>  .BYTE   E&177400/400,E&377
 .IIF   NB,<F>  .BYTE   F&177400/400,F&377
 .IIF   NB,<G>  .BYTE   G&177400/400,G&377
 .IIF   NB,<H>  .BYTE   H&177400/400,H&377
 .IIF   NB,<I>  .BYTE   I&177400/400,I&377
 .IIF   NB,<J>  .BYTE   J&177400/400,J&377
.LIST   SRC
.ENDM   FDB
        ;
.MACRO  ORG     H
.NLIST  SRC
.=<H>
.LIST   SRC
.ENDM   ORG
        ;
.MACRO  RMB     H
.NLIST  SRC
.BLKB   H
.LIST   SRC
.ENDM   RMB
        ;
        ;
        ;
.NLIST  TTM     ;PRINTING MODE
.LIST   MD,MEB  ;ENABLE PRINTING OF MACRO EXPANSIONS
.LIST           ;LIST PROGRAM PROPER
        .PAGE
```

```
R PIP
DISP1.MAC=VARSAV.MAC,INIT.MAC,TRMNL1.MAC,TRMNL2.MAC/U
DISP2.MAC=LCLMON.MAC,ESCAPE.MAC,BOOTS.MAC,KUSARB.MAC/U

R MACRO
DISP[40],DISP=M6800,DISP1,DISP2

@DGPH
```

```
        .SBTTL  6800 TERMINAL SOFTWARE
        ;
        ; THE SOFTWARE CONTAINED IN THIS PROGRAM IS FOR THE
        ;OPERATION OF THE 6800 BASED VIDEO BOARD USED
        ;WITH THE ARB-11 PROCESSOR.
        ; THE SOFTWARE SUPPORTS A STANDARD DL-11 TYPE
        ;INTERFACE FORMAT PLUS A SEPERATE INTERFACE FOR
        ;THE GRAPHICS PROCESSOR.
        ;
        ;       TERMINAL INTERFACE
        ;               KEYBOARD CSR = 177560
        ;                        BUF = 177562
        ;                        VEC = 60
        ;               PRINTER  CSR = 177564
        ;                        BUF = 177566
        ;                        VEC = 64
        ;
        ;       GRAPHICS INTERFACE
        ;               RESULT   CSR = 176670
        ;                        BUF = 176672
        ;                        VEC = 170
        ;               COMMAND  CSR = 176674
        ;                        BUF = 176676
        ;                        VEC = 174
        ;
        .PAGE
        .SBTTL  DEFINITION OF I/O REGISTERS
        ;
        ; THE I/O REGISTERS ARE ALL 6820 PIA CIRCUITS
        ;
        ;       REGISTER ADDRESSING
        ; ___XX
        ;    00 I/O REGISTER A (DIRECTION OF A)
        ;    01 I/O REGISTER B (DIRECTION OF B)
        ;    10 STATUS REGISTER A
        ;    11 STATUS REGISTER B
        ;
        ;       REFER TO MC6820/6821 DATA SHEETS FOR DETAILS
        ;       OF THE CONTROL REGISTERS
        ;
        .PAGE
        .SBTTL  I/O REGISTER LOCATIONS
        ;
        DARHA=0         ;DISPLAY ADDRESS REGISTER H (A-PORT)
        DARLB=1         ;DISPLAY ADDRESS REGISTER L (B-PORT)
        PCRA1=2         ;CSR FOR DARHA [INIT  IRQ1]
        PCRB1=3         ;CSR FOR DARHB
        ;
        CARHA=4         ;CURSOR ADDRESS REGISTER H (A-PORT)
        CARLB=5         ;CURSOR ADDRESS REGISTER L (B-PORT)
        PCRA2=6         ;CSR FOR CARHA
        PCRB2=7         ;CSR FOR CARLB [VERTICAL RETRACE  NMI2]
        ;
        CSL=10          ;CURRENT SCAN LINE
        ISL=11          ;INTERRUPT SCAN LINE
        PCRA3=12        ;CSR FOR CSL
        PCRB3=13        ;CSR FOR ISL [SCAN LINE  NMI3]
        ;
        CHARPL=14       ;CHARACTERS PER LINE
        DSPCTL=15       ;DISPLAY CONTROL
                        ;BIT    0 - GO
                        ;       3 - GRAPHIC ENABLE
                        ;       4 - CONTINUOS SCAN
                        ;       5 - VIDEO POLARITY
                        ;       7 - ODD FRAME
                        ;
        PCRA4=16        ;CSR FOR CHARPL [NMI STROBE]
        PCRB4=17        ;CSR FOR DSPCTL [ODD FRAME  NMI1]
        ;
        WDFBA=20        ;WORD DATA FROM BUS <15:08>
        WDFBB=21        ;WORD DATA FROM BUS <07:00>
        PCRA5=22        ;CSR FOR WDFBA [DATA READY  IRQ8]
        PCRB5=23        ;CSR FOR WDFBB
        ;
        WDTBA=24        ;WORD DATA TO BUS <15:08>
        WDTBB=25        ;WORD DATA TO BUS <07:00>
```

```
        PCRA6=26        ;CSR FOR WDTBA [60HZ CLOCK  IRQ2]
        PCRB6=27        ;CSR FOR WDTBB [DATA READY  IRQ9]
        ;
        TDF11=30        ;TERMNAL DATA FROM ARB-11
        TDT11=31        ;TERMINAL DATA TO ARB-11
        PCRA7=32        ;CSR FOR TDF11 [IRQ6]
        PCRB7=33        ;CSR FOR TDT11 [IRQ7]
        ;
        KASCII=34       ;KEYBOARD
        KBUTN=35        ;KEYBOARD BUTTONS
                        ;BIT    0 - REPEAT
                        ;       1 - BREAK
                        ;       2 - PAPER
                        ;       3 - HEREIS
                        ;       4 - TAPE>
                        ;       5 - <TAPE
                        ;
        PCRA8=36        ;CSR FOR KASCII [IRQ5]
        PCRB8=37        ;CSR FOR KBUTN [BELL CONTROL]
        ;
        HA=40           ;PRINTER CONTROL REGISTER
        HB=41           ;PRINTER DATA REGISTER
        PCRA9=42        ;CSR FOR HA
        PCRB9=43        ;CSR FOR HB
        ;
        PHOTO=44        ;PHOTOREADER DATA
        NPRCSR=45       ;NPR CONTROL REGISTER
                        ;BIT    0 - NPR GO
                        ;       1 - BUS C0 CONTROL
                        ;       2 - BUS C1 CONTROL
                        ;       3 - ADDRESS BIT 16
                        ;       4 - ADDRESS BIT 17
                        ;       5 - NXM ERROR
                        ;       6 - BUS GRANT LATE ERROR
                        ;       7 - NPR DONE
                        ;
        PCRA10=46       ;CSR FOR PHOTO [IRQ3]
        PCRB10=47       ;CSR FOR NPR CONTROL [IRQ10]
        ;
        NPRADA=50       ;NPR ADDRESS <15:08>
        NPRADB=51       ;NPR ADDRESS <07:00>
        PCRA11=52       ;CSR FOR NPRADA [TAPE<  IRQ11]
        PCRB11=53       ;CSR FOR NPRADB [TAPE>  IRQ12]
        ;
        NPRDTA=54       ;NPR DATA TO BUS <15:08>
        NPRDTB=55       ;NPR DATA TO BUS <07:00>
        PCRA12=56       ;CSR FOR NPRDTA [HERE IS  IRQ13]
        PCRB12=57       ;CSR FOR NPRDTB [PAPER  IRQ14]
        ;
        NPRDFA=60       ;NPR DATA FROM BUS <15:08>
        NPRDFB=61       ;NPR DTAT FROM BUS <07:00>
        PCRA13=62       ;CSR FOR NPRDFA [BREAK  IRQ15]
        PCRB13=63       ;CSR FOR NPRDFB [REPEAT  IRQ16]
        ;
        .PAGE
        .SBTTL  MEMORY LAYOUT
        ;
        ;0-77            I/O REGISTERS (6820 PIA'S)
        ;
        ;100-377         DIRECT VARIABLES
        VARSAV=100              ;FIRST LOCATION
        .=VARSAV
        ;
        ;       SPECIFY GLOBLES FIRST
        ;
GRAPH:  .BYTE   0       ;GRAPHICS ON/OFF
RMTON:  .BYTE   0       ;REMOTE LINK ON/OFF FLAG
MODE:   .BYTE   0       ;TEXT MODE
CHARA:  .BYTE   0       ;CHARACTER FROM KEYBOARD ETAL
CHARB:  .BYTE   0       ;CHARACTER BEING PROCESSED BY .PRINT
CHRFLG: .BYTE   0       ;CTLP PRINT INHIBIT FLAG
UPDFLG: .BYTE   0       ;SCREEN UPDATE FLAG
        ;
        ;
        ;
T.60TH: .BYTE   0       ;60TH OF SECOND COUNTER
```

```
T.SCND: .BYTE   0       ;1 SECOND FLAG
T.CHAR: .BYTE   0       ;CHARACTER COUNT THIS INTERVAL
NALL:   .BYTE   0       ;CHAEARCTERS ALLOWED PER 60'TH
DATINH: .BYTE   0       ;DATA INHIBITED FLAG
ONLINE: .BYTE   0       ;ONLINE/LOCAL FLAG
LSTFLG: .BYTE   0       ;LIST GRAPHICS FLAG
ESCFLG: .BYTE   0       ;ESCAPE SEQUENCE FLAG
FL.TO:  .BYTE   0,0     ;FILL TO ADDRESS
FL.FRM: .BYTE   0,0     ;FILL FROM ADDRESS
FL.CNT: .BYTE   0       ;FILL COUNTER
LOCK:   .BYTE   0       ;SCROLL LOCK FLAG
DRCTN:  .BYTE   0       ;SCROLL DIRECTION
CLPNTR: .BYTE   0,0     ;ADDRESS OF CURRENT CHARACTER COUNT
CLBASE: .BYTE   0,0     ;ADDRESS OF CURRENT INPUT LINE
PLPNTR: .BYTE   0,0     ;ADDRESS OF CHARACTER COUNT OF LAST DISPLAY LINE
PLBASE: .BYTE   0,0     ;ADDRESS OF FIRST CHARACTER OF LAST DISPLAY LINE
OVBASE: .BYTE   0,0     ;ADDRESS OF LAST TEXT LINE BEFORE OVER TOP
GLBASE: .BYTE   0,0     ;BASE ADDRESS FOR LINE MODE
GLCNT:  .BYTE   0,0     ;CURSOR POSITION ABOVE GLBASE
GPCNT:  .BYTE   0       ;COLUMN POSITION IN CURRENT LINE
LINES:  .BYTE   0       ;TEXT LINES IN DISPLAY
TXTCBS: .BYTE   0       ;TWO BYTE CURRENT CONTROL BLOCK ADDRESS
CURDSP: .BYTE   0       ;LOW ORDER OF TXTCBS
PGMSTK: .BYTE   0,0     ;SAVE STACK TEMPORARY
CURLOC: .BYTE   0,0     ;CURSOR ADDRESS
CURCNT: .BYTE   0       ;CHARACTER POSITION IN CURRENT LINE
DCTL:   .BYTE   0       ;CONTROL CHARACTER PRINT FLAG
BCRSTK: .BYTE   0,0     ;FIRST DISPLAY CONTROL BLOCK
CRSTK:  .BYTE   0,0     ;DISPLAY POINTER TO NEXT DISPLAY CONTROL BLOCK
DSP0:   .BYTE   0,0     ;NEXT DISPLAY CONTROL BLOCK DATA
DSP1:   .BYTE   0,0
DSP2:   .BYTE   0,0
IBIS:   .BYTE   0       ;OR'D WITH IN DATA
IBIC:   .BYTE   0       ;AND'D WITH IN DATA
OBIS:   .BYTE   0       ;OR'D WITH OUT DATA
OBIC:   .BYTE   0       ;AND'D WITH OUT DATA
RMTFLS: .BYTE   0       ;CURRENT REMOTE FLASH CHARACTER
EJSR:   .BYTE   0,0     ;TRANSFER ADDRESS INTO ESCAPE MONITOR
ESCH:   .BYTE   0       ;ESCAPE SCANNER
ESCL:   .BYTE   0
LJSR:   .BYTE   0,0     ;TRANSFER ADDRESS INTO LOCAL MONITOR
LCLH:   .BYTE   0       ;LOCAL @= SCANNER
LCLL:   .BYTE   0
TEMPLA: .BYTE   0,0     ;TEMPORARIES USED BY LOCAL MONITOR
TEMPLB: .BYTE   0,0
TEMPLC: .BYTE   0,0
VH:     .BYTE   0
VL:     .BYTE   0
VAL:    .BYTE   0
TEMPEA: .BYTE   0,0     ;TEMPORARIES USED BY ESCAPE MONITOR
TEMPEB: .BYTE   0,0
GTABLE: .BYTE   0,0     ;SERVICE ROUTINE ADDRESS
GCNT:   .BYTE   0       ;COUNTER
TEMPGA: .BYTE   0,0     ;TEMPORARIES USED BY LNMODE
TEMPGB: .BYTE   0,0
LRVEC:  .BYTE   0,0     ;SERVICE ROUTINE ADDRESS
LSTPNT: .BYTE   0,0     ;STRING POINTER
LDRPKX: .BYTE   0,0     ;LOADER SAVE POINTER
TRNPKX: .BYTE   0,0     ;TRANSFER TEMPORARY
TRNCNT: .BYTE   0,0     ;TRANSFER WORD COUNTER
        VARSAV=.
        ;
        .PAGE
        .SBTTL  MEMORY DEFINITIONS
        ;
        ;       THE FOLLOWING AREA IS SET UP TO CONTAIN
        ;       THE PARAMETERS TO CONTROL THE DISPLAY
        ;       HARDWARE FOR EACH TEXT LINE OR GRAPHICS
        ;       AREA TO BE DISPLAY. THE ORGANIZATION
        ;       IS AS FOLLOWS:
        ;
        ;  BYTES #1&2   ADDRESS OF FIRST BYTE OF DATA TO BE DISPLAYED
        ;  BYTE #3      BYTES TO DISPLAY IN THIS LINE
        ;  BYTE #4      DISPLAY CONTROL BYTE
        ;  BYTE #5      NEW LINE NUMBER
        ;  BYTE #6      LINE NUMBER FOR NEXT INTERRUPT
```

```
        ;
TXGRCD  =400            ;TEXT LINE CONTROL AREA
CDBOT   =TXGRCD&377             ;LOW ORDER ADDRESS OF BOTTOM
CDTOP   =<TXGRCD+234>&377       ;LOW ORDER ADDRESS OF LAST CONTROL BLOCK
TXTSTP  =642            ;STOP TEXT CONTROL AREA
CLSRT   =650            ;CONTROL LINE START AREA
CLSTP   =656            ;CONTROL LINE STOP AREA
GPSRT   =664            ;GRAPHICS START AREA
GPSTP   =672            ;GRAPHICS STOP AREA
        ;
        ;
        ;       THE FOLLOWING AREA IS A LIST OF POINTERS
        ;       TO THE CONTROL BLOCKS BEING DISPLAYED AT THE
        ;       CURRENT TIME
        ;
DSPSTK  =700            ;DISPLAY STACK EXTENDS FROM 700 TO 777
        ;
        LSTRNG=1000     ;STRING AREA FOR LOCAL MONITOR
        ESTRNG=1120     ;ESCAPE MONITOR STRING AREA
        YTABLE=1200     ;GRAPHIC Y TABLE
        XTABLE=1230     ;GRAPHIC X TABLE
        PTABLE=1260     ;PLOTTING TABLE
        GRPVAR=1400     ;GRAPHIC VARIABLE AREA
        ;
STACK   =6077           ;THE PROCESSOR STACK AREA IS FROM 6000 TO 6077

        ;
        ;       THE FOLLOWING AREAS AND DEFINITIONS
        ;       ARE USED BY THE DISPLAY SECTION
        ;
CLPBOT  =6100           ;BUFFER AREA CONTAINING CHARS IN LINE
CLPTOP  =7731           ;LAST BYTE ADDRESS
        ;
        ;
        ; THESE MEMORY LOACATIONS CONTAIN THE
        ;INTERRUPT VECTOR ADDRESSES AS INITIALIZED
        ;BY THE RUNNING PROGRAM
        ;
        ; NON-MASKABLE INTERRUPT VECTORS
        ;
        NMI1    =7732           ;ODD FRAME
        NMI2    =7734           ;VERTICAL RETRACE
        NMI3    =7736           ;SCAN LINE
        ;
        ; NORMAL INTERRUPT VECTORS
        ;
        IRQ1    =7740           ;ARB-11 BUS INIT
        IRQ2    =7742           ;60 HZ CLOCK
        IRQ3    =7744           ;PHOTO READER DATA
        IRQ4    =7746           ;PRINTER CONTROL
        IRQ5    =7750           ;KEYBOARD DATA
        IRQ6    =7752           ;DATA FROM ARB-11
        IRQ7    =7754           ;DATA TAKEN BY ARB-11
        IRQ8    =7756           ;DISPLAY DATA FROM ARB-11
        IRQ9    =7760           ;DISPLAY DATA TO ARB-11 TAKEN
        IRQ10   =7762           ;ARB-11 BUS NPR DONE
        IRQ11   =7764           ;SCROLL  <TAPE
        IRQ12   =7766           ;SCROLL  >TAPE
        IRQ13   =7770           ;HERE IS BUTTON
        IRQ14   =7772           ;PAPER BUTTON
        IRQ15   =7774           ;BREAK BUTTON
        IRQ16   =7776           ;REPEAT BUTTON
        ;
CHRAM   =10000          ;4096 BYTE CHARACTER DEFINITION BUFFER
                        ;CONTAINING 256 CHARACTER DEFINITIONS
        ;
CTLINE  =20000          ;80 CHARACTER CONTROL LINE
CLBOT   =20120          ;4016 CHARACTER TEXT AREA
CLTOP   =27657          ;LAST TEXT LINE LOCATION
CL.TOP  =27660          ;LAST PLUS 1
        ;
GPBOT   =30000          ;20480 BYTE GRAPHICS AREA
                        ;ALLOWS A 640(X) BY 256(Y) DISPLAY
EVNSCN  =30000          ;SCAN ADDRESS FOR EVEN SCANS
ODDSCN  =54000          ;SCAN ADDRESS FOR ODD SCANS
GPTOP   =77777          ;LAST DISPLAY BYTE
```

```
        ;
        .PAGE
        .SBTTL  ROM LOCATIONS
        ;
        CHROM1  =100000 ;CHARACTER ROM #1 (2K BYTES)
        CHROM2  =104000 ;CHARACTER ROM #2 (2K BYTES)
        ;
        ;       KUS5A ROM FROM 110000-117777 (4K BYTES)
        ;
        ;       GRAPHICS SYSTEM FROM 140000-147777 (4K BYTES)
        GPINIT  =140000         ;INIT ENTRY POINT
        GRPLOT  =140003         ;GRAPHIC ENTRY POINT
        ;
        ;       SPECIAL VECTOR GRAPHICS ROM
        ;       IS FROM 150000-153777 (2K BYTES)
        ;
        ;       SYSTEM SOFTWARE FROM 160000-177777 (8K BYTES)
        ;
        ;
        ;       SYSTEM VECTORS
        ;
        .=177770
        FDB     START   ;IRQ (DUMBY)
        FDB     START   ;SWI
        FDB     START   ;NMI (DUMBY)
        FDB     START   ;POWER FAIL RESTART
        ;
        ;
        PGMSAV=160000
        .=PGMSAV        ;PROGRAM ORIGIN
        ;
```

```
        .PAGE
        .SBTTL  INITIALIZATION ROUTINES
        ;
START:  SEI                     ;HOLD INTERRUPTS DURING SETUP
        LDS     #,STACK         ;STACK AREA
        ;
        .SBTTL  CLOCK INTERRUPT SETUP
        ;
CLKO:   LDX     #,CLKINT        ;ADDRESS OF SERVICE ROUTINE
        STX     IRQ2            ;SAVE AT VECTOR LOCATION
        LDA A   #,5             ;SET CLOCK INTERRUPT CONTROL
        STA A   PCRA6
        LDA A   WDTBA           ;CLEAR ANY PENDING INTERRUPT
        ;
        ; SET UP TIME IN CONTROL LINE
        ;
        LDA A   #,40            ;SPACE CHARACTER
        LDX     #,CTLINE        ;LOCATION OF CONTROL LINE
1$:     STA A   0,X             ;CLEAR OUT LINE
        INX
        CPX     #,CTLINE+70     ;UP TO CLOCK AREA ?
        BNE     1$              ;LOOP UNTIL ALL CLEARED
        ;
        LDA B   #,'M            ;CHECK IF TIME THERE
        CMP B   CTLINE+102
        BEQ     DSINIT          ;IF SO - SKIP TIME INIT
2$:     STA A   0,X             ;ELSE CLEAR REST OF THE LINE
        INX
        CPX     #,CLBOT
        BNE     2$              ;LOOP UNTIL CLEAR
        LDX     #,CTLINE+70     ;ADDRESS OF TIME DISPLAY
        LDA A   #,'0            ;GET ZERO CHARACTER
        STA A   0,X             ;STORE ZERO'S
        STA A   1,X
        STA A   3,X
        STA A   4,X
        STA A   6,X
        STA A   7,X
        LDA A   #,':            ;GET : CHARACTER
        STA A   2,X             ;STORE :'S
        STA A   5,X
        LDA A   #,'A            ;STORE A
        STA A   11,X
        LDA A   #,'M            ;STORE M
        STA A   12,X
        ;
        ; TIME DISPLAY = 00:00:00 AM
        ;
        .PAGE
        .SBTTL  DISPLAY SYSTEM INITIALIZATION
        ;
DSINIT: JSR     RAMSET          ;SET UP CHARACTER RAM
        ;
        LDX     #,CLPBOT        ;INITIALIZE POINTERS
        STX     CLPNTR
        STX     PLPNTR
        LDA A   #,1             ;1 CHARACTER LINES
1$:     STA A   0,X             ;LOAD IN NUMBER
        INX                     ;UPDATE ADDRESS
        CPX     #,CLPTOP+1      ;END OF REGION ?
        BNE     1$              ;LOOP UNTIL FINISHED
        LDX     #,CLBOT         ;GET TEXT AREA
        STX     CLBASE          ;INITIALIZE POINTERS
        STX     PLBASE
        LDA A   #,40            ;PUT SPACES IN ALL CHARACTERS
2$:     STA A   0,X             ;PLACE SPACES
        INX                     ;UPDATE ADDRESS
        CPX     #,CLTOP+120     ;END OF TEXT AREA ?
        BNE     2$              ;LOOP UNTIL FINISHED
        LDX     #,TXGRCD        ;CONTROL BLOCK AREA
        STX     TXTCBS
        CLR A
        STA A   LOCK            ;LOCK DISPLAY TO INCOMING DATA
        STA A   GRAPH           ;GRAPHICS OFF
        INC A
        STA A   CURCNT          ;CURRENT CHARACTER COUNT
```

```
        STA A   UPDFLG          ;DO AN IMMEDIATE UPDATE
        ;
        .PAGE
        ;
        ;       TEXT STOP CONTROL BLOCK
        ;
        LDX     #,CLBOT         ;DUMBY FIRST CHARACTER ADDRESS
        STX     TXTSTP
        LDX     #,1,0           ;TURN DISPLAY OFF
        STX     TXTSTP+2
        LDX     #,7,26          ;RESTART AT CONTROL LINE
        STX     TXTSTP+4
        ;
        ;       CONTROL LINE START CONTROL BLOCK
        ;
        LDX     #,CTLINE        ;ADDRESS OF FIRST CHARACTER
        STX     CLSRT
        LDX     #,120,1         ;80 CHARACTERS AND DISPLAY ON
        STX     CLSRT+2
        LDX     #,7,17          ;NEXT INTERRUPT AT 17
        STX     CLSRT+4
        ;
        ;       CONTROL LINE STOP CONTROL BLOCK
        ;
        LDX     #,CLBOT         ;DUMBY ADDRESS
        STX     CLSTP
        LDX     #,1,0           ;DISPLAY OFF
        STX     CLSTP+2
        LDX     #,7,377         ;INHIBIT FURTHER INTERRUPTS
        STX     CLSTP+4
        ;
        ;       GRAPHICS START CONTROL BLOCK
        ;
        LDX     #,EVNSCN        ;EVEN SCAN
        STX     GPSRT
        LDX     #,120,31        ;640 BITS AND CONTINUOUS SCAN
        STX     GPSRT+2
        LDX     #,377,177       ;128 LINE SCAN
        STX     GPSRT+4
        ;
        ;       GRAPHICS STOP CONTROL BLOCK
        ;
        LDX     #,EVNSCN        ;EVEN SCAN
        STX     GPSTP
        LDX     #,120,0         ;DISPLAY OFF
        STX     GPSTP+2
        LDX     #,377,7         ;RESTART IN 7 LINES
        STX     GPSTP+4
        ;
        .PAGE
        ;
        ;       SET UP DISPLAY STACK FOR TEXT
        ;
        JSR     LDDATA          ;DO FIRST LINE
        LDA A   #,25.           ;SET FOR 25 UPDATES
3$:     PSH A                   ;SAVE COUNT
        INC     CLPNTR+1        ;GO TO NEXT LINE
        INC     CLBASE+1
        JSR     UPDTXT          ;UPDATE THIS LINE
        PUL A                   ;GET COUNT
        DEC A
        BNE     3$              ;LOOP UNTIL FINISHED
        ;
        .PAGE
        .SBTTL  DISPLAY CONTROL REGISTER SETTUP
        ;
        LDX     #,0             ;CLEAR ALL REGISTERS
        STX     PCRA1           ;DISPLAY ADDRESS CONTROL
        STX     PCRA2           ;VERTICAL RETRACE CONTROL
        STX     PCRA3           ;SCAN INTERRUPT CONTROL
        STX     PCRA4           ;DISPLAY CONTROL
        LDX     #,377,377       ;DATA DIRECTION - OUT
        STX     DARHA           ;DISPLAY ADDRESS REGISTER
        STX     CARHA           ;CURSOR ADDRESS REGISTER
        LDX     #,0,377         ;SCAN COUNTER - IN
        STX     CSL             ;INTERRUPT LINE - OUT
```

```
        LDX     #,377,177       ;CHARACTERS PER LINE
        STX     CHARPL          ;DISPLAY CONTROL
        LDX     #,4,4           ;SET TO ACCESS DATA REGISTERS
        STX     PCRA1
        STX     PCRA2
        STX     PCRA3
        STX     PCRA4
        LDA A   #,377           ;INHIBIT SCAN LINE INTERRUPTS
        STA A   ISL
        LDX     #,1,0           ;DISPLAY OFF
        STX     CHARPL
        JSR     CHMODE          ;SET UP IN CHAR MODE
        LDX     CLBASE          ;GET POSITION
        STX     CURLOC          ;SET CURSOR POINTER
        STX     CARHA
        ;
        ;       SET UP INTERRUPT VECTORS
        ;
        LDX     #,DISINT        ;LINE INTERRUPT HANDLER
        STX     NMI3            ;STORE AT VECTOR LOCATION
        LDX     #,VERTIN        ;RETRACE HANDLER
        STX     NMI2            ;STORE AT VECTOR LOCATION
        LDX     #,54,4          ;SET UP NMI STROBE
        STX     PCRA4
        LDX     #,4,7
        STX     PCRA3           ;ENABLE LINE SCAN INTERRUPT
        STX     PCRA2           ;ENABLE RETRACE INTERRUPT
        ;
        ;       DISPLAY IS ON !
        ;
        .PAGE
        .SBTTL  BREAK KEY SET UP
        ;
BRKSET: LDX     #,BRKINT        ;SET UP INTERRUPT VECTOR
        STX     IRQ15
        LDA A   #,7             ;ENABLE INTERRUPT
        STA A   PCRA13
        LDA A   NPRDFA          ;CLEAR PENDING INTERRUPT
        LDA A   #,377           ;PLACE ON LINE
        STA A   ONLINE
        JSR     BREAK           ;DO A BREAK INTO LOCAL
        ;
        .PAGE
        .SBTTL  DATAIN SET UP
        ;
DATSET: LDX     #,TERMDI        ;ADDRESS OF INTERRUPT HANDLER
        STX     IRQ6            ;PLACE IN VECTOR LOCATION
        CLR     PCRA7           ;CLEAR CONTROL REGISTER
        CLR     TDF11           ;DIRECTION IN
        LDA A   #,55            ;SET TO ACCESS DATA AND INTERRUPT
        STA A   PCRA7
        LDA A   TDF11           ;CLEAR PENDING INTERRUPT
        LDA A   #,12            ;SET UP FOR 600 CHARACTERS/SECOND
        STA A   NALL
        ;
        .PAGE
        .SBTTL  KEYBOARD SET UP
        ;
KBSET:  LDX     #,RPINT         ;REPEAT BUTTON INTERRUPT HANDLER
        STX     IRQ16
        LDX     #,KBINT         ;KEYBOARD INTERRUPT HANDLER
        STX     IRQ5
        LDX     #,PHRDR         ;PHOTOREADER INTERRUPT HANDLER
        STX     IRQ3
        ;
        ;       SET UP KEYBOARD
        ;
        CLR     PCRA8           ;CLEAR KEYBOARD CONTROL
        CLR     KASCII          ;SET DIRECTION IN
        LDA A   #,7             ;SET TO INTERUPT ON EVERY CHARACTER
        STA A   PCRA8
        LDA A   KASCII          ;CLEAR PENDING INTERRUPT
        ;
        ;       SET UP BELL
        ;
        CLR     PCRB8           ;CLEAR SWITCH CONTROL
```

```
        CLR     KBUTN           ;SET FOR INPUT
        LDA A   #,54            ;SET TO STROBE BELL
        STA A   PCRB8           ;ON DUMBY WRITE TO KBUTN
        ;
        ;       SET UP REPEAT BUTTON
        ;
        LDA A   #,370
        AND A   PCRB13          ;MASK BITS 0,1,&2
        ORA A   #,5             ;REPEAT CONTROL
        STA A   PCRB13
        LDA A   NPRDFB          ;CLEAR PENDING INTERRUPT
        ;
        ;       SET UP DATAOUT
        ;
        CLR     PCRB7           ;CLEAR OUT CONTROL
        LDA A   #,377           ;SET FOR OUTPUT
        STA A   TDT11
        LDA A   #,54            ;READY FLAG STROBE
        STA A   PCRB7
        ;
        ;       PHOTOREADER SET UP
        ;
        CLR     PCRA10          ;CLEAR PHOTOREADER CONTROL
        CLR     PHOTO           ;SET FOR INPUT
        LDA A   #,7             ;SET FOR INTERRUPT
        STA A   PCRA10
        LDA A   PHOTO           ;CLEAR PENDING INTERRUPT
        ;
        ;       INITIALIZE MASKING VARIABLES
        ;
        CLR A
        STA A   IBIS            ;OR'D WITH INCOMING DATA
        STA A   OBIS            ;OR'D WITH OUTGOING DATA
        LDA A   #,377
        STA A   IBIC            ;AND'D WITH INCOMING DATA
        STA A   OBIC            ;AND'D WITH OUTGOING DATA
        ;
        ;       CONTROL CHARACTER PRINTING
        ;
        CLR     DCTL            ;ALLOW PRINTING
        ;
        .PAGE
        .SBTTL  SCAN BUTTON SET UP
        ;
SETSCN: LDX     #,SCRLUP        ;SCROLL UP INTERRUPT HANDLER
        STX     IRQ12
        LDX     #,SCRLDN        ;SCROLL DOWN INTERRUPT HANDLER
        STX     IRQ11
        LDA A   #,5             ;ENABLE INTERRUPTS
        STA A   PCRA11
        STA A   PCRB11
        LDA A   NPRADA          ;CLEAR PENDING INTERRUPT
        LDA A   NPRADB          ;CLEAR PENDING INTERRUPT
        ;
        .PAGE
        .SBTTL  REMOTE INTERFACE SETUP
        ;
RMTSET: LDX     #,0             ;GET TO DIRECTION REGISTERS
        STX     PCRA9
        LDX     #,140,0         ;SET DIRECTIONS
        STX     HA
        LDX     #,4,4           ;GET TO DATA REGISTERS
        STX     PCRA9
        LDX     #,PPRKEY        ;SETUP INTERRUPT VECTOR
        STX     IRQ14
        LDA A   #,7             ;SET INTERRUPT CONTROL
        STA A   PCRB12
        LDA A   NPRDTB          ;CLEAR PENDING INTERRUPT
        CLR     RMTON           ;REMOTE IS OFF
        LDX     #,REMOTE        ;LOAD REMOTE INTERRUPT VECTOR
        STX     IRQ4
        LDX     #,MSGROF        ;LOAD 'REMOTE  OFF' MESSAGE
        JSR     RMTMOV
        ;
        .PAGE
        .SBTTL  OTHER INITS
```

```
        ;
        JSR     GPINIT          ;INITIALIZE GRAPHICS SYSTEM
        ;
        LDX     #,LCLMON        ;SET UP TRANSFER VECTORS
        STX     LJSR
        LDX     #,ESCAPE
        STX     EJSR
        ;
        CLR     MODE            ;CHMODE INITIALLY
        CLR     LSTFLG          ;DISABLE LIST GRAPHICS
        CLR     ESCFLG          ;DISABLE ESCAPE MONITOR
        ;
        ;       END OF INITIALIZATION
        ;
        CLI                     ;ALLOW INTERRUPTS
        ;
FREVER: BRA     FREVER
        BRA     FREVER
        ;
```

```
        .PAGE
        .SBTTL  CLOCK INTERRUPT HANDLER
        ;
CLKINT: LDA A   DATINH          ;DATA INHIBITED ?
        BEQ     1$              ;IF NOT - SKIP
        LDA A   #,55            ;RESTART TRANSFERS FROM ARB-11
        STA A   PCRA7
        LDA A   TDF11           ;SET READY FLAG
        CLR     DATINH          ;ENABLED FLAG
1$:     BSR     TMCHEK          ;CHECK TIME
        CLR     T.CHAR          ;CHARACTER COUNT FOR NEXT INTERVAL
        RTI                     ;RETURN FROM INTERRUPT
        ;
TMCHEK: DEC     T.60TH          ;ANOTHER 60TH
        BNE     1$              ;NOT 1 SECOND - SKIP
        LDA A   #,60.           ;60 HZ CLOCK
        STA A   T.60TH          ;RESET COUNTER
        INC     T.SCND          ;UPDATE SECOND FLAG
1$:     LDA A   WDTBA           ;CLEAR INTERRUPT FLAG
        RTS                     ;RETURN TO CALLER
        ;
        ;       PROGRAM TIME CHECK ROUTINE
        ;
TMCK:   TST     PCRA6           ;CLOCK FLAG SET ?
        BPL     1$              ;IF NOT - SKIP
        PSH A                   ;SAVE A
        BSR     TMCHEK          ;CHECK TIME
        PUL A                   ;RESTORE A
1$:     RTS                     ;RETURN  TO CALLER
        ;
        .PAGE
        .SBTTL  TIME CLOCK ROUTINE
        ;
        ; THIS ROUTINE IS EXECUTED DURING VERTICAL
        ;RETRACE TO PREVENT NOTICABLE FLECKS ON THE SCREEN DISPLAY
        ;
TIMCLK: LDA A   T.SCND          ;A SECOND YET ?
        BEQ     1$              ;IF NOT - FINISHED
        CLR     T.SCND          ;RESET FLAG
        LDX     #,CTLINE+70     ;TIME SPACE AREA
        INC     7,X             ;UPDATE SECONDS
        LDA A   #,'9            ;ARE WE PAST 9
        CMP A   7,X
        BGE     1$              ;IF NOT - FINISHED
        LDA B   #,'0            ;RESTORE DIGIT TO 0
        STA B   7,X
        INC     6,X             ;ON CARRY - UPDATE 10'S
        LDA A   #,'5            ;ARE WE PAST 5 TENS ?
        CMP A   6,X
        BGE     1$              ;IF NOT - FINISHED
        STA B   6,X             ;RESTORE DIGIT TO 0
        INC     4,X             ;IF CARRY - UPDATE MINUTES
        LDA A   #,'9            ;ARE WE PAST 9 MINUTES ?
        CMP A   4,X
        BGE     1$              ;IF NOT - FINISHED
        STA B   4,X             ;RETORE DIGIT TO 0
        INC     3,X             ;IF CARRY - UPDATE 10'S
        LDA A   #,'5            ;ARE WE PAST 5 TENS ?
        CMP A   3,X
        BGE     1$              ;IF NOT - FINISHED
        STA B   3,X             ;RESTORE DIGIT TO 0
        INC     1,X             ;UPATE HOURS
        LDA A   #,'1            ;ARE WE PAST 10 HOURS
        CMP A   0,X
        BEQ     2$              ;IF SO - USE SPECIAL CHECKS
        LDA A   #,'9            ;ARE WE PAST 9 HOURS ?
        CMP A   1,X
        BGE     1$              ;IF NOT - FINISHED
        STA B   1,X             ;RESTORE DIGIT TO 0
        INC     0,X             ;UPDATE 10'S
1$:     RTS                     ;FINISHED
2$:     LDA A   #,'2            ;PAST 2 HOURS ?
        CMP A   1,X
        BGT     1$              ;IF NOT - FINISHED
        BEQ     3$              ;IF AT 12 HOURS - SKIP
        STA B   0,X             ;ELSE COUNT TO 01 HOURS
```

```
        INC B
        STA B   1,X
        RTS                     ;FINISHED
3$:     LDA A   #,'A            ;AM OR PM ?
        CMP A   11,X
        BEQ     4$              ;ON AM - SKIP
        STA A   11,X            ;ELSE MAKE AM
        RTS                     ;FINISHED
4$:     LDA A   #,'P            ;MAKE PM
        STA A   11,X
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  CHARACTER RAM SET UP
        ;
RAMSET: LDX     #,CHRAM         ;RAM AREA
        CLR     0,X             ;CLEAR FIRST 4
        CLR     1,X
        CLR     2,X
        CLR     3,X
        LDX     #,CHRAM+4       ;RESTORE ADDRESS
        STX     FL.TO
        LDX     #,CHROM1        ;ROM AREA
        STX     FL.FRM
1$:     JSR     TMCK            ;CHECK CLOCK WHILE IN THIS ROUTINE
        LDX     FL.FRM          ;UPDATE ADDRESS
        LDA A   0,X             ;GET BYTE
        LDA B   1,X             ;NEXT BYTE
        INX
        INX
        STX     FL.FRM
        LDX     FL.TO           ;UPDATE ADDRESS
        STA A   0,X             ;MOVE BYTE
        STA B   1,X             ;AND NEXT BYTE
        INX
        INX
        STX     FL.TO
        CPX     #,CHRAM+10000   ;END OF RAM AREA ?
        BNE     1$              ;IF NOT - LOOP
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  DISPLAY INTERRUPT HANDLER
        ;
        ;       LOAD UP PARAMETERS FOR NEW LINE
        ;
DISINT: LDX     DSP0            ;LOAD SCAN ADDRESS
        STX     DARHA
        LDX     DSP1            ;LOAD CHARACTER COUNT AND DISPLAY CONTROL
        STX     CHARPL
        LDX     DSP2            ;LOAD NEW LINE # AND NEXT INTERRUPT LINE #
        STX     CSL
        ;
        ;       NOW SET UP FOR NEXT INTERRUPT
        ;
        BSR     DSPUPD          ;GO LOAD UP DSP0-DSP2
        LDA A   ISL             ;CLEAR INTERRUPT FLAG
        LDA A   CHARPL          ;NMI STROBE
        RTI                     ;RETURN FROM LINE INTERRUPT
        ;
DSPUPD: STS     PGMSTK          ;SAVE STACK POINTER
        LDX     CRSTK           ;POINTER TO CONTROL BLOCKS
        LDX     0,X             ;GET ADDRESS OF NEXT CONTROL BLOCK
        LDS     0,X             ;GET CONTROL BLOCK INTO DSP0-DSP2
        STS     DSP0
        LDS     2,X
        STS     DSP1
        LDS     4,X
        STS     DSP2
        LDX     CRSTK           ;UPDATE CRSTK POINTER
        INX
        INX
        STX     CRSTK           ;SAVE NEW POINTER
        LDS     PGMSTK          ;RESTORE STACK POINTER
        RTS                     ;RETURN TO CALLER
        ;
```

```
        .PAGE
        .SBTTL  CONTROL BLOCK SCROLL ROUTINES
        ;
ROTUP:  BSR     CURUP           ;UPDATE CONTROL BLOCK POINTER
        BSR     LDDATA          ;LOAD CONTROL BLOCK
        RTS                     ;RETURN TO CALLER
        ;
CURUP:  LDA A   CURDSP          ;GET BLOCK POSITION
        CMP A   #,CDTOP         ;AT TOP ?
        BCS     1$              ;IF NOT - SKIP
        LDA A   #,CDBOT-6       ;SET BOTTOM
1$:     ADD A   #,6             ;UPDATE BLOCK POSITION
        STA A   CURDSP          ;SAVE NEW POSITION
        RTS                     ;RETURN TO CALLER
        ;
ROTDWN: BSR     LDDATA          ;LOAD UP CONTROL BLOCK
                                ;FALL THROUGH TO CURDWN
CURDWN: LDA A   CURDSP          ;GET POINTER
        BNE     1$              ;IF NOT AT BOTTOM (CDBOT=0) - SKIP
        LDA A   #,CDTOP+6       ;LOAD TOP
1$:     SUB A   #,6             ;UPDATE POINTER
        STA A   CURDSP          ;SAVE POINTER
        RTS                     ;RETURN TO CALLER
        ;
LDDATA: LDX     PLPNTR          ;GET ADDRESS OF LINE CHARACTER COUNT
        LDA B   0,X             ;GET COUNT
        LDX     TXTCBS          ;GET CONTROL BLOCK ADDRESS
        STA B   2,X             ;SET COUNT
        LDA B   #,1             ;SET DISPLAY CONTROL FOR TEXT LINE
        STA B   3,X
        LDA B   #,17            ;NEXT INTERRUPT SCAN LINE
        STA B   5,X
        LDA B   #,7             ;SET CURRENT LINE
        STA B   4,X
        LDA B   PLBASE+1        ;GET TEXT ADDRESS
        STA B   1,X
        LDA B   PLBASE
        STA B   0,X
        RTS                     ;RETURN TO CALLER
        ;
        .PAGE
        .SBTTL  TEXT POINTER UPDATE
        ;
UPDTXT: TST     LOCK            ;DSPLAY LOCKED ?
        BNE     1$              ;IF NOT - SKIP
        LDX     PLPNTR          ;GET CURRENT COUNT
        LDA A   0,X
        LDX     TXTCBS          ;SAVE IN CONTROL BLOCK
        STA A   2,X
        LDX     CLPNTR          ;CHECK IF SAME LINE OF TEXT
        CPX     PLPNTR
        BEQ     1$              ;IF SO - FINISHED
        STX     PLPNTR          ;ELSE NEW LINE
        LDX     CLBASE          ;SET UP NEW BASE POINTER
        STX     PLBASE
        JSR     ROTUP           ;UPDATE CONTROL BLOCK POINTER
1$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  VERTICAL RETRACE INTERRUPT HANDLER
        ;
VERTIN: TST     UPDFLG          ;NEED AN UPDATE ?
        BEQ     RESTRT          ;IF NOT - GO RESTART DISPLAY
        CLR     UPDFLG          ;DOING UPDATE
        STS     PGMSTK          ;SAVE STACK POINTER
        LDX     #,DSPSTK+58.    ;SETUP DISPLAY STACK
        LDS     #,CLSTP         ;CONTROL STOP
        STS     4,X
        LDS     #,CLSRT         ;CONTROL START
        STS     2,X
        LDS     #,TXTSTP        ;TEXT STOP
        STS     0,X
        LDA B   #,9.            ;9 LINES WITH GRAPHICS
        LDA A   GRAPH           ;GRAPHICS ON ?
        BNE     1$              ;IF SO - SKIP
        ADD B   #,24.-9.        ;ELSE 24 LINES OF TEXT
```

```
1$:     STA B   LINES           ;LINE COUNT
        ;
        ;       SET UP TEXT PORTION OF DISPLAY STACK
        ;
        LDA A   TXTCBS          ;GET POINTER
        LDA B   CURDSP
2$:     DEX                     ;UPDATE STACK POINTER
        DEX
        STA A   0,X             ;LOAD POINTER INTO CONTROL BLOCK
        STA B   1,X
        SUB B   #,6             ;UPDATE POINTER
        BCC     3$              ;SKIP IF NOT PASSED BOTTOM
        LDA B   #,CDTOP         ;ELSE LOAD TOP OF STACK AREA
3$:     DEC     LINES           ;FINISHED ?
        BNE     2$              ;LOOP UNTIL FINISHED
        ;
        ;       NOW CHECK FOR GRAPHICS CONTROL
        ;
        LDA B   #,37            ;START LINE FOR TEXT
        LDA A   GRAPH           ;GRAPHICS ON ?
        BEQ     4$              ;IF NOT - SKIP
        DEX                     ;ELSE PLACE GRAPHIC CONTROLS ON STACK
        DEX
        LDS     #,GPSTP
        STS     0,X
        DEX
        DEX
        LDS     #,GPSRT
        STS     0,X
        LDA B   #,17            ;START LINE FOR GRAPHICS
4$:     STA B   LINES           ;FOR DISPLAY START UP
        STX     BCRSTK          ;SAVE BEGINNING OF DISPLAY STACK
        LDS     PGMSTK          ;RESTORE STACK
        ;
        ;       NOW RESTART DISPLAY FOR CURRENT SCAN
        ;
RESTRT: LDA A   GRAPH           ;GRAPHICS ON ?
        BEQ     2$              ;IF NOT - SKIP
        LDX     #,EVNSCN        ;ASSUME EVEN SCAN
        LDA A   DSPCTL          ;ON ODD SCAN ?
        BPL     1$              ;IF NOT - SKIP
        LDX     #,ODDSCN        ;ELSE ODD SCAN
1$:     STX     GPSRT           ;SET SCAN ADDRESS
2$:     LDX     BCRSTK          ;RESET CRSTK POINTER
        STX     CRSTK
        JSR     DSPUPD          ;SET UP DSP0-DSP2
        LDA A   LINES           ;GET FIRST INTERRUPT LINE
        STA A   ISL             ;STORE IN CONTROL
        LDA A   CARLB           ;CLEAR INTERRUPT FLAG
        LDA A   CHARPL          ;NMI STROBE
        JSR     TIMCLK          ;UPDATE TIME
        LDX     CURLOC          ;UPDATE CURSOR POSTION
        STX     CARHA
        RTI                     ;RETURN FROM INTERRUPT
        ;
        .PAGE
        .SBTTL  BREAK KEY INTERRUPT HANDLER
        ;
BRKINT: BSR     BREAK           ;DO ROUTINE
        LDA A   NPRDFA          ;CLEAR INTERRUPT FLAG
        RTI                     ;RETURN FROM INTERRUPT
        ;
BREAK:  LDX     #,CTLINE+10     ;AREA FOR NOTE
        STX     FL.TO
        LDA A   #,7             ;7 CHARACTERS
        STA A   FL.CNT
        COM     ONLINE          ;CHECK STATUS
        BEQ     1$              ;ON LOCAL - SKIP
        LDA A   #,377           ;DATA INHIBITED FLAG
        STA A   DATINH
        LDX     #,MSGLIN        ;ON LINE MESSAGE
        BRA     2$
1$:     CLR     DATINH          ;DISABLE CLOCK ENABLE
        LDA A   #,5             ;INHIBIT READY FLAG TO ARB-11
        STA A   PCRA7
        LDX     #,MSGLOC        ;LOCAL MESSAGE
```

```
2$:     STX     FL.FRM          ;FROM ADDRESS
                                ;FALL THROUGH TO FILL
        ;
FILL:   LDX     FL.FRM          ;GET A CHARACTER
        LDA A   0,X
        BEQ     1$              ;ON NULL - EXIT
        INX                     ;UPDATE ADDRESS
        STX     FL.FRM
        LDX     FL.TO           ;STORE A CHARACTER
        STA A   0,X
        INX
        STX     FL.TO
        DEC     FL.CNT          ;FINISHED ?
        BNE     FILL            ;LOOP UNTIL ALL TRANSFERRED
1$:     RTS                     ;FINISHED
        ;
MSGLIN: .ASCIZ  /ON LINE/
MSGLOC: .ASCIZ  / LOCAL /
        ;
        .PAGE
        .SBTTL  ARB-11 I/O INTERRUPT HANDLER
        ;
TERMDI: LDA A   #,5             ;INHIBIT READY FLAG
        STA A   PCRA7
        LDA A   TDF11           ;GET CHARATCER
        JSR     CTLP            ;PROCESS CHARACTER
        INC     T.CHAR          ;INCREMENT COUNTER
        CLR A
        TST     ONLINE          ;ON LINE ?
        BEQ     2$              ;IF NOT - SKIP
        LDA A   NALL            ;THIS NUMBER ALLOWED
        CMP A   T.CHAR          ;AT LIMIT ?
        BLE     1$              ;IF SO - SKIP
        LDA A   #,55            ;ELSE ALLOW MORE CHARACTERS
        STA A   PCRA7
        LDA A   TDF11           ;SET READY FLAG
        RTI                     ;FINISHED
1$:     LDA A   #,377
2$:     STA A   DATINH          ;SET FLAG FOR CLOCK
        RTI                     ;FINISHED
        ;
        .PAGE
        .SBTTL  CHARACTER INPUT INTERRUPT HANDLERS
        ;
REMOTE: LDA A   #,53            ;GET TO DIRECTON REGISTERS
        STA A   PCRB9
        CLR B                   ;INPUT DIRECTION
        STA B   HB
        LDA A   #,57            ;GET BACK TO DATA REGISTER
        STA A   PCRB9
        STA B   HA              ;ENABLE READ DATA
        LDA A   HB              ;READ DATA AND CLEAR INTERRUPT FLAG
        LDA B   #,40            ;DISABLE READ
        STA B   HA
        BRA     COMINT          ;TREAT AS A KEYBOARD ENTRY
PHRDR:  LDA A   PHOTO           ;GET CHARACTER
        BRA     COMINT
RPINT:  LDA A   NPRDFB          ;CLEAR REPEAT INTERRUPT FLAG
KBINT:  LDA A   KASCII          ;GET CHARACTER FROM KEYBOARD
COMINT: LDA B   ONLINE          ;ON LINE ?
        BNE     1$              ;IF SO - JUST SEND TO ARB-11
        JSR     CTLP            ;ELSE PROCESS CHARACTER
        RTI                     ;FINISHED
1$:     ORA A   OBIS            ;OR TO SET BIT
        AND A   OBIC            ;AND TO CLEAR BIT
        STA A   TDT11           ;SEND CHARACTER
        RTI                     ;FINISHED
        ;
        .PAGE
        .SBTTL  SCROLL UP INTERRUPT HANDLER
        ;
SCRLUP: LDA A   NPRADB          ;CLEAR INTERRUPT FLAG
        LDA A   KBUTN           ;READ SWITCH VALUES
        BIT A   #,40            ;IS <TAPE DEPRESSED
        BNE     1$              ;IF SO - BOTH DEPRESSED
        LDX     PLPNTR          ;PLPNTR = CLPNTR
```

```
        CPX     CLPNTR
        BEQ     2$              ;IF SO - JUST LOCK DISPLAY
        LDA A   #,377           ;ELSE UNLOCK DISPLAY
        STA A   LOCK
        JSR     UPSCRL          ;SET UP SCROLLING
        JSR     RLPUP           ;ROLL POINTERS
        JSR     ROTUP           ;ROTATE DISPLAY
        BRA     3$
1$:     LDA A   NPRADA          ;CLEAR INTERRUPT FLAG ON TAPE>
2$:     JSR     LCKDSP          ;LOCK DISPLAY
3$:     INC     UPDFLG          ;NEED AN UPDATE
        RTI                     ;RETURN FROM INTERRUPT
        ;
        .PAGE
        .SBTTL  UP SCROLL ROUTINES
        ;
UPSCRL: LDA A   DRCTN           ;CHECK DIRECTION
        BPL     2$              ;IF + WE'RE DONE
        CLR     DRCTN           ;MAKE +
        LDA A   #,CDTOP/6       ;UPDATE ALL TXTCBS LINES
1$:     PSH A                   ;SAVE COUNT
        JSR     RLPUP           ;ROLL POINTERS ONE LINE
        PUL A
        DEC A                   ;ARE WE DONE
        BGT     1$              ;LOOP UNTIL DONE
2$:     RTS                     ;FINISHED
        ;
RLPUP:  LDX     PLPNTR          ;GET CHARACTER COUNT IN LINE
        LDA A   0,X
        CPX     #,CLPTOP        ;ARE WE AT THE TOP ?
        BNE     1$              ;IF NOT - SKIP
        LDX     #,CLPBOT-1      ;ELSE GET BOTTOM ADDRESS
1$:     INX                     ;UPDATE POINTER
        STX     PLPNTR          ;SAVE NEW POINTER
        CLR B                   ;CURRENT COUNT = <B,A>
        ADD A   PLBASE+1        ;COMPUTE NEW BASE
        ADC B   PLBASE
        STA A   PLBASE+1
        STA B   PLBASE
        SUB A   #,CL.TOP&377    ;IS PLBASE PAST CLTOP ?
        SBC B   #,CL.TOP&177400/400
        BCS     2$              ;IF NOT - SKIP
        LDX     #,CLBOT         ;ELSE RESET PLBASE
        STX     PLBASE
2$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  SCROLL DOWN INTERRUPT HANDLER
        ;
SCRLDN: LDA A   NPRADA          ;CLEAR INTERRUPT FLAG
        LDA A   KBUTN           ;READ SWITCHES
        BIT A   #,20            ;TAPE> DEPRESSED ?
        BNE     1$              ;IF SO - LOCK DISPLAY
        LDA A   #,377           ;UNLOCK DISPLAY
        STA A   LOCK
        BSR     DNSCRL          ;SETUP DOWN SCROLLONG
        BSR     RLPDN           ;ROLL POOINTERS DOWN ONE LINE
        JSR     ROTDWN          ;ROTATE DISPLAY ONE LINE
        BRA     2$
1$:     LDA A   NPRADB          ;CLEAR INTERRUPT FLAG ON <TAPE
        JSR     LCKDSP          ;LOCK DISPLAY
2$:     INC     UPDFLG          ;NEED AN UPDATE
        RTI                     ;RETURN FROM INTERRUPT
        ;
        .PAGE
        .SBTTL  DOWN SCROLL ROUTINES
        ;
DNSCRL: LDA A   DRCTN           ;CHECK DIRECTION
        BMI     2$              ;IF NEGATIVE - THEN WE'RE DONE
        LDA A   #,377           ;SET DIRECTION DOWN
        STA A   DRCTN
        LDA A   #,CDTOP/6       ;UPDATE ALL TXTCBS LINES
1$:     PSH A                   ;SAVE COUNTER
        BSR     RLPDN           ;ROLL POINTER ONE LINE
        PUL A
        DEC A                   ;FINISHED ?
```

```
        BGT     1$              ;LOOP UNTIL FINISHED
2$:     RTS                     ;FINISHED
        ;
RLPDN:  LDX     PLPNTR          ;IS POINTER AT BOTTOM ?
        CPX     #,CLPBOT
        BNE     1$              ;IF NOT - SKIP
        LDX     #,CLPTOP+1      ;ELSE GET TOP
1$:     DEX                     ;UPDATE POINTER
        STX     PLPNTR          ;SAVE NEW POINTER
        LDA A   #,CLBOT&377     ;IS PLBASE > CLBOT
        LDA B   #,CLBOT&177400/400
        SUB A   PLBASE+1
        SBC B   PLBASE
        BCS     2$              ;IF SO - SKIP
        LDX     OVBASE          ;ELSE RESET PLBASE
        STX     PLBASE
        RTS                     ;FINISHED
2$:     LDA A   PLBASE+1        ;COMPUTE PREVIOUS BASE ADDRESS
        LDA B   PLBASE
        LDX     PLPNTR
        SUB A   0,X
        SBC B   #,0
        STA A   PLBASE+1
        STA B   PLBASE
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  LOCK DISPLAY ROUTINE
        ;
LCKDSP: LDA A   LOCK            ;IS DISPLAY LOCKED ?
        BEQ     2$              ;IF SO - FINISHED
        CLR     LOCK            ;SET FOR LOCKED DISPLAY
        LDX     CLPNTR          ;SET TO CURRENT POINTERS
        STX     PLPNTR
        LDX     CLBASE
        STX     PLBASE
        CLR     DRCTN           ;SET FOR UP SCROLLING
        JSR     DNSCRL          ;DOWN SCROLL TO 25TH PREVIOUS LINE
        CLR     DRCTN           ;SET FOR UP SCROLLING
        LDA A   #,CDTOP/6       ;UPDATE ALL TXTCBS LINES
1$:     PSH A                   ;SAVE COUNTER
        JSR     ROTUP           ;ROTATE DISPLAY 1 LINE
        JSR     RLPUP           ;ROLL POINTERS 1 LINE
        PUL A
        DEC A                   ;CHECK COUNT
        BGT     1$              ;LOOP UNTIL FINISHED
        JSR     ROTUP           ;UPDATE TO LAST LINE
2$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  PAPER ADVANCE KEY INTERRUPT HANDLER
        ;
PPRKEY: BSR     PAPER           ;GO CHECK KEY
        RTI                     ;RETURN FROM INTERRUPT
        ;
PAPER:  LDA A   PCRB12          ;IS INTERRUPT FLAG SET ?
        BPL     3$              ;IF NOT - JUST LEAVE
        LDA A   NPRDTB          ;CLEAR INTERRUPT FLAG
        COM     RMTON           ;FLIP/FLOP REMOTE FLAG
        BEQ     1$              ;IF OFF - BRANCH
        LDX     #,MSGRON        ;POINT TO MESSAGE
        LDA A   #,57            ;ENABLE REMOTE INTERRUPTS
        BRA     2$
1$:     LDX     #,MSGROF        ;POINT TO MESSAGE
        LDA A   #,4             ;DISABLE INTERRUPTS
2$:     STA A   PCRB9
        LDA A   HB              ;CLEAR PENDNG INTERRUPT
        CLR     HA              ;STROBE UART DAV F/F
        LDA A   #,40            ;SO NEXT CHARACTER WLL BE SEEN
        STA A   HA
        BSR     RMTMOV          ;MOVE MESSAGE
3$:     RTS                     ;FINISHED
        ;
RMTMOV: STX     FL.FRM          ;SAVE FROM POINTER
        LDX     #,CTLINE+42     ;REMOTE AREA
        STX     FL.TO           ;SAVE TO AREA
```

```
        LDA A   #,20            ;CHARACTER COUNT
        STA A   FL.CNT          ;SAVE COUNT
        JSR     FILL            ;MOVE MESSAGE
        RTS                     ;FINISHED
        ;
MSGRON: .ASCIZ  / REMOTE  ON     /
MSGROF: .ASCIZ  / REMOTE  OFF    /
        ;
```

```
        .PAGE
        .SBTTL  CONTROL PROCESSING HANDLER
        ;
        ;       ALL INPUT FROM THE ARB-11, KEYBOARD,
        ;       REPEAT KEY, AND PHOTO READER PASS
        ;       THROUGH THIS HANDLER FOR DISPATCHING TO
        ;       THE REQUIRED HANDLERS
        ;
CTLP:   STA A   CHARA           ;SAVE THE CHARACTER
        CLR     CHRFLG          ;ENABLE PRINTING
        ;
        ;       LOCAL MONITOR CHECK
        ;
        LDA A   ONLINE          ;ON LINE ?
        BNE     1$              ;IF SO - SKIP
        LDX     LJSR            ;ENTRY POINT TO LOCAL MONITOR
        JSR     0,X             ;GO TO MONITOR
        ;
        ;       ESCAPE SEQUENCE CONTROL MONITOR
        ;
1$:     LDA A   ESCFLG          ;ESCAPES ENABLED ?
        BEQ     2$              ;IF NOT - SKIP
        LDX     EJSR            ;ENTRY POINT TO ESCAPE MONITOR
        JSR     0,X             ;GO TO IT
        ;
        ;       LIST GRAPHICS PROCESSING
        ;
2$:     LDA A   LSTFLG          ;GRAPHICS ENABLED ?
        BEQ     3$              ;IF NOT - SKIP
        LDA A   CHARA           ;GET CHARACTER
        JSR     GRPLOT          ;DO GRAPHICS
        ;
        ;       PRINTING CHARACTER PROCESSING
        ;
3$:     LDA A   CHRFLG          ;PRINTING ALLOWED ?
        BNE     4$              ;IF NOT SKIP
        LDA A   CHARA           ;GET CHARACTER
        ORA A   IBIS            ;SET BITS
        AND A   IBIC            ;CLEAR BITS
        JSR     .PRINT          ;PRINT IT
4$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  .PRINT ROUTINE
        ;
        ;       THIS ROUTINE CALLS THE VARIOUS ROUTINES
        ;       THAT ACTUALLY PRINT THE CHARACTER.
        ;       THE ROUTINES USED ARE:
        ;               RMTCHK - FOR THE REMOTE PORT
        ;               CHAR   - FOR THE CHARACTER ORIENTED SCREEN
        ;               LINE   - FOR THE LINE ORIENTED SCREEN
        ;
        ;       THESE ROUTINES MUST NOT BE CALLED INDIVIDUALLY, BUT
        ;       ONLY THROUGH A CALL TO .PRINT
        ;
.PRINT: STA A   CHARB           ;SAVE CHARACTER TO BE PROCESSED
        ;
        ;       ALL CHARACTERS PROCESSED BY REMOTE
        ;
        JSR     RMTCHK
        ;
        ;       CHAR IS INVOKED IF IN CHARACTER MODE
        ;
        LDA A   MODE            ;CHECK MODE
        BNE     1$              ;IF NOT CHARACTER - SKIP
        JSR     CHAR            ;USE CHARACTER HANDLER
        RTS                     ;FINISHED
        ;
        ;       ELSE USE LINE HANDLER
        ;
1$:     JSR     LINE
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  REMOTE OUTPUT HANDLER
        ;
```

```
RMTWT:  LDA B   #,40            ;NO FLASH CHARACTER
        LDA A   T.60TH          ;GET 60'THS COUNTER
        AND A   #,20            ;CHECK BIT FOR FLASH RATE
        BNE     1$
        LDA B   #,377           ;FLASH CHARACTER
1$:     CMP B   RMTFLS          ;SAME AS CURRENT FLASH CHARACTER ?
        BEQ     RMTCHK          ;IF SO - SKIP
        STA B   RMTFLS
        STA B   CTLINE+56       ;PUT IN POSITION
RMTCHK: LDA B   RMTON           ;REMOTE ENABLED ?
        BEQ     2$              ;IF NOT - SKIP OUT
        JSR     PAPER           ;CHECK FOR PAPER KEY
        JSR     TMCK            ;UPDATE TIME
        LDA A   HA              ;IS REMOTE READY ?
        BMI     RMTWT           ;IF NOT - ENTER WAIT LOOP
        AND A   #,20            ;IS UART READY ?
        BEQ     RMTWT           ;IF NOT - ENTER WAIT LOOP
        LDA A   #,53            ;GET TO DIRECTION REGISTERS
        STA A   PCRB9
        LDA A   #,377           ;SET FOR OUTPUT
        STA A   HB
        LDA A   #,57            ;SET FOR DATA REGISTERS
        STA A   PCRB9
        LDA A   CHARB           ;GET CHARACTER TO SEND
        STA A   HB
2$:     LDA A   #,40            ;ELIMINATE FLASH WHEN EXITING
        CMP A   RMTFLS
        BEQ     3$              ;IF A SPACE - SKIP
        STA A   CTLINE+56       ;ELSE MAKE IT A SPACE
        STA A   RMTFLS
3$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  CHARACTER MODE HANDLER
        ;
CHAR:   LDA A   CHARB           ;GET CHARACTER
        CMP A   #,15            ;A CARRIAGE RETURN
        BNE     1$              ;IF NOT - SKIP
        LDA A   #,1             ;POSTION OF 1
        STA A   CURCNT          ;AS CURRENT COUNT
        LDX     CLBASE          ;BEGINNING OF CURRENT LINE
        STX     CURLOC          ;MOVE CURSOR
        RTS                     ;FINISHED
        ;
1$:     CMP A   #,12            ;A LINE FEED ?
        BNE     2$              ;IF NOT - SKIP
        JSR     UPDLIN          ;UPDATE LINE
        LDA A   CURCNT          ;GET CURRENT COUNT
        LDX     CLPNTR          ;GET LINE LENGTH ADDRESS
        STA A   0,X             ;SET LINE LENGTH
        BRA     7$
2$:     CMP A   #,7             ;BELL ?
        BNE     3$              ;IF NOT - SKIP
        STA A   KBUTN           ;RING BELL !
        BRA     5$
3$:     CMP A   #,10            ;BACK SPACE ?
        BNE     5$              ;IF NOT - SKIP
        DEC     CURCNT          ;DELETE A CHARACTER
        BNE     4$              ;IF NOT 0 - SKIP
        INC     CURCNT          ;NEED AT LEAST 1 CHARACTER IN LINE
        RTS                     ;NO FURTHER UPDATE
4$:     LDX     CURLOC          ;MOV CURSOR
        DEX
        STX     CURLOC
        RTS                     ;FINISHED
5$:     CMP A   DCTL            ;PRINTING CONTROLS ?
        BCS     9$              ;IF NOT - FINISHED
        LDX     CURLOC          ;GET CURSOR POSITION
        STA A   0,X             ;PLACE CHARACTER
        INX                     ;UPDATE CURSOR POSITION
        STX     CURLOC
        INC     CURCNT          ;UPDATE COUNTER
        LDA A   CURCNT          ;CURCNT > @CLPNTR ?
        LDX     CLPNTR
        CMP A   0,X
        BLT     6$              ;IF NOT - SKIP
```

```
        STA A   0,X             ;UPDATE @CLPNTR
        LDX     CURLOC          ;AND CLEAR CURSOR POSITION
        LDA A   #,40
        STA A   0,X
        LDX     CLPNTR          ;GET POINTER AGAIN
6$:     LDA A   #,81.           ;ARE WE PAST 80. CHARACTERS ?
        CMP A   0,X
        BGT     8$              ;IF NOT - FINISHED
        STA A   KBUTN           ;ELSE RING BELL !
        STA A   0,X             ;AND SET COUNT TO 81.
        JSR     UPDLIN          ;UPDATE LINE
        LDX     CLPNTR          ;GET POINTER
        LDA A   #,11.           ;NEW LINE OF 11. CHARACTERS
        STA A   0,X
        STA A   CURCNT
7$:     BSR     CLEAR           ;CLEAR LINE
        BSR     POSCUR          ;REPOSITION CURSOR
        JSR     UPDTXT          ;UPDATE TEXT
        INC     UPDFLG          ;DO AN UPDATE
        RTS                     ;FINISHED
8$:     JSR     UPDTXT          ;UPDATE TEXT
9$:     RTS                     ;WITH NO SCREEN UPDATE
        ;
        .PAGE
        ;
POSCUR: LDX     CLPNTR          ;GET ADDRESS OF CURRENT LINE LENGTH
        LDA A   0,X             ;GET LENGTH
        DEC A                   ;DELETE CURSOR POSITION
        CLR B
        ADD A   CLBASE+1        ;COMPUTE NEW POSITION
        ADC B   CLBASE
        STA A   CURLOC+1        ;SAVE NEW POSITION
        STA B   CURLOC
        RTS                     ;FINISHED
        ;
CLEAR:  LDX     CLPNTR          ;GET CHARACTER COUNT AT CURRENT LINE
        LDA A   0,X
        LDX     CLBASE          ;GET FIRST LOCATION
        LDA B   #,40            ;SPACE
1$:     STA B   0,X             ;CLEAR CHARACTER
        INX                     ;UPDATE ADDRESS
        DEC A                   ;FINISHED ?
        BNE     1$              ;LOOP UNTIL ALL CLEARED
        RTS                     ;FINISHED
        ;
        .PAGE
        ;
UPDLIN: LDX     CLPNTR          ;GET CHARACTER COUNT
        LDA A   0,X
        CMP A   #,1             ;1 ?
        BEQ     1$              ;IF SO - SKIP
        DEC     0,X             ;DELETE CURSOR SPACE
        DEC A
1$:     CPX     #,CLPTOP        ;AT TOP ?
        BNE     2$              ;IF NOT - SKIP
        LDX     #,CLPBOT-1      ;ELSE RESET POINTER
2$:     INX                     ;UPDATE POINTER
        STX     CLPNTR          ;SAVE POINTER
        ;
        ;       CHECK STORAGE LIMIT
        ;
        LDX     CLBASE          ;GET PRESENT BASE
        CLR B                   ;COMPUTE NEW BASE
        ADD A   CLBASE+1
        ADC B   CLBASE
        STA A   CLBASE+1
        STA B   CLBASE
        SUB A   #,CL.TOP&377    ;ARE WE PAST TOP OF CHARACTER AREA ?
        SBC B   #,CL.TOP&177400/400
        BCS     4$              ;IF NOT - SKIP
        STX     OVBASE          ;ELSE NOTE LAST BASE BEFORE OVER TOP
        LDX     #,CLBOT         ;AND RESET CLBASE
        STX     CLBASE
4$:     RTS                     ;FINISHED
        ;
        .PAGE
```

```
        .SBTTL  LINE MODE HANDLER
        ;
LINE:   LDA A   CHARB           ;GET CHARACTER
        CMP A   #,40            ;A CONTROL ?
        BCC     4$              ;IF NOT - SKIP
        CMP A   #,15            ;A <CR> ?
        BEQ     CRTRN           ;IF SO - GO
        CMP A   #,12            ;A <LF> ?
        BEQ     ADD80           ;IF SO - GO
        CMP A   #,13            ;A VERTICAL TAB ?
        BEQ     SUB80           ;IF SO - GO
        CMP A   #,11            ;A HORIZONTAL TAB /
        BEQ     ADD1            ;IF SO - GO
        CMP A   #,10            ;A BACKSPACE ?
        BEQ     SUB1            ;IF SO - GO
        ;
        ;       NOW CHECK SPECIALS
        ;
        CMP A   #,36            ;<RS> HOME CURSOR ?
        BNE     1$              ;IF NOT - SKIP
        JMP     HOME            ;ELSE HOME CURSOR
1$:     CMP A   #,32            ;<SUB> CLEAR SCREEN
        BNE     2$              ;IF NOT - SKIP
        JMP     CLRSCN          ;ELSE CLEAR THE SCREEN
2$:     CMP A   #,7             ;BELL /
        BNE     3$              ;IF NOT - SKIP
        STA A   KBUTN           ;SOUND BELL
3$:     LDA B   DCTL            ;PRINT CONTROLS ?
        BNE     5$              ;IF NOT - FINISHED
4$:     LDX     CURLOC          ;ELSE PLACE CHARACTER
        STA A   0,X
        BRA     ADD1            ;GO UPDATE POSITION
5$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  MOTION HANDLERS
        ;
CRTRN:  LDA A   GPCNT            ;GET CURRENT COLUMN POSITION
        CLR B                   ;SET COUNT TO 0
        STA B   GPCNT
        NEG A                   ;MAKE NEGATIVE
        SBC B   #,0             ;EXTEND SIGN OF RESULT
        BRA     UPDGL           ;UPDATE GLCNT
ADD80:  LDA A   #,80.           ;NEXT LINE
        CLR B
        BRA     UPDGL           ;UPDATE GLCNT
SUB80:  LDA A   #,-80.          ;BACK ONE LINE
        LDA B   #,377           ;SIGN EXTENDED -80.
        BRA     UPDGL           ;UPDATE GLCNT
ADD1:   LDA A   #,1             ;1 CHARACTER
        CLR B
        BRA     UPDGP           ;UPDATE GLCNT AND GPCNT
SUB1:   LDA A   #,-1            ;BACK ONE CHARACTER
        TAB                     ;SIGN EXTENDED -1
        ;
UPDGP:  PSH A                   ;SAVE COUNT
        ADD A   GPCNT           ;ADD IN UPDATE
        BPL     2$              ;IF POSITIVE - BRANCH
        ADD A   #,80.           ;ELSE UPDATE
        BRA     3$
1$:     SUB A   #,80.           ;UPDATE
2$:     CMP A   #,80.           ;PAST END OF LINE ?
        BPL     1$              ;IF SO - UPDATE AGAIN
3$:     STA A   GPCNT           ;SAVE COUNT
        PUL A                   ;GET ORIGINAL UPDATE VALUE
        ;
UPDGL:  ADD A   GLCNT+1         ;UPDATE POSITION
        ADC B   GLCNT
1$:     STA A   GLCNT+1         ;SAVE UPDATE
        STA B   GLCNT
        BPL     2$              ;IF POSITIVE - SKIP
        ADD A   #,1920.&377     ;ELSE ADD BUFFER LENGTH
        ADC B   #,1920.&177400/400
        BRA     1$              ;AND LOOP
2$:     SUB A   #,1920.&377     ;SUBTRACT BUFFER LENGTH
        SBC B   #,1920.&177400/400
```

```
        BMI     3$              ;IF WITHIN PAGE - SKIP
        BSR     RPAGE           ;ROTATE PAGE
        LDA A   GPCNT           ;CURRENT COLUMN POSITION
        CLR B
        ADD A   #,1840.&377     ;ADD IN ROW POSITION OF BOTTOM LINE
        ADC B   #,1840.&177400/400
        BRA     1$              ;LOOP
        ;
3$:     LDA A   GLBASE+1        ;COMPUTE CURSOR POSITION
        LDA B   GLBASE
        ADD A   GLCNT+1
        ADC B   GLCNT
        STA A   CURLOC+1
        STA B   CURLOC
        SUB A   #,80.           ;IS POSITION PAST END OF BUFFER ?
        SBC B   #,0
        SUB A   OVBASE+1
        SBC B   OVBASE
        BMI     4$              ;IF NOT - FINISHED
        ADD A   #,CLBOT&377     ;ELSE UPDATE POSITION
        ADC B   #,CLBOT&177400/400
        STA A   CURLOC+1
        STA B   CURLOC
4$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  LINE MODE ROTATE PAGE
        ;
RPAGE:  JSR     UPDLIN          ;UPDATE LINE
        LDX     CLPNTR          ;GET LINE POINTER
        LDA A   #,81.           ;80. CHARACTERS + CURSOR
        STA A   0,X             ;X IS CLPNTR
        JSR     CLEAR           ;CLEAR THE LINE
        JSR     UPDTXT          ;UPDATE TEXT
        INC     UPDFLG          ;UPDATE DISPLAY
        LDA A   #,80.           ;80. CHARACTERS
        CLR B
        ADD A   GLBASE+1        ;UPDATE BASE
        ADC B   GLBASE
        STA A   GLBASE+1
        STA B   GLBASE
        SUB A   #,CL.TOP&377    ;OVER TOP ?
        SBC B   #,CL.TOP&177400/400
        BCS     1$              ;IF NOT - SKIP
        LDX     #,CLBOT         ;ELSE RESET BASE POINTER
        STX     GLBASE
1$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  HOME AND CLSCN ROUTINES
        ;
CLRSCN: LDX     GLBASE          ;BASE ADDRESS
        LDA A   #,24.           ;LINES IN PAGE
        LDA B   #,40            ;FILL WITH SPACES
1$:     PSH A                   ;SAVE COUNTER
        LDA A   #,80.           ;80. CHARACTERS PER LINE
        STX     TEMPGA          ;SAVE FIRST ADDRESS
2$:     STA B   0,X             ;CLEAR LINE
        INX
        DEC A
        BGT     2$              ;LOOP UNTIL LINE CLEARED
        STX     TEMPGB          ;SAVE POINTER
        JSR     TMCK            ;CHECK TIME
        LDX     TEMPGA          ;CHECK BEGINNING ADDRESS
        CPX     OVBASE          ;AT OVBASE ?
        BNE     3$              ;IF NOT SKIP
        LDX     #,CLBOT         ;RESET POINTER
        STX     TEMPGB
3$:     LDX     TEMPGB          ;GET POINTER
        PUL A                   ;GET COUNTER
        DEC A
        BGT     1$              ;LOOP FOR ALL LINES
        STA B   0,X             ;ONE MORE CHARACTER
        ;
HOME:   CLR A                   ;CLEAR MOST EVERYTHING
        STA A   GPCNT
```

```
        STA A   GLCNT+1
        STA A   GLCNT
        LDX     GLBASE          ;REPOSIION CURSOR
        STX     CURLOC
        RTS                     ;FINISHED
        ;
```

```
        .PAGE
        .SBTTL  LOCAL MONTOR SCANNER
        ;
LCLMON: LDA A   CHARA           ;GET CHARACTER
        LDA B   LCLL            ;SHIFT CHARACTERS
        STA B   LCLH
        STA A   LCLL
        LDX     LCLH            ;GET BOTH CHARACTERS
        CPX     #,'@,'=         ;@= SEQUENCE ?
        BEQ     1$              ;IF SO - PROCESS
        RTS                     ;ELSE FINISHED
1$:     JSR     GETSTR          ;GET A CHARACTER STRING
        BSR     CHECK           ;CHECK END OF STRING
        LDX     #,TABLE         ;LIST OF MONITOR COMMANDS
        JSR     SRCHTB          ;GO SEARCH MONITOR TABLE
        BCS     2$              ;ON A MATCH - SKIP
        LDA A   #,'?            ;ELSE PRINT ?
        JSR     .PRINT          ;PRINT IT
        BSR     CHECKX          ;AND TERMINATE SEQUENCE
2$:     LDX     TEMPLA          ;JUMP TO ROUTINE IN TABLE
        LDX     4,X
        JMP     0,X
        ;
        .PAGE
        .SBTTL  TERMINATION CHECK ROUTINE
        ;
CHECK:  LDA A   CHARA           ;GET LAST CHARACTER
        CMP A   #,15            ;A <CR> ?
        BNE     CHECKY          ;IF NOT - SKIP
        LDA B   LSTRNG          ;ANY CHARACTERS ?
        BEQ     CHECKX          ;IF NOT - TERMINATE
CHECKY: CMP A   #,12            ;A <LF> ?
        BNE     CHKEND          ;IF NOT - JUST CONTINUE
CHECKX: LDX     #,LCLMON        ;TERMINATE CURRENT SEQUENCE
        STX     LJSR
        JSR     PLMSG0          ;PRINT <CRLF>*
        PUL A                   ;REMOVE ONE RETURN LEVEL
        PUL A
CHKEND: RTS                     ;RETURN TO CALLER
        ;
        .PAGE
        .SBTTL  GET STRING ROUTINE
        ;
GETSTR: PUL A                   ;SAVE RETURN ADDRESS
        STA A   LRVEC
        PUL A
        STA A   LRVEC+1
        LDX     #,GETS          ;SETUP NEXT TRANSFER ADDRESS
        STX     LJSR
        LDX     #,LSTRNG+80.    ;TOP OF STRING AREA
1$:     DEX                     ;UPDATE ADDRESS
        CLR     0,X             ;CLEAR STRING
        CPX     #,LSTRNG        ;AT BOTTOM ?
        BNE     1$              ;LOOP UNTIL FINISHED
        STX     LSTPNT          ;LOCAL STRING POINTER
        RTS                     ;RETURN 2 LEVELS BACK
        ;
GETS:   LDA A   CHARA           ;GET LAST CHARACTER
        CMP A   #,15            ;A <CR> ?
        BEQ     1$              ;IF SO - HAVE STRING
        CMP A   #,12            ;A <LF> ?
        BNE     2$              ;IF NOT - SKIP
1$:     INC     CHRFLG          ;INHIBIT PRINTING
        LDX     LRVEC           ;RETURN IN LINE AFTER GETSTR
        JMP     0,X
        ;
2$:     LDX     LSTPNT          ;GET STRING POINTER
        CMP A   #,177           ;A DELETE ?
        BEQ     4$              ;IF SO - SKIP
        STA A   0,X             ;ELSE SAVE CHARACTER
        CPX     #,LSTRNG+78.    ;AT END OF STRING ?
        BEQ     3$              ;IF SO - SKIP UPDATE
        INX                     ;ELSE UPDATE POINTER
        STX     LSTPNT
3$:     RTS                     ;FINISHED
        ;
```

```
4$:     DEX                     ;ONE LESS CHARACTER
        CPX     #,LSTRNG-1      ;BELOW BOTTOM ?
        BEQ     6$              ;IF SO - LEAVE
        STX     LSTPNT          ;ELSE SAVE POINTER
        CLR     0,X             ;DELETE CHARACTER
        LDA A   #,10            ;BACKSPACE
        JSR     .PRINT
        LDA A   #,40            ;SPACE
        JSR     .PRINT
        LDA A   #,10            ;BACKSPACE
        JSR     .PRINT
6$:     INC     CHRFLG          ;INHIBIT PRINTING
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  LOCAL MONITOR TABLE SEARCH ROUTINE
        ;
SRCHTB: STX     TEMPLA          ;SAVE POINTER
        LDX     0,X             ;GET 2 BYTES OF STRING
        BEQ     2$              ;END OF TABLE - SKIP
        CPX     LSTRNG          ;COMPARE FIRST TWO CHARACTERS
        BNE     1$              ;NO MATCH - SKIP
        LDX     TEMPLA          ;GET NEXT TWO CHARACTERS
        LDX     2,X
        CPX     LSTRNG+2        ;COMPARE SECOND SET
        BEQ     3$              ;ON MATCH - SKIP
1$:     LDX     TEMPLA          ;UPDATE POINTER TO NEXT TABLE ENTRY
        INX
        INX
        INX
        INX
        INX
        INX
        BRA     SRCHTB          ;AND TRY NEXT ENTRY
2$:     CLC                     ;NO MATCH
        RTS                     ;RETURN
3$:     SEC                     ;MATCH FOUND
        RTS                     ;RETURN
        ;
        .PAGE
        .SBTTL  LOCAL MONITOR DISPATCH TABLE
        ;
TABLE:  .ASCII  /SYST/
        FDB     SYST
        .ASCII  /TIME/
        FDB     TIME
        .ASCII  /CTLC/
        FDB     CTLC
        .ASCII  /PEEK/
        FDB     PEEK
        .ASCII  /POKE/
        FDB     POKE
        .ASCII  /BAUD/
        FDB     BAUD
        .ASCII  /MASK/
        FDB     MASK
        .ASCII  /ESCP/
        FDB     ESCP
        .ASCII  /RK0:/
        FDB     BRK0
        .ASCII  /RK1:/
        FDB     BRK1
        .ASCII  /RK2:/
        FDB     BRK2
        .ASCII  /RK3:/
        FDB     BRK3
        .ASCII  /RK4:/
        FDB     BRK4
        .ASCII  /RK5:/
        FDB     BRK5
        .ASCII  /RK6:/
        FDB     BRK6
        .ASCII  /RK7:/
        FDB     BRK7
        .ASCII  /DY0:/
        FDB     BDY0
```

```
        .ASCII  /DY1:/
        FDB     BDY1
        .ASCII  /DX0:/
        FDB     BDX0
        .ASCII  /DX1:/
        FDB     BDX1
        .ASCII  /CT0:/
        FDB     BCT0
        .ASCII  /CT1:/
        FDB     BCT1
        .ASCII  /TT:/<0>
        FDB     BTT0
        .ASCII  /MT:/<0>
        FDB     BMT0
        .ASCII  /KUS/<0>
        FDB     BKUS
        FDB     0       ;TABLE TERMINATION
        ;
        .PAGE
        .SBTTL  LPRINT ROUTINE AND VARIATIONS
        ;
PLMSG0: LDX     #,LMSG0         ;<CRLF>*
        BRA     LPRINT
PLMSG1: LDX     #,LMSG1         ;<CRLF>* >>
        BRA     LPRINT
PLMSG2: LDX     #,LMSG2         ;<CRLF>*A
        BRA     LPRINT
PLMSG3: LDX     #,LMSG3         ;/   /
        ;
LPRINT: STX     TEMPLA          ;SAVE POINTER
        LDA A   0,X             ;GET CHARACTER
        BEQ     1$              ;NULL IS TERMINATION
        JSR     .PRINT          ;PRINT IT
        LDX     TEMPLA          ;NOW UPDATE POINTER
        INX
        BRA     LPRINT          ;CHECK NEXT CHARACTER
1$:     RTS                     ;FINISHED
        ;
        ;       MESSAGES
        ;
        .NLIST  BIN
LMSG0:  .ASCIZ  <15><12>/*/
LMSG1:  .ASCIZ  <15><12>/* >>  /
LMSG2:  .ASCIZ  <15><12>/*A/
LMSG3:  .ASCIZ  /   /
LIBIS:  .ASCIZ  / IBIS  =/
LIBIC:  .ASCIZ  / IBIC  =/
LOBIS:  .ASCIZ  / OBIS  =/
LOBIC:  .ASCIZ  / OBIC  =/
        .LIST BIN
        ;
        .PAGE
        .SBTTL  SYSTEM NOTES
        ;
SYST:   LDX     #,SYSMSG        ;DUMP SYSTEM MESSAGE
        JSR     LPRINT
        JSR     CHECKX          ;DUMBY ROUTINE
        ;
        .NLIST BIN
SYSMSG: .ASCII  <15><12><12>/MAY 1981/<15><12><12>
        .ASCII  /LOCAL MONITOR COMMANDS/<15><12>
        .ASCII  /@= SYST,TIME,CTLC,MASK,BAUD,PEEK,POKE,ESCP/<15><12><12>
        .ASCII  /DEVICE BOOTS/<15><12>
        .ASCII  /@= RK0:,RK1:,RK2:,RK3:,RK4:,RK5:,RK6:,RK7:/<15><12>
        .ASCII  /   DX0:,DX1:,DY0:,DY1:,CT0:,CT1:,TT:,MT:/<15><12><12>
        .ASCII  /UTILLITY PROGRAM/<15><12>
        .ASCII  /@= KUS/<15><12><12>
        .ASCII  /ESCAPE CODE MONITOR/<15><12>
        .ASCII  /ESC= 0-7/<15><12>
        .ASCII  /CHARACTER SET RESTORE (0), BAUD RATE (1)/<15><12>
        .ASCII  /CHARACTER MODE (2), LINE MODE (3)/<15><12>
        .ASCII  /DEFINE CHARACTER (4), MOV CURSOR (5)/<15><12>
        .ASCII  /TERMINAL GRAPHICS ON (6), TERMINAL GRAPHICS OFF (7)/
        .ASCII  <15><12><12><0>
        .LIST   BIN
        ;
```

```
        .PAGE
        .SBTTL  TIME ROUTINE
        ;
TIME:   JSR     PLMSG1          ;PRINT <CRLF>* >>
        JSR     GETSTR          ;GET A STRING
        JSR     CHECK           ;CHECK TERMINATION
        LDX     #,CTLINE+70     ;CLOCK AREA
        STX     FL.TO
        LDX     #,LSTRNG        ;STRING AREA
        STX     FL.FRM
        LDA A   #,11.           ;ONLY ALLOW 11 CHARACTERS
        STA A   FL.CNT
        JSR     FILL            ;PLACE STRING
        JSR     CHECKX          ;TERMINATE SEQUENCE
        ;
        .PAGE
        .SBTTL  CTLC ROUTINE
        ;
CTLC:   JSR     PLMSG1          ;PRINT <CRLF>* >>
        JSR     GETSTR          ;GET STRING
        JSR     CHECK           ;CHECK TERMINATION
        CLR B                   ;VALUE TO ALLOW PRINTIING
        LDA A   LSTRNG          ;GET FIRST CHARACTER
        CMP A   #,'Y            ;Y IF CONTROLS TO PRINT
        BEQ     1$
        LDA B   #,40            ;ELSE INHIBIT PRINTING
1$:     STA B   DCTL            ;SET FLAG
        JSR     CHECKX          ;END SEQUENCE
        ;
        .PAGE
        .SBTTL  PEEK ROUTINE
        ;
PEEK:   JSR     PLMSG1          ;PRINT <CRLF>* >>
        JSR     GETSTR          ;GET A STRING
        JSR     CHECK           ;CHECK TERMINATION
        JSR     PEVAL           ;EVALUATE STRING
1$:     JSR     PRNTAD          ;PRINT ADDRESS
        LDA A   #,10            ;8 BYTES PER LINE
2$:     PSH A                   ;SAVE COUNT
        JSR     PRNTDT          ;PRINT DATA
        PUL A
        DEC A                   ;MORE TO DO ?
        BNE     2$              ;LOOP UNTIL DONE
        JSR     GETSTR          ;GET ANOTHER STRING
        JSR     CHECKY          ;CHECK FOR TERMINATION
        BRA     1$              ;GO PRINT SOME MORE
        ;
        .PAGE
        .SBTTL  POKE ROUTINE
        ;
POKE:   JSR     PLMSG1          ;PRINT <CRLF>* >>
        JSR     GETSTR          ;GET A STRING
        JSR     CHECK           ;CHECK TERMINATION
        JSR     PEVAL           ;EVALUATE STRING
1$:     JSR     PRNTAD          ;PRINT ADDRESS
        JSR     PLMSG3          ;PRINT /    /
        JSR     GETSTR          ;GET NEW DATA
        JSR     CHECKY          ;CHECK TERMINATION
        LDX     #,LSTRNG        ;EVALUATE THE DATA
        JSR     EVAL
        LDA A   VL              ;GET VALUE
        LDX     TEMPLB          ;POKE ADDRESS
        LDA B   LSTRNG          ;A REAL STRING ?
        BEQ     2$              ;IF NOT - SKIP
        STA A   0,X             ;ELSE POKE DATA
2$:     INX                     ;UPDATE POINTER
        STX     TEMPLB
        BRA     1$              ;LOOP ONE MORE TIME
        ;
        .PAGE
        .SBTTL  EVALUATION ROUTINES
        ;
PEVAL:  LDX     #,LSTRNG        ;EVALUATE STRING
        BSR     EVAL
        LDX     VH              ;EVALUATION
        STX     TEMPLB          ;SAVE VALUE
```

```
        RTS                     ;FINISHED
        ;
EVAL:   CLR     VH              ;CLEAR RESULT
        CLR     VL
1$:     LDA A   0,X             ;GET FIRST CHARACTER
        BEQ     3$              ;IF NULL - END
        SUB A   #,'0            ;COMPUTE BINARY VALUE
        LDA B   #,3             ;3 BIT SHIFT
2$:     BSR     SHIFT
        DEC B                   ;FINISHED ?
        BNE     2$              ;LOOP UNTIL FINISHED
        ADD A   VL              ;COMPUTE UPDATED VALUE
        ADC B   VH
        STA A   VL
        STA B   VH
        INX                     ;UPDATE POINTER
        BRA     1$              ;GET NEXT CHARACTER
3$:     RTS                     ;FINISHED
        ;
SHIFT:  ASL     VL      ;SHIFT A 24. BIT WORD
        ROL     VH
        ROL     VAL
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  PRINT OCTAL VALUE ROUTINES
        ;
PRNTAD: JSR     PLMSG2          ;PRINT <CRLF>*A
        LDX     #,TEMPLB        ;ADDRESS OF DATA
        BRA     PWORD           ;GO PRINT IT
        ;
PRNTDT: JSR     PLMSG3          ;PRINT /   /
        LDX     TEMPLB          ;INCREMENT POINTER
        INX
        STX     TEMPLB
        DEX
        ;
PBYTE:  LDA A   #,3             ;COUNT OFF THREE CHARACTERS
        BRA     PWRD.A
PWORD:  LDA A   #,6             ;COUNT OFF 6 CHARACTERS
PWRD.A: LDX     0,X             ;GET BINARY VALUE TO CONVERT
        STX     VH              ;SAVE VALUE
        PSH A                   ;SAVE COUNT
        CLR     VAL             ;CLEAR HIGH BYTE
        CMP A   #,6             ;WORD CONVERSION ?
        BEQ     2$              ;IF SO - SKIP
1$:     BSR     SHIFT           ;SHIFT A BIT
2$:     BSR     SHIFT           ;SHIFT A BIT
        LDA A   #,'0            ;DEVELOPE A CHARACTER
        ADD A   VAL
        JSR     .PRINT          ;PRINT CHARACTER
        PUL A
        DEC A                   ;MORE CHARACTERS TO DO ?
        BEQ     3$              ;IF NOT - FINISHED
        PSH A                   ;SAVE COUNT
        CLR     VAL             ;CLEAR VALUE
        BSR     SHIFT           ;DO A SHIFT
        BRA     1$              ;LOOP FOR MORE
3$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  BAUD RATE ROUTINE
        ;
BAUD:   JSR     PLMSG1          ;PRINT <CRLF>* >>
        JSR     GETSTR          ;GET A STRING
        JSR     CHECK           ;CHECK TERMINATION
        JSR     PEVAL           ;EVALUATE THE STRING
        LDA A   VL              ;LOW ORDER VALUE
        STA A   NALL            ;CHARACTERS ALLOWED PER 60TH
        JSR     CHECKX          ;TERMINATE SEQUENCE
        ;
        .PAGE
        .SBTTL  CHARACTER MASK ROUTINE
        ;
MASK:   LDX     #,IBIS          ;ADDRESS OF VARIABLES
        STX     TEMPLB          ;SAVE POINTER
```

```
        BSR     PIBIS           ;IBIS SEQUENCE
        JSR     GETSTR          ;GET A STRING
        JSR     CHECKY          ;CHECK TERMINATION
        BSR     PUT             ;PLACE DATA
        BSR     PIBIC           ;IBIC SEQUENCE
        JSR     GETSTR          ;GET A STRING
        JSR     CHECKY          ;CHECK TERMINATION
        BSR     PUT             ;PLACE DATA
        BSR     POBIS           ;OBIS SEQUENCE
        JSR     GETSTR          ;GET A STRING
        JSR     CHECKY          ;CHECK TERMINATION
        BSR     PUT             ;PLACE DATA
        BSR     POBIC           ;OBIC SEQUENCE
        JSR     GETSTR          ;GET A STRING
        JSR     CHECKY          ;CHECK TERMINATION
        BSR     PUT             ;PLACE DATA
        JSR     CHECKX          ;TERMINATE SEQUENCE
        ;
        .PAGE
        .SBTTL  MASK LIST/PUT ROUTINES
        ;
PIBIS:  LDX     #,LIBIS         ;MESSAGE POINTER
        BRA     PCOMB
PIBIC:  LDX     #,LIBIC
        BRA     PCOMB
POBIS:  LDX     #,LOBIS
        BRA     PCOMB
POBIC:  LDX     #,LOBIC
        ;
PCOMB:  STX     TEMPLC          ;SAVE MESSAGE POINTER
        JSR     PLMSG0          ;PRINT <CRLF>*
        JSR     PLMSG3          ;PRINT /   /
        LDX     TEMPLC          ;NOW PRINT MESSAGE
        JSR     LPRINT
        JSR     PRNTDT          ;PRINT DATA
        JSR     PLMSG1          ;PRINT <CRLF>* >>
        RTS                     ;FINISHED
        ;
PUT:    LDA A   LSTRNG          ;A REAL STRING ?
        BEQ     2$              ;IF NOT - LEAVE
        LDX     #,LSTRNG        ;EVALUATE THE STRING
        JSR     EVAL
        LDX     TEMPLB          ;UPDATE ADDRESS
        DEX
        LDA A   VL              ;GET EVALUATION
        STA A   0,X             ;SAVE NEW VALUE
2$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  ESCP ROUTINE
        ;
ESCP:   JSR     PLMSG1          ;PRINT <CRLF>* >>
        JSR     GETSTR          ;GET A STRING
        JSR     CHECK           ;CHECK TERMINATION
        CLR B                   ;DISABLE ESCAPE SEQUENCES
        LDA A   LSTRNG          ;GET CHARACTER
        CMP A   #,'Y            ;ESCAPES ENABLED ?
        BNE     1$              ;IF NOT - SKIP
        LDA B   #,377           ;ELSE ENABLE
1$:     STA B   ESCFLG          ;SET FLAG
        JSR     CHECKX          ;TERMINATE SEQUENCE
        ;
```

```
        .PAGE
        .SBTTL  ESCAPE SEQUENCE MONITOR
        ;
ESCAPE: LDA A   CHARA           ;GET NEW CHARACTER
        LDA B   ESCL            ;TWO CHARACTER SHIFTER
        STA B   ESCH
        STA A   ESCL
        CMP A   #,33            ;NEW CHAR AN ESC ?
        BNE     2$              ;IF NOT - SKIP
        CMP B   #,33            ;OLD CHAR AN ESC ?
        BEQ     1$              ;IF SO - SKIP
        INC     CHRFLG          ;INHIBIT PRINTING
1$:     RTS                     ;FINISHED
2$:     CMP B   #,33            ;OLD CHAR AN ESC ?
        BNE     3$              ;IF NOT - SKIP
        CMP A   #,'=            ;AN = SIGN ?
        BEQ     4$              ;IF SO - SKIP
        TBA                     ;ELSE PRINT OLD ESC
        JSR     .PRINT
3$:     RTS                     ;FINISHED
4$:     INC     CHRFLG          ;INHIBIT PRINTING
        LDX     #,GCTL          ;SET UP NEXT TRANSFER
        STX     EJSR
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  ESCAPE DISPATCHER
        ;
GCTL:   LDA A   CHARA           ;GET CHARACTER
        LDX     #,ESCAPE        ;RESET ENTRY POINTER
        STX     EJSR
        SUB A   #,'0            ;CHECK THAT CHARACTER
        BCS     1$              ;IS BETWEEN 0 AND 7
        CMP A   #,8.
        BCS     2$              ;IF SO - SKIP
1$:     LDA A   ESCH            ;ELSE PRINT SEQUENCE
        JSR     .PRINT
        LDA A   ESCL
        JSR     .PRINT
        CLR     ESCH            ;AND CLEAR SCANNER
        CLR     ESCL
        RTS                     ;FINISHED
        ;
2$:     INC     CHRFLG          ;INHIBIT PRINTING
        TAB                     ;COMPUTE 3*A
        ABA
        ABA
        CLR B
        ADD A   #,GTBLE&377     ;COMPUTE TABLE ADDRESS
        ADC B   #,GTBLE&177400/400
        STA A   GTABLE+1        ;SAVE JUMP LOCATION
        STA B   GTABLE
        LDX     GTABLE          ;GET ADDRESS
        LDA A   0,X             ;GET # OF CHARACTERS NEEDED
        BEQ     3$              ;IF NONE - SKIP
        STA A   GCNT            ;SAVE COUNT NEEDED
        LDX     #,GETLST        ;SETUP TRANSFER VECTOR
        STX     EJSR
        LDX     #,ESTRNG        ;ADDRESS FOR INPUT CHARACTERS
        STX     TEMPEA
        RTS                     ;FINISHED
3$:     LDX     1,X             ;GET VECTOR ADRESS
        JMP     0,X             ;GO TO SELECTED ROUTINE
        ;
        .PAGE
        .SBTTL  GET LIST ROUTINE
        ;
GETLST: INC     CHRFLG          ;INHIBIT PRINTING
        LDX     TEMPEA          ;GET ADDRESS
        LDA A   CHARA           ;GET CHARACTER
        STA A   0,X             ;STORE CHARACTER
        INX                     ;UPDATE POINTER
        STX     TEMPEA          ;SAVE POINTER
        DEC     GCNT            ;ENOUGH YET ?
        BGT     1$              ;IF NOT - SKIP
        LDX     #,ESCAPE        ;RESET TRANSFER VECTOR
```

```
        STX     EJSR
        LDX     GTABLE          ;GET TABLE LOCATION
        LDX     1,X             ;GET TRANSFER ADDRESS
        JMP     0,X             ;AND GO
1$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  ESCAPE DISPATH TABLE
        ;
GTBLE:  .BYTE   0       ;0,RAMSET
        FDB     RAMSET
        .BYTE   1       ;1,GBAUD
        FDB     GBAUD
        .BYTE   0       ;0,CHMODE
        FDB     CHMODE
        .BYTE   0       ;0,LNMODE
        FDB     LNMODE
        .BYTE   17.     ;17,DEFCHR
        FDB     DEFCHR
        .BYTE   2       ;2,MOVCUR
        FDB     MOVCUR
        .BYTE   0       ;0,GPHONN
        FDB     GPHONN
        .BYTE   0       ;0,GPHOFF
        FDB     GPHOFF
        ;
        .PAGE
        .SBTTL  ESCAPE ROUTINES
        ;
GBAUD:  LDA A   CHARA           ;GET CHARACTER
        SUB A   #,'@            ;RANGE A-Z
        STA A   NALL            ;NUMBER CHARS PER 60TH
        RTS                     ;FINISHED
        ;
MOVCUR: LDA A   MODE            ;CHECK MODE
        BEQ     1$              ;NOT LNMODE - SKIP
        LDX     #,ESTRNG        ;GET POINTER TO STRING
        LDA A   0,X             ;GET ROW ADDRESS
        SUB A   #,40            ;SPACE TO 7
        BCS     1$              ;BAD CHARACTER
        CMP A   #,24.
        BCC     1$              ;BAD CHARACTER
        LDA B   1,X             ;GET COLUMN ADDRESS
        SUB B   #,40            ;SPACE TO LITTLE O
        BCS     1$              ;BAD CHARACTER
        CMP B   #,80.
        BCC     1$              ;BAD CHARACTER
        STA B   1,X             ;SAVE COLUMN
        CLR B
        BSR     .MUL20          ;GO COMPUTE 20*<B,A>
        STA A   GLCNT+1         ;SAVE RESULT
        STA B   GLCNT
        BSR     .MUL4           ;COMPUTE 4*(20**<B,A>)
        ADD A   GLCNT+1         ;TOTAL = 120*<B,A>
        ADC B   GLCNT
        STA A   GLCNT+1         ;SAVE RESULT
        STA B   GLCNT
        LDA A   1,X             ;GET COLUMN
        STA A   GPCNT           ;NEW COLUMN
        CLR B
        JSR     UPDGL           ;UPDATE POSITIONS
1$:     RTS                     ;FINISHED
        ;
        .PAGE
        ;
.MUL20: ASL A                   ;20*<B,A>
        ROL B
        ASL A
        ROL B
.MUL4:  ASL A                   ;4*<B,A>
        ROL B
        ASL A
        ROL B
        RTS
        ;
DEFCHR: LDX     #,ESTRNG        ;POINTER TO STRING
```

```
        LDA A   0,X             ;FIRST CHARACTER
        INX                     ;UPDATE PONTER
        STX     TEMPEA          ;DEFINITION ADDRESS
        CLR B                   ;COMPUTE ADDRESS OF CHARACTER IN RAM
        BSR     .MUL20          ;16 BYTES PER CHARACTER
        ADD A   #,CHRAM&377     ;ADD IN CHRAM BASE ADDRESS
        ADC B   #,CHRAM&177400/400
        STA A   TEMPEB+1        ;SAVE ADDRESS
        STA B   TEMPEB
        LDA A   #,20            ;16 BYTE PER CHARACTER
1$:     LDX     TEMPEA          ;GET ADDRESS
        LDA B   0,X             ;AND DEFINITION
        INX                     ;UPDATE POINTER
        STX     TEMPEA
        LDX     TEMPEB          ;GET RAM ADDRESS
        STA B   0,X             ;STORE DEFINITION
        INX                     ;UPDATE ADDRESS
        STX     TEMPEB
        DEC A
        BGT     1$              ;LOOP UNTIL FINISHED
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  LNMODE HANDLER
        ;
LNMODE: JSR     UPDLIN          ;UPDATE LINE
        LDX     CLPNTR          ;GET LINE POINTER
        LDA A   #,81.           ;80. CHARACTERS + CURSOR
        STA A   0,X             ;[X]=CLPNTR
        JSR     CLEAR           ;CLEAR LINE
        JSR     UPDTXT          ;UPDATE TEXT
        LDX     CLBASE          ;SAVE OLD BASE
        STX     GLBASE
        LDA A   #,23.           ;SET FOR 23 MORE LINES IN NEW PAGE
1$:     PSH A                   ;SAVE COUNT
        JSR     UPDLIN          ;UPDATE LINE
        LDX     CLPNTR          ;GET LINE POINTER
        LDA A   #,81.           ;80. CHARACTERS + CURSOR
        STA A   0,X             ;[X]=CLPNTR
        JSR     CLEAR           ;CLEAR LINE
        JSR     UPDTXT          ;UPDATE TEXT
        JSR     TMCK            ;CHECK TIME
        PUL A
        DEC A
        BGT     1$              ;LOOP UNTIL FINISHED
        INC     UPDFLG          ;UPDATE DISPLAY
        JSR     HOME            ;HOME CURSOR
        LDA A   #,377           ;INDICATE NEW MODE
        STA A   MODE
        LDX     #,GMSG1         ;LOAD CONTROL LINE
        BRA     LDMODE
        ;
        .PAGE
        .SBTTL  CHMODE ROUTINE
        ;
CHMODE: LDA A   MODE            ;CHMODE NOW ?
        BEQ     1$              ;DON'T RESET DISPLAY
        CLR     MODE            ;ELSE SET UP
        LDA A   #,15            ;DO A <CR>
        JSR     .PRINT
        LDA A   #,12            ;AND A <LF> INTO CHMODE
        JSR     .PRINT
1$:     LDX     #,GMSG0         ;AND UPDATE CONTROL LINE
LDMODE: STX     FL.FRM
        LDX     #,CTLINE+20
        STX     FL.TO
        LDA A   #,12.           ;12 CHARACTERS
        STA A   FL.CNT
        JSR     FILL            ;MOV MESSAGE
        RTS                     ;FINISHED
        ;
GMSG0:  .ASCIZ  / CHAR MODE  /
GMSG1:  .ASCIZ  / LINE MODE  /
        ;
        .PAGE
        .SBTTL  GRAPHICS CONTROLS
```

```
        ;
GPHONN: LDA A   #,377           ;GRAPHICS ON
        STA A   LSTFLG          ;SET FLAG
        RTS                     ;FINISHED
        ;
GPHOFF: CLR     LSTFLG          ;GRAPHICS OFF
        RTS                     ;FINISHED
        ;
```

```
        .PAGE
        .SBTTL  BOOTSTRAP LOADER BLOCKS
        ;
BKUS:   LDX     #,1$            ;POINTER TO BLOCK
        JMP     LOADER          ;GO LOAD KUS
1$:     FDB     BTKUS           ;START BLOCK
        FDB     KUSLD           ;KUS
        FDB     0               ;END OF BLOCK
        ;
BRK0:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK0           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BRK1:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK1           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BRK2:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK2           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BRK3:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK3           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BRK4:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK4           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BRK5:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK5           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BRK6:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK6           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BRK7:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTRK7           ;START BLOCK
        FDB     RKBOOT          ;BOOT STRAP
        FDB     0               ;END OF BLOCK
        ;
BDY0:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTDY0           ;START BLOCK
        FDB     DYBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
BDY1:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTDY1           ;START BLOCK
        FDB     DYBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
BDX0:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTDX0           ;START BLOCK
        FDB     DXBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
BDX1:   LDX     #,1$            ;POINT TO BLOCK
```

```
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTDX1           ;START BLOCK
        FDB     DXBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
BCT0:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTCT0           ;START BLOCK
        FDB     CTBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
BCT1:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTCT1           ;START BLOCK
        FDB     CTBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
BMT0:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTMT            ;START BLOCK
        FDB     MTBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
BTT0:   LDX     #,1$            ;POINT TO BLOCK
        JMP     LOADER          ;GO LOAD BOOT
1$:     FDB     BTTT0           ;START BLOCK
        FDB     TTBOOT          ;BOOTSTRAP
        FDB     0               ;END OF BLOCK
        ;
        .PAGE
        .SBTTL  LOADER PROGRAM
        ;
        ;       THE BOOT FUNCTIONS ARE PART OF THE LOCAL MONITOR
        ;       PROGRAM AND ADHERES TO ITS PROTOCAL
        ;
LOADER: STX     LDRPKX          ;SAVE POINTER TO TRANSFER BLOCK
        BSR     SETNPR          ;SET UP ALL NPR REGISTERS
        LDA A   PCRB11          ;CAUSE A POWER FAIL INTERRUPT
        AND A   #,367
        STA A   PCRB11
        LDX     #,0             ;WAIT LOOP
1$:     INX
        BNE     1$
        LDA A   PCRA11          ;ENABLE FORCED NPR'S
        AND A   #,367
        STA A   PCRA11
        JSR     TRNSFR          ;TRANSFER DATA
        LDA A   PCRA11          ;DISABLE FORCED NPR'S
        ORA A   #,10
        STA A   PCRA11
        LDA A   PCRB11          ;RESTART ARB-11
        ORA A   #,10
        STA A   PCRB11
        JSR     BREAK           ;GO SET TO 'ON LINE'
        JSR     CHECKX          ;TERMINATE SEQUENCE
        ;
        .PAGE
        .SBTTL  NPR REGISTER SETUP
        ;
SETNPR: LDX     #,PHOTO         ;POINTER TO NPR REGISTERS
        CLR     3,X             ;CLEAR NPR CONTROL
        LDA A   #,37            ;SET DIRECTIONS
        STA A   1,X
        LDA A   #,56            ;SET CONTROL
        STA A   3,X
        CLR     1,X             ;PRESET TO NOP
        LDA B   #,377           ;FOR DIRECTION CONTROL
        BSR     ADDR            ;SET UP ADDRESS REGISTERS
        BSR     ADDR
        BSR     DATO            ;SET UP OUTPUT REGISTERS
        BSR     DATO
        BSR     DATI            ;SET UP INPUT REGISTERS
        BSR     DATI
        RTS                     ;FINISHED WITH SET UP
        ;
ADDR:   LDA A   6,X             ;MASK ALL EXCEPT C1 CONTROLS
```

```
        AND A   #,3
        STA A   6,X
        STA B   4,X             ;SET DIRECTION - OUT
        ORA A   #,74            ;C2 CONTROLS
        STA A   6,X
        INX
        RTS                     ;FINISHED
        ;
DATO:   LDA A   10,X            ;MASK ALL EXCEPT C1 CONTROLS
        AND A   #,3
        STA A   10,X
        STA B   6,X             ;SET DIRECTION - OUT
        ORA A   #,4             ;SET CONTROLS
        STA A   10,X
        INX
        RTS                     ;FINISHED
        ;
DATI:   LDA A   12,X            ;MASK ALL EXCEPT C1 CONTROLS
        AND A   #,3
        STA A   12,X
        CLR     10,X            ;SET DIRECTION - IN
        ORA A   #,4             ;SET CONTROLS
        STA A   12,X
        INX
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  NPR TRANSFER ROUTINE
        ;
TRNSFR: LDX     LDRPKX          ;GET ADDRESS OF FIRST BLOCK
        LDX     0,X
        BEQ     2$              ;IF = 0 THEN FINISHED
        STX     TRNPKX          ;SAVE POINTER
        LDX     0,X             ;FIRST ELEMENT IS WORD COUNT
        BEQ     2$              ;IF = 0 THEN FINISHED
        STX     TRNCNT          ;SAVE COUNT
        LDX     TRNPKX          ;NOW GET NPR ADDRESS
        LDX     2,X
        STX     NPRADA          ;STORE IN NPR ADDRESS REGISTER
        LDX     TRNPKX          ;UPDATE TRNPKX TO DATA
        INX
        INX
        INX
        INX
        STX     TRNPKX
1$:     LDX     TRNPKX          ;GET A WORD OF DATA
        LDX     0,X
        STX     NPRDTA          ;PUT DATA IN NPR OUT REGISTER
        LDA A   #,5             ;START NPR OPERATION
        STA A   NPRCSR
        LDX     TRNPKX          ;UPDATE DATA POINTER
        INX
        INX
        STX     TRNPKX
        LDX     NPRADA          ;UPDATE NPR ADDRESS
        INX
        INX
        STX     NPRADA
        LDX     TRNCNT          ;UPDATE WORD COUNT
        DEX
        STX     TRNCNT
        BNE     1$              ;LOOP UNTIL ALL WORDS TRANSFERRED
        LDX     LDRPKX          ;UPDATE BLOCK POINTER
        INX
        INX
        STX     LDRPKX
        BRA     TRNSFR          ;LOOP FOR MORE
2$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  RK11-D/RK05 BOOTSTRAP
        ;
        ;A SECTION FOR EACH POSSIBLE RK05 DEVICE
        ;
BTRK0:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
```

```
        FDB     157500  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
BTRK1:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157506  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
BTRK2:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157514  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
BTRK3:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157522  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
BTRK4:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157530  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
BTRK5:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157536  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
BTRK6:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157544  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
BTRK7:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157552  ;START ADDRESS
        FDB     340     ;PRIORITY
                        ;
RKBOOT: FDB     41.     ;WORDS IN BOOTSTRAP
        FDB     157500  ;LOAD ADDRESS
                        ;
FDB     012700  000000  ;RK0:   MOV     #0*20000,R0     ;UNIT 0
FDB     000424          ;BR     RKBOOT
FDB     012700  020000  ;RK1:   MOV     #1*20000,R0     ;UNIT 1
FDB     000421          ;BR     RKBOOT
FDB     012700  040000  ;RK2:   MOV     #2*20000,R0     ;UNIT 2
FDB     000416          ;BR     RKBOOT
FDB     012700  060000  ;RK3:   MOV     #3*20000,R0     ;UNIT 3
FDB     000413          ;BR     RKBOOT
FDB     012700  100000  ;RK4:   MOV     #4*20000,R0     ;UNIT 4
FDB     000410          ;BR     RKBOOT
FDB     012700  120000  ;RK5:   MOV     #5*20000,R0     ;UNIT 5
FDB     000405          ;BR     RKBOOT
FDB     012700  140000  ;RK6:   MOV     #6*20000,R0     ;UNIT 6
FDB     000402          ;BR     RKBOOT
FDB     012700  160000  ;RK7:   MOV     #7*20000,R0     ;UNIT Y
                        ;
FDB     105737  177404  ;RKBOOT:TSTB    @#177404        ;CONTROL READY ?
FDB     100375          ;       BPL     1$              ;NO - LOOP
FDB     010037  177412  ;       MOV     R0,@#177412     ;SELECT DRIVE
FDB     122737  000300  ;2$:    CMPB    #300,@#177400   ;CHECK DRIVE STATUS
FDB             177400  ;
FDB     101374          ;       BHI     2$              ;LOOP UNTIL READY
FDB     012700  177406  ;       MOV     #177406,R0      ;WORD COUNT REGISTER
FDB     012710  177400  ;       MOV     #-256.,(R0)     ;1 BLOCK
FDB     012740  000005  ;       MOV     #5,-(R0)        ;READ BLOCK
FDB     105710          ;3$:    TSTB    (R0)            ;LOADED ?
FDB     100376          ;       BPL     3$              ;LOOP UNTIL LOADED
FDB     005007          ;       CLR     PC              ;START SECONDARY BOOT
                        ;
        .PAGE
        .SBTTL  DY BOOTSTRAP
        ;
        ;       THIS SECTION TO BOOT DY0:
        ;
BTDY0:  FDB     2       ;TWO WORD AREA
```

```
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157500  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;
        ;       THIS SECTION TO BOOT DY1:
        ;
BTDY1:  FDB     2       ;TWO WORD AREA
        FDB     24      ;POWER FAIL RESTART ADDRESS
        FDB     157506  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;
        ;       DY BOOTSTRAP ROUTINE
        ;
DYBOOT: FDB     82.     ;WORDS IN BOOTSTRAP
        FDB     157500  ;LOAD ADDRESS
        ;
        ;                               DYCSR   =177160
        ;                               DYDAT   =177162
        ;                               DYDONE  =40
        ;                               DYTR    =200
        ;                               DYDENS  =400
        ;                               DYSIDE  =1000
        ;
DY0:    FDB     012704  000013          ;DY0:   MOV     #13,R4
        FDB     000402                  ;       BR      DYGO
DY1:    FDB     012704  000033          ;DY1:   MOV     #33,R4
        FDB     012700  177160          ;DYGO:  MOV     #DYCSR,R0
        FDB     012701  177162          ;       MOV     #DYDAT,R1
        FDB     005002                  ;       CLR     R2
        FDB     012703  000001          ;       MOV     #1,R3
        FDB     012705  000200          ;       MOV     #DYTR,R5
        FDB     032710  000040          ;1$:    BIT     #DYDONE,(R0)
        FDB     001775                  ;       BEQ     1$
        FDB     010410                  ;       MOV     R4,(R0)
        FDB     042704  177757          ;       BIC     #177757,R4
        FDB     062704  000007          ;       ADD     #7,R4
        FDB     032710  000040          ;2$:    BIT     #DYDONE,(R0)
        FDB     001775                  ;       BEQ     2$
        FDB     032711  000040          ;       BIT     #40,(R1)
        FDB     001402                  ;       BEQ     3$
        FDB     062704  000400          ;       ADD     #DYDENS,R4
        FDB     010410                  ;3$:    MOV     R4,(R0)
        FDB     030510                  ;4$:    BIT     R5,(R0)
        FDB     001776                  ;       BEQ     4$
        FDB     010311                  ;       MOV     R3,(R1)
        FDB     030510                  ;5$:    BIT     R5,(R0)
        FDB     001776                  ;       BEQ     5$
        FDB     012711  000001          ;       MOV     #1,(R1)
        FDB     032710  000040          ;6$:    BIT     #DYDONE,(R0)
        FDB     001775                  ;       BEQ     6$
        FDB     042704  000004          ;       BIC     #4,R4
        FDB     010410                  ;       MOV     R4,(R0)
        FDB     030510                  ;7$:    BIT     R5,(R0)
        FDB     001776                  ;       BEQ     7$
        FDB     032704  000400          ;       BIT     #DYDENS,R4
        FDB     001003                  ;       BNE     8$
        FDB     012711  000100          ;       MOV     #100,(R1)
        FDB     000402                  ;       BR      9$
        FDB     012711  000200          ;8$:    MOV     #200,(R1)
        FDB     030510                  ;9$:    BIT     R5,(R0)
        FDB     001776                  ;       BEQ     9$
        FDB     010211                  ;       MOV     R2,(R1)
        FDB     032710  000040          ;10$:   BIT     #DYDONE,(R0)
        FDB     001775                  ;       BEQ     10$
        FDB     005203                  ;       INC     R3
        FDB     005203                  ;       INC     R3
        FDB     052704  000004          ;       BIS     #4,R4
        FDB     032704  000400          ;       BIT     #DYDENS,R4
        FDB     001402                  ;       BEQ     11$
        FDB     062702  000200          ;       ADD     #200,R2
        FDB     062702  000200          ;11$:   ADD     #200,R2
        FDB     032702  001000          ;       BIT     #DYSIDE,R2
        FDB     001724                  ;       BEQ     3$
        FDB     005000                  ;       CLR     R0
```

```
        FDB     032704  000020          ;       BIT     #20,R4
        FDB     001401                  ;       BEQ     12$
        FDB     005200                  ;       INC     R0
        FDB     012701  177160          ;12$:   MOV     #DYCSR,R1
        FDB     005007                  ;       CLR     PC
        ;
        .PAGE
        .SBTTL  DX BOOTSTRAP
        ;
        ;
        ;       THIS SECTION TO BOOT DX0:
        ;
BTDX0:  FDB     2       ;TWO WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157600  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;
        ;       THIS SECTION TO BOOT DX1:
        ;
BTDX1:  FDB     2       ;TWO WORD TRANSFER
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157604  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;
        ;       DX BOOTSTRAP ROUTINE
        ;
DXBOOT: FDB     35.             ;WORDS IN BOOTSTRAP
        FDB     157600          ;LOAD ADDRESS
        ;
DX0:    FDB     005002          ;       DX0:    CLR     R2
        FDB     000402          ;               BR      0$
DX1:    FDB     012702  000020  ;       DX1:    MOV     #20,R2
        FDB     052702  100247  ;       0$:     BIS     #100247,R2
        FDB     012701  177170  ;       1$:     MOV     #177170,R1
        FDB     130211          ;       2$:     BITB    R2,(R1)
        FDB     001776          ;               BEQ     2$
        FDB     012703  000007  ;               MOV     #7,R3
        FDB     010100          ;               MOV     R1,R0
        FDB     010220          ;               MOV     R2,(R0)+
        FDB     000402          ;               BR      4$
        FDB     012710  000001  ;       3$:     MOV     #1,(R0)
        FDB     006203          ;       4$:     ASR     R3
        FDB     103402          ;               BCS     6$
        FDB     112711          ;               MOVB    (PC)+,(R1)
        FDB     111023          ;       5$:     MOVB    (R0),(R3)+
        FDB     030211          ;       6$:     BIT     R2,(R1)
        FDB     001776          ;               BEQ     6$
        FDB     100756          ;               BMI     1$
        FDB     103766          ;               BCS     3$
        FDB     105711          ;               TSTB    (R1)
        FDB     100771          ;               BMI     5$
        FDB     005000          ;               CLR     R0
        FDB     022710  000240  ;               CMP     #240,(R0)
        FDB     001347          ;               BNE     1$
        FDB     122702  000247  ;               CMPB    #247,R2
        FDB     005500          ;               ADC     R0
        FDB     005007          ;               CLR     PC
        ;
        .PAGE
        .SBTTL  CT BOOTSTRAP
        ;
        ;
        ;       THIS SECTION TO BOOT CT0:
        ;
BTCT0:  FDB     2       ;TWO WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
        FDB     157600  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;
        ;       THIS SECTION TO BOOT CT1:
        ;
BTCT1:  FDB     2       ;TWO WORD AREA
        FDB     24      ;POWER FAIL RESTART VECTOR
```

```
        FDB     157610  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;
        ;       CT BOOTSTRAP ROUTINE
        ;
CTBOOT: FDB     33.     ;WORDS IN BOOTSTRAP
        FDB     157600  ;LOAD ADDRESS
        ;
CT0:    FDB     012700  177500          ;CT0:   MOV     #177500,R0
        FDB     005010                  ;       CLR     (R0)
        FDB     000404                  ;       BR      RESTRT
CT1:    FDB     012700  177500          ;CT1:   MOV     #177500,R0
        FDB     012710  000400          ;       MOV     #400,(R0)
        FDB     010701                  ;RESTRT:        MOV     PC,R1
        FDB     062701  000052          ;       ADD     #TABLE-.,R1
        FDB     012702  000375          ;       MOV     #375,R2
        FDB     112103                  ;       MOVB    (R1)+,R3
        FDB     112110                  ;LOOP1: MOVB    (R1)+,(R0)
        FDB     100413                  ;       BMI     DONE
        FDB     130310                  ;LOOP2: BITB    R3,(R0)
        FDB     001776                  ;       BEQ     LOOP2
        FDB     105202                  ;       INCB    R2
        FDB     100772                  ;       BMI     LOOP1
        FDB     116012  000002          ;       MOVB    2(R0),(R2)
        FDB     120337  000000          ;       CMPB    R3,@#0
        FDB     001767                  ;       BEQ     LOOP2
        FDB     000000                  ;STOP:  HALT
        FDB     000755                  ;       BR      RESTRT
        FDB     005710                  ;DONE:  TST     (R0)
        FDB     100774                  ;       BMI     STOP
        FDB     005007                  ;       CLR     PC
                                        ;
        FDB     017600                  ;TABLE: .WORD   17600
        FDB     002415                  ;       .WORD   2415
        FDB     112024                  ;       .WORD   112024
                                        ;
        .PAGE
        .SBTTL  PAPER TAPE ABSOLUTE LOADER
        ;
        ;       PAPER TAPE
        ;
        ;
BTTT0:  FDB     2       ;TWO WORDS
        FDB     24      ;POWER FAIL VECTOR
        FDB     157500  ;STARTING ADDRESS
        FDB     340     ;PRIORITY
        ;
TTBOOT: FDB     96.     ;BOOT WORDS
        FDB     157500  ;LOAD ADDRESS
        ;
        ;                                       .=157500
        ;
        ;       177570                          L.SR    =177570
        ;
        ;       000000                          L.CKSM  =R0
        ;       000001                          L.ADR   =R1
        ;       000002                          L.BC    =R2
        ;       000003                          L.BYT   =R3
        ;       000005                          L.PTR   =R5
        ;
TT0:    FDB     010706                  ;L.LD1: MOV     PC,SP
        FDB     024646                  ;       CMP     -(SP),-(SP)
        FDB     010705                  ;       MOV     PC,L.PTR
        FDB     062705  000112          ;       ADD     #L.READ-.,L.PTR
        FDB     005001                  ;       CLR     L.ADR
        FDB     013716  177570          ;L.LD1B:        MOV     @#L.SR,(SP)
        FDB     006016                  ;       ROR     (SP)
        FDB     103402                  ;       BCS     L.LD1C
        FDB     005016                  ;       CLR     (SP)
        FDB     000403                  ;       BR      L.LD2
        FDB     006316                  ;L.LD1C:        ASL     (SP)
        FDB     001001                  ;       BNE     L.LD2
        FDB     010116                  ;       MOV     L.ADR,(SP)
        FDB     005000                  ;L.LD2: CLR     L.CKSM
        FDB     004715                  ;       JSR     PC,@L.PTR
```

```
        FDB     105303                  ;       DECB    L.BYT
        FDB     001374                  ;       BNE     L.LD2
        FDB     004715                  ;       JSR     PC,@L.PTR
        FDB     004767  000074          ;       JSR     PC,L.GWRD
        FDB     010402                  ;       MOV     R4,L.BC
        FDB     162702  000004          ;       SUB     #4,L.BC
        FDB     022702  000002          ;       CMP     #2,L.BC
        FDB     001443                  ;       BEQ     L.JMP
        FDB     004767  000054          ;       JSR     PC,L.GWRD
        FDB     061604                  ;       ADD     (SP),R4
        FDB     010401                  ;       MOV     R4,L.ADR
        FDB     004715                  ;L.LD3: JSR     PC,@L.PTR
        FDB     002004                  ;       BGE     L.LD4
        FDB     105700                  ;       TSTB    L.CKSM
        FDB     001753                  ;       BEQ     L.LD2
        FDB     000000                  ;L.BAD: HALT
        FDB     000751                  ;       BR      L.LD2
        FDB     110321                  ;L.LD4: MOVB    L.BYT,(L.ADR)+
        FDB     000770                  ;       BR      L.LD3
        FDB     016703  000152          ;L.READ:        MOV     DEVICE,L.BYT
        FDB     105213                  ;       INCB    (L.BYT)
        FDB     105713                  ;L.R1:  TSTB    (L.BYT)
        FDB     100376                  ;       BPL     L.R1
        FDB     116303  000002          ;       MOVB    2(L.BYT),L.BYT
        FDB     060300                  ;       ADD     L.BYT,L.CKSM
        FDB     042703  177400          ;       BIC     #177400,L.BYT
        FDB     005302                  ;       DEC     L.BC
        FDB     000207                  ;       RTS     PC
        FDB     012667  000052          ;L.GWRD:        MOV     (SP)+,L.TMP
        FDB     004715                  ;       JSR     PC,(L.PTR)
        FDB     010304                  ;       MOV     L.BYT,R4
        FDB     004715                  ;       JSR     PC,(L.PTR)
        FDB     000303                  ;       SWAB    L.BYT
        FDB     042703  000377          ;       BIC     #377,L.BYT
        FDB     050304                  ;       BIS     L.BYT,R4
        FDB     016707  000030          ;       MOV     L.TMP,PC
        FDB     004767  177746          ;L.JMP: JSR     PC,L.GWRD
        FDB     004715                  ;       JSR     PC,(L.PTR)
        FDB     105700                  ;       TSTB    L.CKSM
        FDB     001340                  ;       BNE     L.BAD
        FDB     006204                  ;       ASR     R4
        FDB     103002                  ;       BCC     L.JMP1
        FDB     000000                  ;       HALT
        FDB     000676                  ;       BR      L.LD1B
        FDB     006304                  ;L.JMP1:        ASL     R4
        FDB     061604                  ;       ADD     (SP),R4
        FDB     000114                  ;       JMP     (R4)
        FDB     000000  000000  000000  ;L.TMP: .WORD   0,0,0,0,0,0,0
        FDB     000000  000000  000000
        FDB     000000
        ;
        ;       157744                          .=157744
        ;
        FDB     016701  000026          ;BSLDR: MOV     DEVICE,R1
        FDB     012702  000352          ;LOOP:  MOV     #.-L.LD1+2,R2
        FDB     005211                  ;ENABLE:        INC     (R1)
        FDB     105711                  ;WAIT:  TSTB    (R1)
        FDB     100376                  ;       BPL     WAIT
        FDB     116162  000002  157400  ;       MOVB    2(R1),LOAD(R2)
        FDB     005267  177756          ;       INC     LOOP+2
        FDB     000765                  ;BRNCH: BR      LOOP
        FDB     177560                  ;DEVICE:        .WORD   177560
                                        ;
        .PAGE
        .SBTTL  MT BOOTSTRAP
        ;
        ;
        ;       THIS SECTION FOR MT BOOT
        ;
BTMT:   FDB     2       ;2 WORD SECTION
        FDB     24      ;POWER FAIL RESTART ADDRESS
        FDB     157500  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;
        ;       MT BOOTSTRAP
```

```
        ;
MTBOOT: FDB     96.     ;WORDS IN BOOT
        FDB     157500  ;LOAD ADDESS
        ;
MT0:    FDB     000000          ;HALT AT STARTUP
        FDB     010706 005746 012700 000212 060600 012701 172522 013746
        FDB     177570 006016 103401 005016 006316 005011 105711 100376
        FDB     006061 177776 103375 012711 000017 105711 100376 000000
        FDB     013702 177570 022121 010011 012741 177762 012741 060003
        FDB     105711 100376 020210 001410 012761 177777 000002 012711
        FDB     060011 105711 100376 000756 022020 005710 001417 022121
        FDB     014011 061611 016041 000002 012741 060003 105711 100376
        FDB     005740 005711 100337 011100 000000 000776 014002 005740
        FDB     006202 103721 006302 061602 000112 000000 000000 000000
        FDB     000000 036352 000200 000022 100000 000022 005015 004414
        FDB     004411 040520 000001 000001 000001 000001 000001 000001
        FDB     000001 000001 000001 000001 000001 000001 000001
        ;
```

```
        .PAGE
        .SBTTL  KUSARB PROGRAM
        ;
        .=110000        ;ROM AREA FOR KUSARB
        ;
BTKUS:  FDB     2       ;2 WORD AREA
        FDB     24      ;POWER FAIL RESTART ADDRESS
        FDB     150000  ;START ADDRESS
        FDB     340     ;PRIORITY
        ;
        ;       KUSARB UTILLITY PROGRAM
        ;
KUSLD:  FDB     3750    ;PROGRAM LENGTH
        FDB     150000  ;LOAD ADDRESS
        ;
KUSARB:                 ;BEGINNING OF PROGRAM
        .NLIST          ;SKIP ALL OF LISTING
        FDB     012700 177560 000410 000060 175610 177514 177550 172522
        FDB     177170 177160 177500 010067 002170 005027 000000 012737
        FDB     000300 177776 010700 062700 177730 010006 004567 002172
        FDB     020616 001415 010616 004567 002026 005015 051101 026502
        FDB     052513 026523 030126 026461 050515 055131 000000 105767
        FDB     177712 001171 004567 002070 004567 001764 005015 020052
        FDB     000000 004567 001152 105713 001003 005700 001730 000771
        FDB     004567 000010 004567 002450 000001 000721 010246 010702
        FDB     062702 000342 121327 000056 001434 121327 000101 002427
        FDB     121327 000132 003024 112304 006304 060704 062704 002406
        FDB     061404 016767 177534 001502 012767 001404 001136 005027
        FDB     000000 105713 004514 004567 001626 177415 000660 012602
        FDB     000205 126327 000001 000101 001372 122323 105267 177741
        FDB     000453 005212 042712 000001 000205 004567 001420 004567
        FDB     000516 001001 011201 010112 004567 000504 001001 011201
        FDB     010167 000040 004567 000470 001002 012701 000010 010167
        FDB     000012 105267 177653 004567 000270 012700 000000 004567
        FDB     000324 021227 000000 103146 004567 000232 005300 003367
        FDB     000762 004567 177654 000402 105267 177604 004567 000400
        FDB     001001 011201 010112 105767 177342 001002 004567 000202
        FDB     105767 177555 001051 012767 001407 000700 004567 000600
        FDB     004567 177450 004567 000344 001412 105767 177522 001003
        FDB     010137 000000 000402 110177 177770 004567 000110 005700
        FDB     001742 120027 000012 001642 120027 000177 001005 005412
        FDB     004567 000062 005412 000727 120027 000043 001334 004567
        FDB     000124 004567 000040 000727 004567 001436 004567 001612
        FDB     010405 004567 001424 004567 001600 020405 001607 110477
        FDB     177660 005212 000766 005212 105767 177364 001001 005212
        FDB     000205 105367 177352 002405 004567 001204 005015 000102
        FDB     000404 004567 001172 005015 000101 011201 004567 000026
        FDB     105267 177314 000205 105767 177306 001003 017701 177562
        FDB     000402 117701 177554 010446 005004 012746 000006 105767
        FDB     177256 003407 006216 000301 000402 006301 006104 006301
        FDB     006104 006301 006104 052704 000060 004567 001414 005004
        FDB     005316 003363 004567 001050 000040 005726 012604 000205
        FDB     105713 001004 005700 001402 004567 000224 005046 005027
        FDB     000000 012727 000010 000000 005001 010316 112304 001450
        FDB     105267 177754 120427 000053 001004 012767 000012 177746
        FDB     000763 120427 000055 001003 105167 177725 000755 162704
        FDB     000060 002437 020467 177716 002034 010146 016746 177706
        FDB     005316 001411 066601 000002 103373 022626 004567 001420
        FDB     000002 005726 000721 060401 103770 022626 112304 001347
        FDB     005303 005726 005767 177630 100001 005401 005767 177620
        FDB     000205 126727 177620 000010 001411 020427 177776 001002
        FDB     105713 001756 004567 001330 000003 000743 011603 000705
        FDB     010746 062716 001606 011603 004567 000736 001775 120427
        FDB     000052 001772 002431 120427 000054 001426 120427 000177
        FDB     001411 004567 001064 110423 105713 002356 004567 001240
        FDB     000004 000751 105767 176424 001347 020316 001407 005303
        FDB     004567 000474 177534 000740 004567 001016 120427 000040
        FDB     001002 020316 001731 105013 012603 010400 020027 000015
        FDB     001001 005000 000205 001410 111304 001471 004567 000356
        FDB     000110 000112 000167 176454 010701 062701 001056 010137
        FDB     000004 012737 000340 000006 105713 001024 010701 062701
        FDB     000064 016702 176240 010122 012722 000200 105713 001012
        FDB     010701 062701 001360 010137 000034 012737 000340 000036
        FDB     105713 001425 005203 000730 047513 000124 177720 177672
        FDB     177740 004767 000614 000002 016767 176146 000374 105267
        FDB     176172 004567 000110 001316 016767 000356 000346 000205
```

```
        FDB     004567 000262 176126 016767 176120 000372 000427 105267
        FDB     176356 000424 016704 176100 000405 016704 176076 000402
        FDB     016704 176056 010467 000302 004567 000212 000276 010467
        FDB     000324 062767 000004 000316 010427 000000 005203 111304
        FDB     001405 004567 000070 000010 000016 105713 000205 041120
        FDB     046110 045503 000000 177652 177666 177700 177704 177666
        FDB     177700 004567 177730 001226 016700 000176 000167 175766
        FDB     010446 122315 001217 105725 001374 005303 000427 010046
        FDB     010500 061500 120410 001404 105720 001374 022525 000406
        FDB     160500 162500 006300 061500 060005 061505 012600 000205
        FDB     010446 112504 003403 004567 000320 000773 005205 042705
        FDB     000001 012604 000205 012701 000004 011146 010711 062721
        FDB     000036 011146 005011 010546 062516 013646 005736 012611
        FDB     012641 005705 001013 004567 000416 000007 005005 000002
        FDB     012727 000000 000000 016727 177770 000000 000205 005000
        FDB     005200 100376 000005 005200 100776 016700 177744 022020
        FDB     010027 000000 010027 000000 000205 016704 177726 105767
        FDB     175532 001411 005214 005046 105714 100404 005216 100374
        FDB     000167 175506 005726 105714 100406 020467 175444 001773
        FDB     004767 000076 000770 016446 000002 042716 177600 021627
        FDB     000020 001402 012604 000205 010467 177626 004567 177636
        FDB     004567 177514 005015 050136 000000 000742 004767 000022
        FDB     010446 016704 177574 105714 100002 004567 177710 012604
        FDB     000207 010446 016704 175332 000766 010446 016704 177602
        FDB     105767 175575 003402 016704 177574 004567 000040 011664
        FDB     000002 020427 175614 001402 012604 000205 032716 177740
        FDB     001373 004567 177372 077577 077577 000177 000765 004767
        FDB     177652 020467 175256 001004 032714 000240 001770 000402
        FDB     105714 100365 005714 000205 011601 010706 062706 175202
        FDB     004567 000030 000011 004567 177306 000100 105067 175440
        FDB     004567 176142 000167 175206 105267 175200 004567 177376
        FDB     105367 175415 004567 177250 005015 003477 000000 105267
        FDB     175376 010146 012501 004567 176074 012601 105367 175360
        FDB     105267 175355 105767 175124 001203 000205 004567 177036
        FDB     004567 176134 001002 016701 000002 010127 000000 105267
        FDB     175317 032701 000001 001016 010702 062702 176650 020277
        FDB     175024 001003 052777 000100 175010 005067 175000 005037
        FDB     177776 010107 000167 175026 004567 177024 000103 004567
        FDB     176036 010102 004567 176030 001001 010601 005022 020201
        FDB     101775 000760 175376 175402 177740 000444 175246 001214
        FDB     175104 000664 176544 175076 174710 000164 004000 175066
        FDB     176722 175230 002606 177606 176356 175052 175050 175046
        FDB     175044 175042 003174 004160 000000 000000 000000 000000
        FDB     177400 016637 000002 177776 010566 000002 011605 116505
        FDB     177776 006305 060705 062705 000010 061505 000205 000022
        FDB     176720 175564 177434 177244 175660 174620 177056 000014
        FDB     004567 176674 005015 000000 000205 010702 062702 175306
        FDB     012512 012567 175156 012567 175142 000167 175130 004567
        FDB     176474 004567 175572 010705 062705 000124 042701 000001
        FDB     010146 016704 176444 004715 105300 001375 004715 005700
        FDB     001373 012767 000001 000146 004767 000146 010103 004767
        FDB     000140 061601 162703 000006 001004 004715 001010 000167
        FDB     177360 004715 110021 005303 001374 004715 001744 004567
        FDB     177232 000010 016701 000066 000167 177172 005214 005002
        FDB     020467 174356 001402 004767 177010 020467 176564 001402
        FDB     004767 176754 105714 100405 005202 001362 004567 177150
        FDB     000005 016400 000002 060067 000004 042727 177400 000000
        FDB     000207 004715 010001 004715 000300 060001 000207 004567
        FDB     176234 062767 000004 176222 004567 175324 010102 004567
        FDB     175316 001016 004767 000160 012701 000024 005000 004767
        FDB     000112 005301 003374 005302 003367 004767 000132 000205
        FDB     004767 000124 005003 012700 000001 004767 000076 010100
        FDB     160200 010001 062700 000007 004767 000060 010200 004767
        FDB     000052 005201 001405 112200 004767 000020 005301 000772
        FDB     010300 004767 000006 004767 000036 000205 016704 176052
        FDB     004567 176632 010064 000002 160003 000207 004767 177754
        FDB     000300 004767 177746 000207 026767 176016 176324 001002
        FDB     004567 176326 000207 121327 000103 001402 000167 174204
        FDB     004567 176162 174030 016704 174022 005214 004767 000030
        FDB     004767 000024 004767 000020 004567 176522 005764 177774
        FDB     100666 010264 000002 000766 042704 000006 004767 176354
        FDB     032724 100200 001771 100653 010102 010001 012400 005264
        FDB     177774 000207 000000 000000 000000 000000 000000 000000
        FDB     000000 000000 000000 000000 010046 004567 001422 010067
        FDB     000372 016767 001234 000354 016767 001222 000352 012600
        FDB     010167 001212 010467 001212 004567 175772 001206 010702
```

```
        FDB     062702 177704 112304 001404 004567 175662 000010 000020
        FDB     000167 173760 042111 046527 050114 042532 000124 000302
        FDB     000404 000402 000420 000554 000660 000672 000710 000002
        FDB     004567 174614 016700 001106 060700 062700 000007 110110
        FDB     000205 000000 000000 000000 000000 004567 175542 047503
        FDB     054520 000000 010702 062702 177556 005067 177550 005012
        FDB     005067 177562 010600 005400 062700 000020 010046 026727
        FDB     001014 000002 003406 012700 100000 006200 021600 101374
        FDB     010016 011267 000616 011667 177512 004567 000756 003070
        FDB     000000 004567 000030 004567 000742 003054 000002 004567
        FDB     000014 004567 001066 005713 001754 000167 173400 004567
        FDB     001076 010046 016746 000706 016746 000706 016767 000024
        FDB     000700 001455 016767 000020 000664 016701 000016 004567
        FDB     177542 012627 000000 012627 000000 012627 000000 000205
        FDB     004567 174334 010105 004567 000660 004567 000702 000100
        FDB     010146 004567 000416 000000 021613 001021 005213 060567
        FDB     177306 005767 177304 001007 004567 000560 002530 016701
        FDB     177266 000167 176134 004567 000422 000000 000752 004567
        FDB     176006 000005 004567 000562 004567 000604 000026 004567
        FDB     175766 000006 004567 174210 004567 000564 177762 004567
        FDB     000530 010113 010346 004567 175226 005015 040511 000040
        FDB     004567 174164 010167 177160 004567 175204 005015 040506
        FDB     000040 004567 174142 005101 066701 177134 010167 177132
        FDB     012603 004567 000216 000002 004567 000260 000002 005767
        FDB     177110 001340 004567 000364 002440 004567 000356 002326
        FDB     000205 004567 174750 105267 173247 004567 000336 002306
        FDB     004567 000140 000000 005767 177052 001757 004567 000314
        FDB     002416 004567 175052 005015 000000 011301 004567 173706
        FDB     016701 177004 004567 173676 016701 176776 005101 066701
        FDB     176766 004567 173660 000744 004567 173744 004567 000320
        FDB     177516 000205 004567 174710 051105 000117 004567 000214
        FDB     002164 000404 004567 174670 043117 000000 004567 000174
        FDB     002250 000205 010367 176716 026727 000162 000002 003412
        FDB     005212 011227 000000 016767 176660 176670 004567 000274
        FDB     010267 176662 012767 177762 176652 000411 005267 177744
        FDB     016767 176624 176640 016767 176620 176630 001763 026727
        FDB     000074 000002 003406 005767 176600 001403 000261 006067
        FDB     176570 012567 000006 004567 000042 002154 000000 026727
        FDB     000034 000002 003406 006367 176540 005712 001002 005067
        FDB     176542 005767 000026 001103 000167 172462 012700 000000
        FDB     012704 000000 010203 005723 012727 000010 000000 005725
        FDB     010546 064505 005300 003375 000205 004567 174502 005015
        FDB     044506 042514 000040 004567 173436 010203 005723 000205
        FDB     005725 004567 177706 001656 004567 177510 000000 005767
        FDB     176422 001002 064505 000406 021301 001404 004567 177654
        FDB     001756 000762 004567 177644 001670 000205 012746 000777
        FDB     166716 176360 000241 006016 105016 000316 062612 000205
        FDB     016700 177612 060700 062700 176513 111000 000241 000205
        FDB     004567 177754 110064 000001 004567 174742 105014 052514
        FDB     000167 174732 000002 005012 000205 000002 016712 177364
        FDB     005312 000205 000002 000205 000002 005212 011267 177344
        FDB     005012 004567 177352 000002 000205 026725 177326 002007
        FDB     005767 176220 001405 000261 006067 176210 000205 005213
        FDB     005067 176200 012605 005725 000205 016704 172102 012701
        FDB     000004 000167 176162 001540 010346 010246 010146 004567
        FDB     177710 000764 012767 000010 177422 016700 177226 006300
        FDB     006300 016701 176116 006301 016702 176112 012703 000001
        FDB     005715 001005 004567 000124 000007 012703 000003 004767
        FDB     000254 012703 177600 160301 003401 060103 005715 001007
        FDB     105714 001776 116422 000002 005203 002772 000406 105714
        FDB     001776 112264 000002 005203 002772 105714 001776 100003
        FDB     005564 000002 000772 004567 000130 005715 001403 004567
        FDB     000012 000005 005200 005701 002721 000454 010046 022700
        FDB     001750 101002 062700 176030 006100 012703 000010 022700
        FDB     006400 101002 062700 171400 006100 005303 003370 000300
        FDB     105200 012503 004767 000066 105714 001776 110064 000002
        FDB     105714 001776 000300 110064 000002 012600 105714 001776
        FDB     100402 005714 100013 012605 005367 177120 003236 004567
        FDB     174356 000026 012601 012602 012603 005725 000205 010046
        FDB     004567 177234 001402 052703 000020 012600 010314 000207
        FDB     004767 000420 016704 171472 012701 000003 000167 175550
        FDB     000000 177366 010346 010246 010146 004567 177274 000764
        FDB     012767 000010 177006 016700 176612 032767 000400 177740
        FDB     001001 006300 006300 016701 175472 016702 175470 012703
        FDB     000001 005715 001005 004567 000116 000007 012703 000003
        FDB     004767 000402 012703 177600 032767 000400 177662 001001
```

```
        FDB     006203 160301 003401 060103 105714 001776 005403 010364
        FDB     000002 105714 001776 010264 000002 060302 060302 105714
        FDB     001776 004567 000156 005715 001403 004567 000012 000005
        FDB     005200 005701 002724 000471 010046 042767 001000 177560
        FDB     022700 003720 101005 052767 001000 177544 162700 003720
        FDB     022700 001750 101002 062700 176030 006100 012703 000010
        FDB     022700 006400 101002 062700 171400 006100 005303 003370
        FDB     000300 105200 012503 004767 000174 105714 001776 110064
        FDB     000002 105714 001776 000300 110064 000002 012600 105714
        FDB     001776 100402 005714 100015 012605 004767 000026 005367
        FDB     176452 003221 004567 173710 000026 012601 012602 012603
        FDB     005725 000205 010346 010446 016704 171046 012703 000013
        FDB     004767 000062 012767 000764 177364 005067 177342 105714
        FDB     001776 032764 000002 000002 001402 006367 177340 032764
        FDB     000040 000002 001405 006367 177324 012767 000400 177300
        FDB     012604 012603 000207 010046 004567 176464 001402 052703
        FDB     000020 012600 056703 177250 010314 000207 016704 170716
        FDB     012701 000002 000167 175000 176476 004567 176442 060017
        FDB     000205 176472 012764 177777 000002 004567 176422 060013
        FDB     000205 176470 004567 176410 060007 000205 176452 012764
        FDB     177777 000002 112714 000011 000205 177152 004567 173336
        FDB     012700 060003 062500 010067 000022 016764 174670 000004
        FDB     016764 174660 000002 001746 004567 176324 000000 100011
        FDB     006164 177776 100415 005367 176122 001007 004567 173360
        FDB     000021 166467 000002 174614 000205 004567 177626 000745
        FDB     005067 174600 000205 016704 170512 012701 000001 000167
        FDB     174566 177570 004567 176230 000017 012746 175602 060716
        FDB     020526 001007 004567 176210 000003 152714 000020 000167
        FDB     173154 004567 176172 000015 152714 000020 000205 177526
        FDB     011446 004567 176152 000011 032726 004000 001403 004567
        FDB     176136 000015 000205 177516 032714 010000 001403 004567
        FDB     173166 000022 004567 176110 000001 000205 177502 112714
        FDB     000015 000205 177510 012700 000005 162500 010067 000022
        FDB     016767 174402 000152 016767 174372 000156 001745 004567
        FDB     176036 000000 004567 173012 122714 000204 001444 122714
        FDB     000202 001445 032714 004000 001033 005714 100022 032714
        FDB     020000 001411 004567 173050 000122 004567 172474 004567
        FDB     175332 000167 177572 005367 175562 001007 004567 173020
        FDB     000022 166767 000044 174254 000205 004567 177542 000720
        FDB     005067 174240 000205 116477 000002 000004 000403 113764
        FDB     000000 000002 005267 177770 005227 000000 001002 052714
        FDB     000020 000710 000000 000000 000000 000000 000000 000000
        .LIST   MD,MEB
        .LIST
        ;
        .END
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35

Table of contents

```
   38-   1      CHARACTER MODE HANDLER
   41-   1      LINE MODE HANDLER
   42-   1      MOTION HANDLERS
   43-   1      LINE MODE ROTATE PAGE
   44-   1      HOME AND CLSCN ROUTINES
   46-   1      LOCAL MONTOR SCANNER
   47-   1      TERMINATION CHECK ROUTINE
   48-   1      GET STRING ROUTINE
   49-   1      LOCAL MONITOR TABLE SEARCH ROUTINE
   50-   1      LOCAL MONITOR DISPATCH TABLE
   51-   1      LPRINT ROUTINE AND VARIATIONS
   52-   1      SYSTEM NOTES
   53-   1      TIME ROUTINE
   54-   1      CTLC ROUTINE
   55-   1      PEEK ROUTINE
   56-   1      POKE ROUTINE
   57-   1      EVALUATION ROUTINES
   58-   1      PRINT OCTAL VALUE ROUTINES
   59-   1      BAUD RATE ROUTINE
   60-   1      CHARACTER MASK ROUTINE
   61-   1      MASK LIST/PUT ROUTINES
   62-   1      ESCP ROUTINE
```

Table of contents

```
                                .SBTTL  6800 TERMINAL SOFTWARE
                                ;
                                ; THE SOFTWARE CONTAINED IN THIS PROGRAM IS FOR THE
                                ;OPERATION OF THE 6800 BASED VIDEO BOARD USED
                                ;WITH THE ARB-11 PROCESSOR.
                                ; THE SOFTWARE SUPPORTS A STANDARD DL-11 TYPE
                                ;INTERFACE FORMAT PLUS A SEPERATE INTERFACE FOR
                                ;THE GRAPHICS PROCESSOR.
                                ;
                                ;       TERMINAL INTERFACE
                                ;               KEYBOARD CSR = 177560
                                ;                        BUF = 177562
                                ;                        VEC = 60
                                ;               PRINTER  CSR = 177564
                                ;                        BUF = 177566
                                ;                        VEC = 64
                                ;
                                ;       GRAPHICS INTERFACE
                                ;               RESULT   CSR = 176670
                                ;                        BUF = 176672
                                ;                        VEC = 170
                                ;               COMMAND  CSR = 176674
                                ;                        BUF = 176676
                                ;                        VEC = 174
                                ;
```

DEFINITION OF I/O REGISTERS

```
                                .SBTTL  DEFINITION OF I/O REGISTERS
                                ;
                                ; THE I/O REGISTERS ARE ALL 6820 PIA CIRCUITS
                                ;
                                ;       REGISTER ADDRESSING
                                ; ___XX
                                ;    00 I/O REGISTER A (DIRECTION OF A)
                                ;    01 I/O REGISTER B (DIRECTION OF B)
                                ;    10 STATUS REGISTER A
                                ;    11 STATUS REGISTER B
                                ;
                                ;       REFER TO MC6820/6821 DATA SHEETS FOR DETAILS
                                ;       OF THE CONTROL REGISTERS
                                ;
```

I/O REGISTER LOCATIONS

```
                                .SBTTL  I/O REGISTER LOCATIONS
                                ;
000000                          DARHA=0         ;DISPLAY ADDRESS REGISTER H (A-PORT)
000001                          DARLB=1         ;DISPLAY ADDRESS REGISTER L (B-PORT)
000002                          PCRA1=2         ;CSR FOR DARHA [INIT  IRQ1]
000003                          PCRB1=3         ;CSR FOR DARHB
                                ;
000004                          CARHA=4         ;CURSOR ADDRESS REGISTER H (A-PORT)
000005                          CARLB=5         ;CURSOR ADDRESS REGISTER L (B-PORT)
000006                          PCRA2=6         ;CSR FOR CARHA
000007                          PCRB2=7         ;CSR FOR CARLB [VERTICAL RETRACE  NMI2]
                                ;
000010                          CSL=10          ;CURRENT SCAN LINE
000011                          ISL=11          ;INTERRUPT SCAN LINE
000012                          PCRA3=12        ;CSR FOR CSL
000013                          PCRB3=13        ;CSR FOR ISL [SCAN LINE  NMI3]
                                ;
000014                          CHARPL=14       ;CHARACTERS PER LINE
000015                          DSPCTL=15       ;DISPLAY CONTROL
                                                ;BIT    0 - GO
                                                ;       3 - GRAPHIC ENABLE
                                                ;       4 - CONTINUOS SCAN
                                                ;       5 - VIDEO POLARITY
                                                ;       7 - ODD FRAME
                                                ;
000016                          PCRA4=16        ;CSR FOR CHARPL [NMI STROBE]
000017                          PCRB4=17        ;CSR FOR DSPCTL [ODD FRAME  NMI1]
                                ;
000020                          WDFBA=20        ;WORD DATA FROM BUS <15:08>
000021                          WDFBB=21        ;WORD DATA FROM BUS <07:00>
000022                          PCRA5=22        ;CSR FOR WDFBA [DATA READY  IRQ8]
000023                          PCRB5=23        ;CSR FOR WDFBB
                                ;
000024                          WDTBA=24        ;WORD DATA TO BUS <15:08>
000025                          WDTBB=25        ;WORD DATA TO BUS <07:00>
```

```
37          000027                      PCRB6=27        ;CSR FOR WDTBB [DATA READY  IRQ9]
38                                      ;
39          000030                      TDF11=30        ;TERMNAL DATA FROM ARB-11
40          000031                      TDT11=31        ;TERMINAL DATA TO ARB-11
41          000032                      PCRA7=32        ;CSR FOR TDF11 [IRQ6]
42          000033                      PCRB7=33        ;CSR FOR TDT11 [IRQ7]
43                                      ;
44          000034                      KASCII=34       ;KEYBOARD
45          000035                      KBUTN=35        ;KEYBOARD BUTTONS
46                                                      ;BIT    0 - REPEAT
47                                                      ;       1 - BREAK
48                                                      ;       2 - PAPER
49                                                      ;       3 - HEREIS
50                                                      ;       4 - TAPE>
51                                                      ;       5 - <TAPE
52                                                      ;
53          000036                      PCRA8=36        ;CSR FOR KASCII [IRQ5]
54          000037                      PCRB8=37        ;CSR FOR KBUTN [BELL CONTROL]
55                                      ;
56          000040                      HA=40           ;PRINTER CONTROL REGISTER
57          000041                      HB=41           ;PRINTER DATA REGISTER
```

I/O REGISTER LOCATIONS

```
     58         000042                          PCRA9=42        ;CSR FOR HA
     59         000043                          PCRB9=43        ;CSR FOR HB
     60                                         ;
     61         000044                          PHOTO=44        ;PHOTOREADER DATA
     62         000045                          NPRCSR=45       ;NPR CONTROL REGISTER
     63                                                         ;BIT    0 - NPR GO
     64                                                         ;       1 - BUS C0 CONTROL
     65                                                         ;       2 - BUS C1 CONTROL
     66                                                         ;       3 - ADDRESS BIT 16
     67                                                         ;       4 - ADDRESS BIT 17
     68                                                         ;       5 - NXM ERROR
     69                                                         ;       6 - BUS GRANT LATE ERROR
     70                                                         ;       7 - NPR DONE
     71                                                         ;
     72         000046                          PCRA10=46       ;CSR FOR PHOTO [IRQ3]
     73         000047                          PCRB10=47       ;CSR FOR NPR CONTROL [IRQ10]
     74                                         ;
     75         000050                          NPRADA=50       ;NPR ADDRESS <15:08>
     76         000051                          NPRADB=51       ;NPR ADDRESS <07:00>
     77         000052                          PCRA11=52       ;CSR FOR NPRADA [TAPE<  IRQ11]
     78         000053                          PCRB11=53       ;CSR FOR NPRADB [TAPE>  IRQ12]
     79                                         ;
     80         000054                          NPRDTA=54       ;NPR DATA TO BUS <15:08>
     81         000055                          NPRDTB=55       ;NPR DATA TO BUS <07:00>
     82         000056                          PCRA12=56       ;CSR FOR NPRDTA [HERE IS  IRQ13]
     83         000057                          PCRB12=57       ;CSR FOR NPRDTB [PAPER  IRQ14]
     84                                         ;
     85         000060                          NPRDFA=60       ;NPR DATA FROM BUS <15:08>
     86         000061                          NPRDFB=61       ;NPR DTAT FROM BUS <07:00>
     87         000062                          PCRA13=62       ;CSR FOR NPRDFA [BREAK  IRQ15]
     88         000063                          PCRB13=63       ;CSR FOR NPRDFB [REPEAT  IRQ16]
     89                                         ;
```

MEMORY LAYOUT

```
                                        .SBTTL  MEMORY LAYOUT
                                        ;
                                        ;0-77            I/O REGISTERS (6820 PIA'S)
                                        ;
                                        ;100-377         DIRECT VARIABLES
         000100                         VARSAV=100              ;FIRST LOCATION
         000100                         .=VARSAV
                                        ;
                                        ;       SPECIFY GLOBLES FIRST
                                        ;
 000100     000                GRAPH:  .BYTE   0       ;GRAPHICS ON/OFF
 000101     000                RMTON:  .BYTE   0       ;REMOTE LINK ON/OFF FLAG
 000102     000                MODE:   .BYTE   0       ;TEXT MODE
 000103     000                CHARA:  .BYTE   0       ;CHARACTER FROM KEYBOARD ETAL
 000104     000                CHARB:  .BYTE   0       ;CHARACTER BEING PROCESSED BY .PRINT
 000105     000                CHRFLG: .BYTE   0       ;CTLP PRINT INHIBIT FLAG
 000106     000                UPDFLG: .BYTE   0       ;SCREEN UPDATE FLAG
                                        ;
                                        ;
                                        ;
 000107     000                T.60TH: .BYTE   0       ;60TH OF SECOND COUNTER
 000110     000                T.SCND: .BYTE   0       ;1 SECOND FLAG
 000111     000                T.CHAR: .BYTE   0       ;CHARACTER COUNT THIS INTERVAL
 000112     000                NALL:   .BYTE   0       ;CHAEARCTERS ALLOWED PER 60'TH
 000113     000                DATINH: .BYTE   0       ;DATA INHIBITED FLAG
 000114     000                ONLINE: .BYTE   0       ;ONLINE/LOCAL FLAG
 000115     000                LSTFLG: .BYTE   0       ;LIST GRAPHICS FLAG
 000116     000                ESCFLG: .BYTE   0       ;ESCAPE SEQUENCE FLAG
 000117     000     000        FL.TO:  .BYTE   0,0     ;FILL TO ADDRESS
 000121     000     000        FL.FRM: .BYTE   0,0     ;FILL FROM ADDRESS
 000123     000                FL.CNT: .BYTE   0       ;FILL COUNTER
 000124     000                LOCK:   .BYTE   0       ;SCROLL LOCK FLAG
 000125     000                DRCTN:  .BYTE   0       ;SCROLL DIRECTION
 000126     000     000        CLPNTR: .BYTE   0,0     ;ADDRESS OF CURRENT CHARACTER COUNT
 000130     000     000        CLBASE: .BYTE   0,0     ;ADDRESS OF CURRENT INPUT LINE
```

```
     36 000132     000     000          PLPNTR: .BYTE   0,0     ;ADDRESS OF CHARACTER COUNT OF LAST DISPL
AY LINE

     37 000134     000     000          PLBASE: .BYTE   0,0     ;ADDRESS OF FIRST CHARACTER OF LAST DISPL
AY LINE

     38 000136     000     000          OVBASE: .BYTE   0,0     ;ADDRESS OF LAST TEXT LINE BEFORE OVER TO
P

     39 000140     000     000          GLBASE: .BYTE   0,0     ;BASE ADDRESS FOR LINE MODE

     40 000142     000     000          GLCNT:  .BYTE   0,0     ;CURSOR POSITION ABOVE GLBASE

     41 000144     000                  GPCNT:  .BYTE   0       ;COLUMN POSITION IN CURRENT LINE

     42 000145     000                  LINES:  .BYTE   0       ;TEXT LINES IN DISPLAY

     43 000146     000                  TXTCBS: .BYTE   0       ;TWO BYTE CURRENT CONTROL BLOCK ADDRESS

     44 000147     000                  CURDSP: .BYTE   0       ;LOW ORDER OF TXTCBS

     45 000150     000     000          PGMSTK: .BYTE   0,0     ;SAVE STACK TEMPORARY

     46 000152     000     000          CURLOC: .BYTE   0,0     ;CURSOR ADDRESS

     47 000154     000                  CURCNT: .BYTE   0       ;CHARACTER POSITION IN CURRENT LINE

     48 000155     000                  DCTL:   .BYTE   0       ;CONTROL CHARACTER PRINT FLAG

     49 000156     000     000          BCRSTK: .BYTE   0,0     ;FIRST DISPLAY CONTROL BLOCK

     50 000160     000     000          CRSTK:  .BYTE   0,0     ;DISPLAY POINTER TO NEXT DISPLAY CONTROL
BLOCK

     51 000162     000     000          DSP0:   .BYTE   0,0     ;NEXT DISPLAY CONTROL BLOCK DATA

     52 000164     000     000          DSP1:   .BYTE   0,0

     53 000166     000     000          DSP2:   .BYTE   0,0

     54 000170     000                  IBIS:   .BYTE   0       ;OR'D WITH IN DATA

     55 000171     000                  IBIC:   .BYTE   0       ;AND'D WITH IN DATA

     56 000172     000                  OBIS:   .BYTE   0       ;OR'D WITH OUT DATA

     57 000173     000                  OBIC:   .BYTE   0       ;AND'D WITH OUT DATA
```

MEMORY LAYOUT

```
     58 000174     000                RMTFLS: .BYTE   0       ;CURRENT REMOTE FLASH CHARACTER
     59 000175     000     000        EJSR:   .BYTE   0,0     ;TRANSFER ADDRESS INTO ESCAPE MONITOR
     60 000177     000                ESCH:   .BYTE   0       ;ESCAPE SCANNER
     61 000200     000                ESCL:   .BYTE   0
     62 000201     000     000        LJSR:   .BYTE   0,0     ;TRANSFER ADDRESS INTO LOCAL MONITOR
     63 000203     000                LCLH:   .BYTE   0       ;LOCAL @= SCANNER
     64 000204     000                LCLL:   .BYTE   0
     65 000205     000     000        TEMPLA: .BYTE   0,0     ;TEMPORARIES USED BY LOCAL MONITOR
     66 000207     000     000        TEMPLB: .BYTE   0,0
     67 000211     000     000        TEMPLC: .BYTE   0,0
     68 000213     000                VH:     .BYTE   0
     69 000214     000                VL:     .BYTE   0
     70 000215     000                VAL:    .BYTE   0
     71 000216     000     000        TEMPEA: .BYTE   0,0     ;TEMPORARIES USED BY ESCAPE MONITOR
     72 000220     000     000        TEMPEB: .BYTE   0,0
     73 000222     000     000        GTABLE: .BYTE   0,0     ;SERVICE ROUTINE ADDRESS
     74 000224     000                GCNT:   .BYTE   0       ;COUNTER
     75 000225     000     000        TEMPGA: .BYTE   0,0     ;TEMPORARIES USED BY LNMODE
     76 000227     000     000        TEMPGB: .BYTE   0,0
     77 000231     000     000        LRVEC:  .BYTE   0,0     ;SERVICE ROUTINE ADDRESS
     78 000233     000     000        LSTPNT: .BYTE   0,0     ;STRING POINTER
     79 000235     000     000        LDRPKX: .BYTE   0,0     ;LOADER SAVE POINTER
     80 000237     000     000        TRNPKX: .BYTE   0,0     ;TRANSFER TEMPORARY
     81 000241     000     000        TRNCNT: .BYTE   0,0     ;TRANSFER WORD COUNTER
     82          000243                     VARSAV=.
     83                                     ;
```

MEMORY DEFINITIONS

```
                                        .SBTTL  MEMORY DEFINITIONS
                                        ;
                                        ;       THE FOLLOWING AREA IS SET UP TO CONTAIN
                                        ;       THE PARAMETERS TO CONTROL THE DISPLAY
                                        ;       HARDWARE FOR EACH TEXT LINE OR GRAPHICS
                                        ;       AREA TO BE DISPLAY. THE ORGANIZATION
                                        ;       IS AS FOLLOWS:
                                        ;
                                        ;  BYTES #1&2   ADDRESS OF FIRST BYTE OF DATA TO BE DISPL
AYED
                                        ;  BYTE #3      BYTES TO DISPLAY IN THIS LINE
                                        ;  BYTE #4      DISPLAY CONTROL BYTE
                                        ;  BYTE #5      NEW LINE NUMBER
                                        ;  BYTE #6      LINE NUMBER FOR NEXT INTERRUPT
                                        ;
000400                  TXGRCD  =400            ;TEXT LINE CONTROL AREA
000000                  CDBOT   =TXGRCD&377             ;LOW ORDER ADDRESS OF BOTTOM
000234                  CDTOP   =<TXGRCD+234>&377       ;LOW ORDER ADDRESS OF LAST CONTRO
L BLOCK
000642                  TXTSTP  =642            ;STOP TEXT CONTROL AREA
000650                  CLSRT   =650            ;CONTROL LINE START AREA
000656                  CLSTP   =656            ;CONTROL LINE STOP AREA
000664                  GPSRT   =664            ;GRAPHICS START AREA
000672                  GPSTP   =672            ;GRAPHICS STOP AREA
                                        ;
                                        ;
                                        ;       THE FOLLOWING AREA IS A LIST OF POINTERS
                                        ;       TO THE CONTROL BLOCKS BEING DISPLAYED AT THE
                                        ;       CURRENT TIME
                                        ;
000700                  DSPSTK  =700            ;DISPLAY STACK EXTENDS FROM 700 TO 777
                                        ;
001000                          LSTRNG=1000     ;STRING AREA FOR LOCAL MONITOR
001120                          ESTRNG=1120     ;ESCAPE MONITOR STRING AREA
001200                          YTABLE=1200     ;GRAPHIC Y TABLE
001230                          XTABLE=1230     ;GRAPHIC X TABLE
```

```
    36         001400                       GRPVAR=1400     ;GRAPHIC VARIABLE AREA
    37                                      ;
    38         006077             STACK     =6077           ;THE PROCESSOR STACK AREA IS FROM 6000 TO
6077
    39
    40                                      ;
    41                                      ;       THE FOLLOWING AREAS AND DEFINITIONS
    42                                      ;       ARE USED BY THE DISPLAY SECTION
    43                                      ;
    44         006100             CLPBOT    =6100           ;BUFFER AREA CONTAINING CHARS IN LINE
    45         007731             CLPTOP    =7731           ;LAST BYTE ADDRESS
    46                                      ;
    47                                      ;
    48                                      ; THESE MEMORY LOACATIONS CONTAIN THE
    49                                      ;INTERRUPT VECTOR ADDRESSES AS INITIALIZED
    50                                      ;BY THE RUNNING PROGRAM
    51                                      ;
    52                                      ; NON-MASKABLE INTERRUPT VECTORS
    53                                      ;
    54         007732                       NMI1    =7732           ;ODD FRAME
    55         007734                       NMI2    =7734           ;VERTICAL RETRACE
    56         007736                       NMI3    =7736           ;SCAN LINE
    57                                      ;
```

MEMORY DEFINITIONS

```
                                ; NORMAL INTERRUPT VECTORS
                                ;
007740                  IRQ1    =7740           ;ARB-11 BUS INIT
007742                  IRQ2    =7742           ;60 HZ CLOCK
007744                  IRQ3    =7744           ;PHOTO READER DATA
007746                  IRQ4    =7746           ;PRINTER CONTROL
007750                  IRQ5    =7750           ;KEYBOARD DATA
007752                  IRQ6    =7752           ;DATA FROM ARB-11
007754                  IRQ7    =7754           ;DATA TAKEN BY ARB-11
007756                  IRQ8    =7756           ;DISPLAY DATA FROM ARB-11
007760                  IRQ9    =7760           ;DISPLAY DATA TO ARB-11 TAKEN
007762                  IRQ10   =7762           ;ARB-11 BUS NPR DONE
007764                  IRQ11   =7764           ;SCROLL  <TAPE
007766                  IRQ12   =7766           ;SCROLL  >TAPE
007770                  IRQ13   =7770           ;HERE IS BUTTON
007772                  IRQ14   =7772           ;PAPER BUTTON
007774                  IRQ15   =7774           ;BREAK BUTTON
007776                  IRQ16   =7776           ;REPEAT BUTTON
                                ;
010000          CHRAM   =10000          ;4096 BYTE CHARACTER DEFINITION BUFFER
                                        ;CONTAINING 256 CHARACTER DEFINITIONS
                                ;
020000          CTLINE  =20000          ;80 CHARACTER CONTROL LINE
020120          CLBOT   =20120          ;4016 CHARACTER TEXT AREA
027657          CLTOP   =27657          ;LAST TEXT LINE LOCATION
027660          CL.TOP  =27660          ;LAST PLUS 1
                                ;
030000          GPBOT   =30000          ;20480 BYTE GRAPHICS AREA
                                        ;ALLOWS A 640(X) BY 256(Y) DISPLAY
030000          EVNSCN  =30000          ;SCAN ADDRESS FOR EVEN SCANS
054000          ODDSCN  =54000          ;SCAN ADDRESS FOR ODD SCANS
077777          GPTOP   =77777          ;LAST DISPLAY BYTE
                                ;
```

ROM LOCATIONS

```
                                        .SBTTL  ROM LOCATIONS
                                        ;
         100000                         CHROM1  =100000 ;CHARACTER ROM #1 (2K BYTES)
         104000                         CHROM2  =104000 ;CHARACTER ROM #2 (2K BYTES)
                                        ;
                                        ;       KUS5A ROM FROM 110000-117777 (4K BYTES)
                                        ;
                                        ;       GRAPHICS SYSTEM FROM 140000-147777 (4K BYTES)
         140000                         GPINIT  =140000         ;INIT ENTRY POINT
         140003                         GRPLOT  =140003         ;GRAPHIC ENTRY POINT
                                        ;
                                        ;       SPECIAL VECTOR GRAPHICS ROM
                                        ;       IS FROM 150000-153777 (2K BYTES)
                                        ;
                                        ;       SYSTEM SOFTWARE FROM 160000-177777 (8K BYTES)
                                        ;
                                        ;
                                        ;       SYSTEM VECTORS
                                        ;
         177770                         .=177770
 177770                                 FDB     START   ;IRQ (DUMBY)
    177770     340     000
 177772                                 FDB     START   ;SWI
    177772     340     000
 177774                                 FDB     START   ;NMI (DUMBY)
    177774     340     000
 177776                                 FDB     START   ;POWER FAIL RESTART
    177776     340     000
                                        ;
                                        ;
         160000                         PGMSAV=160000
         160000                         .=PGMSAV        ;PROGRAM ORIGIN
                                        ;
```

INITIALIZATION ROUTINES

```
                                        .SBTTL  INITIALIZATION ROUTINES
                                        ;
160000                          START:  SEI                     ;HOLD INTERRUPTS DURING SETUP
  160000     017
160001                                  LDS     #,STACK         ;STACK AREA
  160001     216     014     077
                                        ;
                                        .SBTTL  CLOCK INTERRUPT SETUP
                                        ;
160004                          CLKO:   LDX     #,CLKINT        ;ADDRESS OF SERVICE ROUTINE
  160004     316     342     030
160007                                  STX     IRQ2            ;SAVE AT VECTOR LOCATION
  160007     377     017     342
160012                                  LDA A   #,5             ;SET CLOCK INTERRUPT CONTROL
  160012     206     005
160014                                  STA A   PCRA6
  160014     227     026
160016                                  LDA A   WDTBA           ;CLEAR ANY PENDING INTERRUPT
  160016     226     024
                                        ;
                                        ; SET UP TIME IN CONTROL LINE
                                        ;
160020                                  LDA A   #,40            ;SPACE CHARACTER
  160020     206     040
160022                                  LDX     #,CTLINE        ;LOCATION OF CONTROL LINE
  160022     316     040     000
160025                          1$:     STA A   0,X             ;CLEAR OUT LINE
  160025     247     000
160027                                  INX
  160027     010
160030                                  CPX     #,CTLINE+70     ;UP TO CLOCK AREA ?
  160030     214     040     070
160033                                  BNE     1$              ;LOOP UNTIL ALL CLEARED
  160033     046     370
                                        ;
```

```
   160035     306     115
24 160037                          CMP B   CTLINE+102
   160037     361     040     102
25 160042                          BEQ     DSINIT          ;IF SO - SKIP TIME INIT
   160042     047     047
26 160044                     2$:  STA A   0,X             ;ELSE CLEAR REST OF THE LINE
   160044     247     000
27 160046                          INX
   160046     010
28 160047                          CPX     #,CLBOT
   160047     214     040     120
29 160052                          BNE     2$              ;LOOP UNTIL CLEAR
   160052     046     370
30 160054                          LDX     #,CTLINE+70     ;ADDRESS OF TIME DISPLAY
   160054     316     040     070
31 160057                          LDA A   #,'0            ;GET ZERO CHARACTER
   160057     206     060
32 160061                          STA A   0,X             ;STORE ZERO'S
   160061     247     000
33 160063                          STA A   1,X
   160063     247     001
```

CLOCK INTERRUPT SETUP

```
     34 160065                              STA A   3,X
        160065     247     003
     35 160067                              STA A   4,X
        160067     247     004
     36 160071                              STA A   6,X
        160071     247     006
     37 160073                              STA A   7,X
        160073     247     007
     38 160075                              LDA A   #,':            ;GET : CHARACTER
        160075     206     072
     39 160077                              STA A   2,X             ;STORE :'S
        160077     247     002
     40 160101                              STA A   5,X
        160101     247     005
     41 160103                              LDA A   #,'A            ;STORE A
        160103     206     101
     42 160105                              STA A   11,X
        160105     247     011
     43 160107                              LDA A   #,'M            ;STORE M
        160107     206     115
     44 160111                              STA A   12,X
        160111     247     012
     45                                     ;
     46                                     ; TIME DISPLAY = 00:00:00 AM
     47                                     ;
```

DISPLAY SYSTEM INITIALIZATION

```
                                        .SBTTL  DISPLAY SYSTEM INITIALIZATION
                                        ;
160113                          DSINIT: JSR     RAMSET          ;SET UP CHARACTER RAM
  160113     275     342     247
                                        ;
160116                                  LDX     #,CLPBOT        ;INITIALIZE POINTERS
  160116     316     014     100
160121                                  STX     CLPNTR
  160121     337     126
160123                                  STX     PLPNTR
  160123     337     132
160125                                  LDA A   #,1             ;1 CHARACTER LINES
  160125     206     001
160127                          1$:     STA A   0,X             ;LOAD IN NUMBER
  160127     247     000
160131                                  INX                     ;UPDATE ADDRESS
  160131     010
160132                                  CPX     #,CLPTOP+1      ;END OF REGION ?
  160132     214     017     332
160135                                  BNE     1$              ;LOOP UNTIL FINISHED
  160135     046     370
160137                                  LDX     #,CLBOT         ;GET TEXT AREA
  160137     316     040     120
160142                                  STX     CLBASE          ;INITIALIZE POINTERS
  160142     337     130
160144                                  STX     PLBASE
  160144     337     134
160146                                  LDA A   #,40            ;PUT SPACES IN ALL CHARACTERS
  160146     206     040
160150                          2$:     STA A   0,X             ;PLACE SPACES
  160150     247     000
160152                                  INX                     ;UPDATE ADDRESS
  160152     010
160153                                  CPX     #,CLTOP+120     ;END OF TEXT AREA ?
  160153     214     057     377
```

```
   160156     046     370
21 160160                              LDX     #,TXGRCD        ;CONTROL BLOCK AREA
   160160     316     001     000
22 160163                              STX     TXTCBS
   160163     337     146
23 160165                              CLR A
   160165     117
24 160166                              STA A   LOCK            ;LOCK DISPLAY TO INCOMING DATA
   160166     227     124
25 160170                              STA A   GRAPH           ;GRAPHICS OFF
   160170     227     100
26 160172                              INC A
   160172     114
27 160173                              STA A   CURCNT          ;CURRENT CHARACTER COUNT
   160173     227     154
28 160175                              STA A   UPDFLG          ;DO AN IMMEDIATE UPDATE
   160175     227     106
29                                     ;
```

DISPLAY SYSTEM INITIALIZATION

```
      1                                    ;
      2                                    ;       TEXT STOP CONTROL BLOCK
      3                                    ;
      4 160177                             LDX     #,CLBOT         ;DUMBY FIRST CHARACTER ADDRESS
        160177     316     040     120
      5 160202                             STX     TXTSTP
        160202     377     001     242
      6 160205                             LDX     #,1,0           ;TURN DISPLAY OFF
        160205     316     001     000
      7 160210                             STX     TXTSTP+2
        160210     377     001     244
      8 160213                             LDX     #,7,26          ;RESTART AT CONTROL LINE
        160213     316     007     026
      9 160216                             STX     TXTSTP+4
        160216     377     001     246
     10                                    ;
     11                                    ;       CONTROL LINE START CONTROL BLOCK
     12                                    ;
     13 160221                             LDX     #,CTLINE        ;ADDRESS OF FIRST CHARACTER
        160221     316     040     000
     14 160224                             STX     CLSRT
        160224     377     001     250
     15 160227                             LDX     #,120,1         ;80 CHARACTERS AND DISPLAY ON
        160227     316     120     001
     16 160232                             STX     CLSRT+2
        160232     377     001     252
     17 160235                             LDX     #,7,17          ;NEXT INTERRUPT AT 17
        160235     316     007     017
     18 160240                             STX     CLSRT+4
        160240     377     001     254
     19                                    ;
     20                                    ;       CONTROL LINE STOP CONTROL BLOCK
     21                                    ;
     22 160243                             LDX     #,CLBOT         ;DUMBY ADDRESS
        160243     316     040     120
```

```
   160246     377     001     256
24 160251                          LDX     #,1,0           ;DISPLAY OFF
   160251     316     001     000
25 160254                          STX     CLSTP+2
   160254     377     001     260
26 160257                          LDX     #,7,377         ;INHIBIT FURTHER INTERRUPTS
   160257     316     007     377
27 160262                          STX     CLSTP+4
   160262     377     001     262
28                                 ;
29                                 ;       GRAPHICS START CONTROL BLOCK
30                                 ;
31 160265                          LDX     #,EVNSCN        ;EVEN SCAN
   160265     316     060     000
32 160270                          STX     GPSRT
   160270     377     001     264
33 160273                          LDX     #,120,31        ;640 BITS AND CONTINUOUS SCAN
   160273     316     120     031
34 160276                          STX     GPSRT+2
   160276     377     001     266
35 160301                          LDX     #,377,177       ;128 LINE SCAN
```

DISPLAY SYSTEM INITIALIZATION

```
   160301     316     377     177
36 160304                              STX     GPSRT+4
   160304     377     001     270
37                                     ;
38                                     ;       GRAPHICS STOP CONTROL BLOCK
39                                     ;
40 160307                              LDX     #,EVNSCN        ;EVEN SCAN
   160307     316     060     000
41 160312                              STX     GPSTP
   160312     377     001     272
42 160315                              LDX     #,120,0         ;DISPLAY OFF
   160315     316     120     000
43 160320                              STX     GPSTP+2
   160320     377     001     274
44 160323                              LDX     #,377,7         ;RESTART IN 7 LINES
   160323     316     377     007
45 160326                              STX     GPSTP+4
   160326     377     001     276
46                                     ;
```

DISPLAY SYSTEM INITIALIZATION

```
      1                                     ;
      2                                     ;       SET UP DISPLAY STACK FOR TEXT
      3                                     ;
      4 160331                              JSR     LDDATA          ;DO FIRST LINE
        160331     275     343     046
      5 160334                              LDA A   #,25.           ;SET FOR 25 UPDATES
        160334     206     031
      6 160336                         3$:  PSH A                   ;SAVE COUNT
        160336     066
      7 160337                              INC     CLPNTR+1        ;GO TO NEXT LINE
        160337     174     000     127
      8 160342                              INC     CLBASE+1
        160342     174     000     131
      9 160345                              JSR     UPDTXT          ;UPDATE THIS LINE
        160345     275     343     103
     10 160350                              PUL A                   ;GET COUNT
        160350     062
     11 160351                              DEC A
        160351     112
     12 160352                              BNE     3$              ;LOOP UNTIL FINISHED
        160352     046     362
     13                                     ;
```

DISPLAY CONTROL REGISTER SETTUP

```
      1                                  .SBTTL  DISPLAY CONTROL REGISTER SETTUP
      2                                  ;
      3 160354                           LDX     #,0             ;CLEAR ALL REGISTERS
        160354     316     000     000
      4 160357                           STX     PCRA1           ;DISPLAY ADDRESS CONTROL
        160357     337     002
      5 160361                           STX     PCRA2           ;VERTICAL RETRACE CONTROL
        160361     337     006
      6 160363                           STX     PCRA3           ;SCAN INTERRUPT CONTROL
        160363     337     012
      7 160365                           STX     PCRA4           ;DISPLAY CONTROL
        160365     337     016
      8 160367                           LDX     #,377,377       ;DATA DIRECTION - OUT
        160367     316     377     377
      9 160372                           STX     DARHA           ;DISPLAY ADDRESS REGISTER
        160372     337     000
     10 160374                           STX     CARHA           ;CURSOR ADDRESS REGISTER
        160374     337     004
     11 160376                           LDX     #,0,377         ;SCAN COUNTER - IN
        160376     316     000     377
     12 160401                           STX     CSL             ;INTERRUPT LINE - OUT
        160401     337     010
     13 160403                           LDX     #,377,177       ;CHARACTERS PER LINE
        160403     316     377     177
     14 160406                           STX     CHARPL          ;DISPLAY CONTROL
        160406     337     014
     15 160410                           LDX     #,4,4           ;SET TO ACCESS DATA REGISTERS
        160410     316     004     004
     16 160413                           STX     PCRA1
        160413     337     002
     17 160415                           STX     PCRA2
        160415     337     006
     18 160417                           STX     PCRA3
        160417     337     012
     19 160421                           STX     PCRA4
```

```
    20 160423                          LDA A   #,377          ;INHIBIT SCAN LINE INTERRUPTS
       160423     206     377
    21 160425                          STA A   ISL
       160425     227     011
    22 160427                          LDX     #,1,0          ;DISPLAY OFF
       160427     316     001     000
    23 160432                          STX     CHARPL
       160432     337     014
    24 160434                          JSR     CHMODE         ;SET UP IN CHAR MODE
       160434     275     356     056
    25 160437                          LDX     CLBASE         ;GET POSITION
       160437     336     130
    26 160441                          STX     CURLOC         ;SET CURSOR POINTER
       160441     337     152
    27 160443                          STX     CARHA
       160443     337     004
    28                                 ;
    29                                 ;       SET UP INTERRUPT VECTORS
    30                                 ;
    31 160445                          LDX     #,DISINT       ;LINE INTERRUPT HANDLER
       160445     316     342     331
```

DISPLAY CONTROL REGISTER SETTUP

```
     32 160450                              STX     NMI3            ;STORE AT VECTOR LOCATION
        160450     377     017     336
     33 160453                              LDX     #,VERTIN        ;RETRACE HANDLER
        160453     316     343     140
     34 160456                              STX     NMI2            ;STORE AT VECTOR LOCATION
        160456     377     017     334
     35 160461                              LDX     #,54,4          ;SET UP NMI STROBE
        160461     316     054     004
     36 160464                              STX     PCRA4
        160464     337     016
     37 160466                              LDX     #,4,7
        160466     316     004     007
     38 160471                              STX     PCRA3           ;ENABLE LINE SCAN INTERRUPT
        160471     337     012
     39 160473                              STX     PCRA2           ;ENABLE RETRACE INTERRUPT
        160473     337     006
     40                                     ;
     41                                     ;       DISPLAY IS ON !
     42                                     ;
```

BREAK KEY SET UP

```
      1                                    .SBTTL  BREAK KEY SET UP
      2                                    ;
      3 160475                       BRKSET: LDX     #,BRKINT        ;SET UP INTERRUPT VECTOR
        160475     316     343     337
      4 160500                             STX     IRQ15
        160500     377     017     374
      5 160503                             LDA A   #,7             ;ENABLE INTERRUPT
        160503     206     007
      6 160505                             STA A   PCRA13
        160505     227     062
      7 160507                             LDA A   NPRDFA          ;CLEAR PENDING INTERRUPT
        160507     226     060
      8 160511                             LDA A   #,377           ;PLACE ON LINE
        160511     206     377
      9 160513                             STA A   ONLINE
        160513     227     114
     10 160515                             JSR     BREAK           ;DO A BREAK INTO LOCAL
        160515     275     343     344
     11                                    ;
```

DATAIN SET UP

```
      1                                        .SBTTL  DATAIN SET UP
      2                                        ;
      3 160520                          DATSET: LDX     #,TERMDI        ;ADDRESS OF INTERRUPT HANDLER
        160520     316     344     055
      4 160523                                 STX     IRQ6            ;PLACE IN VECTOR LOCATION
        160523     377     017     352
      5 160526                                 CLR     PCRA7           ;CLEAR CONTROL REGISTER
        160526     177     000     032
      6 160531                                 CLR     TDF11           ;DIRECTION IN
        160531     177     000     030
      7 160534                                 LDA A   #,55            ;SET TO ACCESS DATA AND INTERRUPT
        160534     206     055
      8 160536                                 STA A   PCRA7
        160536     227     032
      9 160540                                 LDA A   TDF11           ;CLEAR PENDING INTERRUPT
        160540     226     030
     10 160542                                 LDA A   #,12            ;SET UP FOR 600 CHARACTERS/SECOND
        160542     206     012
     11 160544                                 STA A   NALL
        160544     227     112
     12                                        ;
```

KEYBOARD SET UP

```
      1                                          .SBTTL  KEYBOARD SET UP
      2                                          ;
      3 160546                           KBSET:  LDX     #,RPINT         ;REPEAT BUTTON INTERRUPT HANDLER
        160546     316     344     152
      4 160551                                   STX     IRQ16
        160551     377     017     376
      5 160554                                   LDX     #,KBINT         ;KEYBOARD INTERRUPT HANDLER
        160554     316     344     154
      6 160557                                   STX     IRQ5
        160557     377     017     350
      7 160562                                   LDX     #,PHRDR         ;PHOTOREADER INTERRUPT HANDLER
        160562     316     344     146
      8 160565                                   STX     IRQ3
        160565     377     017     344
      9                                          ;
     10                                          ;       SET UP KEYBOARD
     11                                          ;
     12 160570                                   CLR     PCRA8           ;CLEAR KEYBOARD CONTROL
        160570     177     000     036
     13 160573                                   CLR     KASCII          ;SET DIRECTION IN
        160573     177     000     034
     14 160576                                   LDA A   #,7             ;SET TO INTERUPT ON EVERY CHARACT
ER
        160576     206     007
     15 160600                                   STA A   PCRA8
        160600     227     036
     16 160602                                   LDA A   KASCII          ;CLEAR PENDING INTERRUPT
        160602     226     034
     17                                          ;
     18                                          ;       SET UP BELL
     19                                          ;
     20 160604                                   CLR     PCRB8           ;CLEAR SWITCH CONTROL
        160604     177     000     037
     21 160607                                   CLR     KBUTN           ;SET FOR INPUT
        160607     177     000     035
     22 160612                                   LDA A   #,54            ;SET TO STROBE BELL
```

```
   160612     206     054
23 160614                          STA A   PCRB8           ;ON DUMBY WRITE TO KBUTN
   160614     227     037
24                                 ;
25                                 ;       SET UP REPEAT BUTTON
26                                 ;
27 160616                          LDA A   #,370
   160616     206     370
28 160620                          AND A   PCRB13          ;MASK BITS 0,1,&2
   160620     224     063
29 160622                          ORA A   #,5             ;REPEAT CONTROL
   160622     212     005
30 160624                          STA A   PCRB13
   160624     227     063
31 160626                          LDA A   NPRDFB          ;CLEAR PENDING INTERRUPT
   160626     226     061
32                                 ;
33                                 ;       SET UP DATAOUT
34                                 ;
35 160630                          CLR     PCRB7           ;CLEAR OUT CONTROL
   160630     177     000     033
36 160633                          LDA A   #,377           ;SET FOR OUTPUT
```

KEYBOARD SET UP

```
        160633     206     377
     37 160635                              STA A   TDT11
        160635     227     031
     38 160637                              LDA A   #,54            ;READY FLAG STROBE
        160637     206     054
     39 160641                              STA A   PCRB7
        160641     227     033
     40                                     ;
     41                                     ;       PHOTOREADER SET UP
     42                                     ;
     43 160643                              CLR     PCRA10          ;CLEAR PHOTOREADER CONTROL
        160643     177     000     046
     44 160646                              CLR     PHOTO           ;SET FOR INPUT
        160646     177     000     044
     45 160651                              LDA A   #,7             ;SET FOR INTERRUPT
        160651     206     007
     46 160653                              STA A   PCRA10
        160653     227     046
     47 160655                              LDA A   PHOTO           ;CLEAR PENDING INTERRUPT
        160655     226     044
     48                                     ;
     49                                     ;       INITIALIZE MASKING VARIABLES
     50                                     ;
     51 160657                              CLR A
        160657     117
     52 160660                              STA A   IBIS            ;OR'D WITH INCOMING DATA
        160660     227     170
     53 160662                              STA A   OBIS            ;OR'D WITH OUTGOING DATA
        160662     227     172
     54 160664                              LDA A   #,377
        160664     206     377
     55 160666                              STA A   IBIC            ;AND'D WITH INCOMING DATA
        160666     227     171
     56 160670                              STA A   OBIC            ;AND'D WITH OUTGOING DATA
        160670     227     173
```

```
    58                                  ;       CONTROL CHARACTER PRINTING
    59                                  ;
    60 160672                           CLR     DCTL            ;ALLOW PRINTING
       160672     177     000     155
    61                                  ;
```

SCAN BUTTON SET UP

```
      1                                       .SBTTL  SCAN BUTTON SET UP
      2                                       ;
      3 160675                          SETSCN: LDX     #,SCRLUP        ;SCROLL UP INTERRUPT HANDLER
        160675     316     344     175
      4 160700                                  STX     IRQ12
        160700     377     017     366
      5 160703                                  LDX     #,SCRLDN        ;SCROLL DOWN INTERRUPT HANDLER
        160703     316     344     331
      6 160706                                  STX     IRQ11
        160706     377     017     364
      7 160711                                  LDA A   #,5             ;ENABLE INTERRUPTS
        160711     206     005
      8 160713                                  STA A   PCRA11
        160713     227     052
      9 160715                                  STA A   PCRB11
        160715     227     053
     10 160717                                  LDA A   NPRADA          ;CLEAR PENDING INTERRUPT
        160717     226     050
     11 160721                                  LDA A   NPRADB          ;CLEAR PENDING INTERRUPT
        160721     226     051
     12                                         ;
```

REMOTE INTERFACE SETUP

```
      1                                      .SBTTL  REMOTE INTERFACE SETUP
      2                                      ;
      3 160723                       RMTSET: LDX     #,0             ;GET TO DIRECTION REGISTERS
        160723     316     000     000
      4 160726                               STX     PCRA9
        160726     337     042
      5 160730                               LDX     #,140,0         ;SET DIRECTIONS
        160730     316     140     000
      6 160733                               STX     HA
        160733     337     040
      7 160735                               LDX     #,4,4           ;GET TO DATA REGISTERS
        160735     316     004     004
      8 160740                               STX     PCRA9
        160740     337     042
      9 160742                               LDX     #,PPRKEY        ;SETUP INTERRUPT VECTOR
        160742     316     345     135
     10 160745                               STX     IRQ14
        160745     377     017     372
     11 160750                               LDA A   #,7             ;SET INTERRUPT CONTROL
        160750     206     007
     12 160752                               STA A   PCRB12
        160752     227     057
     13 160754                               LDA A   NPRDTB          ;CLEAR PENDING INTERRUPT
        160754     226     055
     14 160756                               CLR     RMTON           ;REMOTE IS OFF
        160756     177     000     101
     15 160761                               LDX     #,REMOTE        ;LOAD REMOTE INTERRUPT VECTOR
        160761     316     344     121
     16 160764                               STX     IRQ4
        160764     377     017     346
     17 160767                               LDX     #,MSGROF        ;LOAD 'REMOTE  OFF' MESSAGE
        160767     316     345     245
     18 160772                               JSR     RMTMOV
        160772     275     345     205
     19                                      ;
```

OTHER INITS

```
      1                                     .SBTTL  OTHER INITS
      2                                     ;
      3 160775                              JSR     GPINIT          ;INITIALIZE GRAPHICS SYSTEM
        160775     275     300     000
      4                                     ;
      5 161000                              LDX     #,LCLMON        ;SET UP TRANSFER VECTORS
        161000     316     350     004
      6 161003                              STX     LJSR
        161003     337     201
      7 161005                              LDX     #,ESCAPE
        161005     316     354     340
      8 161010                              STX     EJSR
        161010     337     175
      9                                     ;
     10 161012                              CLR     MODE            ;CHMODE INITIALLY
        161012     177     000     102
     11 161015                              CLR     LSTFLG          ;DISABLE LIST GRAPHICS
        161015     177     000     115
     12 161020                              CLR     ESCFLG          ;DISABLE ESCAPE MONITOR
        161020     177     000     116
     13                                     ;
     14                                     ;       END OF INITIALIZATION
     15                                     ;
     16 161023                              CLI                     ;ALLOW INTERRUPTS
        161023     016
     17                                     ;
     18 161024                         FREVER: BRA     FREVER
        161024     040     376
     19 161026                              BRA     FREVER
        161026     040     374
     20                                     ;
```

CLOCK INTERRUPT HANDLER

```
      1                                        .SBTTL  CLOCK INTERRUPT HANDLER
      2                                        ;
      3 161030                          CLKINT: LDA A   DATINH          ;DATA INHIBITED ?
        161030     226     113
      4 161032                                  BEQ     1$              ;IF NOT - SKIP
        161032     047     011
      5 161034                                  LDA A   #,55            ;RESTART TRANSFERS FROM ARB-11
        161034     206     055
      6 161036                                  STA A   PCRA7
        161036     227     032
      7 161040                                  LDA A   TDF11           ;SET READY FLAG
        161040     226     030
      8 161042                                  CLR     DATINH          ;ENABLED FLAG
        161042     177     000     113
      9 161045                               1$:     BSR     TMCHEK          ;CHECK TIME
        161045     215     004
     10 161047                                  CLR     T.CHAR          ;CHARACTER COUNT FOR NEXT INTERVA
L
        161047     177     000     111
     11 161052                                  RTI                     ;RETURN FROM INTERRUPT
        161052     073
     12                                         ;
     13 161053                          TMCHEK: DEC     T.60TH          ;ANOTHER 60TH
        161053     172     000     107
     14 161056                                  BNE     1$              ;NOT 1 SECOND - SKIP
        161056     046     007
     15 161060                                  LDA A   #,60.           ;60 HZ CLOCK
        161060     206     074
     16 161062                                  STA A   T.60TH          ;RESET COUNTER
        161062     227     107
     17 161064                                  INC     T.SCND          ;UPDATE SECOND FLAG
        161064     174     000     110
     18 161067                               1$:     LDA A   WDTBA           ;CLEAR INTERRUPT FLAG
        161067     226     024
     19 161071                                  RTS                     ;RETURN TO CALLER
        161071     071
```

```
    20                                      ;
    21                                      ;       PROGRAM TIME CHECK ROUTINE
    22                                      ;
    23 161072                       TMCK:   TST     PCRA6           ;CLOCK FLAG SET ?
       161072     175     000     026
    24 161075                               BPL     1$              ;IF NOT - SKIP
       161075     052     004
    25 161077                               PSH A                   ;SAVE A
       161077     066
    26 161100                               BSR     TMCHEK          ;CHECK TIME
       161100     215     351
    27 161102                               PUL A                   ;RESTORE A
       161102     062
    28 161103                       1$:     RTS                     ;RETURN  TO CALLER
       161103     071
    29                                      ;
```

TIME CLOCK ROUTINE

```
      1                                    .SBTTL  TIME CLOCK ROUTINE
      2                                    ;
      3                                    ; THIS ROUTINE IS EXECUTED DURING VERTICAL
      4                                    ;RETRACE TO PREVENT NOTICABLE FLECKS ON THE SCREEN DISPLA
Y
      5                                    ;
      6 161104                      TIMCLK: LDA A   T.SCND          ;A SECOND YET ?
        161104     226     110
      7 161106                              BEQ     1$              ;IF NOT - FINISHED
        161106     047     102
      8 161110                              CLR     T.SCND          ;RESET FLAG
        161110     177     000     110
      9 161113                              LDX     #,CTLINE+70     ;TIME SPACE AREA
        161113     316     040     070
     10 161116                              INC     7,X             ;UPDATE SECONDS
        161116     154     007
     11 161120                              LDA A   #,'9            ;ARE WE PAST 9
        161120     206     071
     12 161122                              CMP A   7,X
        161122     241     007
     13 161124                              BGE     1$              ;IF NOT - FINISHED
        161124     054     064
     14 161126                              LDA B   #,'0            ;RESTORE DIGIT TO 0
        161126     306     060
     15 161130                              STA B   7,X
        161130     347     007
     16 161132                              INC     6,X             ;ON CARRY - UPDATE 10'S
        161132     154     006
     17 161134                              LDA A   #,'5            ;ARE WE PAST 5 TENS ?
        161134     206     065
     18 161136                              CMP A   6,X
        161136     241     006
     19 161140                              BGE     1$              ;IF NOT - FINISHED
        161140     054     050
     20 161142                              STA B   6,X             ;RESTORE DIGIT TO 0
        161142     347     006
```

```
    21 161144                          INC     4,X             ;IF CARRY - UPDATE MINUTES
       161144     154     004
    22 161146                          LDA A   #,'9            ;ARE WE PAST 9 MINUTES ?
       161146     206     071
    23 161150                          CMP A   4,X
       161150     241     004
    24 161152                          BGE     1$              ;IF NOT - FINISHED
       161152     054     036
    25 161154                          STA B   4,X             ;RETORE DIGIT TO 0
       161154     347     004
    26 161156                          INC     3,X             ;IF CARRY - UPDATE 10'S
       161156     154     003
    27 161160                          LDA A   #,'5            ;ARE WE PAST 5 TENS ?
       161160     206     065
    28 161162                          CMP A   3,X
       161162     241     003
    29 161164                          BGE     1$              ;IF NOT - FINISHED
       161164     054     024
    30 161166                          STA B   3,X             ;RESTORE DIGIT TO 0
       161166     347     003
    31 161170                          INC     1,X             ;UPATE HOURS
       161170     154     001
```

TIME CLOCK ROUTINE

```
     32 161172                              LDA A   #,'1            ;ARE WE PAST 10 HOURS
        161172     206     061
     33 161174                              CMP A   0,X
        161174     241     000
     34 161176                              BEQ     2$              ;IF SO - USE SPECIAL CHECKS
        161176     047     013
     35 161200                              LDA A   #,'9            ;ARE WE PAST 9 HOURS ?
        161200     206     071
     36 161202                              CMP A   1,X
        161202     241     001
     37 161204                              BGE     1$              ;IF NOT - FINISHED
        161204     054     004
     38 161206                              STA B   1,X             ;RESTORE DIGIT TO 0
        161206     347     001
     39 161210                              INC     0,X             ;UPDATE 10'S
        161210     154     000
     40 161212                      1$:     RTS                     ;FINISHED
        161212     071
     41 161213                      2$:     LDA A   #,'2            ;PAST 2 HOURS ?
        161213     206     062
     42 161215                              CMP A   1,X
        161215     241     001
     43 161217                              BGT     1$              ;IF NOT - FINISHED
        161217     056     371
     44 161221                              BEQ     3$              ;IF AT 12 HOURS - SKIP
        161221     047     006
     45 161223                              STA B   0,X             ;ELSE COUNT TO 01 HOURS
        161223     347     000
     46 161225                              INC B
        161225     134
     47 161226                              STA B   1,X
        161226     347     001
     48 161230                              RTS                     ;FINISHED
        161230     071
     49 161231                      3$:     LDA A   #,'A            ;AM OR PM ?
```

```
    50 161233                              CMP A   11,X
       161233     241     011
    51 161235                              BEQ     4$              ;ON AM - SKIP
       161235     047     003
    52 161237                              STA A   11,X            ;ELSE MAKE AM
       161237     247     011
    53 161241                              RTS                     ;FINISHED
       161241     071
    54 161242                         4$:  LDA A   #,'P            ;MAKE PM
       161242     206     120
    55 161244                              STA A   11,X
       161244     247     011
    56 161246                              RTS                     ;FINISHED
       161246     071
    57                                     ;
```

CHARACTER RAM SET UP

```
      1                                      .SBTTL  CHARACTER RAM SET UP
      2                                      ;
      3 161247                        RAMSET: LDX     #,CHRAM         ;RAM AREA
        161247     316     020     000
      4 161252                                CLR     0,X             ;CLEAR FIRST 4
        161252     157     000
      5 161254                                CLR     1,X
        161254     157     001
      6 161256                                CLR     2,X
        161256     157     002
      7 161260                                CLR     3,X
        161260     157     003
      8 161262                                LDX     #,CHRAM+4       ;RESTORE ADDRESS
        161262     316     020     004
      9 161265                                STX     FL.TO
        161265     337     117
     10 161267                                LDX     #,CHROM1        ;ROM AREA
        161267     316     200     000
     11 161272                                STX     FL.FRM
        161272     337     121
     12 161274                           1$:  JSR     TMCK            ;CHECK CLOCK WHILE IN THIS ROUTIN
E
        161274     275     342     072
     13 161277                                LDX     FL.FRM          ;UPDATE ADDRESS
        161277     336     121
     14 161301                                LDA A   0,X             ;GET BYTE
        161301     246     000
     15 161303                                LDA B   1,X             ;NEXT BYTE
        161303     346     001
     16 161305                                INX
        161305     010
     17 161306                                INX
        161306     010
     18 161307                                STX     FL.FRM
        161307     337     121
     19 161311                                LDX     FL.TO           ;UPDATE ADDRESS
```

```
        161311     336     117
    20  161313                             STA A   0,X             ;MOVE BYTE
        161313     247     000
    21  161315                             STA B   1,X             ;AND NEXT BYTE
        161315     347     001
    22  161317                             INX
        161317     010
    23  161320                             INX
        161320     010
    24  161321                             STX     FL.TO
        161321     337     117
    25  161323                             CPX     #,CHRAM+10000   ;END OF RAM AREA ?
        161323     214     040     000
    26  161326                             BNE     1$              ;IF NOT - LOOP
        161326     046     344
    27  161330                             RTS                     ;FINISHED
        161330     071
    28                                     ;
```

DISPLAY INTERRUPT HANDLER

```
      1                                     .SBTTL  DISPLAY INTERRUPT HANDLER
      2                                     ;
      3                                     ;       LOAD UP PARAMETERS FOR NEW LINE
      4                                     ;
      5 161331                        DISINT: LDX     DSP0            ;LOAD SCAN ADDRESS
        161331     336     162
      6 161333                                STX     DARHA
        161333     337     000
      7 161335                                LDX     DSP1            ;LOAD CHARACTER COUNT AND DISPLAY
 CONTROL
        161335     336     164
      8 161337                                STX     CHARPL
        161337     337     014
      9 161341                                LDX     DSP2            ;LOAD NEW LINE # AND NEXT INTERRU
PT LINE #
        161341     336     166
     10 161343                                STX     CSL
        161343     337     010
     11                                     ;
     12                                     ;       NOW SET UP FOR NEXT INTERRUPT
     13                                     ;
     14 161345                                BSR     DSPUPD          ;GO LOAD UP DSP0-DSP2
        161345     215     005
     15 161347                                LDA A   ISL             ;CLEAR INTERRUPT FLAG
        161347     226     011
     16 161351                                LDA A   CHARPL          ;NMI STROBE
        161351     226     014
     17 161353                                RTI                     ;RETURN FROM LINE INTERRUPT
        161353     073
     18                                     ;
     19 161354                       DSPUPD: STS     PGMSTK          ;SAVE STACK POINTER
        161354     237     150
     20 161356                                LDX     CRSTK           ;POINTER TO CONTROL BLOCKS
        161356     336     160
     21 161360                                LDX     0,X             ;GET ADDRESS OF NEXT CONTROL BLOC
K
        161360     356     000
```

```
    22 161362                          LDS     0,X             ;GET CONTROL BLOCK INTO DSP0-DSP2
       161362     256     000
    23 161364                          STS     DSP0
       161364     237     162
    24 161366                          LDS     2,X
       161366     256     002
    25 161370                          STS     DSP1
       161370     237     164
    26 161372                          LDS     4,X
       161372     256     004
    27 161374                          STS     DSP2
       161374     237     166
    28 161376                          LDX     CRSTK           ;UPDATE CRSTK POINTER
       161376     336     160
    29 161400                          INX
       161400     010
    30 161401                          INX
       161401     010
    31 161402                          STX     CRSTK           ;SAVE NEW POINTER
       161402     337     160
    32 161404                          LDS     PGMSTK          ;RESTORE STACK POINTER
       161404     236     150
    33 161406                          RTS                     ;RETURN TO CALLER
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35 Page 22-1

DISPLAY INTERRUPT HANDLER

```
        161406     071
     34                                      ;
```

CONTROL BLOCK SCROLL ROUTINES

```
      1                                    .SBTTL  CONTROL BLOCK SCROLL ROUTINES
      2                                    ;
      3 161407                     ROTUP:  BSR     CURUP           ;UPDATE CONTROL BLOCK POINTER
        161407     215     003
      4 161411                             BSR     LDDATA          ;LOAD CONTROL BLOCK
        161411     215     033
      5 161413                             RTS                     ;RETURN TO CALLER
        161413     071
      6                                    ;
      7 161414                     CURUP:  LDA A   CURDSP          ;GET BLOCK POSITION
        161414     226     147
      8 161416                             CMP A   #,CDTOP         ;AT TOP ?
        161416     201     234
      9 161420                             BCS     1$              ;IF NOT - SKIP
        161420     045     002
     10 161422                             LDA A   #,CDBOT-6       ;SET BOTTOM
        161422     206     372
     11 161424                     1$:     ADD A   #,6             ;UPDATE BLOCK POSITION
        161424     213     006
     12 161426                             STA A   CURDSP          ;SAVE NEW POSITION
        161426     227     147
     13 161430                             RTS                     ;RETURN TO CALLER
        161430     071
     14                                    ;
     15 161431                     ROTDWN: BSR     LDDATA          ;LOAD UP CONTROL BLOCK
        161431     215     013
     16                                                            ;FALL THROUGH TO CURDWN
     17 161433                     CURDWN: LDA A   CURDSP          ;GET POINTER
        161433     226     147
     18 161435                             BNE     1$              ;IF NOT AT BOTTOM (CDBOT=0) - SKI
P
        161435     046     002
     19 161437                             LDA A   #,CDTOP+6       ;LOAD TOP
        161437     206     242
     20 161441                     1$:     SUB A   #,6             ;UPDATE POINTER
        161441     200     006
```

```
     21 161443                                STA A   CURDSP          ;SAVE POINTER
        161443     227     147
     22 161445                                RTS                     ;RETURN TO CALLER
        161445     071
     23                                       ;
     24 161446                          LDDATA: LDX     PLPNTR          ;GET ADDRESS OF LINE CHARACTER CO
UNT
        161446     336     132
     25 161450                                LDA B   0,X             ;GET COUNT
        161450     346     000
     26 161452                                LDX     TXTCBS          ;GET CONTROL BLOCK ADDRESS
        161452     336     146
     27 161454                                STA B   2,X             ;SET COUNT
        161454     347     002
     28 161456                                LDA B   #,1             ;SET DISPLAY CONTROL FOR TEXT LIN
E
        161456     306     001
     29 161460                                STA B   3,X
        161460     347     003
     30 161462                                LDA B   #,17            ;NEXT INTERRUPT SCAN LINE
        161462     306     017
     31 161464                                STA B   5,X
        161464     347     005
     32 161466                                LDA B   #,7             ;SET CURRENT LINE
```

CONTROL BLOCK SCROLL ROUTINES

```
   161466     306     007
33 161470                              STA B   4,X
   161470     347     004
34 161472                              LDA B   PLBASE+1        ;GET TEXT ADDRESS
   161472     326     135
35 161474                              STA B   1,X
   161474     347     001
36 161476                              LDA B   PLBASE
   161476     326     134
37 161500                              STA B   0,X
   161500     347     000
38 161502                              RTS                     ;RETURN TO CALLER
   161502     071
39                                     ;
```

TEXT POINTER UPDATE

```
      1                                     .SBTTL  TEXT POINTER UPDATE
      2                                     ;
      3 161503                        UPDTXT: TST     LOCK            ;DSPLAY LOCKED ?
        161503     175     000     124
      4 161506                                BNE     1$              ;IF NOT - SKIP
        161506     046     027
      5 161510                                LDX     PLPNTR          ;GET CURRENT COUNT
        161510     336     132
      6 161512                                LDA A   0,X
        161512     246     000
      7 161514                                LDX     TXTCBS          ;SAVE IN CONTROL BLOCK
        161514     336     146
      8 161516                                STA A   2,X
        161516     247     002
      9 161520                                LDX     CLPNTR          ;CHECK IF SAME LINE OF TEXT
        161520     336     126
     10 161522                                CPX     PLPNTR
        161522     234     132
     11 161524                                BEQ     1$              ;IF SO - FINISHED
        161524     047     011
     12 161526                                STX     PLPNTR          ;ELSE NEW LINE
        161526     337     132
     13 161530                                LDX     CLBASE          ;SET UP NEW BASE POINTER
        161530     336     130
     14 161532                                STX     PLBASE
        161532     337     134
     15 161534                                JSR     ROTUP           ;UPDATE CONTROL BLOCK POINTER
        161534     275     343     007
     16 161537                           1$:  RTS                     ;FINISHED
        161537     071
     17                                     ;
```

VERTICAL RETRACE INTERRUPT HANDLER

```
                                        .SBTTL  VERTICAL RETRACE INTERRUPT HANDLER
                                        ;
161540                          VERTIN: TST     UPDFLG          ;NEED AN UPDATE ?
  161540     175     000     106
161543                                  BEQ     RESTRT          ;IF NOT - GO RESTART DISPLAY
  161543     047     122
161545                                  CLR     UPDFLG          ;DOING UPDATE
  161545     177     000     106
161550                                  STS     PGMSTK          ;SAVE STACK POINTER
  161550     237     150
161552                                  LDX     #,DSPSTK+58.    ;SETUP DISPLAY STACK
  161552     316     001     372
161555                                  LDS     #,CLSTP         ;CONTROL STOP
  161555     216     001     256
161560                                  STS     4,X
  161560     257     004
161562                                  LDS     #,CLSRT         ;CONTROL START
  161562     216     001     250
161565                                  STS     2,X
  161565     257     002
161567                                  LDS     #,TXTSTP        ;TEXT STOP
  161567     216     001     242
161572                                  STS     0,X
  161572     257     000
161574                                  LDA B   #,9.            ;9 LINES WITH GRAPHICS
  161574     306     011
161576                                  LDA A   GRAPH           ;GRAPHICS ON ?
  161576     226     100
161600                                  BNE     1$              ;IF SO - SKIP
  161600     046     002
161602                                  ADD B   #,24.-9.        ;ELSE 24 LINES OF TEXT
  161602     313     017
161604                          1$:     STA B   LINES           ;LINE COUNT
  161604     327     145
                                        ;
```

```
21                                      ;
22 161606                               LDA A   TXTCBS          ;GET POINTER
   161606     226     146
23 161610                               LDA B   CURDSP
   161610     326     147
24 161612                       2$:     DEX                     ;UPDATE STACK POINTER
   161612     011
25 161613                               DEX
   161613     011
26 161614                               STA A   0,X             ;LOAD POINTER INTO CONTROL BLOCK
   161614     247     000
27 161616                               STA B   1,X
   161616     347     001
28 161620                               SUB B   #,6             ;UPDATE POINTER
   161620     300     006
29 161622                               BCC     3$              ;SKIP IF NOT PASSED BOTTOM
   161622     044     002
30 161624                               LDA B   #,CDTOP         ;ELSE LOAD TOP OF STACK AREA
   161624     306     234
31 161626                       3$:     DEC     LINES           ;FINISHED ?
   161626     172     000     145
```

VERTICAL RETRACE INTERRUPT HANDLER

```
     32 161631                                  BNE     2$              ;LOOP UNTIL FINISHED
        161631     046     357
     33                                         ;
     34                                         ;       NOW CHECK FOR GRAPHICS CONTROL
     35                                         ;
     36 161633                                  LDA B   #,37            ;START LINE FOR TEXT
        161633     306     037
     37 161635                                  LDA A   GRAPH           ;GRAPHICS ON ?
        161635     226     100
     38 161637                                  BEQ     4$              ;IF NOT - SKIP
        161637     047     020
     39 161641                                  DEX                     ;ELSE PLACE GRAPHIC CONTROLS ON S
TACK
        161641     011
     40 161642                                  DEX
        161642     011
     41 161643                                  LDS     #,GPSTP
        161643     216     001     272
     42 161646                                  STS     0,X
        161646     257     000
     43 161650                                  DEX
        161650     011
     44 161651                                  DEX
        161651     011
     45 161652                                  LDS     #,GPSRT
        161652     216     001     264
     46 161655                                  STS     0,X
        161655     257     000
     47 161657                                  LDA B   #,17            ;START LINE FOR GRAPHICS
        161657     306     017
     48 161661                          4$:     STA B   LINES           ;FOR DISPLAY START UP
        161661     327     145
     49 161663                                  STX     BCRSTK          ;SAVE BEGINNING OF DISPLAY STACK
        161663     337     156
     50 161665                                  LDS     PGMSTK          ;RESTORE STACK
        161665     236     150
```

```
    51                                      ;
    52                                      ;       NOW RESTART DISPLAY FOR CURRENT SCAN
    53                                      ;
    54 161667                       RESTRT: LDA A   GRAPH           ;GRAPHICS ON ?
       161667     226     100
    55 161671                               BEQ     2$              ;IF NOT - SKIP
       161671     047     015
    56 161673                               LDX     #,EVNSCN        ;ASSUME EVEN SCAN
       161673     316     060     000
    57 161676                               LDA A   DSPCTL          ;ON ODD SCAN ?
       161676     226     015
    58 161700                               BPL     1$              ;IF NOT - SKIP
       161700     052     003
    59 161702                               LDX     #,ODDSCN        ;ELSE ODD SCAN
       161702     316     130     000
    60 161705                          1$:  STX     GPSRT           ;SET SCAN ADDRESS
       161705     377     001     264
    61 161710                          2$:  LDX     BCRSTK          ;RESET CRSTK POINTER
       161710     336     156
    62 161712                               STX     CRSTK
       161712     337     160
    63 161714                               JSR     DSPUPD          ;SET UP DSP0-DSP2
```

VERTICAL RETRACE INTERRUPT HANDLER

```
        161714     275     342     354
     64 161717                              LDA A   LINES           ;GET FIRST INTERRUPT LINE
        161717     226     145
     65 161721                              STA A   ISL             ;STORE IN CONTROL
        161721     227     011
     66 161723                              LDA A   CARLB           ;CLEAR INTERRUPT FLAG
        161723     226     005
     67 161725                              LDA A   CHARPL          ;NMI STROBE
        161725     226     014
     68 161727                              JSR     TIMCLK          ;UPDATE TIME
        161727     275     342     104
     69 161732                              LDX     CURLOC          ;UPDATE CURSOR POSTION
        161732     336     152
     70 161734                              STX     CARHA
        161734     337     004
     71 161736                              RTI                     ;RETURN FROM INTERRUPT
        161736     073
     72                                     ;
```

BREAK KEY INTERRUPT HANDLER

```
                                        .SBTTL  BREAK KEY INTERRUPT HANDLER
                                        ;
161737                          BRKINT: BSR     BREAK           ;DO ROUTINE
  161737     215     003
161741                                  LDA A   NPRDFA          ;CLEAR INTERRUPT FLAG
  161741     226     060
161743                                  RTI                     ;RETURN FROM INTERRUPT
  161743     073
                                        ;
161744                          BREAK:  LDX     #,CTLINE+10     ;AREA FOR NOTE
  161744     316     040     010
161747                                  STX     FL.TO
  161747     337     117
161751                                  LDA A   #,7             ;7 CHARACTERS
  161751     206     007
161753                                  STA A   FL.CNT
  161753     227     123
161755                                  COM     ONLINE          ;CHECK STATUS
  161755     163     000     114
161760                                  BEQ     1$              ;ON LOCAL - SKIP
  161760     047     011
161762                                  LDA A   #,377           ;DATA INHIBITED FLAG
  161762     206     377
161764                                  STA A   DATINH
  161764     227     113
161766                                  LDX     #,MSGLIN        ;ON LINE MESSAGE
  161766     316     344     035
161771                                  BRA     2$
  161771     040     012
161773                          1$:     CLR     DATINH          ;DISABLE CLOCK ENABLE
  161773     177     000     113
161776                                  LDA A   #,5             ;INHIBIT READY FLAG TO ARB-11
  161776     206     005
162000                                  STA A   PCRA7
  162000     227     032
```

```
   162002     316     344     045
21 162005                          2$:     STX     FL.FRM          ;FROM ADDRESS
   162005     337     121
22                                                                 ;FALL THROUGH TO FILL
23                                         ;
24 162007                          FILL:   LDX     FL.FRM          ;GET A CHARACTER
   162007     336     121
25 162011                                  LDA A   0,X
   162011     246     000
26 162013                                  BEQ     1$              ;ON NULL - EXIT
   162013     047     017
27 162015                                  INX                     ;UPDATE ADDRESS
   162015     010
28 162016                                  STX     FL.FRM
   162016     337     121
29 162020                                  LDX     FL.TO           ;STORE A CHARACTER
   162020     336     117
30 162022                                  STA A   0,X
   162022     247     000
31 162024                                  INX
   162024     010
```

BREAK KEY INTERRUPT HANDLER

```
     32 162025                                  STX     FL.TO
        162025     337     117
     33 162027                                  DEC     FL.CNT          ;FINISHED ?
        162027     172     000     123
     34 162032                                  BNE     FILL            ;LOOP UNTIL ALL TRANSFERRED
        162032     046     353
     35 162034                          1$:     RTS                     ;FINISHED
        162034     071
     36                                         ;
     37 162035     117     116     040  MSGLIN: .ASCIZ  /ON LINE/
        162040     114     111     116
        162043     105     000
     38 162045     040     114     117  MSGLOC: .ASCIZ  / LOCAL /
        162050     103     101     114
        162053     040     000
     39                                         ;
```

```
                                        .SBTTL  ARB-11 I/O INTERRUPT HANDLER
                                        ;
162055                          TERMDI: LDA A   #,5             ;INHIBIT READY FLAG
  162055     206     005
162057                                  STA A   PCRA7
  162057     227     032
162061                                  LDA A   TDF11           ;GET CHARATCER
  162061     226     030
162063                                  JSR     CTLP            ;PROCESS CHARACTER
  162063     275     345     266
162066                                  INC     T.CHAR          ;INCREMENT COUNTER
  162066     174     000     111
162071                                  CLR A
  162071     117
162072                                  TST     ONLINE          ;ON LINE ?
  162072     175     000     114
162075                                  BEQ     2$              ;IF NOT - SKIP
  162075     047     017
162077                                  LDA A   NALL            ;THIS NUMBER ALLOWED
  162077     226     112
162101                                  CMP A   T.CHAR          ;AT LIMIT ?
  162101     221     111
162103                                  BLE     1$              ;IF SO - SKIP
  162103     057     007
162105                                  LDA A   #,55            ;ELSE ALLOW MORE CHARACTERS
  162105     206     055
162107                                  STA A   PCRA7
  162107     227     032
162111                                  LDA A   TDF11           ;SET READY FLAG
  162111     226     030
162113                                  RTI                     ;FINISHED
  162113     073
162114                          1$:     LDA A   #,377
  162114     206     377
162116                          2$:     STA A   DATINH          ;SET FLAG FOR CLOCK
```

```
    20 162120                               RTI                     ;FINISHED
       162120     073
    21                                      ;
```

CHARACTER INPUT INTERRUPT HANDLERS

```
      1                                        .SBTTL  CHARACTER INPUT INTERRUPT HANDLERS
      2                                        ;
      3 162121                          REMOTE: LDA A   #,53            ;GET TO DIRECTON REGISTERS
        162121     206     053
      4 162123                                  STA A   PCRB9
        162123     227     043
      5 162125                                  CLR B                   ;INPUT DIRECTION
        162125     137
      6 162126                                  STA B   HB
        162126     327     041
      7 162130                                  LDA A   #,57            ;GET BACK TO DATA REGISTER
        162130     206     057
      8 162132                                  STA A   PCRB9
        162132     227     043
      9 162134                                  STA B   HA              ;ENABLE READ DATA
        162134     327     040
     10 162136                                  LDA A   HB              ;READ DATA AND CLEAR INTERRUPT FL
AG
        162136     226     041
     11 162140                                  LDA B   #,40            ;DISABLE READ
        162140     306     040
     12 162142                                  STA B   HA
        162142     327     040
     13 162144                                  BRA     COMINT          ;TREAT AS A KEYBOARD ENTRY
        162144     040     010
     14 162146                          PHRDR:  LDA A   PHOTO           ;GET CHARACTER
        162146     226     044
     15 162150                                  BRA     COMINT
        162150     040     004
     16 162152                          RPINT:  LDA A   NPRDFB          ;CLEAR REPEAT INTERRUPT FLAG
        162152     226     061
     17 162154                          KBINT:  LDA A   KASCII          ;GET CHARACTER FROM KEYBOARD
        162154     226     034
     18 162156                          COMINT: LDA B   ONLINE          ;ON LINE ?
        162156     326     114
     19 162160                                  BNE     1$              ;IF SO - JUST SEND TO ARB-11
```

```
        162160     046     004
     20 162162                              JSR     CTLP            ;ELSE PROCESS CHARACTER
        162162     275     345     266
     21 162165                              RTI                     ;FINISHED
        162165     073
     22 162166                         1$:  ORA A   OBIS            ;OR TO SET BIT
        162166     232     172
     23 162170                              AND A   OBIC            ;AND TO CLEAR BIT
        162170     224     173
     24 162172                              STA A   TDT11           ;SEND CHARACTER
        162172     227     031
     25 162174                              RTI                     ;FINISHED
        162174     073
     26                                     ;
```

SCROLL UP INTERRUPT HANDLER

```
                                        .SBTTL  SCROLL UP INTERRUPT HANDLER
                                        ;
162175                          SCRLUP: LDA A   NPRADB          ;CLEAR INTERRUPT FLAG
  162175     226     051
162177                                  LDA A   KBUTN           ;READ SWITCH VALUES
  162177     226     035
162201                                  BIT A   #,40            ;IS <TAPE DEPRESSED
  162201     205     040
162203                                  BNE     1$              ;IF SO - BOTH DEPRESSED
  162203     046     025
162205                                  LDX     PLPNTR          ;PLPNTR = CLPNTR
  162205     336     132
162207                                  CPX     CLPNTR
  162207     234     126
162211                                  BEQ     2$              ;IF SO - JUST LOCK DISPLAY
  162211     047     021
162213                                  LDA A   #,377           ;ELSE UNLOCK DISPLAY
  162213     206     377
162215                                  STA A   LOCK
  162215     227     124
162217                                  JSR     UPSCRL          ;SET UP SCROLLING
  162217     275     344     243
162222                                  JSR     RLPUP           ;ROLL POINTERS
  162222     275     344     265
162225                                  JSR     ROTUP           ;ROTATE DISPLAY
  162225     275     343     007
162230                                  BRA     3$
  162230     040     005
162232                          1$:     LDA A   NPRADA          ;CLEAR INTERRUPT FLAG ON TAPE>
  162232     226     050
162234                          2$:     JSR     LCKDSP          ;LOCK DISPLAY
  162234     275     345     064
162237                          3$:     INC     UPDFLG          ;NEED AN UPDATE
  162237     174     000     106
162242                                  RTI                     ;RETURN FROM INTERRUPT
```

```
     20                                        ;
```

UP SCROLL ROUTINES

```
      1                                        .SBTTL  UP SCROLL ROUTINES
      2                                        ;
      3 162243                         UPSCRL: LDA A   DRCTN           ;CHECK DIRECTION
        162243     226     125
      4 162245                                 BPL     2$             ;IF + WE'RE DONE
        162245     052     015
      5 162247                                 CLR     DRCTN          ;MAKE +
        162247     177     000     125
      6 162252                                 LDA A   #,CDTOP/6      ;UPDATE ALL TXTCBS LINES
        162252     206     032
      7 162254                         1$:     PSH A                  ;SAVE COUNT
        162254     066
      8 162255                                 JSR     RLPUP          ;ROLL POINTERS ONE LINE
        162255     275     344     265
      9 162260                                 PUL A
        162260     062
     10 162261                                 DEC A                  ;ARE WE DONE
        162261     112
     11 162262                                 BGT     1$             ;LOOP UNTIL DONE
        162262     056     370
     12 162264                         2$:     RTS                    ;FINISHED
        162264     071
     13                                        ;
     14 162265                         RLPUP:  LDX     PLPNTR         ;GET CHARACTER COUNT IN LINE
        162265     336     132
     15 162267                                 LDA A   0,X
        162267     246     000
     16 162271                                 CPX     #,CLPTOP       ;ARE WE AT THE TOP ?
        162271     214     017     331
     17 162274                                 BNE     1$             ;IF NOT - SKIP
        162274     046     003
     18 162276                                 LDX     #,CLPBOT-1     ;ELSE GET BOTTOM ADDRESS
        162276     316     014     077
     19 162301                         1$:     INX                    ;UPDATE POINTER
        162301     010
```

```
   162302     337     132
21 162304                          CLR B                   ;CURRENT COUNT = <B,A>
   162304     137
22 162305                          ADD A   PLBASE+1        ;COMPUTE NEW BASE
   162305     233     135
23 162307                          ADC B   PLBASE
   162307     331     134
24 162311                          STA A   PLBASE+1
   162311     227     135
25 162313                          STA B   PLBASE
   162313     327     134
26 162315                          SUB A   #,CL.TOP&377    ;IS PLBASE PAST CLTOP ?
   162315     200     260
27 162317                          SBC B   #,CL.TOP&177400/400
   162317     302     057
28 162321                          BCS     2$              ;IF NOT - SKIP
   162321     045     005
29 162323                          LDX     #,CLBOT         ;ELSE RESET PLBASE
   162323     316     040     120
30 162326                          STX     PLBASE
   162326     337     134
```

UP SCROLL ROUTINES

```
     31 162330                          2$:     RTS                     ;FINISHED
        162330     071
     32                                         ;
```

SCROLL DOWN INTERRUPT HANDLER

```
      1                                      .SBTTL  SCROLL DOWN INTERRUPT HANDLER
      2                                      ;
      3 162331                        SCRLDN: LDA A   NPRADA          ;CLEAR INTERRUPT FLAG
        162331     226     050
      4 162333                                LDA A   KBUTN           ;READ SWITCHES
        162333     226     035
      5 162335                                BIT A   #,20            ;TAPE> DEPRESSED ?
        162335     205     020
      6 162337                                BNE     1$              ;IF SO - LOCK DISPLAY
        162337     046     015
      7 162341                                LDA A   #,377           ;UNLOCK DISPLAY
        162341     206     377
      8 162343                                STA A   LOCK
        162343     227     124
      9 162345                                BSR     DNSCRL          ;SETUP DOWN SCROLLONG
        162345     215     020
     10 162347                                BSR     RLPDN           ;ROLL POOINTERS DOWN ONE LINE
        162347     215     040
     11 162351                                JSR     ROTDWN          ;ROTATE DISPLAY ONE LINE
        162351     275     343     031
     12 162354                                BRA     2$
        162354     040     005
     13 162356                           1$:  LDA A   NPRADB          ;CLEAR INTERRUPT FLAG ON <TAPE
        162356     226     051
     14 162360                                JSR     LCKDSP          ;LOCK DISPLAY
        162360     275     345     064
     15 162363                           2$:  INC     UPDFLG          ;NEED AN UPDATE
        162363     174     000     106
     16 162366                                RTI                     ;RETURN FROM INTERRUPT
        162366     073
     17                                       ;
```

DOWN SCROLL ROUTINES

```
      1                                     .SBTTL  DOWN SCROLL ROUTINES
      2                                     ;
      3 162367                      DNSCRL: LDA A   DRCTN           ;CHECK DIRECTION
        162367     226     125
      4 162371                              BMI     2$              ;IF NEGATIVE - THEN WE'RE DONE
        162371     053     015
      5 162373                              LDA A   #,377           ;SET DIRECTION DOWN
        162373     206     377
      6 162375                              STA A   DRCTN
        162375     227     125
      7 162377                              LDA A   #,CDTOP/6       ;UPDATE ALL TXTCBS LINES
        162377     206     032
      8 162401                      1$:     PSH A                   ;SAVE COUNTER
        162401     066
      9 162402                              BSR     RLPDN           ;ROLL POINTER ONE LINE
        162402     215     005
     10 162404                              PUL A
        162404     062
     11 162405                              DEC A                   ;FINISHED ?
        162405     112
     12 162406                              BGT     1$              ;LOOP UNTIL FINISHED
        162406     056     371
     13 162410                      2$:     RTS                     ;FINISHED
        162410     071
     14                                     ;
     15 162411                      RLPDN:  LDX     PLPNTR          ;IS POINTER AT BOTTOM ?
        162411     336     132
     16 162413                              CPX     #,CLPBOT
        162413     214     014     100
     17 162416                              BNE     1$              ;IF NOT - SKIP
        162416     046     003
     18 162420                              LDX     #,CLPTOP+1      ;ELSE GET TOP
        162420     316     017     332
     19 162423                      1$:     DEX                     ;UPDATE POINTER
        162423     011
```

```
   162424     337     132
21 162426                          LDA A   #,CLBOT&377     ;IS PLBASE > CLBOT
   162426     206     120
22 162430                          LDA B   #,CLBOT&177400/400
   162430     306     040
23 162432                          SUB A   PLBASE+1
   162432     220     135
24 162434                          SBC B   PLBASE
   162434     322     134
25 162436                          BCS     2$              ;IF SO - SKIP
   162436     045     005
26 162440                          LDX     OVBASE          ;ELSE RESET PLBASE
   162440     336     136
27 162442                          STX     PLBASE
   162442     337     134
28 162444                          RTS                     ;FINISHED
   162444     071
29 162445                    2$:   LDA A   PLBASE+1        ;COMPUTE PREVIOUS BASE ADDRESS
   162445     226     135
30 162447                          LDA B   PLBASE
   162447     326     134
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35 Page 32-1

DOWN SCROLL ROUTINES

```
     31 162451                              LDX     PLPNTR
        162451     336     132
     32 162453                              SUB A   0,X
        162453     240     000
     33 162455                              SBC B   #,0
        162455     302     000
     34 162457                              STA A   PLBASE+1
        162457     227     135
     35 162461                              STA B   PLBASE
        162461     327     134
     36 162463                              RTS                     ;FINISHED
        162463     071
     37                                     ;
```

LOCK DISPLAY ROUTINE

```
      1                                        .SBTTL  LOCK DISPLAY ROUTINE
      2                                        ;
      3 162464                          LCKDSP: LDA A   LOCK            ;IS DISPLAY LOCKED ?
        162464     226     124
      4 162466                                  BEQ     2$              ;IF SO - FINISHED
        162466     047     044
      5 162470                                  CLR     LOCK            ;SET FOR LOCKED DISPLAY
        162470     177     000     124
      6 162473                                  LDX     CLPNTR          ;SET TO CURRENT POINTERS
        162473     336     126
      7 162475                                  STX     PLPNTR
        162475     337     132
      8 162477                                  LDX     CLBASE
        162477     336     130
      9 162501                                  STX     PLBASE
        162501     337     134
     10 162503                                  CLR     DRCTN           ;SET FOR UP SCROLLING
        162503     177     000     125
     11 162506                                  JSR     DNSCRL          ;DOWN SCROLL TO 25TH PREVIOUS LIN
E
        162506     275     344     367
     12 162511                                  CLR     DRCTN           ;SET FOR UP SCROLLING
        162511     177     000     125
     13 162514                                  LDA A   #,CDTOP/6       ;UPDATE ALL TXTCBS LINES
        162514     206     032
     14 162516                          1$:     PSH A                   ;SAVE COUNTER
        162516     066
     15 162517                                  JSR     ROTUP           ;ROTATE DISPLAY 1 LINE
        162517     275     343     007
     16 162522                                  JSR     RLPUP           ;ROLL POINTERS 1 LINE
        162522     275     344     265
     17 162525                                  PUL A
        162525     062
     18 162526                                  DEC A                   ;CHECK COUNT
        162526     112
     19 162527                                  BGT     1$              ;LOOP UNTIL FINISHED
```

```
        162527     056     365
     20 162531                              JSR     ROTUP           ;UPDATE TO LAST LINE
        162531     275     343     007
     21 162534                          2$:     RTS                     ;FINISHED
        162534     071
     22                                         ;
```

PAPER ADVANCE KEY INTERRUPT HANDLER

```
                                        .SBTTL  PAPER ADVANCE KEY INTERRUPT HANDLER
                                        ;
162535                          PPRKEY: BSR     PAPER           ;GO CHECK KEY
  162535     215     001
162537                                  RTI                     ;RETURN FROM INTERRUPT
  162537     073
                                        ;
162540                          PAPER:  LDA A   PCRB12          ;IS INTERRUPT FLAG SET ?
  162540     226     057
162542                                  BPL     3$              ;IF NOT - JUST LEAVE
  162542     052     040
162544                                  LDA A   NPRDTB          ;CLEAR INTERRUPT FLAG
  162544     226     055
162546                                  COM     RMTON           ;FLIP/FLOP REMOTE FLAG
  162546     163     000     101
162551                                  BEQ     1$              ;IF OFF - BRANCH
  162551     047     007
162553                                  LDX     #,MSGRON        ;POINT TO MESSAGE
  162553     316     345     224
162556                                  LDA A   #,57            ;ENABLE REMOTE INTERRUPTS
  162556     206     057
162560                                  BRA     2$
  162560     040     005
162562                          1$:     LDX     #,MSGROF        ;POINT TO MESSAGE
  162562     316     345     245
162565                                  LDA A   #,4             ;DISABLE INTERRUPTS
  162565     206     004
162567                          2$:     STA A   PCRB9
  162567     227     043
162571                                  LDA A   HB              ;CLEAR PENDNG INTERRUPT
  162571     226     041
162573                                  CLR     HA              ;STROBE UART DAV F/F
  162573     177     000     040
162576                                  LDA A   #,40            ;SO NEXT CHARACTER WLL BE SEEN
  162576     206     040
```

```
   162600     227     040
21 162602                          BSR     RMTMOV          ;MOVE MESSAGE
   162602     215     001
22 162604                    3$:   RTS                     ;FINISHED
   162604     071
23                                 ;
24 162605                    RMTMOV: STX     FL.FRM          ;SAVE FROM POINTER
   162605     337     121
25 162607                          LDX     #,CTLINE+42     ;REMOTE AREA
   162607     316     040     042
26 162612                          STX     FL.TO           ;SAVE TO AREA
   162612     337     117
27 162614                          LDA A   #,20            ;CHARACTER COUNT
   162614     206     020
28 162616                          STA A   FL.CNT          ;SAVE COUNT
   162616     227     123
29 162620                          JSR     FILL            ;MOVE MESSAGE
   162620     275     344     007
30 162623                          RTS                     ;FINISHED
   162623     071
31                                 ;
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35 Page 34-1

PAPER ADVANCE KEY INTERRUPT HANDLER

```
     32 162624     040     122     105  MSGRON: .ASCIZ  / REMOTE  ON     /
        162627     115     117     124
        162632     105     040     040
        162635     117     116     040
        162640     040     040     040
        162643     040     000
     33 162645     040     122     105  MSGROF: .ASCIZ  / REMOTE  OFF    /
        162650     115     117     124
        162653     105     040     040
        162656     117     106     106
        162661     040     040     040
        162664     040     000
     34                                      ;
```

CONTROL PROCESSING HANDLER

```
                                        .SBTTL  CONTROL PROCESSING HANDLER
                                        ;
                                        ;       ALL INPUT FROM THE ARB-11, KEYBOARD,
                                        ;       REPEAT KEY, AND PHOTO READER PASS
                                        ;       THROUGH THIS HANDLER FOR DISPATCHING TO
                                        ;       THE REQUIRED HANDLERS
                                        ;
162666                          CTLP:   STA A   CHARA           ;SAVE THE CHARACTER
  162666     227     103
162670                                  CLR     CHRFLG          ;ENABLE PRINTING
  162670     177     000     105
                                        ;
                                        ;       LOCAL MONITOR CHECK
                                        ;
162673                                  LDA A   ONLINE          ;ON LINE ?
  162673     226     114
162675                                  BNE     1$              ;IF SO - SKIP
  162675     046     004
162677                                  LDX     LJSR            ;ENTRY POINT TO LOCAL MONITOR
  162677     336     201
162701                                  JSR     0,X             ;GO TO MONITOR
  162701     255     000
                                        ;
                                        ;       ESCAPE SEQUENCE CONTROL MONITOR
                                        ;
162703                          1$:     LDA A   ESCFLG          ;ESCAPES ENABLED ?
  162703     226     116
162705                                  BEQ     2$              ;IF NOT - SKIP
  162705     047     004
162707                                  LDX     EJSR            ;ENTRY POINT TO ESCAPE MONITOR
  162707     336     175
162711                                  JSR     0,X             ;GO TO IT
  162711     255     000
                                        ;
                                        ;       LIST GRAPHICS PROCESSING
```

```
 27 162713                          2$:     LDA A   LSTFLG          ;GRAPHICS ENABLED ?
    162713     226     115
 28 162715                                  BEQ     3$              ;IF NOT - SKIP
    162715     047     005
 29 162717                                  LDA A   CHARA           ;GET CHARACTER
    162717     226     103
 30 162721                                  JSR     GRPLOT          ;DO GRAPHICS
    162721     275     300     003
 31                                         ;
 32                                         ;       PRINTING CHARACTER PROCESSING
 33                                         ;
 34 162724                          3$:     LDA A   CHRFLG          ;PRINTING ALLOWED ?
    162724     226     105
 35 162726                                  BNE     4$              ;IF NOT SKIP
    162726     046     011
 36 162730                                  LDA A   CHARA           ;GET CHARACTER
    162730     226     103
 37 162732                                  ORA A   IBIS            ;SET BITS
    162732     232     170
 38 162734                                  AND A   IBIC            ;CLEAR BITS
    162734     224     171
```

CONTROL PROCESSING HANDLER

```
     39 162736                                  JSR     .PRINT          ;PRINT IT
        162736     275     345     342
     40 162741                          4$:     RTS                     ;FINISHED
        162741     071
     41                                         ;
```

.PRINT ROUTINE

```
      1                                       .SBTTL  .PRINT ROUTINE
      2                                       ;
      3                                       ;       THIS ROUTINE CALLS THE VARIOUS ROUTINES
      4                                       ;       THAT ACTUALLY PRINT THE CHARACTER.
      5                                       ;       THE ROUTINES USED ARE:
      6                                       ;               RMTCHK - FOR THE REMOTE PORT
      7                                       ;               CHAR   - FOR THE CHARACTER ORIENTED SCREE
N
      8                                       ;               LINE   - FOR THE LINE ORIENTED SCREEN
      9                                       ;
     10                                       ;       THESE ROUTINES MUST NOT BE CALLED INDIVIDUALLY, B
UT
     11                                       ;       ONLY THROUGH A CALL TO .PRINT
     12                                       ;
     13 162742                          .PRINT: STA A   CHARB           ;SAVE CHARACTER TO BE PROCESSED
        162742     227     104
     14                                       ;
     15                                       ;       ALL CHARACTERS PROCESSED BY REMOTE
     16                                       ;
     17 162744                                  JSR     RMTCHK
        162744     275     346     006
     18                                       ;
     19                                       ;       CHAR IS INVOKED IF IN CHARACTER MODE
     20                                       ;
     21 162747                                  LDA A   MODE            ;CHECK MODE
        162747     226     102
     22 162751                                  BNE     1$              ;IF NOT CHARACTER - SKIP
        162751     046     004
     23 162753                                  JSR     CHAR            ;USE CHARACTER HANDLER
        162753     275     346     064
     24 162756                                  RTS                     ;FINISHED
        162756     071
     25                                       ;
     26                                       ;       ELSE USE LINE HANDLER
     27                                       ;
     28 162757                          1$:     JSR     LINE
```

```
    29 162762                                RTS                    ;FINISHED
       162762     071
    30                                       ;
```

REMOTE OUTPUT HANDLER

```
      1                                        .SBTTL  REMOTE OUTPUT HANDLER
      2                                        ;
      3 162763                         RMTWT:  LDA B   #,40            ;NO FLASH CHARACTER
        162763     306     040
      4 162765                                 LDA A   T.60TH          ;GET 60'THS COUNTER
        162765     226     107
      5 162767                                 AND A   #,20            ;CHECK BIT FOR FLASH RATE
        162767     204     020
      6 162771                                 BNE     1$
        162771     046     002
      7 162773                                 LDA B   #,377           ;FLASH CHARACTER
        162773     306     377
      8 162775                         1$:     CMP B   RMTFLS          ;SAME AS CURRENT FLASH CHARACTER
?
        162775     321     174
      9 162777                                 BEQ     RMTCHK          ;IF SO - SKIP
        162777     047     005
     10 163001                                 STA B   RMTFLS
        163001     327     174
     11 163003                                 STA B   CTLINE+56       ;PUT IN POSITION
        163003     367     040     056
     12 163006                         RMTCHK: LDA B   RMTON           ;REMOTE ENABLED ?
        163006     326     101
     13 163010                                 BEQ     2$              ;IF NOT - SKIP OUT
        163010     047     036
     14 163012                                 JSR     PAPER           ;CHECK FOR PAPER KEY
        163012     275     345     140
     15 163015                                 JSR     TMCK            ;UPDATE TIME
        163015     275     342     072
     16 163020                                 LDA A   HA              ;IS REMOTE READY ?
        163020     226     040
     17 163022                                 BMI     RMTWT           ;IF NOT - ENTER WAIT LOOP
        163022     053     337
     18 163024                                 AND A   #,20            ;IS UART READY ?
        163024     204     020
     19 163026                                 BEQ     RMTWT           ;IF NOT - ENTER WAIT LOOP
```

```
        163026     047     333
     20 163030                              LDA A   #,53            ;GET TO DIRECTION REGISTERS
        163030     206     053
     21 163032                              STA A   PCRB9
        163032     227     043
     22 163034                              LDA A   #,377           ;SET FOR OUTPUT
        163034     206     377
     23 163036                              STA A   HB
        163036     227     041
     24 163040                              LDA A   #,57            ;SET FOR DATA REGISTERS
        163040     206     057
     25 163042                              STA A   PCRB9
        163042     227     043
     26 163044                              LDA A   CHARB           ;GET CHARACTER TO SEND
        163044     226     104
     27 163046                              STA A   HB
        163046     227     041
     28 163050                      2$:     LDA A   #,40            ;ELIMINATE FLASH WHEN EXITING
        163050     206     040
     29 163052                              CMP A   RMTFLS
        163052     221     174
     30 163054                              BEQ     3$              ;IF A SPACE - SKIP
```

REMOTE OUTPUT HANDLER

```
        163054     047     005
     31 163056                                STA A   CTLINE+56       ;ELSE MAKE IT A SPACE
        163056     267     040     056
     32 163061                                STA A   RMTFLS
        163061     227     174
     33 163063                          3$:   RTS                     ;FINISHED
        163063     071
     34                                       ;
```

```
      1                                      .SBTTL  CHARACTER MODE HANDLER
      2                                      ;
      3 163064                         CHAR: LDA A   CHARB           ;GET CHARACTER
        163064     226     104
      4 163066                               CMP A   #,15            ;A CARRIAGE RETURN
        163066     201     015
      5 163070                               BNE     1$              ;IF NOT - SKIP
        163070     046     011
      6 163072                               LDA A   #,1             ;POSTION OF 1
        163072     206     001
      7 163074                               STA A   CURCNT          ;AS CURRENT COUNT
        163074     227     154
      8 163076                               LDX     CLBASE          ;BEGINNING OF CURRENT LINE
        163076     336     130
      9 163100                               STX     CURLOC          ;MOVE CURSOR
        163100     337     152
     10 163102                               RTS                     ;FINISHED
        163102     071
     11                                      ;
     12 163103                         1$:   CMP A   #,12            ;A LINE FEED ?
        163103     201     012
     13 163105                               BNE     2$              ;IF NOT - SKIP
        163105     046     013
     14 163107                               JSR     UPDLIN          ;UPDATE LINE
        163107     275     346     317
     15 163112                               LDA A   CURCNT          ;GET CURRENT COUNT
        163112     226     154
     16 163114                               LDX     CLPNTR          ;GET LINE LENGTH ADDRESS
        163114     336     126
     17 163116                               STA A   0,X             ;SET LINE LENGTH
        163116     247     000
     18 163120                               BRA     7$
        163120     040     120
     19 163122                         2$:   CMP A   #,7             ;BELL ?
        163122     201     007
```

```
        163124     046     004
     21 163126                              STA A   KBUTN           ;RING BELL !
        163126     227     035
     22 163130                              BRA     5$
        163130     040     023
     23 163132                          3$: CMP A   #,10            ;BACK SPACE ?
        163132     201     010
     24 163134                              BNE     5$              ;IF NOT - SKIP
        163134     046     017
     25 163136                              DEC     CURCNT          ;DELETE A CHARACTER
        163136     172     000     154
     26 163141                              BNE     4$              ;IF NOT 0 - SKIP
        163141     046     004
     27 163143                              INC     CURCNT          ;NEED AT LEAST 1 CHARACTER IN LIN
E
        163143     174     000     154
     28 163146                              RTS                     ;NO FURTHER UPDATE
        163146     071
     29 163147                          4$: LDX     CURLOC          ;MOV CURSOR
        163147     336     152
     30 163151                              DEX
        163151     011
```

CHARACTER MODE HANDLER

```
     31 163152                              STX     CURLOC
        163152     337     152
     32 163154                              RTS                     ;FINISHED
        163154     071
     33 163155                         5$:  CMP A   DCTL            ;PRINTING CONTROLS ?
        163155     221     155
     34 163157                              BCS     9$              ;IF NOT - FINISHED
        163157     045     077
     35 163161                              LDX     CURLOC          ;GET CURSOR POSITION
        163161     336     152
     36 163163                              STA A   0,X             ;PLACE CHARACTER
        163163     247     000
     37 163165                              INX                     ;UPDATE CURSOR POSITION
        163165     010
     38 163166                              STX     CURLOC
        163166     337     152
     39 163170                              INC     CURCNT          ;UPDATE COUNTER
        163170     174     000     154
     40 163173                              LDA A   CURCNT          ;CURCNT > @CLPNTR ?
        163173     226     154
     41 163175                              LDX     CLPNTR
        163175     336     126
     42 163177                              CMP A   0,X
        163177     241     000
     43 163201                              BLT     6$              ;IF NOT - SKIP
        163201     055     012
     44 163203                              STA A   0,X             ;UPDATE @CLPNTR
        163203     247     000
     45 163205                              LDX     CURLOC          ;AND CLEAR CURSOR POSITION
        163205     336     152
     46 163207                              LDA A   #,40
        163207     206     040
     47 163211                              STA A   0,X
        163211     247     000
     48 163213                              LDX     CLPNTR          ;GET POINTER AGAIN
```

```
 49 163215                          6$:     LDA A   #,81.           ;ARE WE PAST 80. CHARACTERS ?
    163215     206     121
 50 163217                                  CMP A   0,X
    163217     241     000
 51 163221                                  BGT     8$              ;IF NOT - FINISHED
    163221     056     032
 52 163223                                  STA A   KBUTN           ;ELSE RING BELL !
    163223     227     035
 53 163225                                  STA A   0,X             ;AND SET COUNT TO 81.
    163225     247     000
 54 163227                                  JSR     UPDLIN          ;UPDATE LINE
    163227     275     346     317
 55 163232                                  LDX     CLPNTR          ;GET POINTER
    163232     336     126
 56 163234                                  LDA A   #,11.           ;NEW LINE OF 11. CHARACTERS
    163234     206     013
 57 163236                                  STA A   0,X
    163236     247     000
 58 163240                                  STA A   CURCNT
    163240     227     154
 59 163242                          7$:     BSR     CLEAR           ;CLEAR LINE
```

CHARACTER MODE HANDLER

```
        163242     215     034
     60 163244                              BSR     POSCUR          ;REPOSITION CURSOR
        163244     215     013
     61 163246                              JSR     UPDTXT          ;UPDATE TEXT
        163246     275     343     103
     62 163251                              INC     UPDFLG          ;DO AN UPDATE
        163251     174     000     106
     63 163254                              RTS                     ;FINISHED
        163254     071
     64 163255                         8$:  JSR     UPDTXT          ;UPDATE TEXT
        163255     275     343     103
     65 163260                         9$:  RTS                     ;WITH NO SCREEN UPDATE
        163260     071
     66                                     ;
```

CHARACTER MODE HANDLER

```
      1                                     ;
      2 163261                      POSCUR: LDX     CLPNTR          ;GET ADDRESS OF CURRENT LINE LENG
TH
        163261     336     126
      3 163263                              LDA A   0,X             ;GET LENGTH
        163263     246     000
      4 163265                              DEC A                   ;DELETE CURSOR POSITION
        163265     112
      5 163266                              CLR B
        163266     137
      6 163267                              ADD A   CLBASE+1        ;COMPUTE NEW POSITION
        163267     233     131
      7 163271                              ADC B   CLBASE
        163271     331     130
      8 163273                              STA A   CURLOC+1        ;SAVE NEW POSITION
        163273     227     153
      9 163275                              STA B   CURLOC
        163275     327     152
     10 163277                              RTS                     ;FINISHED
        163277     071
     11                                     ;
     12 163300                      CLEAR:  LDX     CLPNTR          ;GET CHARACTER COUNT AT CURRENT L
INE
        163300     336     126
     13 163302                              LDA A   0,X
        163302     246     000
     14 163304                              LDX     CLBASE          ;GET FIRST LOCATION
        163304     336     130
     15 163306                              LDA B   #,40            ;SPACE
        163306     306     040
     16 163310                      1$:     STA B   0,X             ;CLEAR CHARACTER
        163310     347     000
     17 163312                              INX                     ;UPDATE ADDRESS
        163312     010
     18 163313                              DEC A                   ;FINISHED ?
        163313     112
```

```
        163314     046     372
     20 163316                              RTS                     ;FINISHED
        163316     071
     21                                     ;
```

```
      1                                     ;
      2 163317                        UPDLIN: LDX     CLPNTR          ;GET CHARACTER COUNT
        163317     336     126
      3 163321                                LDA A   0,X
        163321     246     000
      4 163323                                CMP A   #,1             ;1 ?
        163323     201     001
      5 163325                                BEQ     1$              ;IF SO - SKIP
        163325     047     003
      6 163327                                DEC     0,X             ;DELETE CURSOR SPACE
        163327     152     000
      7 163331                                DEC A
        163331     112
      8 163332                           1$:  CPX     #,CLPTOP        ;AT TOP ?
        163332     214     017     331
      9 163335                                BNE     2$              ;IF NOT - SKIP
        163335     046     003
     10 163337                                LDX     #,CLPBOT-1      ;ELSE RESET POINTER
        163337     316     014     077
     11 163342                           2$:  INX                     ;UPDATE POINTER
        163342     010
     12 163343                                STX     CLPNTR          ;SAVE POINTER
        163343     337     126
     13                                     ;
     14                                     ;       CHECK STORAGE LIMIT
     15                                     ;
     16 163345                                LDX     CLBASE          ;GET PRESENT BASE
        163345     336     130
     17 163347                                CLR B                   ;COMPUTE NEW BASE
        163347     137
     18 163350                                ADD A   CLBASE+1
        163350     233     131
     19 163352                                ADC B   CLBASE
        163352     331     130
     20 163354                                STA A   CLBASE+1
```

```
     21 163356                                STA B   CLBASE
        163356     327     130
     22 163360                                SUB A   #,CL.TOP&377    ;ARE WE PAST TOP OF CHARACTER ARE
A ?
        163360     200     260
     23 163362                                SBC B   #,CL.TOP&177400/400
        163362     302     057
     24 163364                                BCS     4$              ;IF NOT - SKIP
        163364     045     007
     25 163366                                STX     OVBASE          ;ELSE NOTE LAST BASE BEFORE OVER
TOP
        163366     337     136
     26 163370                                LDX     #,CLBOT         ;AND RESET CLBASE
        163370     316     040     120
     27 163373                                STX     CLBASE
        163373     337     130
     28 163375                          4$:   RTS                     ;FINISHED
        163375     071
     29                                       ;
```

LINE MODE HANDLER

```
      1                                      .SBTTL  LINE MODE HANDLER
      2                                      ;
      3 163376                         LINE: LDA A   CHARB           ;GET CHARACTER
        163376     226     104
      4 163400                               CMP A   #,40            ;A CONTROL ?
        163400     201     040
      5 163402                               BCC     4$              ;IF NOT - SKIP
        163402     044     054
      6 163404                               CMP A   #,15            ;A <CR> ?
        163404     201     015
      7 163406                               BEQ     CRTRN           ;IF SO - GO
        163406     047     057
      8 163410                               CMP A   #,12            ;A <LF> ?
        163410     201     012
      9 163412                               BEQ     ADD80           ;IF SO - GO
        163412     047     065
     10 163414                               CMP A   #,13            ;A VERTICAL TAB ?
        163414     201     013
     11 163416                               BEQ     SUB80           ;IF SO - GO
        163416     047     066
     12 163420                               CMP A   #,11            ;A HORIZONTAL TAB /
        163420     201     011
     13 163422                               BEQ     ADD1            ;IF SO - GO
        163422     047     070
     14 163424                               CMP A   #,10            ;A BACKSPACE ?
        163424     201     010
     15 163426                               BEQ     SUB1            ;IF SO - GO
        163426     047     071
     16                                      ;
     17                                      ;       NOW CHECK SPECIALS
     18                                      ;
     19 163430                               CMP A   #,36            ;<RS> HOME CURSOR ?
        163430     201     036
     20 163432                               BNE     1$              ;IF NOT - SKIP
        163432     046     003
```

```
   163434     176     347     370
22 163437                          1$:     CMP A   #,32            ;<SUB> CLEAR SCREEN
   163437     201     032
23 163441                                  BNE     2$              ;IF NOT - SKIP
   163441     046     003
24 163443                                  JMP     CLRSCN          ;ELSE CLEAR THE SCREEN
   163443     176     347     317
25 163446                          2$:     CMP A   #,7             ;BELL /
   163446     201     007
26 163450                                  BNE     3$              ;IF NOT - SKIP
   163450     046     002
27 163452                                  STA A   KBUTN           ;SOUND BELL
   163452     227     035
28 163454                          3$:     LDA B   DCTL            ;PRINT CONTROLS ?
   163454     326     155
29 163456                                  BNE     5$              ;IF NOT - FINISHED
   163456     046     006
30 163460                          4$:     LDX     CURLOC          ;ELSE PLACE CHARACTER
   163460     336     152
31 163462                                  STA A   0,X
   163462     247     000
```

LINE MODE HANDLER

```
     32 163464                                  BRA     ADD1            ;GO UPDATE POSITION
        163464     040     026
     33 163466                          5$:     RTS                     ;FINISHED
        163466     071
     34                                         ;
```

MOTION HANDLERS

```
      1                                        .SBTTL  MOTION HANDLERS
      2                                        ;
      3 163467                         CRTRN:  LDA A   GPCNT           ;GET CURRENT COLUMN POSITION
        163467     226     144
      4 163471                                 CLR B                  ;SET COUNT TO 0
        163471     137
      5 163472                                 STA B   GPCNT
        163472     327     144
      6 163474                                 NEG A                  ;MAKE NEGATIVE
        163474     100
      7 163475                                 SBC B   #,0            ;EXTEND SIGN OF RESULT
        163475     302     000
      8 163477                                 BRA     UPDGL          ;UPDATE GLCNT
        163477     040     045
      9 163501                         ADD80:  LDA A   #,80.          ;NEXT LINE
        163501     206     120
     10 163503                                 CLR B
        163503     137
     11 163504                                 BRA     UPDGL          ;UPDATE GLCNT
        163504     040     040
     12 163506                         SUB80:  LDA A   #,-80.         ;BACK ONE LINE
        163506     206     260
     13 163510                                 LDA B   #,377          ;SIGN EXTENDED -80.
        163510     306     377
     14 163512                                 BRA     UPDGL          ;UPDATE GLCNT
        163512     040     032
     15 163514                         ADD1:   LDA A   #,1            ;1 CHARACTER
        163514     206     001
     16 163516                                 CLR B
        163516     137
     17 163517                                 BRA     UPDGP          ;UPDATE GLCNT AND GPCNT
        163517     040     003
     18 163521                         SUB1:   LDA A   #,-1           ;BACK ONE CHARACTER
        163521     206     377
     19 163523                                 TAB                    ;SIGN EXTENDED -1
```

```
    20                                      ;
    21 163524                       UPDGP:  PSH A                   ;SAVE COUNT
       163524     066
    22 163525                               ADD A   GPCNT           ;ADD IN UPDATE
       163525     233     144
    23 163527                               BPL     2$              ;IF POSITIVE - BRANCH
       163527     052     006
    24 163531                               ADD A   #,80.           ;ELSE UPDATE
       163531     213     120
    25 163533                               BRA     3$
       163533     040     006
    26 163535                       1$:     SUB A   #,80.           ;UPDATE
       163535     200     120
    27 163537                       2$:     CMP A   #,80.           ;PAST END OF LINE ?
       163537     201     120
    28 163541                               BPL     1$              ;IF SO - UPDATE AGAIN
       163541     052     372
    29 163543                       3$:     STA A   GPCNT           ;SAVE COUNT
       163543     227     144
    30 163545                               PUL A                   ;GET ORIGINAL UPDATE VALUE
       163545     062
```

MOTION HANDLERS

```
     31                                     ;
     32 163546                      UPDGL:  ADD A   GLCNT+1         ;UPDATE POSITION
        163546     233     143
     33 163550                              ADC B   GLCNT
        163550     331     142
     34 163552                      1$:     STA A   GLCNT+1         ;SAVE UPDATE
        163552     227     143
     35 163554                              STA B   GLCNT
        163554     327     142
     36 163556                              BPL     2$              ;IF POSITIVE - SKIP
        163556     052     006
     37 163560                              ADD A   #,1920.&377     ;ELSE ADD BUFFER LENGTH
        163560     213     200
     38 163562                              ADC B   #,1920.&177400/400
        163562     311     007
     39 163564                              BRA     1$              ;AND LOOP
        163564     040     364
     40 163566                      2$:     SUB A   #,1920.&377     ;SUBTRACT BUFFER LENGTH
        163566     200     200
     41 163570                              SBC B   #,1920.&177400/400
        163570     302     007
     42 163572                              BMI     3$              ;IF WITHIN PAGE - SKIP
        163572     053     013
     43 163574                              BSR     RPAGE           ;ROTATE PAGE
        163574     215     050
     44 163576                              LDA A   GPCNT           ;CURRENT COLUMN POSITION
        163576     226     144
     45 163600                              CLR B
        163600     137
     46 163601                              ADD A   #,1840.&377     ;ADD IN ROW POSITION OF BOTTOM LI
NE
        163601     213     060
     47 163603                              ADC B   #,1840.&177400/400
        163603     311     007
     48 163605                              BRA     1$              ;LOOP
        163605     040     343
```

```
    49                                      ;
    50 163607                       3$:     LDA A   GLBASE+1        ;COMPUTE CURSOR POSITION
       163607     226     141
    51 163611                               LDA B   GLBASE
       163611     326     140
    52 163613                               ADD A   GLCNT+1
       163613     233     143
    53 163615                               ADC B   GLCNT
       163615     331     142
    54 163617                               STA A   CURLOC+1
       163617     227     153
    55 163621                               STA B   CURLOC
       163621     327     152
    56 163623                               SUB A   #,80.           ;IS POSITION PAST END OF BUFFER ?
       163623     200     120
    57 163625                               SBC B   #,0
       163625     302     000
    58 163627                               SUB A   OVBASE+1
       163627     220     137
    59 163631                               SBC B   OVBASE
       163631     322     136
    60 163633                               BMI     4$              ;IF NOT - FINISHED
```

MOTION HANDLERS

```
           163633     053     010
     61    163635                             ADD A   #,CLBOT&377     ;ELSE UPDATE POSITION
           163635     213     120
     62    163637                             ADC B   #,CLBOT&177400/400
           163637     311     040
     63    163641                             STA A   CURLOC+1
           163641     227     153
     64    163643                             STA B   CURLOC
           163643     327     152
     65    163645                       4$:   RTS                     ;FINISHED
           163645     071
     66                                       ;
```

LINE MODE ROTATE PAGE

```
      1                                     .SBTTL  LINE MODE ROTATE PAGE
      2                                     ;
      3 163646                        RPAGE:  JSR     UPDLIN          ;UPDATE LINE
        163646     275     346     317
      4 163651                                LDX     CLPNTR          ;GET LINE POINTER
        163651     336     126
      5 163653                                LDA A   #,81.           ;80. CHARACTERS + CURSOR
        163653     206     121
      6 163655                                STA A   0,X             ;X IS CLPNTR
        163655     247     000
      7 163657                                JSR     CLEAR           ;CLEAR THE LINE
        163657     275     346     300
      8 163662                                JSR     UPDTXT          ;UPDATE TEXT
        163662     275     343     103
      9 163665                                INC     UPDFLG          ;UPDATE DISPLAY
        163665     174     000     106
     10 163670                                LDA A   #,80.           ;80. CHARACTERS
        163670     206     120
     11 163672                                CLR B
        163672     137
     12 163673                                ADD A   GLBASE+1        ;UPDATE BASE
        163673     233     141
     13 163675                                ADC B   GLBASE
        163675     331     140
     14 163677                                STA A   GLBASE+1
        163677     227     141
     15 163701                                STA B   GLBASE
        163701     327     140
     16 163703                                SUB A   #,CL.TOP&377    ;OVER TOP ?
        163703     200     260
     17 163705                                SBC B   #,CL.TOP&177400/400
        163705     302     057
     18 163707                                BCS     1$              ;IF NOT - SKIP
        163707     045     005
     19 163711                                LDX     #,CLBOT         ;ELSE RESET BASE POINTER
```

```
    20 163714                                STX     GLBASE
       163714     337     140
    21 163716                          1$:   RTS                     ;FINISHED
       163716     071
    22                                       ;
```

```
      1                                     .SBTTL  HOME AND CLSCN ROUTINES
      2                                     ;
      3 163717                      CLRSCN: LDX     GLBASE          ;BASE ADDRESS
        163717     336     140
      4 163721                              LDA A   #,24.           ;LINES IN PAGE
        163721     206     030
      5 163723                              LDA B   #,40            ;FILL WITH SPACES
        163723     306     040
      6 163725                      1$:     PSH A                   ;SAVE COUNTER
        163725     066
      7 163726                              LDA A   #,80.           ;80. CHARACTERS PER LINE
        163726     206     120
      8 163730                              STX     TEMPGA          ;SAVE FIRST ADDRESS
        163730     337     225
      9 163732                      2$:     STA B   0,X             ;CLEAR LINE
        163732     347     000
     10 163734                              INX
        163734     010
     11 163735                              DEC A
        163735     112
     12 163736                              BGT     2$              ;LOOP UNTIL LINE CLEARED
        163736     056     372
     13 163740                              STX     TEMPGB          ;SAVE POINTER
        163740     337     227
     14 163742                              JSR     TMCK            ;CHECK TIME
        163742     275     342     072
     15 163745                              LDX     TEMPGA          ;CHECK BEGINNING ADDRESS
        163745     336     225
     16 163747                              CPX     OVBASE          ;AT OVBASE ?
        163747     234     136
     17 163751                              BNE     3$              ;IF NOT SKIP
        163751     046     005
     18 163753                              LDX     #,CLBOT         ;RESET POINTER
        163753     316     040     120
     19 163756                              STX     TEMPGB
```

```
    20 163760                      3$:     LDX     TEMPGB          ;GET POINTER
       163760     336     227
    21 163762                              PUL A                   ;GET COUNTER
       163762     062
    22 163763                              DEC A
       163763     112
    23 163764                              BGT     1$              ;LOOP FOR ALL LINES
       163764     056     337
    24 163766                              STA B   0,X             ;ONE MORE CHARACTER
       163766     347     000
    25                                     ;
    26 163770                      HOME:   CLR A                   ;CLEAR MOST EVERYTHING
       163770     117
    27 163771                              STA A   GPCNT
       163771     227     144
    28 163773                              STA A   GLCNT+1
       163773     227     143
    29 163775                              STA A   GLCNT
       163775     227     142
    30 163777                              LDX     GLBASE          ;REPOSIION CURSOR
       163777     336     140
```

HOME AND CLSCN ROUTINES

```
     31 164001                                  STX     CURLOC
        164001     337     152
     32 164003                                  RTS                     ;FINISHED
        164003     071
     33                                         ;
```

LOCAL MONTOR SCANNER

```
      2                                     .SBTTL  LOCAL MONTOR SCANNER
      3                                     ;
      4 164004                        LCLMON: LDA A   CHARA           ;GET CHARACTER
        164004     226     103
      5 164006                                LDA B   LCLL            ;SHIFT CHARACTERS
        164006     326     204
      6 164010                                STA B   LCLH
        164010     327     203
      7 164012                                STA A   LCLL
        164012     227     204
      8 164014                                LDX     LCLH            ;GET BOTH CHARACTERS
        164014     336     203
      9 164016                                CPX     #,'@,'=         ;@= SEQUENCE ?
        164016     214     100     075
     10 164021                                BEQ     1$              ;IF SO - PROCESS
        164021     047     001
     11 164023                                RTS                     ;ELSE FINISHED
        164023     071
     12 164024                           1$:  JSR     GETSTR          ;GET A CHARACTER STRING
        164024     275     350     110
     13 164027                                BSR     CHECK           ;CHECK END OF STRING
        164027     215     025
     14 164031                                LDX     #,TABLE         ;LIST OF MONITOR COMMANDS
        164031     316     350     302
     15 164034                                JSR     SRCHTB          ;GO SEARCH MONITOR TABLE
        164034     275     350     240
     16 164037                                BCS     2$              ;ON A MATCH - SKIP
        164037     045     007
     17 164041                                LDA A   #,'?            ;ELSE PRINT ?
        164041     206     077
     18 164043                                JSR     .PRINT          ;PRINT IT
        164043     275     345     342
     19 164046                                BSR     CHECKX          ;AND TERMINATE SEQUENCE
        164046     215     025
     20 164050                           2$:  LDX     TEMPLA          ;JUMP TO ROUTINE IN TABLE
```

```
    21 164052                              LDX     4,X
       164052     356     004
    22 164054                              JMP     0,X
       164054     156     000
    23                                     ;
```

TERMINATION CHECK ROUTINE

```
      1                                    .SBTTL  TERMINATION CHECK ROUTINE
      2                                    ;
      3 164056                      CHECK: LDA A   CHARA           ;GET LAST CHARACTER
        164056     226     103
      4 164060                             CMP A   #,15            ;A <CR> ?
        164060     201     015
      5 164062                             BNE     CHECKY          ;IF NOT - SKIP
        164062     046     005
      6 164064                             LDA B   LSTRNG          ;ANY CHARACTERS ?
        164064     366     002     000
      7 164067                             BEQ     CHECKX          ;IF NOT - TERMINATE
        164067     047     004
      8 164071                      CHECKY: CMP A   #,12            ;A <LF> ?
        164071     201     012
      9 164073                             BNE     CHKEND          ;IF NOT - JUST CONTINUE
        164073     046     012
     10 164075                      CHECKX: LDX     #,LCLMON        ;TERMINATE CURRENT SEQUENCE
        164075     316     350     004
     11 164100                             STX     LJSR
        164100     337     201
     12 164102                             JSR     PLMSG0          ;PRINT <CRLF>*
        164102     275     351     132
     13 164105                             PUL A                   ;REMOVE ONE RETURN LEVEL
        164105     062
     14 164106                             PUL A
        164106     062
     15 164107                      CHKEND: RTS                    ;RETURN TO CALLER
        164107     071
     16                                    ;
```

GET STRING ROUTINE

```
      1                                     .SBTTL  GET STRING ROUTINE
      2                                     ;
      3 164110                        GETSTR: PUL A                   ;SAVE RETURN ADDRESS
        164110     062
      4 164111                                STA A   LRVEC
        164111     227     231
      5 164113                                PUL A
        164113     062
      6 164114                                STA A   LRVEC+1
        164114     227     232
      7 164116                                LDX     #,GETS          ;SETUP NEXT TRANSFER ADDRESS
        164116     316     350     141
      8 164121                                STX     LJSR
        164121     337     201
      9 164123                                LDX     #,LSTRNG+80.    ;TOP OF STRING AREA
        164123     316     002     120
     10 164126                             1$:     DEX                     ;UPDATE ADDRESS
        164126     011
     11 164127                                CLR     0,X             ;CLEAR STRING
        164127     157     000
     12 164131                                CPX     #,LSTRNG        ;AT BOTTOM ?
        164131     214     002     000
     13 164134                                BNE     1$              ;LOOP UNTIL FINISHED
        164134     046     370
     14 164136                                STX     LSTPNT          ;LOCAL STRING POINTER
        164136     337     233
     15 164140                                RTS                     ;RETURN 2 LEVELS BACK
        164140     071
     16                                      ;
     17 164141                        GETS:   LDA A   CHARA           ;GET LAST CHARACTER
        164141     226     103
     18 164143                                CMP A   #,15            ;A <CR> ?
        164143     201     015
     19 164145                                BEQ     1$              ;IF SO - HAVE STRING
        164145     047     004
```

```
   164147     201     012
21 164151                          BNE     2$              ;IF NOT - SKIP
   164151     046     007
22 164153                    1$:   INC     CHRFLG          ;INHIBIT PRINTING
   164153     174     000     105
23 164156                          LDX     LRVEC           ;RETURN IN LINE AFTER GETSTR
   164156     336     231
24 164160                          JMP     0,X
   164160     156     000
25                                 ;
26 164162                    2$:   LDX     LSTPNT          ;GET STRING POINTER
   164162     336     233
27 164164                          CMP A   #,177           ;A DELETE ?
   164164     201     177
28 164166                          BEQ     4$              ;IF SO - SKIP
   164166     047     013
29 164170                          STA A   0,X             ;ELSE SAVE CHARACTER
   164170     247     000
30 164172                          CPX     #,LSTRNG+78.    ;AT END OF STRING ?
   164172     214     002     116
31 164175                          BEQ     3$              ;IF SO - SKIP UPDATE
```

GET STRING ROUTINE

```
   164175     047     003
32 164177                              INX                     ;ELSE UPDATE POINTER
   164177     010
33 164200                              STX     LSTPNT
   164200     337     233
34 164202                         3$:  RTS                     ;FINISHED
   164202     071
35                                     ;
36 164203                         4$:  DEX                     ;ONE LESS CHARACTER
   164203     011
37 164204                              CPX     #,LSTRNG-1      ;BELOW BOTTOM ?
   164204     214     001     377
38 164207                              BEQ     6$              ;IF SO - LEAVE
   164207     047     023
39 164211                              STX     LSTPNT          ;ELSE SAVE POINTER
   164211     337     233
40 164213                              CLR     0,X             ;DELETE CHARACTER
   164213     157     000
41 164215                              LDA A   #,10            ;BACKSPACE
   164215     206     010
42 164217                              JSR     .PRINT
   164217     275     345     342
43 164222                              LDA A   #,40            ;SPACE
   164222     206     040
44 164224                              JSR     .PRINT
   164224     275     345     342
45 164227                              LDA A   #,10            ;BACKSPACE
   164227     206     010
46 164231                              JSR     .PRINT
   164231     275     345     342
47 164234                         6$:  INC     CHRFLG          ;INHIBIT PRINTING
   164234     174     000     105
48 164237                              RTS                     ;FINISHED
   164237     071
49                                     ;
```

LOCAL MONITOR TABLE SEARCH ROUTINE

```
      1                                        .SBTTL  LOCAL MONITOR TABLE SEARCH ROUTINE
      2                                        ;
      3 164240                         SRCHTB: STX     TEMPLA          ;SAVE POINTER
        164240     337     205
      4 164242                                 LDX     0,X             ;GET 2 BYTES OF STRING
        164242     356     000
      5 164244                                 BEQ     2$              ;END OF TABLE - SKIP
        164244     047     030
      6 164246                                 CPX     LSTRNG          ;COMPARE FIRST TWO CHARACTERS
        164246     274     002     000
      7 164251                                 BNE     1$              ;NO MATCH - SKIP
        164251     046     011
      8 164253                                 LDX     TEMPLA          ;GET NEXT TWO CHARACTERS
        164253     336     205
      9 164255                                 LDX     2,X
        164255     356     002
     10 164257                                 CPX     LSTRNG+2        ;COMPARE SECOND SET
        164257     274     002     002
     11 164262                                 BEQ     3$              ;ON MATCH - SKIP
        164262     047     014
     12 164264                             1$: LDX     TEMPLA          ;UPDATE POINTER TO NEXT TABLE ENT
RY
        164264     336     205
     13 164266                                 INX
        164266     010
     14 164267                                 INX
        164267     010
     15 164270                                 INX
        164270     010
     16 164271                                 INX
        164271     010
     17 164272                                 INX
        164272     010
     18 164273                                 INX
        164273     010
     19 164274                                 BRA     SRCHTB          ;AND TRY NEXT ENTRY
```

```
        164274     040     342
     20 164276                          2$:     CLC                     ;NO MATCH
        164276     014
     21 164277                                  RTS                     ;RETURN
        164277     071
     22 164300                          3$:     SEC                     ;MATCH FOUND
        164300     015
     23 164301                                  RTS                     ;RETURN
        164301     071
     24                                         ;
```

LOCAL MONITOR DISPATCH TABLE

```
      1                                     .SBTTL  LOCAL MONITOR DISPATCH TABLE
      2                                     ;
      3 164302     123     131     123  TABLE:  .ASCII  /SYST/
        164305     124
      4 164306                             FDB     SYST
        164306     351     265
      5 164310     124     111     115          .ASCII  /TIME/
        164313     105
      6 164314                             FDB     TIME
        164314     353     132
      7 164316     103     124     114          .ASCII  /CTLC/
        164321     103
      8 164322                             FDB     CTLC
        164322     353     167
      9 164324     120     105     105          .ASCII  /PEEK/
        164327     113
     10 164330                             FDB     PEEK
        164330     353     217
     11 164332     120     117     113          .ASCII  /POKE/
        164335     105
     12 164336                             FDB     POKE
        164336     353     260
     13 164340     102     101     125          .ASCII  /BAUD/
        164343     104
     14 164344                             FDB     BAUD
        164344     354     114
     15 164346     115     101     123          .ASCII  /MASK/
        164351     113
     16 164352                             FDB     MASK
        164352     354     137
     17 164354     105     123     103          .ASCII  /ESCP/
        164357     120
     18 164360                             FDB     ESCP
        164360     354     310
     19 164362     122     113     060          .ASCII  /RK0:/
```

```
    20 164366                                  FDB     BRK0
       164366     356     200
    21 164370     122     113     061         .ASCII  /RK1:/
       164373     072
    22 164374                                  FDB     BRK1
       164374     356     214
    23 164376     122     113     062         .ASCII  /RK2:/
       164401     072
    24 164402                                  FDB     BRK2
       164402     356     230
    25 164404     122     113     063         .ASCII  /RK3:/
       164407     072
    26 164410                                  FDB     BRK3
       164410     356     244
    27 164412     122     113     064         .ASCII  /RK4:/
       164415     072
    28 164416                                  FDB     BRK4
       164416     356     260
    29 164420     122     113     065         .ASCII  /RK5:/
       164423     072
    30 164424                                  FDB     BRK5
```

LOCAL MONITOR DISPATCH TABLE

```
           164424     356     274
        31 164426     122     113     066          .ASCII  /RK6:/
           164431     072
        32 164432                                  FDB     BRK6
           164432     356     310
        33 164434     122     113     067          .ASCII  /RK7:/
           164437     072
        34 164440                                  FDB     BRK7
           164440     356     324
        35 164442     104     131     060          .ASCII  /DY0:/
           164445     072
        36 164446                                  FDB     BDY0
           164446     356     340
        37 164450     104     131     061          .ASCII  /DY1:/
           164453     072
        38 164454                                  FDB     BDY1
           164454     356     354
        39 164456     104     130     060          .ASCII  /DX0:/
           164461     072
        40 164462                                  FDB     BDX0
           164462     356     370
        41 164464     104     130     061          .ASCII  /DX1:/
           164467     072
        42 164470                                  FDB     BDX1
           164470     357     004
        43 164472     103     124     060          .ASCII  /CT0:/
           164475     072
        44 164476                                  FDB     BCT0
           164476     357     020
        45 164500     103     124     061          .ASCII  /CT1:/
           164503     072
        46 164504                                  FDB     BCT1
           164504     357     034
        47 164506     124     124     072          .ASCII  /TT:/<0>
           164511     000
```

```
   164512     357     064
49 164514     115     124     072          .ASCII  /MT:/<0>
   164517     000
50 164520                                  FDB     BMT0
   164520     357     050
51 164522     113     125     123          .ASCII  /KUS/<0>
   164525     000
52 164526                                  FDB     BKUS
   164526     356     164
53 164530                                  FDB     0       ;TABLE TERMINATION
   164530     000     000
54                                         ;
```

LPRINT ROUTINE AND VARIATIONS

```
                                        .SBTTL  LPRINT ROUTINE AND VARIATIONS
                                        ;
164532                          PLMSG0: LDX     #,LMSG0         ;<CRLF>*
  164532     316     351     173
164535                                  BRA     LPRINT
  164535     040     015
164537                          PLMSG1: LDX     #,LMSG1         ;<CRLF>* >>
  164537     316     351     177
164542                                  BRA     LPRINT
  164542     040     010
164544                          PLMSG2: LDX     #,LMSG2         ;<CRLF>*A
  164544     316     351     210
164547                                  BRA     LPRINT
  164547     040     003
164551                          PLMSG3: LDX     #,LMSG3         ;/   /
  164551     316     351     215
                                        ;
164554                          LPRINT: STX     TEMPLA          ;SAVE POINTER
  164554     337     205
164556                                  LDA A   0,X             ;GET CHARACTER
  164556     246     000
164560                                  BEQ     1$              ;NULL IS TERMINATION
  164560     047     010
164562                                  JSR     .PRINT          ;PRINT IT
  164562     275     345     342
164565                                  LDX     TEMPLA          ;NOW UPDATE POINTER
  164565     336     205
164567                                  INX
  164567     010
164570                                  BRA     LPRINT          ;CHECK NEXT CHARACTER
  164570     040     362
164572                          1$:     RTS                     ;FINISHED
  164572     071
                                        ;
                                        ;       MESSAGES
```

```
    22                .NLIST  BIN
    23 164573  LMSG0:  .ASCIZ  <15><12>/*/
    24 164577  LMSG1:  .ASCIZ  <15><12>/* >>  /
    25 164610  LMSG2:  .ASCIZ  <15><12>/*A/
    26 164615  LMSG3:  .ASCIZ  /   /
    27 164621  LIBIS:  .ASCIZ  / IBIS  =/
    28 164632  LIBIC:  .ASCIZ  / IBIC  =/
    29 164643  LOBIS:  .ASCIZ  / OBIS  =/
    30 164654  LOBIC:  .ASCIZ  / OBIC  =/
    31                                      .LIST BIN
    32                                      ;
```

SYSTEM NOTES

```
                                        .SBTTL  SYSTEM NOTES
                                        ;
164665                          SYST:   LDX     #,SYSMSG        ;DUMP SYSTEM MESSAGE
164665     316     351     276
164670                                  JSR     LPRINT
164670     275     351     154
164673                                  JSR     CHECKX          ;DUMBY ROUTINE
164673     275     350     075
                                        ;
                .NLIST BIN
164676  SYSMSG: .ASCII  <15><12><12>/MAY 1981/<15><12><12>
164714          .ASCII  /LOCAL MONITOR COMMANDS/<15><12>
164744          .ASCII  /@= SYST,TIME,CTLC,MASK,BAUD,PEEK,POKE,ESCP/<15><12><12>
165021          .ASCII  /DEVICE BOOTS/<15><12>
165037          .ASCII  /@= RK0:,RK1:,RK2:,RK3:,RK4:,RK5:,RK6:,RK7:/<15><12>
165113          .ASCII  /   DX0:,DX1:,DY0:,DY1:,CT0:,CT1:,TT:,MT:/<15><12><12>
165166          .ASCII  /UTILLITY PROGRAM/<15><12>
165210          .ASCII  /@= KUS/<15><12><12>
165221          .ASCII  /ESCAPE CODE MONITOR/<15><12>
165246          .ASCII  /ESC= 0-7/<15><12>
165260          .ASCII  /CHARACTER SET RESTORE (0), BAUD RATE (1)/<15><12>
165332          .ASCII  /CHARACTER MODE (2), LINE MODE (3)/<15><12>
165375          .ASCII  /DEFINE CHARACTER (4), MOV CURSOR (5)/<15><12>
165443          .ASCII  /TERMINAL GRAPHICS ON (6), TERMINAL GRAPHICS OFF (7)/
165526          .ASCII  <15><12><12><0>
                                        .LIST   BIN
                                        ;
```

TIME ROUTINE

```
      1                                     .SBTTL  TIME ROUTINE
      2                                     ;
      3 165532                        TIME: JSR     PLMSG1          ;PRINT <CRLF>* >>
        165532     275     351     137
      4 165535                              JSR     GETSTR          ;GET A STRING
        165535     275     350     110
      5 165540                              JSR     CHECK           ;CHECK TERMINATION
        165540     275     350     056
      6 165543                              LDX     #,CTLINE+70     ;CLOCK AREA
        165543     316     040     070
      7 165546                              STX     FL.TO
        165546     337     117
      8 165550                              LDX     #,LSTRNG        ;STRING AREA
        165550     316     002     000
      9 165553                              STX     FL.FRM
        165553     337     121
     10 165555                              LDA A   #,11.           ;ONLY ALLOW 11 CHARACTERS
        165555     206     013
     11 165557                              STA A   FL.CNT
        165557     227     123
     12 165561                              JSR     FILL            ;PLACE STRING
        165561     275     344     007
     13 165564                              JSR     CHECKX          ;TERMINATE SEQUENCE
        165564     275     350     075
     14                                     ;
```

CTLC ROUTINE

```
      1                                        .SBTTL  CTLC ROUTINE
      2                                        ;
      3 165567                           CTLC: JSR     PLMSG1          ;PRINT <CRLF>* >>
        165567     275     351     137
      4 165572                                 JSR     GETSTR          ;GET STRING
        165572     275     350     110
      5 165575                                 JSR     CHECK           ;CHECK TERMINATION
        165575     275     350     056
      6 165600                                 CLR B                   ;VALUE TO ALLOW PRINTIING
        165600     137
      7 165601                                 LDA A   LSTRNG          ;GET FIRST CHARACTER
        165601     266     002     000
      8 165604                                 CMP A   #,'Y            ;Y IF CONTROLS TO PRINT
        165604     201     131
      9 165606                                 BEQ     1$
        165606     047     002
     10 165610                                 LDA B   #,40            ;ELSE INHIBIT PRINTING
        165610     306     040
     11 165612                           1$:   STA B   DCTL            ;SET FLAG
        165612     327     155
     12 165614                                 JSR     CHECKX          ;END SEQUENCE
        165614     275     350     075
     13                                        ;
```

PEEK ROUTINE

```
      1                                        .SBTTL  PEEK ROUTINE
      2                                        ;
      3 165617                           PEEK: JSR     PLMSG1          ;PRINT <CRLF>* >>
        165617     275     351     137
      4 165622                                 JSR     GETSTR          ;GET A STRING
        165622     275     350     110
      5 165625                                 JSR     CHECK           ;CHECK TERMINATION
        165625     275     350     056
      6 165630                                 JSR     PEVAL           ;EVALUATE STRING
        165630     275     353     336
      7 165633                           1$:   JSR     PRNTAD          ;PRINT ADDRESS
        165633     275     354     021
      8 165636                                 LDA A   #,10            ;8 BYTES PER LINE
        165636     206     010
      9 165640                           2$:   PSH A                   ;SAVE COUNT
        165640     066
     10 165641                                 JSR     PRNTDT          ;PRINT DATA
        165641     275     354     031
     11 165644                                 PUL A
        165644     062
     12 165645                                 DEC A                   ;MORE TO DO ?
        165645     112
     13 165646                                 BNE     2$              ;LOOP UNTIL DONE
        165646     046     370
     14 165650                                 JSR     GETSTR          ;GET ANOTHER STRING
        165650     275     350     110
     15 165653                                 JSR     CHECKY          ;CHECK FOR TERMINATION
        165653     275     350     071
     16 165656                                 BRA     1$              ;GO PRINT SOME MORE
        165656     040     353
     17                                        ;
```

POKE ROUTINE

```
      1                                      .SBTTL  POKE ROUTINE
      2                                      ;
      3 165660                         POKE: JSR     PLMSG1          ;PRINT <CRLF>* >>
        165660     275     351     137
      4 165663                               JSR     GETSTR          ;GET A STRING
        165663     275     350     110
      5 165666                               JSR     CHECK           ;CHECK TERMINATION
        165666     275     350     056
      6 165671                               JSR     PEVAL           ;EVALUATE STRING
        165671     275     353     336
      7 165674                         1$:   JSR     PRNTAD          ;PRINT ADDRESS
        165674     275     354     021
      8 165677                               JSR     PLMSG3          ;PRINT /    /
        165677     275     351     151
      9 165702                               JSR     GETSTR          ;GET NEW DATA
        165702     275     350     110
     10 165705                               JSR     CHECKY          ;CHECK TERMINATION
        165705     275     350     071
     11 165710                               LDX     #,LSTRNG        ;EVALUATE THE DATA
        165710     316     002     000
     12 165713                               JSR     EVAL
        165713     275     353     350
     13 165716                               LDA A   VL              ;GET VALUE
        165716     226     214
     14 165720                               LDX     TEMPLB          ;POKE ADDRESS
        165720     336     207
     15 165722                               LDA B   LSTRNG          ;A REAL STRING ?
        165722     366     002     000
     16 165725                               BEQ     2$              ;IF NOT - SKIP
        165725     047     002
     17 165727                               STA A   0,X             ;ELSE POKE DATA
        165727     247     000
     18 165731                         2$:   INX                     ;UPDATE POINTER
        165731     010
     19 165732                               STX     TEMPLB
```

```
    20 165734                           BRA     1$              ;LOOP ONE MORE TIME
       165734     040     336
    21                                  ;
```

EVALUATION ROUTINES

```
      1                                      .SBTTL  EVALUATION ROUTINES
      2                                      ;
      3 165736                        PEVAL: LDX     #,LSTRNG        ;EVALUATE STRING
        165736     316     002     000
      4 165741                               BSR     EVAL
        165741     215     005
      5 165743                               LDX     VH              ;EVALUATION
        165743     336     213
      6 165745                               STX     TEMPLB          ;SAVE VALUE
        165745     337     207
      7 165747                               RTS                     ;FINISHED
        165747     071
      8                                      ;
      9 165750                        EVAL:  CLR     VH              ;CLEAR RESULT
        165750     177     000     213
     10 165753                               CLR     VL
        165753     177     000     214
     11 165756                        1$:    LDA A   0,X             ;GET FIRST CHARACTER
        165756     246     000
     12 165760                               BEQ     3$              ;IF NULL - END
        165760     047     024
     13 165762                               SUB A   #,'0            ;COMPUTE BINARY VALUE
        165762     200     060
     14 165764                               LDA B   #,3             ;3 BIT SHIFT
        165764     306     003
     15 165766                        2$:    BSR     SHIFT
        165766     215     017
     16 165770                               DEC B                   ;FINISHED ?
        165770     132
     17 165771                               BNE     2$              ;LOOP UNTIL FINISHED
        165771     046     373
     18 165773                               ADD A   VL              ;COMPUTE UPDATED VALUE
        165773     233     214
     19 165775                               ADC B   VH
        165775     331     213
```

```
   165777     227     214
21 166001                              STA B   VH
   166001     327     213
22 166003                              INX                     ;UPDATE POINTER
   166003     010
23 166004                              BRA     1$              ;GET NEXT CHARACTER
   166004     040     350
24 166006                      3$:     RTS                     ;FINISHED
   166006     071
25                                     ;
26 166007                      SHIFT:  ASL     VL      ;SHIFT A 24. BIT WORD
   166007     170     000     214
27 166012                              ROL     VH
   166012     171     000     213
28 166015                              ROL     VAL
   166015     171     000     215
29 166020                              RTS                     ;FINISHED
   166020     071
30                                     ;
```

PRINT OCTAL VALUE ROUTINES

```
                                        .SBTTL  PRINT OCTAL VALUE ROUTINES
                                        ;
166021                          PRNTAD: JSR     PLMSG2          ;PRINT <CRLF>*A
  166021     275     351     144
166024                                  LDX     #,TEMPLB        ;ADDRESS OF DATA
  166024     316     000     207
166027                                  BRA     PWORD           ;GO PRINT IT
  166027     040     015
                                        ;
166031                          PRNTDT: JSR     PLMSG3          ;PRINT /   /
  166031     275     351     151
166034                                  LDX     TEMPLB          ;INCREMENT POINTER
  166034     336     207
166036                                  INX
  166036     010
166037                                  STX     TEMPLB
  166037     337     207
166041                                  DEX
  166041     011
                                        ;
166042                          PBYTE:  LDA A   #,3             ;COUNT OFF THREE CHARACTERS
  166042     206     003
166044                                  BRA     PWRD.A
  166044     040     002
166046                          PWORD:  LDA A   #,6             ;COUNT OFF 6 CHARACTERS
  166046     206     006
166050                          PWRD.A: LDX     0,X             ;GET BINARY VALUE TO CONVERT
  166050     356     000
166052                                  STX     VH              ;SAVE VALUE
  166052     337     213
166054                                  PSH A                   ;SAVE COUNT
  166054     066
166055                                  CLR     VAL             ;CLEAR HIGH BYTE
  166055     177     000     215
166060                                  CMP A   #,6             ;WORD CONVERSION ?
```

```
    21 166062                              BEQ     2$              ;IF SO - SKIP
       166062     047     002
    22 166064                      1$:     BSR     SHIFT           ;SHIFT A BIT
       166064     215     321
    23 166066                      2$:     BSR     SHIFT           ;SHIFT A BIT
       166066     215     317
    24 166070                              LDA A   #,'0            ;DEVELOPE A CHARACTER
       166070     206     060
    25 166072                              ADD A   VAL
       166072     233     215
    26 166074                              JSR     .PRINT          ;PRINT CHARACTER
       166074     275     345     342
    27 166077                              PUL A
       166077     062
    28 166100                              DEC A                   ;MORE CHARACTERS TO DO ?
       166100     112
    29 166101                              BEQ     3$              ;IF NOT - FINISHED
       166101     047     010
    30 166103                              PSH A                   ;SAVE COUNT
       166103     066
    31 166104                              CLR     VAL             ;CLEAR VALUE
```

PRINT OCTAL VALUE ROUTINES

```
   166104     177     000     215
32 166107                                BSR     SHIFT           ;DO A SHIFT
   166107     215     276
33 166111                                BRA     1$              ;LOOP FOR MORE
   166111     040     351
34 166113                          3$:   RTS                     ;FINISHED
   166113     071
35                                       ;
```

BAUD RATE ROUTINE

```
      1                                      .SBTTL  BAUD RATE ROUTINE
      2                                      ;
      3 166114                         BAUD: JSR     PLMSG1          ;PRINT <CRLF>* >>
        166114     275     351     137
      4 166117                               JSR     GETSTR          ;GET A STRING
        166117     275     350     110
      5 166122                               JSR     CHECK           ;CHECK TERMINATION
        166122     275     350     056
      6 166125                               JSR     PEVAL           ;EVALUATE THE STRING
        166125     275     353     336
      7 166130                               LDA A   VL              ;LOW ORDER VALUE
        166130     226     214
      8 166132                               STA A   NALL            ;CHARACTERS ALLOWED PER 60TH
        166132     227     112
      9 166134                               JSR     CHECKX          ;TERMINATE SEQUENCE
        166134     275     350     075
     10                                      ;
```

CHARACTER MASK ROUTINE

```
      1                                      .SBTTL  CHARACTER MASK ROUTINE
      2                                      ;
      3 166137                         MASK: LDX     #,IBIS          ;ADDRESS OF VARIABLES
        166137     316     000     170
      4 166142                               STX     TEMPLB          ;SAVE POINTER
        166142     337     207
      5 166144                               BSR     PIBIS           ;IBIS SEQUENCE
        166144     215     051
      6 166146                               JSR     GETSTR          ;GET A STRING
        166146     275     350     110
      7 166151                               JSR     CHECKY          ;CHECK TERMINATION
        166151     275     350     071
      8 166154                               BSR     PUT             ;PLACE DATA
        166154     215     107
      9 166156                               BSR     PIBIC           ;IBIC SEQUENCE
        166156     215     044
     10 166160                               JSR     GETSTR          ;GET A STRING
        166160     275     350     110
     11 166163                               JSR     CHECKY          ;CHECK TERMINATION
        166163     275     350     071
     12 166166                               BSR     PUT             ;PLACE DATA
        166166     215     075
     13 166170                               BSR     POBIS           ;OBIS SEQUENCE
        166170     215     037
     14 166172                               JSR     GETSTR          ;GET A STRING
        166172     275     350     110
     15 166175                               JSR     CHECKY          ;CHECK TERMINATION
        166175     275     350     071
     16 166200                               BSR     PUT             ;PLACE DATA
        166200     215     063
     17 166202                               BSR     POBIC           ;OBIC SEQUENCE
        166202     215     032
     18 166204                               JSR     GETSTR          ;GET A STRING
        166204     275     350     110
     19 166207                               JSR     CHECKY          ;CHECK TERMINATION
```

```
    20 166212                              BSR     PUT             ;PLACE DATA
       166212     215     051
    21 166214                              JSR     CHECKX          ;TERMINATE SEQUENCE
       166214     275     350     075
    22                                     ;
```

MASK LIST/PUT ROUTINES

```
      1                                      .SBTTL  MASK LIST/PUT ROUTINES
      2                                      ;
      3 166217                       PIBIS:  LDX     #,LIBIS         ;MESSAGE POINTER
        166217     316     351     221
      4 166222                               BRA     PCOMB
        166222     040     015
      5 166224                       PIBIC:  LDX     #,LIBIC
        166224     316     351     232
      6 166227                               BRA     PCOMB
        166227     040     010
      7 166231                       POBIS:  LDX     #,LOBIS
        166231     316     351     243
      8 166234                               BRA     PCOMB
        166234     040     003
      9 166236                       POBIC:  LDX     #,LOBIC
        166236     316     351     254
     10                                      ;
     11 166241                       PCOMB:  STX     TEMPLC          ;SAVE MESSAGE POINTER
        166241     337     211
     12 166243                               JSR     PLMSG0          ;PRINT <CRLF>*
        166243     275     351     132
     13 166246                               JSR     PLMSG3          ;PRINT /   /
        166246     275     351     151
     14 166251                               LDX     TEMPLC          ;NOW PRINT MESSAGE
        166251     336     211
     15 166253                               JSR     LPRINT
        166253     275     351     154
     16 166256                               JSR     PRNTDT          ;PRINT DATA
        166256     275     354     031
     17 166261                               JSR     PLMSG1          ;PRINT <CRLF>* >>
        166261     275     351     137
     18 166264                               RTS                     ;FINISHED
        166264     071
     19                                      ;
     20 166265                       PUT:    LDA A   LSTRNG          ;A REAL STRING ?
```

```
 21 166270                              BEQ     2$              ;IF NOT - LEAVE
    166270     047     015
 22 166272                              LDX     #,LSTRNG        ;EVALUATE THE STRING
    166272     316     002     000
 23 166275                              JSR     EVAL
    166275     275     353     350
 24 166300                              LDX     TEMPLB          ;UPDATE ADDRESS
    166300     336     207
 25 166302                              DEX
    166302     011
 26 166303                              LDA A   VL              ;GET EVALUATION
    166303     226     214
 27 166305                              STA A   0,X             ;SAVE NEW VALUE
    166305     247     000
 28 166307                      2$:     RTS                     ;FINISHED
    166307     071
 29                                     ;
```

ESCP ROUTINE

```
      1                                        .SBTTL  ESCP ROUTINE
      2                                        ;
      3 166310                           ESCP: JSR     PLMSG1          ;PRINT <CRLF>* >>
        166310     275     351     137
      4 166313                                 JSR     GETSTR          ;GET A STRING
        166313     275     350     110
      5 166316                                 JSR     CHECK           ;CHECK TERMINATION
        166316     275     350     056
      6 166321                                 CLR B                   ;DISABLE ESCAPE SEQUENCES
        166321     137
      7 166322                                 LDA A   LSTRNG          ;GET CHARACTER
        166322     266     002     000
      8 166325                                 CMP A   #,'Y            ;ESCAPES ENABLED ?
        166325     201     131
      9 166327                                 BNE     1$              ;IF NOT - SKIP
        166327     046     002
     10 166331                                 LDA B   #,377           ;ELSE ENABLE
        166331     306     377
     11 166333                           1$:   STA B   ESCFLG          ;SET FLAG
        166333     327     116
     12 166335                                 JSR     CHECKX          ;TERMINATE SEQUENCE
        166335     275     350     075
     13                                        ;
```

ESCAPE SEQUENCE MONITOR

```
      1                                       .SBTTL  ESCAPE SEQUENCE MONITOR
      2                                       ;
      3 166340                        ESCAPE: LDA A   CHARA           ;GET NEW CHARACTER
        166340     226     103
      4 166342                                LDA B   ESCL            ;TWO CHARACTER SHIFTER
        166342     326     200
      5 166344                                STA B   ESCH
        166344     327     177
      6 166346                                STA A   ESCL
        166346     227     200
      7 166350                                CMP A   #,33            ;NEW CHAR AN ESC ?
        166350     201     033
      8 166352                                BNE     2$              ;IF NOT - SKIP
        166352     046     010
      9 166354                                CMP B   #,33            ;OLD CHAR AN ESC ?
        166354     301     033
     10 166356                                BEQ     1$              ;IF SO - SKIP
        166356     047     003
     11 166360                                INC     CHRFLG          ;INHIBIT PRINTING
        166360     174     000     105
     12 166363                           1$:  RTS                     ;FINISHED
        166363     071
     13 166364                           2$:  CMP B   #,33            ;OLD CHAR AN ESC ?
        166364     301     033
     14 166366                                BNE     3$              ;IF NOT - SKIP
        166366     046     010
     15 166370                                CMP A   #,'=            ;AN = SIGN ?
        166370     201     075
     16 166372                                BEQ     4$              ;IF SO - SKIP
        166372     047     005
     17 166374                                TBA                     ;ELSE PRINT OLD ESC
        166374     027
     18 166375                                JSR     .PRINT
        166375     275     345     342
     19 166400                           3$:  RTS                     ;FINISHED
```

```
20 166401                          4$:     INC     CHRFLG          ;INHIBIT PRINTING
   166401     174     000     105
21 166404                                  LDX     #,GCTL          ;SET UP NEXT TRANSFER
   166404     316     355     012
22 166407                                  STX     EJSR
   166407     337     175
23 166411                                  RTS                     ;FINISHED
   166411     071
24                                         ;
```

ESCAPE DISPATCHER

```
      1                                      .SBTTL  ESCAPE DISPATCHER
      2                                      ;
      3 166412                         GCTL: LDA A   CHARA           ;GET CHARACTER
        166412     226     103
      4 166414                               LDX     #,ESCAPE       ;RESET ENTRY POINTER
        166414     316     354     340
      5 166417                               STX     EJSR
        166417     337     175
      6 166421                               SUB A   #,'0           ;CHECK THAT CHARACTER
        166421     200     060
      7 166423                               BCS     1$             ;IS BETWEEN 0 AND 7
        166423     045     004
      8 166425                               CMP A   #,8.
        166425     201     010
      9 166427                               BCS     2$             ;IF SO - SKIP
        166427     045     021
     10 166431                         1$:   LDA A   ESCH           ;ELSE PRINT SEQUENCE
        166431     226     177
     11 166433                               JSR     .PRINT
        166433     275     345     342
     12 166436                               LDA A   ESCL
        166436     226     200
     13 166440                               JSR     .PRINT
        166440     275     345     342
     14 166443                               CLR     ESCH           ;AND CLEAR SCANNER
        166443     177     000     177
     15 166446                               CLR     ESCL
        166446     177     000     200
     16 166451                               RTS                    ;FINISHED
        166451     071
     17                                      ;
     18 166452                         2$:   INC     CHRFLG         ;INHIBIT PRINTING
        166452     174     000     105
     19 166455                               TAB                    ;COMPUTE 3*A
        166455     026
```

```
   166456     033
21 166457                              ABA
   166457     033
22 166460                              CLR B
   166460     137
23 166461                              ADD A   #,GTBLE&377     ;COMPUTE TABLE ADDRESS
   166461     213     155
24 166463                              ADC B   #,GTBLE&177400/400
   166463     311     355
25 166465                              STA A   GTABLE+1        ;SAVE JUMP LOCATION
   166465     227     223
26 166467                              STA B   GTABLE
   166467     327     222
27 166471                              LDX     GTABLE          ;GET ADDRESS
   166471     336     222
28 166473                              LDA A   0,X             ;GET # OF CHARACTERS NEEDED
   166473     246     000
29 166475                              BEQ     3$              ;IF NONE - SKIP
   166475     047     015
30 166477                              STA A   GCNT            ;SAVE COUNT NEEDED
   166477     227     224
```

ESCAPE DISPATCHER

```
 31 166501                                LDX     #,GETLST        ;SETUP TRANSFER VECTOR
    166501     316     355     120
 32 166504                                STX     EJSR
    166504     337     175
 33 166506                                LDX     #,ESTRNG        ;ADDRESS FOR INPUT CHARACTERS
    166506     316     002     120
 34 166511                                STX     TEMPEA
    166511     337     216
 35 166513                                RTS                     ;FINISHED
    166513     071
 36 166514                          3$:   LDX     1,X             ;GET VECTOR ADRESS
    166514     356     001
 37 166516                                JMP     0,X             ;GO TO SELECTED ROUTINE
    166516     156     000
 38                                       ;
```

GET LIST ROUTINE

```
                                        .SBTTL  GET LIST ROUTINE
                                        ;
166520                          GETLST: INC     CHRFLG          ;INHIBIT PRINTING
  166520     174     000     105
166523                                  LDX     TEMPEA          ;GET ADDRESS
  166523     336     216
166525                                  LDA A   CHARA           ;GET CHARACTER
  166525     226     103
166527                                  STA A   0,X             ;STORE CHARACTER
  166527     247     000
166531                                  INX                     ;UPDATE POINTER
  166531     010
166532                                  STX     TEMPEA          ;SAVE POINTER
  166532     337     216
166534                                  DEC     GCNT            ;ENOUGH YET ?
  166534     172     000     224
166537                                  BGT     1$              ;IF NOT - SKIP
  166537     056     013
166541                                  LDX     #,ESCAPE        ;RESET TRANSFER VECTOR
  166541     316     354     340
166544                                  STX     EJSR
  166544     337     175
166546                                  LDX     GTABLE          ;GET TABLE LOCATION
  166546     336     222
166550                                  LDX     1,X             ;GET TRANSFER ADDRESS
  166550     356     001
166552                                  JMP     0,X             ;AND GO
  166552     156     000
166554                          1$:     RTS                     ;FINISHED
  166554     071
                                        ;
```

ESCAPE DISPATH TABLE

```
                                        .SBTTL  ESCAPE DISPATH TABLE
                                        ;
166555  000             GTBLE:  .BYTE   0       ;0,RAMSET
166556                          FDB     RAMSET
166556  342  247
166560  001                     .BYTE   1       ;1,GBAUD
166561                          FDB     GBAUD
166561  355  205
166563  000                     .BYTE   0       ;0,CHMODE
166564                          FDB     CHMODE
166564  356  056
166566  000                     .BYTE   0       ;0,LNMODE
166567                          FDB     LNMODE
166567  355  363
166571  021                     .BYTE   17.     ;17,DEFCHR
166572                          FDB     DEFCHR
166572  355  314
166574  002                     .BYTE   2       ;2,MOVCUR
166575                          FDB     MOVCUR
166575  355  214
166577  000                     .BYTE   0       ;0,GPHONN
166600                          FDB     GPHONN
166600  356  153
166602  000                     .BYTE   0       ;0,GPHOFF
166603                          FDB     GPHOFF
166603  356  160
                                        ;
```

ESCAPE ROUTINES

```
      1                                    .SBTTL  ESCAPE ROUTINES
      2                                    ;
      3 166605                      GBAUD: LDA A   CHARA           ;GET CHARACTER
        166605     226     103
      4 166607                             SUB A   #,'@            ;RANGE A-Z
        166607     200     100
      5 166611                             STA A   NALL            ;NUMBER CHARS PER 60TH
        166611     227     112
      6 166613                             RTS                     ;FINISHED
        166613     071
      7                                    ;
      8 166614                      MOVCUR: LDA A   MODE           ;CHECK MODE
        166614     226     102
      9 166616                             BEQ     1$             ;NOT LNMODE - SKIP
        166616     047     062
     10 166620                             LDX     #,ESTRNG       ;GET POINTER TO STRING
        166620     316     002     120
     11 166623                             LDA A   0,X            ;GET ROW ADDRESS
        166623     246     000
     12 166625                             SUB A   #,40           ;SPACE TO 7
        166625     200     040
     13 166627                             BCS     1$             ;BAD CHARACTER
        166627     045     051
     14 166631                             CMP A   #,24.
        166631     201     030
     15 166633                             BCC     1$             ;BAD CHARACTER
        166633     044     045
     16 166635                             LDA B   1,X            ;GET COLUMN ADDRESS
        166635     346     001
     17 166637                             SUB B   #,40           ;SPACE TO LITTLE O
        166637     300     040
     18 166641                             BCS     1$             ;BAD CHARACTER
        166641     045     037
     19 166643                             CMP B   #,80.
        166643     301     120
```

```
   166645     044     033
21 166647                          STA B   1,X             ;SAVE COLUMN
   166647     347     001
22 166651                          CLR B
   166651     137
23 166652                          BSR     .MUL20          ;GO COMPUTE 20*<B,A>
   166652     215     027
24 166654                          STA A   GLCNT+1         ;SAVE RESULT
   166654     227     143
25 166656                          STA B   GLCNT
   166656     327     142
26 166660                          BSR     .MUL4           ;COMPUTE 4*(20**<B,A>)
   166660     215     025
27 166662                          ADD A   GLCNT+1         ;TOTAL = 120*<B,A>
   166662     233     143
28 166664                          ADC B   GLCNT
   166664     331     142
29 166666                          STA A   GLCNT+1         ;SAVE RESULT
   166666     227     143
30 166670                          STA B   GLCNT
   166670     327     142
```

ESCAPE ROUTINES

```
     31 166672                                LDA A   1,X             ;GET COLUMN
        166672     246     001
     32 166674                                STA A   GPCNT           ;NEW COLUMN
        166674     227     144
     33 166676                                CLR B
        166676     137
     34 166677                                JSR     UPDGL           ;UPDATE POSITIONS
        166677     275     347     146
     35 166702                           1$:  RTS                     ;FINISHED
        166702     071
     36                                       ;
```

ESCAPE ROUTINES

```
      1                                     ;
      2 166703                      .MUL20: ASL A                   ;20*<B,A>
        166703     110
      3 166704                              ROL B
        166704     131
      4 166705                              ASL A
        166705     110
      5 166706                              ROL B
        166706     131
      6 166707                      .MUL4:  ASL A                   ;4*<B,A>
        166707     110
      7 166710                              ROL B
        166710     131
      8 166711                              ASL A
        166711     110
      9 166712                              ROL B
        166712     131
     10 166713                              RTS
        166713     071
     11                                     ;
     12 166714                      DEFCHR: LDX     #,ESTRNG        ;POINTER TO STRING
        166714     316     002     120
     13 166717                              LDA A   0,X             ;FIRST CHARACTER
        166717     246     000
     14 166721                              INX                     ;UPDATE PONTER
        166721     010
     15 166722                              STX     TEMPEA          ;DEFINITION ADDRESS
        166722     337     216
     16 166724                              CLR B                   ;COMPUTE ADDRESS OF CHARACTER IN
RAM
        166724     137
     17 166725                              BSR     .MUL20          ;16 BYTES PER CHARACTER
        166725     215     354
     18 166727                              ADD A   #,CHRAM&377     ;ADD IN CHRAM BASE ADDRESS
        166727     213     000
     19 166731                              ADC B   #,CHRAM&177400/400
```

```
   166731     311     020
20 166733                          STA A   TEMPEB+1        ;SAVE ADDRESS
   166733     227     221
21 166735                          STA B   TEMPEB
   166735     327     220
22 166737                          LDA A   #,20            ;16 BYTE PER CHARACTER
   166737     206     020
23 166741                    1$:   LDX     TEMPEA          ;GET ADDRESS
   166741     336     216
24 166743                          LDA B   0,X             ;AND DEFINITION
   166743     346     000
25 166745                          INX                     ;UPDATE POINTER
   166745     010
26 166746                          STX     TEMPEA
   166746     337     216
27 166750                          LDX     TEMPEB          ;GET RAM ADDRESS
   166750     336     220
28 166752                          STA B   0,X             ;STORE DEFINITION
   166752     347     000
29 166754                          INX                     ;UPDATE ADDRESS
   166754     010
30 166755                          STX     TEMPEB
```

ESCAPE ROUTINES

```
           166755     337     220
        31 166757                                DEC A
           166757     112
        32 166760                                BGT     1$              ;LOOP UNTIL FINISHED
           166760     056     357
        33 166762                                RTS                     ;FINISHED
           166762     071
        34                                       ;
```

LNMODE HANDLER

```
      1                                       .SBTTL  LNMODE HANDLER
      2                                       ;
      3 166763                          LNMODE: JSR     UPDLIN          ;UPDATE LINE
        166763     275     346     317
      4 166766                                  LDX     CLPNTR          ;GET LINE POINTER
        166766     336     126
      5 166770                                  LDA A   #,81.           ;80. CHARACTERS + CURSOR
        166770     206     121
      6 166772                                  STA A   0,X             ;[X]=CLPNTR
        166772     247     000
      7 166774                                  JSR     CLEAR           ;CLEAR LINE
        166774     275     346     300
      8 166777                                  JSR     UPDTXT          ;UPDATE TEXT
        166777     275     343     103
      9 167002                                  LDX     CLBASE          ;SAVE OLD BASE
        167002     336     130
     10 167004                                  STX     GLBASE
        167004     337     140
     11 167006                                  LDA A   #,23.           ;SET FOR 23 MORE LINES IN NEW PAG
E
        167006     206     027
     12 167010                          1$:     PSH A                   ;SAVE COUNT
        167010     066
     13 167011                                  JSR     UPDLIN          ;UPDATE LINE
        167011     275     346     317
     14 167014                                  LDX     CLPNTR          ;GET LINE POINTER
        167014     336     126
     15 167016                                  LDA A   #,81.           ;80. CHARACTERS + CURSOR
        167016     206     121
     16 167020                                  STA A   0,X             ;[X]=CLPNTR
        167020     247     000
     17 167022                                  JSR     CLEAR           ;CLEAR LINE
        167022     275     346     300
     18 167025                                  JSR     UPDTXT          ;UPDATE TEXT
        167025     275     343     103
     19 167030                                  JSR     TMCK            ;CHECK TIME
```

```
        167030     275     342     072
     20 167033                             PUL A
        167033     062
     21 167034                             DEC A
        167034     112
     22 167035                             BGT     1$              ;LOOP UNTIL FINISHED
        167035     056     351
     23 167037                             INC     UPDFLG          ;UPDATE DISPLAY
        167037     174     000     106
     24 167042                             JSR     HOME            ;HOME CURSOR
        167042     275     347     370
     25 167045                             LDA A   #,377           ;INDICATE NEW MODE
        167045     206     377
     26 167047                             STA A   MODE
        167047     227     102
     27 167051                             LDX     #,GMSG1         ;LOAD CONTROL LINE
        167051     316     356     136
     28 167054                             BRA     LDMODE
        167054     040     024
     29                                    ;
```

CHMODE ROUTINE

```
      1                                        .SBTTL  CHMODE ROUTINE
      2                                        ;
      3 167056                          CHMODE: LDA A   MODE            ;CHMODE NOW ?
        167056     226     102
      4 167060                                  BEQ     1$              ;DON'T RESET DISPLAY
        167060     047     015
      5 167062                                  CLR     MODE            ;ELSE SET UP
        167062     177     000     102
      6 167065                                  LDA A   #,15            ;DO A <CR>
        167065     206     015
      7 167067                                  JSR     .PRINT
        167067     275     345     342
      8 167072                                  LDA A   #,12            ;AND A <LF> INTO CHMODE
        167072     206     012
      9 167074                                  JSR     .PRINT
        167074     275     345     342
     10 167077                          1$:     LDX     #,GMSG0         ;AND UPDATE CONTROL LINE
        167077     316     356     121
     11 167102                          LDMODE: STX     FL.FRM
        167102     337     121
     12 167104                                  LDX     #,CTLINE+20
        167104     316     040     020
     13 167107                                  STX     FL.TO
        167107     337     117
     14 167111                                  LDA A   #,12.           ;12 CHARACTERS
        167111     206     014
     15 167113                                  STA A   FL.CNT
        167113     227     123
     16 167115                                  JSR     FILL            ;MOV MESSAGE
        167115     275     344     007
     17 167120                                  RTS                     ;FINISHED
        167120     071
     18                                         ;
     19 167121     040     103     110  GMSG0:  .ASCIZ  / CHAR MODE  /
        167124     101     122     040
```

GRAPHICS CONTROLS

```
      1                                    .SBTTL  GRAPHICS CONTROLS
      2                                    ;
      3 167153                       GPHONN: LDA A   #,377           ;GRAPHICS ON
        167153     206     377
      4 167155                               STA A   LSTFLG          ;SET FLAG
        167155     227     115
      5 167157                               RTS                     ;FINISHED
        167157     071
      6                                    ;
      7 167160                       GPHOFF: CLR     LSTFLG          ;GRAPHICS OFF
        167160     177     000     115
      8 167163                               RTS                     ;FINISHED
        167163     071
      9                                    ;
```

BOOTSTRAP LOADER BLOCKS

```
                                        .SBTTL  BOOTSTRAP LOADER BLOCKS
                                        ;
167164                          BKUS:   LDX     #,1$            ;POINTER TO BLOCK
  167164     316     356     172
167167                                  JMP     LOADER          ;GO LOAD KUS
  167167     176     357     100
167172                          1$:     FDB     BTKUS           ;START BLOCK
  167172     220     000
167174                                  FDB     KUSLD           ;KUS
  167174     220     010
167176                                  FDB     0               ;END OF BLOCK
  167176     000     000
                                        ;
167200                          BRK0:   LDX     #,1$            ;POINT TO BLOCK
  167200     316     356     206
167203                                  JMP     LOADER          ;GO LOAD BOOT
  167203     176     357     100
167206                          1$:     FDB     BTRK0           ;START BLOCK
  167206     357     365
167210                                  FDB     RKBOOT          ;BOOT STRAP
  167210     360     065
167212                                  FDB     0               ;END OF BLOCK
  167212     000     000
                                        ;
167214                          BRK1:   LDX     #,1$            ;POINT TO BLOCK
  167214     316     356     222
167217                                  JMP     LOADER          ;GO LOAD BOOT
  167217     176     357     100
167222                          1$:     FDB     BTRK1           ;START BLOCK
  167222     357     375
167224                                  FDB     RKBOOT          ;BOOT STRAP
  167224     360     065
167226                                  FDB     0               ;END OF BLOCK
  167226     000     000
                                        ;
```

BOOTSTRAP LOADER BLOCKS

```
     33 167260                          BRK4:   LDX     #,1$            ;POINT TO BLOCK
        167260     316     356     266
     34 167263                                  JMP     LOADER          ;GO LOAD BOOT
        167263     176     357     100
     35 167266                          1$:     FDB     BTRK4           ;START BLOCK
        167266     360     025
     36 167270                                  FDB     RKBOOT          ;BOOT STRAP
        167270     360     065
     37 167272                                  FDB     0               ;END OF BLOCK
        167272     000     000
     38                                         ;
     39 167274                          BRK5:   LDX     #,1$            ;POINT TO BLOCK
        167274     316     356     302
     40 167277                                  JMP     LOADER          ;GO LOAD BOOT
        167277     176     357     100
     41 167302                          1$:     FDB     BTRK5           ;START BLOCK
        167302     360     035
     42 167304                                  FDB     RKBOOT          ;BOOT STRAP
        167304     360     065
     43 167306                                  FDB     0               ;END OF BLOCK
        167306     000     000
     44                                         ;
     45 167310                          BRK6:   LDX     #,1$            ;POINT TO BLOCK
        167310     316     356     316
     46 167313                                  JMP     LOADER          ;GO LOAD BOOT
        167313     176     357     100
     47 167316                          1$:     FDB     BTRK6           ;START BLOCK
        167316     360     045
     48 167320                                  FDB     RKBOOT          ;BOOT STRAP
        167320     360     065
     49 167322                                  FDB     0               ;END OF BLOCK
        167322     000     000
     50                                         ;
     51 167324                          BRK7:   LDX     #,1$            ;POINT TO BLOCK
        167324     316     356     332
```

BOOTSTRAP LOADER BLOCKS

```
 64 167357                              JMP     LOADER          ;GO LOAD BOOT
    167357     176     357     100
 65 167362                      1$:     FDB     BTDY1           ;START BLOCK
    167362     360     223
 66 167364                              FDB     DYBOOT          ;BOOTSTRAP
    167364     360     233
 67 167366                              FDB     0               ;END OF BLOCK
    167366     000     000
 68                                     ;
 69 167370                      BDX0:   LDX     #,1$            ;POINT TO BLOCK
    167370     316     356     376
 70 167373                              JMP     LOADER          ;GO LOAD BOOT
    167373     176     357     100
 71 167376                      1$:     FDB     BTDX0           ;START BLOCK
    167376     361     103
 72 167400                              FDB     DXBOOT          ;BOOTSTRAP
    167400     361     123
 73 167402                              FDB     0               ;END OF BLOCK
    167402     000     000
 74                                     ;
 75 167404                      BDX1:   LDX     #,1$            ;POINT TO BLOCK
    167404     316     357     012
 76 167407                              JMP     LOADER          ;GO LOAD BOOT
    167407     176     357     100
 77 167412                      1$:     FDB     BTDX1           ;START BLOCK
    167412     361     113
 78 167414                              FDB     DXBOOT          ;BOOTSTRAP
    167414     361     123
 79 167416                              FDB     0               ;END OF BLOCK
    167416     000     000
 80                                     ;
 81 167420                      BCT0:   LDX     #,1$            ;POINT TO BLOCK
    167420     316     357     026
 82 167423                              JMP     LOADER          ;GO LOAD BOOT
    167423     176     357     100
```

BOOTSTRAP LOADER BLOCKS

```
 95 167456                          1$:     FDB     BTMT            ;START BLOCK
    167456     362     277
 96 167460                                  FDB     MTBOOT          ;BOOTSTRAP
    167460     362     307
 97 167462                                  FDB     0               ;END OF BLOCK
    167462     000     000
 98                                         ;
 99 167464                          BTT0:   LDX     #,1$            ;POINT TO BLOCK
    167464     316     357     072
100 167467                                  JMP     LOADER          ;GO LOAD BOOT
    167467     176     357     100
101 167472                          1$:     FDB     BTTT0           ;START BLOCK
    167472     361     363
102 167474                                  FDB     TTBOOT          ;BOOTSTRAP
    167474     361     373
103 167476                                  FDB     0               ;END OF BLOCK
    167476     000     000
104                                         ;
```

LOADER PROGRAM

```
                                        .SBTTL  LOADER PROGRAM
                                        ;
                                        ;       THE BOOT FUNCTIONS ARE PART OF THE LOCAL MONITOR
                                        ;       PROGRAM AND ADHERES TO ITS PROTOCAL
                                        ;
167500                          LOADER: STX     LDRPKX          ;SAVE POINTER TO TRANSFER BLOCK
167500  337  235
167502                                  BSR     SETNPR          ;SET UP ALL NPR REGISTERS
167502  215  047
167504                                  LDA A   PCRB11          ;CAUSE A POWER FAIL INTERRUPT
167504  226  053
167506                                  AND A   #,367
167506  204  367
167510                                  STA A   PCRB11
167510  227  053
167512                                  LDX     #,0             ;WAIT LOOP
167512  316  000  000
167515                               1$:        INX
167515  010
167516                                  BNE     1$
167516  046  375
167520                                  LDA A   PCRA11          ;ENABLE FORCED NPR'S
167520  226  052
167522                                  AND A   #,367
167522  204  367
167524                                  STA A   PCRA11
167524  227  052
167526                                  JSR     TRNSFR          ;TRANSFER DATA
167526  275  357  263
167531                                  LDA A   PCRA11          ;DISABLE FORCED NPR'S
167531  226  052
167533                                  ORA A   #,10
167533  212  010
167535                                  STA A   PCRA11
167535  227  052
```

NPR REGISTER SETUP

```
      1                                     .SBTTL  NPR REGISTER SETUP
      2                                     ;
      3 167553                        SETNPR: LDX     #,PHOTO         ;POINTER TO NPR REGISTERS
        167553     316     000     044
      4 167556                                CLR     3,X             ;CLEAR NPR CONTROL
        167556     157     003
      5 167560                                LDA A   #,37            ;SET DIRECTIONS
        167560     206     037
      6 167562                                STA A   1,X
        167562     247     001
      7 167564                                LDA A   #,56            ;SET CONTROL
        167564     206     056
      8 167566                                STA A   3,X
        167566     247     003
      9 167570                                CLR     1,X             ;PRESET TO NOP
        167570     157     001
     10 167572                                LDA B   #,377           ;FOR DIRECTION CONTROL
        167572     306     377
     11 167574                                BSR     ADDR            ;SET UP ADDRESS REGISTERS
        167574     215     013
     12 167576                                BSR     ADDR
        167576     215     011
     13 167600                                BSR     DATO            ;SET UP OUTPUT REGISTERS
        167600     215     025
     14 167602                                BSR     DATO
        167602     215     023
     15 167604                                BSR     DATI            ;SET UP INPUT REGISTERS
        167604     215     037
     16 167606                                BSR     DATI
        167606     215     035
     17 167610                                RTS                     ;FINISHED WITH SET UP
        167610     071
     18                                     ;
     19 167611                        ADDR:   LDA A   6,X             ;MASK ALL EXCEPT C1 CONTROLS
        167611     246     006
```

NPR REGISTER SETUP

```
   167635     347     006
32 167637                              ORA A   #,4             ;SET CONTROLS
   167637     212     004
33 167641                              STA A   10,X
   167641     247     010
34 167643                              INX
   167643     010
35 167644                              RTS                     ;FINISHED
   167644     071
36                                     ;
37 167645                      DATI:   LDA A   12,X            ;MASK ALL EXCEPT C1 CONTROLS
   167645     246     012
38 167647                              AND A   #,3
   167647     204     003
39 167651                              STA A   12,X
   167651     247     012
40 167653                              CLR     10,X            ;SET DIRECTION - IN
   167653     157     010
41 167655                              ORA A   #,4             ;SET CONTROLS
   167655     212     004
42 167657                              STA A   12,X
   167657     247     012
43 167661                              INX
   167661     010
44 167662                              RTS                     ;FINISHED
   167662     071
45                                     ;
```

NPR TRANSFER ROUTINE

```
      1                                     .SBTTL  NPR TRANSFER ROUTINE
      2                                     ;
      3 167663                      TRNSFR: LDX     LDRPKX          ;GET ADDRESS OF FIRST BLOCK
        167663     336     235
      4 167665                              LDX     0,X
        167665     356     000
      5 167667                              BEQ     2$              ;IF = 0 THEN FINISHED
        167667     047     073
      6 167671                              STX     TRNPKX          ;SAVE POINTER
        167671     337     237
      7 167673                              LDX     0,X             ;FIRST ELEMENT IS WORD COUNT
        167673     356     000
      8 167675                              BEQ     2$              ;IF = 0 THEN FINISHED
        167675     047     065
      9 167677                              STX     TRNCNT          ;SAVE COUNT
        167677     337     241
     10 167701                              LDX     TRNPKX          ;NOW GET NPR ADDRESS
        167701     336     237
     11 167703                              LDX     2,X
        167703     356     002
     12 167705                              STX     NPRADA          ;STORE IN NPR ADDRESS REGISTER
        167705     337     050
     13 167707                              LDX     TRNPKX          ;UPDATE TRNPKX TO DATA
        167707     336     237
     14 167711                              INX
        167711     010
     15 167712                              INX
        167712     010
     16 167713                              INX
        167713     010
     17 167714                              INX
        167714     010
     18 167715                              STX     TRNPKX
        167715     337     237
     19 167717                      1$:     LDX     TRNPKX          ;GET A WORD OF DATA
```

NPR TRANSFER ROUTINE

```
                167742     010
     31 167743                                STX     NPRADA
                167743     337     050
     32 167745                                LDX     TRNCNT          ;UPDATE WORD COUNT
                167745     336     241
     33 167747                                DEX
                167747     011
     34 167750                                STX     TRNCNT
                167750     337     241
     35 167752                                BNE     1$              ;LOOP UNTIL ALL WORDS TRANSFERRED
                167752     046     343
     36 167754                                LDX     LDRPKX          ;UPDATE BLOCK POINTER
                167754     336     235
     37 167756                                INX
                167756     010
     38 167757                                INX
                167757     010
     39 167760                                STX     LDRPKX
                167760     337     235
     40 167762                                BRA     TRNSFR          ;LOOP FOR MORE
                167762     040     277
     41 167764                           2$:  RTS                     ;FINISHED
                167764     071
     42                                       ;
```

RK11-D/RK05 BOOTSTRAP

```
                                        .SBTTL  RK11-D/RK05 BOOTSTRAP
                                        ;
                                        ;A SECTION FOR EACH POSSIBLE RK05 DEVICE
                                        ;
167765                          BTRK0:  FDB     2       ;2 WORD AREA
  167765    000     002
167767                                  FDB     24      ;POWER FAIL RESTART VECTOR
  167767    000     024
167771                                  FDB     157500  ;START ADDRESS
  167771    337     100
167773                                  FDB     340     ;PRIORITY
  167773    000     340
                                                        ;
167775                          BTRK1:  FDB     2       ;2 WORD AREA
  167775    000     002
167777                                  FDB     24      ;POWER FAIL RESTART VECTOR
  167777    000     024
170001                                  FDB     157506  ;START ADDRESS
  170001    337     106
170003                                  FDB     340     ;PRIORITY
  170003    000     340
                                                        ;
170005                          BTRK2:  FDB     2       ;2 WORD AREA
  170005    000     002
170007                                  FDB     24      ;POWER FAIL RESTART VECTOR
  170007    000     024
170011                                  FDB     157514  ;START ADDRESS
  170011    337     114
170013                                  FDB     340     ;PRIORITY
  170013    000     340
                                                        ;
170015                          BTRK3:  FDB     2       ;2 WORD AREA
  170015    000     002
170017                                  FDB     24      ;POWER FAIL RESTART VECTOR
  170017    000     024
```

RK11-D/RK05 BOOTSTRAP

```
     34                                                         ;
     35 170045                        BTRK6:  FDB     2         ;2 WORD AREA
        170045     000     002
     36 170047                                FDB     24        ;POWER FAIL RESTART VECTOR
        170047     000     024
     37 170051                                FDB     157544    ;START ADDRESS
        170051     337     144
     38 170053                                FDB     340       ;PRIORITY
        170053     000     340
     39                                                         ;
     40 170055                        BTRK7:  FDB     2         ;2 WORD AREA
        170055     000     002
     41 170057                                FDB     24        ;POWER FAIL RESTART VECTOR
        170057     000     024
     42 170061                                FDB     157552    ;START ADDRESS
        170061     337     152
     43 170063                                FDB     340       ;PRIORITY
        170063     000     340
     44                                                         ;
     45 170065                        RKBOOT: FDB     41.       ;WORDS IN BOOTSTRAP
        170065     000     051
     46 170067                                FDB     157500    ;LOAD ADDRESS
        170067     337     100
     47                                                         ;
     48 170071                        FDB     012700  000000    ;RK0:   MOV     #0*20000,R0     ;UNIT 0
        170071     025     300
        170073     000     000
     49 170075                        FDB     000424            ;BR     RKBOOT
        170075     001     024
     50 170077                        FDB     012700  020000    ;RK1:   MOV     #1*20000,R0     ;UNIT 1
        170077     025     300
        170101     040     000
     51 170103                        FDB     000421            ;BR     RKBOOT
        170103     001     021
     52 170105                        FDB     012700  040000    ;RK2:   MOV     #2*20000,R0     ;UNIT 2
```

RK11-D/RK05 BOOTSTRAP

```
     61 170141                          FDB     000402          ;BR     RKBOOT
        170141     001     002
     62 170143                          FDB     012700  160000  ;RK7:   MOV     #7*20000,R0     ;UNIT Y
        170143     025     300
        170145     340     000
     63                                                         ;
     64 170147                          FDB     105737  177404  ;RKBOOT:TSTB    @#177404        ;CONTROL
READY ?
        170147     213     337
        170151     377     004
     65 170153                          FDB     100375          ;       BPL     1$              ;NO - LOO
P
        170153     200     375
     66 170155                          FDB     010037  177412  ;       MOV     R0,@#177412     ;SELECT D
RIVE
        170155     020     037
        170157     377     012
     67 170161                          FDB     122737  000300  ;2$:    CMPB    #300,@#177400   ;CHECK DR
IVE STATUS
        170161     245     337
        170163     000     300
     68 170165                          FDB             177400  ;
        170165     377     000
     69 170167                          FDB     101374          ;       BHI     2$              ;LOOP UNT
IL READY
        170167     202     374
     70 170171                          FDB     012700  177406  ;       MOV     #177406,R0      ;WORD COU
NT REGISTER
        170171     025     300
        170173     377     006
     71 170175                          FDB     012710  177400  ;       MOV     #-256.,(R0)     ;1 BLOCK
        170175     025     310
        170177     377     000
     72 170201                          FDB     012740  000005  ;       MOV     #5,-(R0)        ;READ BLO
CK
        170201     025     340
        170203     000     005
     73 170205                          FDB     105710          ;3$:    TSTB    (R0)            ;LOADED ?
        170205     213     310
```

```
                                        .SBTTL  DY BOOTSTRAP
                                        ;
                                        ;       THIS SECTION TO BOOT DY0:
                                        ;
170213                          BTDY0:  FDB     2       ;TWO WORD AREA
  170213    000     002
170215                                  FDB     24      ;POWER FAIL RESTART VECTOR
  170215    000     024
170217                                  FDB     157500  ;START ADDRESS
  170217    337     100
170221                                  FDB     340     ;PRIORITY
  170221    000     340
                                        ;
                                        ;
                                        ;       THIS SECTION TO BOOT DY1:
                                        ;
170223                          BTDY1:  FDB     2       ;TWO WORD AREA
  170223    000     002
170225                                  FDB     24      ;POWER FAIL RESTART ADDRESS
  170225    000     024
170227                                  FDB     157506  ;START ADDRESS
  170227    337     106
170231                                  FDB     340     ;PRIORITY
  170231    000     340
                                        ;
                                        ;
                                        ;       DY BOOTSTRAP ROUTINE
                                        ;
170233                          DYBOOT: FDB     82.     ;WORDS IN BOOTSTRAP
  170233    000     122
170235                                  FDB     157500  ;LOAD ADDRESS
  170235    337     100
                                        ;
                                        ;                               DYCSR   =177160
                                        ;                               DYDAT   =177162
```

```
        170263     025     303
        170265     000     001
     38 170267                          FDB     012705  000200          ;       MOV     #DYTR,R5
        170267     025     305
        170271     000     200
     39 170273                          FDB     032710  000040          ;1$:    BIT     #DYDONE,(
R0)
        170273     065     310
        170275     000     040
     40 170277                          FDB     001775                  ;       BEQ     1$
        170277     003     375
     41 170301                          FDB     010410                  ;       MOV     R4,(R0)
        170301     021     010
     42 170303                          FDB     042704  177757          ;       BIC     #177757,R
4
        170303     105     304
        170305     377     357
     43 170307                          FDB     062704  000007          ;       ADD     #7,R4
        170307     145     304
        170311     000     007
     44 170313                          FDB     032710  000040          ;2$:    BIT     #DYDONE,(
R0)
        170313     065     310
        170315     000     040
     45 170317                          FDB     001775                  ;       BEQ     2$
        170317     003     375
     46 170321                          FDB     032711  000040          ;       BIT     #40,(R1)
        170321     065     311
        170323     000     040
     47 170325                          FDB     001402                  ;       BEQ     3$
        170325     003     002
     48 170327                          FDB     062704  000400          ;       ADD     #DYDENS,R
4
        170327     145     304
        170331     001     000
     49 170333                          FDB     010410                  ;3$:    MOV     R4,(R0)
        170333     021     010
```

```
        170367     061     110
     61 170371                          FDB     001776              ;       BEQ     7$
        170371     003     376
     62 170373                          FDB     032704  000400      ;       BIT     #DYDENS,R
4
        170373     065     304
        170375     001     000
     63 170377                          FDB     001003              ;       BNE     8$
        170377     002     003
     64 170401                          FDB     012711  000100      ;       MOV     #100,(R1)
        170401     025     311
        170403     000     100
     65 170405                          FDB     000402              ;       BR      9$
        170405     001     002
     66 170407                          FDB     012711  000200      ;8$:    MOV     #200,(R1)
        170407     025     311
        170411     000     200
     67 170413                          FDB     030510              ;9$:    BIT     R5,(R0)
        170413     061     110
     68 170415                          FDB     001776              ;       BEQ     9$
        170415     003     376
     69 170417                          FDB     010211              ;       MOV     R2,(R1)
        170417     020     211
     70 170421                          FDB     032710  000040      ;10$:   BIT     #DYDONE,(
R0)
        170421     065     310
        170423     000     040
     71 170425                          FDB     001775              ;       BEQ     10$
        170425     003     375
     72 170427                          FDB     005203              ;       INC     R3
        170427     012     203
     73 170431                          FDB     005203              ;       INC     R3
        170431     012     203
     74 170433                          FDB     052704  000004      ;       BIS     #4,R4
        170433     125     304
        170435     000     004
```

DY BOOTSTRAP

```
    84 170473                              FDB     005200                  ;       INC     R0
       170473     012     200
    85 170475                              FDB     012701  177160          ;12$:   MOV     #DYCSR,R1
       170475     025     301
       170477     376     160
    86 170501                              FDB     005007                  ;       CLR     PC
       170501     012     007
    87                                     ;
```

DX BOOTSTRAP

```
                                        .SBTTL  DX BOOTSTRAP
                                        ;
                                        ;
                                        ;       THIS SECTION TO BOOT DX0:
                                        ;
170503                          BTDX0:  FDB     2       ;TWO WORD AREA
170503  000     002
170505                                  FDB     24      ;POWER FAIL RESTART VECTOR
170505  000     024
170507                                  FDB     157600  ;START ADDRESS
170507  337     200
170511                                  FDB     340     ;PRIORITY
170511  000     340
                                        ;
                                        ;
                                        ;       THIS SECTION TO BOOT DX1:
                                        ;
170513                          BTDX1:  FDB     2       ;TWO WORD TRANSFER
170513  000     002
170515                                  FDB     24      ;POWER FAIL RESTART VECTOR
170515  000     024
170517                                  FDB     157604  ;START ADDRESS
170517  337     204
170521                                  FDB     340     ;PRIORITY
170521  000     340
                                        ;
                                        ;
                                        ;       DX BOOTSTRAP ROUTINE
                                        ;
170523                          DXBOOT: FDB     35.             ;WORDS IN BOOTSTRAP
170523  000     043
170525                                  FDB     157600          ;LOAD ADDRESS
170525  337     200
                                        ;
170527                          DX0:    FDB     005002          ;       DX0:    CLR     R2
```

DX BOOTSTRAP

```
        170561     020     220
     35 170563                          FDB     000402          ;               BR      4$
        170563     001     002
     36 170565                          FDB     012710  000001  ;       3$:     MOV     #1,(R0)
        170565     025     310
        170567     000     001
     37 170571                          FDB     006203          ;       4$:     ASR     R3
        170571     014     203
     38 170573                          FDB     103402          ;               BCS     6$
        170573     207     002
     39 170575                          FDB     112711          ;               MOVB    (PC)+,(R1
)
        170575     225     311
     40 170577                          FDB     111023          ;       5$:     MOVB    (R0),(R3)
+
        170577     222     023
     41 170601                          FDB     030211          ;       6$:     BIT     R2,(R1)
        170601     060     211
     42 170603                          FDB     001776          ;               BEQ     6$
        170603     003     376
     43 170605                          FDB     100756          ;               BMI     1$
        170605     201     356
     44 170607                          FDB     103766          ;               BCS     3$
        170607     207     366
     45 170611                          FDB     105711          ;               TSTB    (R1)
        170611     213     311
     46 170613                          FDB     100771          ;               BMI     5$
        170613     201     371
     47 170615                          FDB     005000          ;               CLR     R0
        170615     012     000
     48 170617                          FDB     022710  000240  ;               CMP     #240,(R0)
        170617     045     310
        170621     000     240
     49 170623                          FDB     001347          ;               BNE     1$
        170623     002     347
     50 170625                          FDB     122702  000247  ;               CMPB    #247,R2
```

CT BOOTSTRAP

```
                                        .SBTTL  CT BOOTSTRAP
                                        ;
                                        ;
                                        ;       THIS SECTION TO BOOT CT0:
                                        ;
170635                          BTCT0:  FDB     2       ;TWO WORD AREA
170635  000     002
170637                                  FDB     24      ;POWER FAIL RESTART VECTOR
170637  000     024
170641                                  FDB     157600  ;START ADDRESS
170641  337     200
170643                                  FDB     340     ;PRIORITY
170643  000     340
                                        ;
                                        ;
                                        ;       THIS SECTION TO BOOT CT1:
                                        ;
170645                          BTCT1:  FDB     2       ;TWO WORD AREA
170645  000     002
170647                                  FDB     24      ;POWER FAIL RESTART VECTOR
170647  000     024
170651                                  FDB     157610  ;START ADDRESS
170651  337     210
170653                                  FDB     340     ;PRIORITY
170653  000     340
                                        ;
                                        ;
                                        ;       CT BOOTSTRAP ROUTINE
                                        ;
170655                          CTBOOT: FDB     33.     ;WORDS IN BOOTSTRAP
170655  000     041
170657                                  FDB     157600  ;LOAD ADDRESS
170657  337     200
                                        ;
170661                          CT0:    FDB     012700  177500          ;CT0:   MOV     #177500,R
0
```

CT BOOTSTRAP

```
     34 170715                          FDB     112110              ;LOOP1: MOVB    (R1)+,(R0
)
        170715     224     110
     35 170717                          FDB     100413              ;       BMI     DONE
        170717     201     013
     36 170721                          FDB     130310              ;LOOP2: BITB    R3,(R0)
        170721     260     310
     37 170723                          FDB     001776              ;       BEQ     LOOP2
        170723     003     376
     38 170725                          FDB     105202              ;       INCB    R2
        170725     212     202
     39 170727                          FDB     100772              ;       BMI     LOOP1
        170727     201     372
     40 170731                          FDB     116012  000002      ;       MOVB    2(R0),(R2
)
        170731     234     012
        170733     000     002
     41 170735                          FDB     120337  000000      ;       CMPB    R3,@#0
        170735     240     337
        170737     000     000
     42 170741                          FDB     001767              ;       BEQ     LOOP2
        170741     003     367
     43 170743                          FDB     000000              ;STOP:  HALT
        170743     000     000
     44 170745                          FDB     000755              ;       BR      RESTRT
        170745     001     355
     45 170747                          FDB     005710              ;DONE:  TST     (R0)
        170747     013     310
     46 170751                          FDB     100774              ;       BMI     STOP
        170751     201     374
     47 170753                          FDB     005007              ;       CLR     PC
        170753     012     007
     48                                                             ;
     49 170755                          FDB     017600              ;TABLE: .WORD   17600
        170755     037     200
     50 170757                          FDB     002415              ;       .WORD   2415
```

PAPER TAPE ABSOLUTE LOADER

```
      1                                     .SBTTL  PAPER TAPE ABSOLUTE LOADER
      2                                     ;
      3                                     ;       PAPER TAPE
      4                                     ;
      5                                     ;
      6 170763                        BTTT0:  FDB     2       ;TWO WORDS
        170763     000     002
      7 170765                                FDB     24      ;POWER FAIL VECTOR
        170765     000     024
      8 170767                                FDB     157500  ;STARTING ADDRESS
        170767     337     100
      9 170771                                FDB     340     ;PRIORITY
        170771     000     340
     10                                     ;
     11 170773                        TTBOOT: FDB     96.     ;BOOT WORDS
        170773     000     140
     12 170775                                FDB     157500  ;LOAD ADDRESS
        170775     337     100
     13                                     ;
     14                                     ;                                       .=157500
     15                                     ;
     16                                     ;       177570                          L.SR    =177570
     17                                     ;
     18                                     ;       000000                          L.CKSM  =R0
     19                                     ;       000001                          L.ADR   =R1
     20                                     ;       000002                          L.BC    =R2
     21                                     ;       000003                          L.BYT   =R3
     22                                     ;       000005                          L.PTR   =R5
     23                                     ;
     24 170777                        TT0:    FDB     010706                  ;L.LD1: MOV     PC,SP
        170777     021     306
     25 171001                                FDB     024646                  ;       CMP     -(SP),-(S
P)
        171001     051     246
     26 171003                                FDB     010705                  ;       MOV     PC,L.PTR
        171003     021     305
```

PAPER TAPE ABSOLUTE LOADER

```
     37 171035                              FDB     005000              ;L.LD2: CLR     L.CKSM
        171035     012     000
     38 171037                              FDB     004715              ;       JSR     PC,@L.PTR
        171037     011     315
     39 171041                              FDB     105303              ;       DECB    L.BYT
        171041     212     303
     40 171043                              FDB     001374              ;       BNE     L.LD2
        171043     002     374
     41 171045                              FDB     004715              ;       JSR     PC,@L.PTR
        171045     011     315
     42 171047                              FDB     004767  000074      ;       JSR     PC,L.GWRD
        171047     011     367
        171051     000     074
     43 171053                              FDB     010402              ;       MOV     R4,L.BC
        171053     021     002
     44 171055                              FDB     162702  000004      ;       SUB     #4,L.BC
        171055     345     302
        171057     000     004
     45 171061                              FDB     022702  000002      ;       CMP     #2,L.BC
        171061     045     302
        171063     000     002
     46 171065                              FDB     001443              ;       BEQ     L.JMP
        171065     003     043
     47 171067                              FDB     004767  000054      ;       JSR     PC,L.GWRD
        171067     011     367
        171071     000     054
     48 171073                              FDB     061604              ;       ADD     (SP),R4
        171073     143     204
     49 171075                              FDB     010401              ;       MOV     R4,L.ADR
        171075     021     001
     50 171077                              FDB     004715              ;L.LD3: JSR     PC,@L.PTR
        171077     011     315
     51 171101                              FDB     002004              ;       BGE     L.LD4
        171101     004     004
     52 171103                              FDB     105700              ;       TSTB    L.CKSM
```

PAPER TAPE ABSOLUTE LOADER

```
        171133     000     002
     63 171135                          FDB     060300               ;       ADD     L.BYT,L.C
KSM
        171135     140     300
     64 171137                          FDB     042703  177400       ;       BIC     #177400,L
.BYT
        171137     105     303
        171141     377     000
     65 171143                          FDB     005302               ;       DEC     L.BC
        171143     012     302
     66 171145                          FDB     000207               ;       RTS     PC
        171145     000     207
     67 171147                          FDB     012667  000052       ;L.GWRD:        MOV     (
SP)+,L.TMP
        171147     025     267
        171151     000     052
     68 171153                          FDB     004715               ;       JSR     PC,(L.PTR
)
        171153     011     315
     69 171155                          FDB     010304               ;       MOV     L.BYT,R4
        171155     020     304
     70 171157                          FDB     004715               ;       JSR     PC,(L.PTR
)
        171157     011     315
     71 171161                          FDB     000303               ;       SWAB    L.BYT
        171161     000     303
     72 171163                          FDB     042703  000377       ;       BIC     #377,L.BY
T
        171163     105     303
        171165     000     377
     73 171167                          FDB     050304               ;       BIS     L.BYT,R4
        171167     120     304
     74 171171                          FDB     016707  000030       ;       MOV     L.TMP,PC
        171171     035     307
        171173     000     030
     75 171175                          FDB     004767  177746       ;L.JMP: JSR     PC,L.GWRD
        171175     011     367
        171177     377     346
```

PAPER TAPE ABSOLUTE LOADER

```
        171233     000     000
        171235     000     000
        171237     000     000
     88 171241                             FDB     000000
        171241     000     000
     89                                    ;
     90                                    ;       157744                          .=157744
     91                                    ;
     92 171243                             FDB     016701  000026          ;BSLDR: MOV     DEVICE,R1
        171243     035     301
        171245     000     026
     93 171247                             FDB     012702  000352          ;LOOP:  MOV     #.-L.LD1+
2,R2
        171247     025     302
        171251     000     352
     94 171253                             FDB     005211                  ;ENABLE:        INC     (
R1)
        171253     012     211
     95 171255                             FDB     105711                  ;WAIT:  TSTB    (R1)
        171255     213     311
     96 171257                             FDB     100376                  ;       BPL     WAIT
        171257     200     376
     97 171261                             FDB     116162  000002  157400  ;       MOVB    2(R1),LOA
D(R2)
        171261     234     162
        171263     000     002
        171265     337     000
     98 171267                             FDB     005267  177756          ;       INC     LOOP+2
        171267     012     267
        171271     377     356
     99 171273                             FDB     000765                  ;BRNCH: BR      LOOP
        171273     001     365
    100 171275                             FDB     177560                  ;DEVICE:        .WORD   1
77560
        171275     377     160
    101                                                                    ;
```

MT BOOTSTRAP

```
      1                                     .SBTTL  MT BOOTSTRAP
      2                                     ;
      3                                     ;
      4                                     ;       THIS SECTION FOR MT BOOT
      5                                     ;
      6 171277                        BTMT:   FDB     2       ;2 WORD SECTION
        171277     000     002
      7 171301                                FDB     24      ;POWER FAIL RESTART ADDRESS
        171301     000     024
      8 171303                                FDB     157500  ;START ADDRESS
        171303     337     100
      9 171305                                FDB     340     ;PRIORITY
        171305     000     340
     10                                     ;
     11                                     ;
     12                                     ;       MT BOOTSTRAP
     13                                     ;
     14 171307                       MTBOOT: FDB     96.     ;WORDS IN BOOT
        171307     000     140
     15 171311                               FDB     157500  ;LOAD ADDESS
        171311     337     100
     16                                     ;
     17 171313                       MT0:    FDB     000000          ;HALT AT STARTUP
        171313     000     000
     18 171315                               FDB     010706 005746 012700 000212 060600 012701 172522
013746
        171315     021     306
        171317     013     346
        171321     025     300
        171323     000     212
        171325     141     200
        171327     025     301
        171331     365     122
        171333     027     346
     19 171335                               FDB     177570 006016 103401 005016 006316 005011 105711
100376
```

MT BOOTSTRAP

```
        171407     377     362
        171411     025     341
        171413     140     003
     22 171415                              FDB     105711 100376 020210 001410 012761 177777 000002
012711
        171415     213     311
        171417     200     376
        171421     040     210
        171423     003     010
        171425     025     361
        171427     377     377
        171431     000     002
        171433     025     311
     23 171435                              FDB     060011 105711 100376 000756 022020 005710 001417
022121
        171435     140     011
        171437     213     311
        171441     200     376
        171443     001     356
        171445     044     020
        171447     013     310
        171451     003     017
        171453     044     121
     24 171455                              FDB     014011 061611 016041 000002 012741 060003 105711
100376
        171455     030     011
        171457     143     211
        171461     034     041
        171463     000     002
        171465     025     341
        171467     140     003
        171471     213     311
        171473     200     376
     25 171475                              FDB     005740 005711 100337 011100 000000 000776 014002
005740
        171475     013     340
        171477     013     311
```

MT BOOTSTRAP

```
     28 171555                                  FDB     004411 040520 000001 000001 000001 000001 000001
000001

        171555     011     011

        171557     101     120

        171561     000     001

        171563     000     001

        171565     000     001

        171567     000     001

        171571     000     001

        171573     000     001

     29 171575                                  FDB     000001 000001 000001 000001 000001 000001 000001

        171575     000     001

        171577     000     001

        171601     000     001

        171603     000     001

        171605     000     001

        171607     000     001

        171611     000     001

     30                                         ;
```

KUSARB PROGRAM

```
      1                                     .SBTTL  KUSARB PROGRAM
      2                                     ;
      3         110000                      .=110000        ;ROM AREA FOR KUSARB
      4                                     ;
      5 110000                      BTKUS:  FDB     2       ;2 WORD AREA
        110000     000     002
      6 110002                              FDB     24      ;POWER FAIL RESTART ADDRESS
        110002     000     024
      7 110004                              FDB     150000  ;START ADDRESS
        110004     320     000
      8 110006                              FDB     340     ;PRIORITY
        110006     000     340
      9                                     ;
     10                                     ;       KUSARB UTILLITY PROGRAM
     11                                     ;
     12 110010                      KUSLD:  FDB     3750    ;PROGRAM LENGTH
        110010     007     350
     13 110012                              FDB     150000  ;LOAD ADDRESS
        110012     320     000
     14                                     ;
     15 110014                      KUSARB:                 ;BEGINNING OF PROGRAM
    272                                     ;
    273         000001                      .END
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35 Page 81-1

Symbol table

```
ADDR    167611          CHROM1= 100000          ESCH    000177          KBINT   162154          PBYTE   1
66042

ADD1    163514          CHROM2= 104000          ESCL    000200          KBSET   160546          PCOMB   1
66241

ADD80   163501          CLBASE  000130          ESCP    166310          KBUTN = 000035          PCRA1 = 0
00002

BAUD    166114          CLBOT = 020120          ESTRNG= 001120          KUSARB  110014          PCRA10= 0
00046

BCRSTK  000156          CLEAR   163300          EVAL    165750          KUSLD   110010          PCRA11= 0
00052

BCT0    167420          CLKINT  161030          EVNSCN= 030000          LCKDSP  162464          PCRA12= 0
00056

BCT1    167434          CLKO    160004          FILL    162007          LCLH    000203          PCRA13= 0
00062

BDX0    167370          CLPBOT= 006100          FL.CNT  000123          LCLL    000204          PCRA2 = 0
00006

BDX1    167404          CLPNTR  000126          FL.FRM  000121          LCLMON  164004          PCRA3 = 0
00012

BDY0    167340          CLPTOP= 007731          FL.TO   000117          LDDATA  161446          PCRA4 = 0
00016

BDY1    167354          CLRSCN  163717          FREVER  161024          LDMODE  167102          PCRA5 = 0
00022

BKUS    167164          CLSRT = 000650          GBAUD   166605          LDRPKX  000235          PCRA6 = 0
00026

BMT0    167450          CLSTP = 000656          GCNT    000224          LIBIC   164632          PCRA7 = 0
00032

BREAK   161744          CLTOP = 027657          GCTL    166412          LIBIS   164621          PCRA8 = 0
00036

BRKINT  161737          CL.TOP= 027660          GETLST  166520          LINE    163376          PCRA9 = 0
00042

BRKSET  160475          COMINT  162156          GETS    164141          LINES   000145          PCRB1 = 0
00003

BRK0    167200          CRSTK   000160          GETSTR  164110          LJSR    000201          PCRB10= 0
00047

BRK1    167214          CRTRN   163467          GLBASE  000140          LMSG0   164573          PCRB11= 0
00053

BRK2    167230          CSL   = 000010          GLCNT   000142          LMSG1   164577          PCRB12= 0
00057

BRK3    167244          CTBOOT  170655          GMSG0   167121          LMSG2   164610          PCRB13= 0
00063

BRK4    167260          CTLC    165567          GMSG1   167136          LMSG3   164615          PCRB2 = 0
00007

BRK5    167274          CTLINE= 020000          GPBOT = 030000          LNMODE  166763          PCRB3 = 0
00013

BRK6    167310          CTLP    162666          GPCNT   000144          LOADER  167500          PCRB4 = 0
00017
```

Symbol table

```
RLPDN   162411          SETSCN  160675          TEMPGA  000225          TT0     170777          VARSAV= 0
00243

RLPUP   162265          SHIFT   166007          TEMPGB  000227          TXGRCD= 000400          VERTIN  1
61540

RMTCHK  163006          SRCHTB  164240          TEMPLA  000205          TXTCBS  000146          VH      0
00213

RMTFLS  000174          STACK = 006077          TEMPLB  000207          TXTSTP= 000642          VL      0
00214

RMTMOV  162605          START   160000          TEMPLC  000211          T.CHAR  000111          WDFBA = 0
00020

RMTON   000101          SUB1    163521          TERMDI  162055          T.SCND  000110          WDFBB = 0
00021

RMTSET  160723          SUB80   163506          TIMCLK  161104          T.60TH  000107          WDTBA = 0
00024

RMTWT   162763          SYSMSG  164676          TIME    165532          UPDFLG  000106          WDTBB = 0
00025

ROTDWN  161431          SYST    164665          TMCHEK  161053          UPDGL   163546          XTABLE= 0
01230

ROTUP   161407          TABLE   164302          TMCK    161072          UPDGP   163524          YTABLE= 0
01200

RPAGE   163646          TDF11 = 000030          TRNCNT  000241          UPDLIN  163317          .MUL20  1
66703

RPINT   162152          TDT11 = 000031          TRNPKX  000237          UPDTXT  161503          .MUL4   1
66707

SCRLDN  162331          TEMPEA  000216          TRNSFR  167663          UPSCRL  162243          .PRINT  1
62742

SCRLUP  162175          TEMPEB  000220          TTBOOT  170773          VAL     000215          ...A  = 1
77677

SETNPR  167553


. ABS.  177776    000   (RW,I,GBL,ABS,OVR)

        000000    001   (RW,I,LCL,REL,CON)

Errors detected:  0


*** Assembler statistics



Work  file  reads: 0

Work  file writes: 0

Size of work file: 15214 Words  ( 60 Pages)

Size of core pool: 18176 Words  ( 71 Pages)

Operating  system: RT-11
```

Elapsed time: 00:00:07.02

DISP[40],DISP=M6800,DISP1,DISP2

```
R MACRO
GRP[40],GRP=M6800,GRAPH1,GRAPH2,GRAPH3,GRAPH4
```

```
        .SBTTL  6800 TERMINAL SOFTWARE
        ;
        ; THE SOFTWARE CONTAINED IN THIS PROGRAM IS FOR THE
        ;OPERATION OF THE 6800 BASED VIDEO BOARD USED
        ;WITH THE ARB-11 PROCESSOR.
        ; THE SOFTWARE SUPPORTS A STANDARD DL-11 TYPE
        ;INTERFACE FORMAT PLUS A SEPERATE INTERFACE FOR
        ;THE GRAPHICS PROCESSOR.
        ;
        ;       TERMINAL INTERFACE
        ;               KEYBOARD CSR = 177560
        ;                        BUF = 177562
        ;                        VEC = 60
        ;               PRINTER  CSR = 177564
        ;                        BUF = 177566
        ;                        VEC = 64
        ;
        ;       GRAPHICS INTERFACE
        ;               RESULT   CSR = 176670
        ;                        BUF = 176672
        ;                        VEC = 170
        ;               COMMAND  CSR = 176674
        ;                        BUF = 176676
        ;                        VEC = 174
        ;
        .PAGE
        .SBTTL  DEFINITION OF I/O REGISTERS
        ;
        ; THE I/O REGISTERS ARE ALL 6820 PIA CIRCUITS
        ;
        ;       REGISTER ADDRESSING
        ; ___XX
        ;    00 I/O REGISTER A (DIRECTION OF A)
        ;    01 I/O REGISTER B (DIRECTION OF B)
        ;    10 STATUS REGISTER A
        ;    11 STATUS REGISTER B
        ;
        ;       REFER TO MC6820/6821 DATA SHEETS FOR DETAILS
        ;       OF THE CONTROL REGISTERS
        ;
        .PAGE
        .SBTTL  I/O REGISTER LOCATIONS
        ;
        DARHA=0         ;DISPLAY ADDRESS REGISTER H (A-PORT)
        DARLB=1         ;DISPLAY ADDRESS REGISTER L (B-PORT)
        PCRA1=2         ;CSR FOR DARHA [INIT  IRQ1]
        PCRB1=3         ;CSR FOR DARHB
        ;
        CARHA=4         ;CURSOR ADDRESS REGISTER H (A-PORT)
        CARLB=5         ;CURSOR ADDRESS REGISTER L (B-PORT)
        PCRA2=6         ;CSR FOR CARHA
        PCRB2=7         ;CSR FOR CARLB [VERTICAL RETRACE  NMI2]
        ;
        CSL=10          ;CURRENT SCAN LINE
        ISL=11          ;INTERRUPT SCAN LINE
        PCRA3=12        ;CSR FOR CSL
        PCRB3=13        ;CSR FOR ISL [SCAN LINE  NMI3]
        ;
        CHARPL=14       ;CHARACTERS PER LINE
        DSPCTL=15       ;DISPLAY CONTROL
                        ;BIT    0 - GO
                        ;       3 - GRAPHIC ENABLE
                        ;       4 - CONTINUOS SCAN
                        ;       5 - VIDEO POLARITY
                        ;       7 - ODD FRAME
                        ;
        PCRA4=16        ;CSR FOR CHARPL [NMI STROBE]
        PCRB4=17        ;CSR FOR DSPCTL [ODD FRAME  NMI1]
        ;
        WDFBA=20        ;WORD DATA FROM BUS <15:08>
        WDFBB=21        ;WORD DATA FROM BUS <07:00>
        PCRA5=22        ;CSR FOR WDFBA [DATA READY  IRQ8]
        PCRB5=23        ;CSR FOR WDFBB
        ;
        WDTBA=24        ;WORD DATA TO BUS <15:08>
        WDTBB=25        ;WORD DATA TO BUS <07:00>
```

```
        PCRA6=26        ;CSR FOR WDTBA [60HZ CLOCK  IRQ2]
        PCRB6=27        ;CSR FOR WDTBB [DATA READY  IRQ9]
        ;
        TDF11=30        ;TERMNAL DATA FROM ARB-11
        TDT11=31        ;TERMINAL DATA TO ARB-11
        PCRA7=32        ;CSR FOR TDF11 [IRQ6]
        PCRB7=33        ;CSR FOR TDT11 [IRQ7]
        ;
        KASCII=34       ;KEYBOARD
        KBUTN=35        ;KEYBOARD BUTTONS
                        ;BIT    0 - REPEAT
                        ;       1 - BREAK
                        ;       2 - PAPER
                        ;       3 - HEREIS
                        ;       4 - TAPE>
                        ;       5 - <TAPE
                        ;
        PCRA8=36        ;CSR FOR KASCII [IRQ5]
        PCRB8=37        ;CSR FOR KBUTN [BELL CONTROL]
        ;
        HA=40           ;PRINTER CONTROL REGISTER
        HB=41           ;PRINTER DATA REGISTER
        PCRA9=42        ;CSR FOR HA
        PCRB9=43        ;CSR FOR HB
        ;
        PHOTO=44        ;PHOTOREADER DATA
        NPRCSR=45       ;NPR CONTROL REGISTER
                        ;BIT    0 - NPR GO
                        ;       1 - BUS C0 CONTROL
                        ;       2 - BUS C1 CONTROL
                        ;       3 - ADDRESS BIT 16
                        ;       4 - ADDRESS BIT 17
                        ;       5 - NXM ERROR
                        ;       6 - BUS GRANT LATE ERROR
                        ;       7 - NPR DONE
                        ;
        PCRA10=46       ;CSR FOR PHOTO [IRQ3]
        PCRB10=47       ;CSR FOR NPR CONTROL [IRQ10]
        ;
        NPRADA=50       ;NPR ADDRESS <15:08>
        NPRADB=51       ;NPR ADDRESS <07:00>
        PCRA11=52       ;CSR FOR NPRADA [TAPE<  IRQ11]
        PCRB11=53       ;CSR FOR NPRADB [TAPE>  IRQ12]
        ;
        NPRDTA=54       ;NPR DATA TO BUS <15:08>
        NPRDTB=55       ;NPR DATA TO BUS <07:00>
        PCRA12=56       ;CSR FOR NPRDTA [HERE IS  IRQ13]
        PCRB12=57       ;CSR FOR NPRDTB [PAPER  IRQ14]
        ;
        NPRDFA=60       ;NPR DATA FROM BUS <15:08>
        NPRDFB=61       ;NPR DTAT FROM BUS <07:00>
        PCRA13=62       ;CSR FOR NPRDFA [BREAK  IRQ15]
        PCRB13=63       ;CSR FOR NPRDFB [REPEAT  IRQ16]
        ;
        .PAGE
        .SBTTL  GRAPHICS VARIABLES
        ;
        .=1200  ;VARIABLE AREA
        ;
YTABLE: .BLKB   30      ;Y AXIS TABLE
XTABLE: .BLKB   30      ;X AXIS TABLE
PTABLE: .BLKB   20      ;PLOTTING TABLE
        ;
OPR:    .BYTE   0,0     ;OPERATION CODE
CHK:    .BYTE   0,0     ;XY PLOT BOUNDS CHECK
SAVEX:  .BYTE   0,0     ;TEMPORARY
GTEMPA: .BYTE   0,0
GTEMPB: .BYTE   0,0
VCNTR:  .BYTE   0,0     ;VECTOR LENGTH COUNT
DIVCNT: .BYTE   0       ;DIVIDE COUNTER
DND:    .BYTE   0,0     ;DIVIDEND
DSR:    .BYTE   0,0     ;DIVISOR
QT:     .BYTE   0,0     ;QUOTIENT
GCSR:   .BYTE   0       ;CURRENT COMMAND CODE
BT:     .BYTE   0       ;BIT COUNTER
LT:     .BYTE   0       ;LINE COUNTER
```

```
CWORD:  .BYTE   0       ;BIT PATTERN FROM CHARACTER RAM
WTBYTE: .BYTE   0       ;DATA FOR WRITE FUNCTION
RADD:   .BYTE   0,0     ;CHARACTER ADDRESS IN ASCII SELECT
HADD:   .BYTE   0       ;HIGH BITS ADDRESS
RDADD:  .BYTE   0,0     ;BYTE ADDRESS
CADR:   .BYTE   0,0     ;COLUMN ADDRESS
ROWUPD: .BYTE   0,0     ;CHARACTER ROW VALUES
ROWADD: .BYTE   0,0
ROWRST: .BYTE   0,0
COLUPD: .BYTE   0,0     ;CHARACTER COLUMN VALUES
COLADD: .BYTE   0,0
COLRST: .BYTE   0,0
        ;
        ;       LIST PROCESSOR VARIABLES
        ;
LSBUFF: .BLKB   9.              ;CHARACTER BUFFER
LSTCNT: .BYTE   0               ;CHARS IN LSBUFF
LPNTR:  .BYTE   0,0             ;POINTER TO CHARS
FORMAT: .BYTE   0               ;INTEGER FORMAT LENGTH
GOSUB:  .BYTE   0,0             ;ADDRESS OF PROCESS
CHARG:  .BYTE   0               ;CURRENT CHARACTER
PCHAR:  .BYTE   0               ;PREVIOUS CHARACTER
SCHFLG: .BYTE   0               ;SEEN A CHARACTER FLAG
STPFLG: .BYTE   0               ;STRIPPING FLAG
        ;
        ;       EVALN VARIABLES
        ;
NSIGN:  .BYTE   0               ;SIGN OF NUMBER
NUMBER: .BYTE   0,0,0,0         ;16-BIT RESULT / 4 CHARACTER BUFFER
NCNTR:  .BYTE   0               ;DIGITS IN NUMBER
NPNTR:  .BYTE   0,0             ;POINTER FOR DIGITS
T0:     .BYTE   0               ;TEMPS
T1:     .BYTE   0
T2:     .BYTE   0
T3:     .BYTE   0
        ;
PNTSKP: .BYTE   0               ;POINTS TO SKIP
CHPNTR: .BYTE   0,0             ;CHARACTER POINTER
PROW:   .BYTE   0               ;CHARACTER DOTS
SVGSUB: .BYTE   0,0             ;GOSUB SAVE
SVSTAT: .BYTE   0               ;RSTAT0 SAVE
        ;
        .PAGE
        .SBTTL  RUN AND LIST STATUS DEFINITION
        ;
RSTAT0: .BYTE   0               ;RUN STATUS
        ;
        ;       RSTAT0
        ;
        ;       7-      RUNNING (1)
        ;       6-      RELATIVE ORIGIN (0) / RELATIVE POSITION (1)
        ;       5-      AUTO X STEP (1)
        ;       4-      AUTO Y STEP (1)
        ;       3-      VECTOR (0) / POINT (1)
        ;       2-      HOLD FLAG
        ;       1-      PEN UP (0) / PEN DOWN (1)
        ;
LSTAT:  .BYTE   0               ;LIST PROCESSOR STATUS
        ;
        ;       LSTAT
        ;
        ;       7- SINGLE CHARACTER MODE
        ;       6- LIST SCAN
        ;       5- BUILD MODE
        ;       3- SIZING FLAG
        ;       2- LOWER/GREEK
        ;       1- SCAN/PLOT
        ;       0- HORIZONTAL/VERTICAL
        ;
        ;
        .PAGE
        .SBTTL  ABSOLUTE DEFINITIONS
        ;
        CTLINE=20000    ;CONTROL LINE LOCATION
        ;
        GRAPH=100       ;GLOBLE GRAPH ON/OFF FLAG
```

```
        CHRFLG=105      ;GLOBLE INHIBIT PRINTING FLAG
        UPDFLG=106      ;GLOBLE DISPLAY UPDATE FLAG
        ;
        ;
        ; THESE MEMORY LOACATIONS CONTAIN THE
        ;INTERRUPT VECTOR ADDRESSES AS INITIALIZED
        ;BY THE RUNNING PROGRAM
        ;
        ; NON-MASKABLE INTERRUPT VECTORS
        ;
        NMI1    =7732           ;ODD FRAME
        NMI2    =7734           ;VERTICAL RETRACE
        NMI3    =7736           ;SCAN LINE
        ;
        ; NORMAL INTERRUPT VECTORS
        ;
        IRQ1    =7740           ;ARB-11 BUS INIT
        IRQ2    =7742           ;60 HZ CLOCK
        IRQ3    =7744           ;PHOTO READER DATA
        IRQ4    =7746           ;PRINTER CONTROL
        IRQ5    =7750           ;KEYBOARD DATA
        IRQ6    =7752           ;DATA FROM ARB-11
        IRQ7    =7754           ;DATA TAKEN BY ARB-11
        IRQ8    =7756           ;DISPLAY DATA FROM ARB-11
        IRQ9    =7760           ;DISPLAY DATA TO ARB-11 TAKEN
        IRQ10   =7762           ;ARB-11 BUS NPR DONE
        IRQ11   =7764           ;SCROLL  <TAPE
        IRQ12   =7766           ;SCROLL  >TAPE
        IRQ13   =7770           ;HERE IS BUTTON
        IRQ14   =7772           ;PAPER BUTTON
        IRQ15   =7774           ;BREAK BUTTON
        IRQ16   =7776           ;REPEAT BUTTON
        ;
CHRAM   =10000          ;4096 BYTE CHARACTER DEFINITION BUFFER
                        ;CONTAINING 256 CHARACTER DEFINITIONS
        ;
GPBOT   =30000          ;20480 BYTE GRAPHICS AREA
                        ;ALLOWS A 640(X) BY 256(Y) DISPLAY
EVNSCN  =30000          ;SCAN ADDRESS FOR EVEN SCANS
ODDSCN  =54000          ;SCAN ADDRESS FOR ODD SCANS
GPTOP   =77777          ;LAST DISPLAY BYTE
        ;
        CHROM1  =100000 ;CHARACTER ROM #1 (2K BYTES)
        CHROM2  =104000 ;CHARACTER ROM #2 (2K BYTES)
        ;
        .PAGE
        .SBTTL  SUMMARY OF GRAPHICS TABLES
        ;
        ;       THREE PARAMETER TABLES ARE USED BY
        ;       THE GRAPHICS PROCESSOR.
        ;
        ;       1       XTABLE
        ;       2       YTABLE
        ;       3       PLTBLE
        ;
        ;       THE X AND Y TABLES ARE ORGANIZED AS FOLLOWS:
        ;
        ;_VAL   0,X     PREVIOUS VALUE (16 BITS)
        ;F_VAL  2,X     NEW VALUE (16BITS)
        ;_STEP  4,X     RELATIVE REGISTER (16 BITS)
        ;VCTL_  6,X     VECTOR LENGTH (16 BITS)
        ;VAL_   10,X    RUNNING VALUE DURING VECTOR PLOT (16 BITS)
        ;       12,X    RUNNING FRACTIONAL VALUE DURING VECTOR PLOT (16 BITS)
        ;FRCT_  14,X    UPDATE VALUE DURING VECTOR PLOT (16 BITS)
        ;       16,X    FRACTIONAL UPDATE VALUE DURING VECTOR PLOT (16 BITS)
        ;SING_  20,X    SIGN OF UPDATE
        ;_SIZE  21,X    CHARACTER SIZING
        ;_ORG   22,X    ORIGIN
        ;_CNT   24,X    POINT POSITION
        ;P_CNT  25,X    PREVIOUS POSITION
        ;SAV_   26,X    SAVE STEP SIZE
        ;
        ;
        ;       THE PLOTTING TABLE IS ORGANIZED AS FOLLOWS:
        ;
        ;       0,X     Y POSITION VALUE (16 BITS)
```

```
;       2,X     X POSITION VALUE (16 BITS)
;       4,X     HORIZONTAL LIN BASE ADDRESS
;       6,X     BYTE OFFSET IN LINE
;       10,X    EXPLICIT BIT POSITION
;       11,X    CONTROL STATUS DURING PLOT
;       12,X    ABSOLUTE LOCATION OF BYTE TO BE UPDATED
;       14,X    TEMPORARES (- 17,X)
;
.PAGE
.SBTTL  SUMMARY OF PLOTTER COMMANDS
;
;       $R_     INITIATE VIDEO GRAPHICS
;               _ IS AN OPTIONAL INTEGER FORMAT SPECIFIER
;               FROM 0 TO 9. 0 IS THE DEFAULT VALUE
;               AND ALLOWS FREE FORMAT INPUT
;
;       $O      DEFINE ORIGIN OF PLOT
;               DATA FOLLOWING THE COMMAND IS STORED AS
;               XORIGIN, YORIGIN. IF MORE DATA IS SPECIFIED
;               THEN DATA IS SEQUENCIALLY STORED IN
;               XO, YO, XO, YO,  ETC.
;
;       $M      MOVE TO POINT RELATIVE TO ORIGIN MODE.
;               DATA FORMAT IS X, Y, X, Y,  ETC.
;               POSITION IS (XO+X,YO+Y),  ETC.
;
;       $L      MOVE RELATIVE TO CURRENT POSITION
;               DATA FORMAT IS DX, DY, DX, DY,  ETC.
;               POSITION IS  (X+DX,Y+DY) THEN X=X+DX, Y=Y+DY
;
;       $D      PEN DOWN COMMAND
;
;       $U      PEN UP COMMAND
;
;       $X      AUTO X STEP MODE
;               FIRST DATA IS X STEP VALUE
;               SUBSEQUENT DATA IS Y VALUE (RELATIVE TO ORIGIN)
;               PLOTTED DATA IS THUS:
;                       X,YO+Y  THEN X=X+XSTEP
;                       X,YO+Y  THEN X=X+XSTEP  ETC.
;
;       $Y      AUTO Y STEP MODE
;               FIRST DATA IS Y STEP VALUE
;               SUBSEQUENT DATA IS X VALUE (RELATIVE TO ORIGIN)
;               PLOTTED DATA IS THUS:
;                       XO+X,Y  THEN Y=Y+YSTEP
;                       XO+X,Y  THEN Y=Y+YSTEP  ETC.
;
.PAGE
;
;       $F__... CHARACTER FORMAT SPECIFIER
;               THE $F COMMAND IS FOLLOWED (IN ANY ORDER)
;               BY CHARACTERS H (HORIZONTAL), V (VERTICAL),
;               L (LOWER CASE), G (GREEK CHARACTERS),
;               OR DIGITS 0-9. THE FIRST DIGIT IS THE
;               CHARACTER HEIGHT SPECIFIER AND THE SECOND
;               IS THE CHARACTER WIDTH SPECIFIER.
;               ANY OTHER CHARACTER TERMINATES THIS COMMAND
;
;       $C      CHARACTER MODE PLOTTING
;               THE $C COMMAND INITIATES THE CHARACTER
;               DRAWING SEQUENCE.
;                       1. THE PEN IS LIFTED
;                       2. SEARCH MODE IS INITIATED FOR A +
;                       3. ALL CHARACTERS AFTER THE + ARE DRAWN
;                       4. ANY CONTROL CHARACTER <CR>,<LF>
;                               TERMINATES THE $C COMMAND
;
;       $G_     GRAPHICAL PLOT COMMAND
;               THIS COMMAND PROVIDES 10 GRAPHICAL
;               SYMBOLS WHICH MAY BE PLOTTED AS
;               DATA MARKERS. THE COMMAND DOES NOT
;               CHANGE THE CURRENT PLOTTING
;               MODE DURRING EXECUTION. HOWEVER, THE
;               COMMAND PLOTS IN VECTOR MODE AND ALWAYS LIFTS
;               THE PEN. _ IS A DIGIT 0-9 SPECIFYING THE
```

```
;               SYMBOL TO BE PLOTTED.
;
.PAGE
;       $V      VECTOR PLOTTING
;               DRAW LINE BETWEEN END POINTS
;
;       $P      POINT PLOTTING
;               PLOT ONLY END POINT
;
;       $H      HOLD COMMAND
;               WAIT FOR OPERATOR TO PUSH 'HERE IS' BUTTON
;
;       $K      CLEAR VIDEO COMMAND
;
;       $W      INCLUDED FOR COMPATABILITY ONLY
;
;       $Z      SET PLOTTING INTENSITY
;
;       $S      STOP PLOTTING OPERATIONS
;
;       $A      TURNS DISPLAY SCREEN ON
;
;       $B      TURNS DISPLAY SCREEN OFF
;
;       $I      INHIBITS PRINTING OF INPUT DATA FROM $R-$S
;
;       $J      ENABLES PRINTING OF INPUT DATA (WHEN USING TERMINAL)
;
.PAGE
.SBTTL  INPUT DATA FORMAT
;
;         THE NUMERICAL DATA REQUIRED BY THE PLOTTING
;       COMMANDS MAY BE SPECIFIED IN ONE OF TWO FORMATS:
;
;       1.   IMMEDIATELY FOLLOWING A COMMAND, IF THE
;       FIRST CHARACTER IS A  -  OR A  DIGIT
;       DATA IS ASSUMED TO FOLLOW IN THE SPECIFIED INTEGER
;       FORMAT UNTIL A <CR>,<LF> OR OTHER CONTROL CHARACTER
;       IS ENCOUNTERED. (THIS MODE DOES NOT APPLY TO $R, $F, OR $C)
;            IN FREE FORMAT DATA MAY FOLLOW WITHOUT
;       THE ABOVE RESTRICTIONS
;
;       2.   IF THE FIRST CHARACTER FOLLOWING A COMMAND IS NOT
;       ONE OF THE ABOVE, THE GRAPHICS CONTROL ENTERS
;       THE SEARCH MODE. THIS MODE SEARCHES FOR THE
;       CHARACTER + . ALL CHARACTERS FOLLOWING THE DELIMITER +
;       ARE ASSUMED TO BE IN THE SPECIFIED INTEGER FORMAT.
;       THE TERMINATION OF THE DATA BY A <CR>,<LF> OR
;       OTHER CONTROL CHARACTER CAUSES THE CONTROLLER TO
;       AGAIN ENTER THE SEARCH MODE.
;          A NEW PLOTTER COMMAND IS ALWAYS RECOGNIZED
;       WITH THE CONTROLLER IN SEARCH MODE.
;
```

```
        .PAGE
        .SBTTL  GLOBLE ENTRY POINT
        ;
        .=140000                ;LOCATION OF GRAPHICS PROGRAMS
        ;
GPINIT: JMP     GPSET           ;INITIALIZE
GRPLOT: JMP     LSTENT          ;PROCESS DATA
GPSET:  LDX     #,0             ;CLEAR GOSUB POINTER
        STX     GOSUB
        LDX     #,HEREIS        ;SET UP 'HERE IS' BUTTON
        STX     IRQ13
        LDA A   #,7             ;ENABLE INTERRUPT
        STA A   PCRA12
        LDA A   NPRDTA          ;CLEAR PENDING INTERRUPT
        ;
        LDX     #,0             ;CLEAR CONTROL REGISTERS
        STX     PCRA5           ;GRAPHIC 'IN' PIA
        STX     WDFBA           ;DIRECTION - IN
        STX     PCRA6           ;GRAPHIC 'OUT' PIA
        LDX     #,377,377       ;DIRECTION - OUT
        STX     WDTBA
        LDX     #,GRPIN         ;SETUP IN INTERRUPT
        STX     IRQ8
        LDX     #,GRPOUT        ;SETUP OUT INTERRUPT
        STX     IRQ9
        LDX     #,55,4          ;ENABLE IN INTERRUPT
        STX     PCRA5
        LDX     #,5,4           ;DISABLE OUT INTERRUPT
        STX     PCRA6           ;REENABLE CLOCK
        LDA A   WDFBA           ;CLEAR PENDING INTERRUPT
        LDA A   WDTBB           ;CLEAR PENDING INTERRUPT
        CLR     STPFLG          ;DISABLE STRIPPING
        RTS                     ;RETURN TO MAIN LINE
        ;
        .PAGE
        .SBTTL  HERE IS INTERRUPT HANDLER
        ;
HEREIS: COM     GRAPH           ;FLIP-FLOP GRAPHIC
        INC     UPDFLG          ;DISPLAY UPDATE
        LDA A   NPRDTA          ;CLEAR INTERRUPT FLAG
        RTI                     ;FINISHED
        ;
        .PAGE
        .SBTTL  GRAPHICS DISPATCHER
        ;
GRPIN:
GRPOUT: LDA A   #,4             ;DISABLE FURTHER INTERRUPTS
        STA A   PCRA5
        CLI                     ;ENABLE ALL OTHERS
        BSR     1$              ;PLACE RETURN POINT ON STACK
        SEI                     ;INHIBIT INTERRUPTS
        LDA A   #,55            ;ENABLE GRAPHICS INTERRUPT
        STA A   PCRA5
        LDA A   WDFBA           ;SET READY FLAG
        RTI                     ;RETURN FROM INTERRUPT
1$:     LDA A   WDFBB           ;GET CODED INSTRUCTION
        LDA B   WDFBA
        STA A   WDTBB           ;STORE AT OUTPUT PORT
        STA B   WDTBA
        STA A   OPR+1           ;SAVE FOR OPERATIONS
        STA B   OPR
        AND B   #,340           ;XY FUNCTION ?
        BMI     2$              ;IF NOT - SKIP
        JMP     XYFNCT
2$:     CMP B   #,200           ;CHARACTER FUNCTION ?
        BNE     3$              ;IF NOT - SKIP
        JMP     CWRITE
3$:     CMP B   #,240           ;SELECT FUNCTION ?
        BNE     4$              ;IF NOT - SKIP
        JMP     SELECT
4$:     CMP B   #,300           ;CONTROL FUNCTION ?
        BNE     5$              ;IF NOT - SKIP
        JMP     CNTROL
5$:     JMP     READ            ;READ/WRITE FUNCTION
        ;
        .PAGE
```

```
        .SBTTL  X-COORDINATE CALCULATION
        ;
        ;       ENTER WITH [X] = TABLE ADDRESS
        ;
NCXVAL: CLR     CHK+1           ;XPLOT OK
        LDA B   2,X             ;CHECK HIGH ORDER OF XVALUE
        BMI     1$              ;IF <0 - DON'T PLOT
        BSR     BXADC           ;COMPUTE BIT POSITION
        BNE     1$              ;IF /=0 - DON'T PLOT
        SUB A   #,80.           ;WITHIN LINE ?
        BCS     2$              ;IF SO - PLOT
1$:     COM     CHK+1           ;ELSE INHIBIT PLOT
2$:     RTS                     ;FINISHED
        ;
BXADC:  LDA A   3,X             ;GET LOW BYTE OF XV
        CLR B
        AND A   #,7             ;SAVE LOW THREE BITS
        ADD A   #,BPTBLE&377    ;COMPUTE LOCATION IN TABLE
        ADC B   #,BPTBLE&177400/400
        STA A   15,X            ;SAVE IN TEMPORARY
        STA B   14,X
        STX     SAVEX           ;SAVE POINTER
        LDX     14,X            ;GET TABLE ADRDRESS
        LDA A   0,X             ;GET BIT PATTERN
        LDX     SAVEX           ;GET POINTER BACK
        STA A   10,X            ;SAVE BIT PATTERN
        LDA A   3,X             ;GET XV TO COMPUTE BYTE ADDRESS
        LDA B   2,X
        CLC
        ROR B                   ;DIVIDE BY 8
        ROR A
        ASR B
        ROR A
        ASR B
        ROR A
        STA A   7,X             ;SAVE RESULT IN XVAL
        STA B   6,X
        RTS                     ;FINISHED
        ;
BPTBLE: .BYTE   200,100,40,20,10,4,2,1
        ;
        .PAGE
        .SBTTL  Y-COORDINATE CALCULATION
        ;
        ;       ENTER WITH [X] = TABLE ADDRESS
        ;
NCYVAL: CLR     CHK             ;YPLOT OK
        LDA B   0,X             ;MUST = 0
        BEQ     1$              ;IF SO - SKIP
        COM     CHK             ;ELSE INHIBIT PLOT
        RTS                     ;FINISHED
1$:     LDA B   1,X             ;GET LOW ORDER VALUE
        CLC
        ROR B                   ;DIVIDE BY 2
        LDA A   #,177           ;COMPUTE <B,A>=127.-YV/2
        SBA
        CLR B
        ASL A                   ;COMPUTE 16*<B,A>
        ROL B
        ASL A
        ROL B
        ASL A
        ROL B
        ASL A
        ROL B
        STA A   5,X             ;SAVE TEMPORARILY
        STA B   4,X
        ASL A                   ;COMPUTE 64*<B,A>
        ROL B
        ASL A
        ROL B
        ADD A   5,X             ;COMPUTE 80*<B,A>
        ADC B   4,X
        ROR     1,X             ;CHECK EVEN/ODD SCAN
        BCC     2$              ;IF EVEN - SKIP
        ROL     1,X             ;RESTORE VALUE
```

```
        ADD A   #,ODDSCN&377    ;COMPUTE ABSOLUTE ADDRESS
        ADC B   #,ODDSCN&177400/400
        BRA     3$
2$:     ROL     1,X             ;RESTORE VALUE
        ADD A   #,EVNSCN&377    ;COMPUTE ABSOLUTE ADDRESS
        ADC B   #,EVNSCN&177400/400
3$:     STA A   5,X             ;SAVE VALUE IN YVAL
        STA B   4,X
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  PLOTXY POINT ROUTINE
        ;
PLOTXY: LDX     YTABLE+2        ;NEW Y VALUE
        STX     PTABLE
        LDX     XTABLE+2        ;NEW X VALUE
        STX     PTABLE+2
VECTXY: LDX     #,PTABLE        ;TABLE POINTER
        JSR     NCXVAL          ;COMPUTE XVAL AND BIT POSITION
        JSR     NCYVAL          ;COMPUTE YVAL
NXYPLT: LDX     CHK             ;OK TO PLOT ?
        BNE     1$              ;IF NOT - SKIP
        LDX     #,PTABLE        ;TABLE POINTER
        BSR     XYBIS           ;DO A POINT
1$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  XY BIT SET ROUTINE
        ;
XYBIS:  LDA A   5,X             ;GET YVAL
        LDA B   4,X
        ADD A   7,X             ;ADD XVAL
        ADC B   6,X
        STA A   13,X            ;SAVE XYVAL
        STA B   12,X
        LDA A   10,X            ;GET BIT POSITION
        LDA B   11,X            ;GET CONTROL
        ROR B                   ;CHECK ENABLE BIT
        BCC     3$              ;IF =0 NO PLOTTING
        LDX     12,X            ;GET XYVAL AS ADDRESS
        ROL B                   ;CHECK FOR ADD/DELETE
        BMI     1$              ;IF DELETE - SKIP
        ORA A   0,X             ;ADD BY ORING
        BRA     2$
1$:     COM A                   ;BIT TO CLEAR IS '0'
        AND A   0,X             ;DELETE BY ANDING
2$:     STA A   0,X             ;MOVE INTO DISPLAY AREA
3$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  AUTO-INCREMENT ROUTINE
        ;
        ;       ENTER WITH [X] = TABLE ADDRESS
        ;
AI:     LDA A   3,X             ;P_ = N_
        LDA B   2,X
        STA A   1,X
        STA B   0,X
        ADD A   5,X             ;N_ = N_ + DEL
        ADC B   4,X
        STA A   3,X
        STA B   2,X
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  LOAD NEW VALUE ROUTINE
        ;
        ;       ENTER WITH [X] = TABLE ADDRESS
        ;
LOADV:  LDA B   OPR             ;CHECK OPERATION CODE
        BIT B   #,40            ;DEL MODE ?
        BNE     1$              ;IF SO - SKIP
        LDA A   3,X             ;P_ = N_
        STA A   1,X
        LDA A   2,X
        STA A   0,X
```

```
        LDA A   OPR+1           ;GET HIGH ORDER
        AND B   #,3             ;MASK VALUE
        STA A   3,X             ;SAVE NEW VALUE
        STA B   2,X
        RTS                     ;FINISHED
        ;
1$:     LDA A   OPR+1           ;GET DEL VALUE
        AND B   #,3
        BIT B   #,2             ;A NEGATIVE DEL ?
        BEQ     2$              ;IF NOT SKIP
        ORA B   #,374           ;SIGN EXTEND VALUE
2$:     STA A   5,X             ;NEW DEL VALUE
        STA B   4,X
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  ABSOLUTE VECTOR LENGTH CALCULATION
        ;
        ;       ENTER WITH [X] = TABLE ADDDRESS
        ;
ABSD:   CLR     20,X            ;SET SIGN (+)
        LDA A   3,X             ;COMPUTE N_ - P_
        LDA B   2,X
        SUB A   1,X
        SBC B   0,X
        BPL     1$              ;IF (+) - SKIP
        COM     20,X            ;ELSE SIGN IS (-)
        COM A                   ;COMPUTE TWO'S COMPLEMENT
        COM B
        ADD A   #,1
        ADC B   #,0
1$:     STA A   7,X             ;SAVE ABS(VECT)
        STA B   6,X
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  VECTOR SET CONSTANT ROUTINE
        ;
        ;       ENTER WITH [X] = TABLE ADDRESS
        ;
VSETC:  CLR B
        STA B   14,X            ;SET STEP VALUE SIZE
        STA A   15,X
        STA B   16,X
        STA B   17,X
        STA B   13,X            ;SET FRACTION = 1/2
        LDA B   #,200
        STA B   12,X
        LDA A   1,X             ;MOV START VALUE TO CP_
        LDA B   0,X
        STA A   11,X
        STA B   10,X
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  SET SIGN OF INCREMENT VALUE ROUTINE
        ;
        ;       ENTER WITH [X] = TABLE ADDRESS
        ;
SETSGN: TST     20,X            ;CHECK SIGN
        BEQ     1$              ;IF (+) - SKIP
        COM     17,X            ;ELSE COMPUTE 32 BIT 2'S COMPLEMENT
        COM     16,X
        COM     15,X
        COM     14,X
        INC     17,X
        BNE     1$
        INC     16,X
        BNE     1$
        INC     15,X
        BNE     1$
        INC     14,X
1$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  UPDATE VALUE ROUTINE
```

```
        ;
        ;       ENTER WITH [X] = TABLE ADDRESS
        ;
UPDATE: LDA A   13,X            ;32 BIT ADDITION
        LDA B   12,X
        ADD A   17,X
        ADC B   16,X
        STA A   13,X
        STA B   12,X
        LDA A   11,X
        LDA B   10,X
        ADC A   15,X
        ADC B   14,X
        STA A   11,X
        STA B   10,X
        LDX     10,X            ;GET RESULT
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  CLEAR GRAPHIC MEMORY ROUTINE
        ;
GRPCLR: LDA A   #,GPBOT&377     ;BASE ADDRESS
        LDA B   #,GPBOT&177400/400
1$:     STA B   GTEMPA          ;SAVE
2$:     STA A   GTEMPA+1
        LDX     GTEMPA          ;GET INDEX ADDRESS
        CLR     0,X             ;CLEAR 8 BYTES QUICKLY
        CLR     1,X
        CLR     2,X
        CLR     3,X
        CLR     4,X
        CLR     5,X
        CLR     6,X
        CLR     7,X
        ADD A   #,10            ;UPDATE ADDRESS
        BCC     2$              ;LOOP
        ADC B   #,0
        BPL     1$              ;END OF GRAPHIC IS 100000 (-)
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  PLOT VECTOR HANDLER
        ;
PLOTVC: LDX     XTABLE+10       ;LOAD RUNNING VALUES
        STX     PTABLE+2
        LDX     YTABLE+10
        STX     PTABLE
        JSR     VECTXY          ;PLOT FIRST POINT
1$:     LDX     VCNTR           ;GET COUNT
        BEQ     4$              ;IF 0 - FINISHED
        DEX                     ;UPDATE COUNT
        STX     VCNTR
        LDX     #,XTABLE        ;DO X UPDATES
        JSR     UPDATE
        CPX     PTABLE+2        ;HAS VALUE CHANGED ?
        BEQ     2$              ;IF NOT - SKIP CALCULATION
        STX     PTABLE+2        ;ELSE SAVE NEW VALUE
        LDX     #,PTABLE        ;COMPUTE NEW XVAL
        JSR     NCXVAL
2$:     LDX     #,YTABLE        ;DO Y UPDATES
        JSR     UPDATE
        CPX     PTABLE          ;HAS VALUE CHANGED ?
        BEQ     3$              ;IF NOT - SKIP CALCULATION
        STX     PTABLE          ;ELSE SAVE NEW VALUE
        LDX     #,PTABLE        ;COMPUTE YVAL
        JSR     NCYVAL
3$:     JSR     NXYPLT          ;PLOT POINT
        BRA     1$              ;LOOP AGAIN
4$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  DIVISION ROUTINE
        ;
DIVIDE: LDA A   #,16.            ;16 BIT DIVIDE
        STA A   DIVCNT          ;SAVE COUNTER
        LDA A   DND+1           ;GET DIVIDEND
```

```
        LDA B   DND
        CLC                     ;DO INITIAL SHIFT
        ROL A
        ROL B
1$:     SUB A   DSR+1           ;DO INITIAL DIVISION
        SBC B   DSR
        BCC     2$              ;IF DND>DSR - BRANCH
        ADD A   DSR+1           ;ELSE RESTORE
        ADC B   DSR
2$:     ROL     QT+1            ;SHIFT IN COMPUTED BIT
        ROL     QT
        ASL A                   ;SHIFT DIVIDEND
        ROL B
        DEC     DIVCNT          ;16 BITS YET ?
        BNE     1$              ;LOOP UNTIL FINISHED
        COM     QT+1            ;GET TRUE RESULT
        COM     QT
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  XY FUNCTION SCAN ROUTINE
        ;
XYFNCT: LDA A   GCSR            ;USE CURRENT STATUS INFO
        STA A   PTABLE+11       ;SAVE IN PTABLE
        LDA A   OPR             ;CHECK FOR X OPERATION
        BIT A   #,100
        BNE     2$              ;IF NOT - SKIP
        BIT A   #,20            ;AUTO-INCREMENT ?
        BEQ     1$              ;IF NOT - SKIP
        LDX     #,YTABLE        ;UPDATE VALUES
        JSR     AI
1$:     LDX     #,XTABLE        ;LOAD VALUES
        JSR     LOADV
        BRA     4$
2$:     BIT A   #,20            ;AUTO-INCREMENT ?
        BEQ     3$              ;IF NOT - SKIP
        LDX     #,XTABLE        ;UPDATE VALUES
        JSR     AI
3$:     LDX     #,YTABLE        ;LOAD NEW VALUES
        JSR     LOADV
        ;
4$:     LDA A   OPR             ;GET OPERATION
        BIT A   #,14            ;PL OR V SET ?
        BEQ     6$              ;IF NOT - FINISHED
        BIT A   #,40            ;RELATIVE MODE ?
        BEQ     5$              ;IF NOT - SKIP
        LDX     #,XTABLE        ;P_=N_ ; N_=N_+DEL
        JSR     AI
        LDX     #,YTABLE        ;P_=N_ ; N_=N_+DEL
        JSR     AI
        LDA A   OPR             ;GET OPERATION
5$:     BIT A   #,10            ;VECTOR ?
        BNE     VECTOR          ;IF SO - SKIP
        JSR     PLOTXY          ;ELSE PLOT POINT
6$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  VECTOR DRAWER
        ;
VECTOR: LDX     #,XTABLE        ;COMPUTE /X/
        JSR     ABSD
CHEK1:  LDX     #,YTABLE        ;COMPUTE /Y/
        JSR     ABSD
        LDX     XTABLE+6        ;DX=DY ?
        CPX     YTABLE+6
        BNE     CHEK2           ;IF NOT - SKIP
        STX     VCNTR           ;COUNT
        LDX     #,XTABLE        ;UPDATE COUNT = 1
        LDA A   #,1
        JSR     VSETC
        JSR     SETSGN          ;SET SIGN
        LDX     #,YTABLE        ;UPDATE COUNT = 1
        LDA A   #,1
        JSR     VSETC
        JSR     SETSGN          ;SET SIGN
        JSR     PLOTVC          ;PLOT THE VECTOR
```

```
        ;
VECTDN: LDA A   OPR             ;CHECK FOR RELATIIVE OPERATION
        BIT A   #,40
        BNE     1$              ;IF NOT - SKIP
        LDX     YTABLE+2
        STX     YTABLE
        LDX     XTABLE+2
        STX     XTABLE
1$:     RTS                     ;FINISHED
        ;
CHEK2:  LDX     XTABLE+6        ;DX=0 ?
        BNE     CHEK3           ;IF NOT - SKIP AHEAD
        LDX     YTABLE+6        ;DY IS COUNT
        STX     VCNTR
        LDX     #,XTABLE        ;UPDATE VALUE = 0
        CLR A
        JSR     VSETC
        LDX     #,YTABLE        ;UPDATE OF 1
        LDA A   #,1
        JSR     VSETC
        JSR     SETSGN          ;SET SIGN
        JSR     PLOTVC          ;GO PLOT VECTOR
        JMP     VECTDN          ;FINISH UP
        ;
CHEK3:  LDX     YTABLE+6        ;DY = 0 ?
        BNE     CHEK4           ;IF NOT - SKIP
        LDX     XTABLE+6        ;DX IS COUNT
        STX     VCNTR
        LDX     #,XTABLE        ;UPDATE = 1
        LDA A   #,1
        JSR     VSETC
        JSR     SETSGN          ;SET SIGN
        LDX     #,YTABLE        ;UPDATE = 0
        CLR A
        JSR     VSETC
        JSR     PLOTVC          ;PLOT VECTOR
        JMP     VECTDN          ;FINISH UP
        ;
CHEK4:  LDA A   XTABLE+7        ;WHICH IS LARGER ?
        LDA B   XTABLE+6
        SUB A   YTABLE+7
        SBC B   YTABLE+6
        BCS     CHEK5           ;IF DY>DX BRANCH
        LDX     XTABLE+6        ;DX IS COUNT
        STX     VCNTR
        STX     DSR             ;DO DIVISION
        LDX     YTABLE+6
        STX     DND
        JSR     DIVIDE
        LDX     #,XTABLE        ;UPDATE = 1
        LDA A   #,1
        JSR     VSETC
        JSR     SETSGN          ;SET SIGN
        LDX     #,YTABLE        ;UPDATE = 0
        CLR A
        JSR     VSETC
        LDX     QT              ;SET FRACTIONAL PART
        STX     YTABLE+16
        LDX     #,YTABLE
        JSR     SETSGN          ;SET SIGN
        JSR     PLOTVC          ;PLOT VECTOR
        JMP     VECTDN          ;FINISH UP
        ;
CHEK5:  LDX     YTABLE+6        ;DY IS COUNT
        STX     VCNTR
        STX     DSR             ;DO DIVISION
        LDX     XTABLE+6
        STX     DND
        JSR     DIVIDE
        LDX     #,XTABLE        ;UPDATE = 0
        CLR A
        JSR     VSETC
        LDX     QT              ;FRACTIONAL VALUE
        STX     XTABLE+16
        LDX     #,XTABLE
        JSR     SETSGN          ;SET SIGN
```

```
        LDX     #,YTABLE        ;UPDATE = 1
        LDA A   #,1
        JSR     VSETC
        JSR     SETSGN          ;SET SIGN
        JSR     PLOTVC          ;PLOT VECTOR
        JMP     VECTDN          ;FINISH UP
        ;
        .PAGE
        .SBTTL  CHARACTER WRITING HANDLER
        ;
CWRITE: LDA A   OPR             ;GET CODE
        TAB
        AND A   #,4             ;CHACK REVERSE BIT
        BNE     5$              ;IF SO - SKIP
        LDX     #,PTABLE+2      ;ROW ALONG X
        STX     ROWADD
        LDX     XTABLE+2        ;SAVE ROW POSITION
        STX     ROWRST
        LDX     #,PTABLE        ;COLUMN ALONG Y
        STX     COLADD
        LDX     YTABLE+2        ;SAVE RCOLUMN POSITION
        STX     COLRST
        TBA                     ;GET CODE
        AND A   #,2             ;CHECK ROW DIRECTION
        BNE     1$              ;IF REVERSED - SKIP
        LDX     #,1             ;(+)
        BRA     2$
1$:     LDX     #,-1            ;(-)
2$:     STX     ROWUPD          ;SAVE UPDATING VALUE
        TBA                     ;GET CODE
        AND A   #,1             ;COLUMN REVERSED ?
        BNE     3$              ;IF SO - SKIP
        LDX     #,1             ;(+)
        BRA     4$
3$:     LDX     #,-1            ;(-)
4$:     STX     COLUPD          ;SAVE UPDATE VALUE
        JMP     CXWRT           ;DO CHARACTER
        ;
5$:     LDX     #,PTABLE+2      ;COLUMN ALONG X
        STX     COLADD
        LDX     XTABLE+2        ;COLUMN RESET VALUE
        STX     COLRST
        LDX     #,PTABLE        ;ROW ALONG Y
        STX     ROWADD
        LDX     YTABLE+2        ;ROW RESET VALUE
        STX     ROWRST
        TBA                     ;GET CODE
        AND A   #,2             ;REVERSED DIRECTION ?
        BNE     6$              ;IF SO - SKIP
        LDX     #,1             ;(+)
        BRA     7$
6$:     LDX     #,-1            ;(-)
7$:     STX     COLUPD          ;SAVE UPDATING VALUE
        TBA                     ;GET CODE
        AND A   #,1             ;REVERSED ROW ?
        BNE     8$              ;IF SO - SKIP
        LDX     #,1             ;(+)
        BRA     9$
8$:     LDX     #,-1            ;(-)
9$:     STX     ROWUPD          ;SAVE UPDATE VALUE
                                ;FALL THROUGH TO CXWRT
        ;
        .PAGE
        ;
CXWRT:  LDA A   OPR+1           ;GET ASCII CODE
        CLR B
        ASL A                   ;16.*ASCII CODE
        ROL B
        ASL A
        ROL B
        ASL A
        ROL B
        ASL A
        ROL B
        ORA A   #,17            ;FILL IN REST IF ADDRESS
        ORA B   #,CHRAM&177400/400
```

```
        STA A   CADR+1          ;SAVE ADDRESS
        STA B   CADR
        LDA A   GCSR            ;SET CURRENT STATUS IN PTABLE
        STA A   PTABLE+11
        LDX     COLADD          ;SET UP COLUMN PARAMETER
        LDA B   COLRST
        STA B   0,X
        LDA A   COLRST+1
        STA A   1,X
        LDA A   #,16.           ;16 ROWS
        STA A   LT
1$:     LDX     CADR            ;GET CHARACTER ADDRESS
        LDA A   0,X             ;GET ROW OF DOTS
        STA A   CWORD           ;AND SAVE
        LDX     ROWADD          ;SET UP ROW PARAMETER
        LDA B   ROWRST
        STA B   0,X
        LDA A   ROWRST+1
        STA A   1,X
        LDA A   #,8.            ;8. BITS PER ROW
        STA A   BT
2$:     ROL     CWORD           ;IS BIT SET ?
        BCC     3$              ;IF NOT - SKIP
        JSR     VECTXY          ;ELSE PLOT POINT
3$:     LDX     ROWADD          ;UPDATE ROW POSITION
        LDA A   1,X
        ADD A   ROWUPD+1
        STA A   1,X
        LDA B   0,X
        ADC B   ROWUPD
        STA B   0,X
        DEC     BT              ;ONE LESS BIT
        BNE     2$              ;LOOP FOR ALL 8 BITS
        LDX     COLADD          ;UPDATE COLUMN POSITION
        LDA A   1,X
        ADD A   COLUPD+1
        STA A   1,X
        LDA B   0,X
        ADC B   COLUPD
        STA B   0,X
        LDX     CADR            ;UPDATE CHARACTER ROW ADDRESS
        DEX
        STX     CADR
        DEC     LT              ;MORE ROWS ?
        BNE     1$              ;LOOP FOR ALL LINES
        ;
        LDA A   OPR             ;UPDATE ALONG X AXIS ?
        AND A   #,20
        BEQ     4$              ;IF NOT - SKIP
        LDX     XTABLE+2        ;PX=NX
        STX     XTABLE
        LDX     PTABLE+2        ;NX=NX+DELX
        STX     XTABLE+2
4$:     LDA A   OPR             ;UPDATE ALONG Y AXIS ?
        AND A   #,10
        BEQ     5$              ;IF NOT - KIP
        LDX     YTABLE+2        ;PY=NY
        STX     YTABLE
        LDX     PTABLE          ;NY=NY+DELY
        STX     YTABLE+2
5$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  CHARACTER RAM SET UP
        ;
RSTRAM: LDX     #,CHRAM         ;RAM AREA
        CLR     0,X             ;CLEAR FIRST 4
        CLR     1,X
        CLR     2,X
        CLR     3,X
        LDX     #,CHRAM+4       ;FILL ADDRESS
        STX     GTEMPB
        LDX     #,CHROM1        ;ROM AREA
        STX     GTEMPA
1$:     LDX     GTEMPA          ;UPDATE ADDRESS
        LDA A   0,X             ;GET BYTE
```

```
        LDA B   1,X             ;NEXT BYTE
        INX
        INX
        STX     GTEMPA
        LDX     GTEMPB          ;UPDATE ADDRESS
        STA A   0,X             ;MOVE BYTE
        STA B   1,X             ;AND NEXT BYTE
        INX
        INX
        STX     GTEMPB
        CPX     #,CHRAM+10000   ;END OF RAM AREA ?
        BNE     1$              ;IF NOT - LOOP
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  SELECT ROUTINE
        ;
SELECT: LDA A   OPR             ;GET OPERATION
        BIT A   #,20            ;REPLACING ?
        BEQ     1$              ;IF SO - SKIP
        AND A   #,17            ;CHECK FUNCTION BITS
        BNE     2$              ;IF NOT WORD SELECT - SKIP
        LDA A   OPR+1           ;ELSE SPECIFYING A WORD
        STA A   WTBYTE          ;SAVE FOR WRITE FUNCTION
        CLR B                   ;COMPUTE ADDRESS IN CHRAM
        ASL A
        ROL B
        ASL A
        ROL B
        ASL A
        ROL B
        ASL A
        ROL B
        ADD A   #,CHRAM&377
        ADC B   #,CHRAM&177400/400
        STA A   RADD+1          ;SAVE CHAR ADDRESS
        STA B   RADD
        RTS                     ;FINISHED
        ;
1$:     AND A   #,17            ;MASK LINE SELECT
        LDA B   RADD+1          ;GET LOW BYTE ADDRESS
        AND B   #,360           ;MASK LINE SELECT
        ABA                     ;MASK IN NEW SELECT
        STA A   RADD+1
        LDX     RADD            ;GET COMPLETE ADDRESS
        LDA A   OPR+1           ;GET DATA
        STA A   0,X             ;STORE IN PLACE
        RTS                     ;FINISHED
        ;
2$:     LDA A   OPR+1           ;THIS IS CHARACTER
        JSR     LSTGRP          ;PROCESS IT
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  CONTROL ROUTINE
        ;
CNTROL: LDA A   OPR             ;GET OPERATION
        BIT A   #,20            ;CLEAR SCREEN ?
        BEQ     1$              ;IF NOT - SKIP
        JSR     GRPCLR          ;ELSE CLEAR SCREEN
        LDA A   OPR
1$:     AND A   #,17            ;MASK HIGH BITS
        ASL A
        ASL A
        ASL A
        ASL A
        STA A   HADD            ;SAVE BITS FOR R/W FUNCTION
        LDA A   OPR+1           ;SAVE NEW CONTROL
        STA A   GCSR
        BIT A   #,20            ;RESTORE  CHARACTERS ?
        BEQ     2$              ;IF NOT - SKIP
        JSR     RSTRAM          ;RESTORE CHARACTER SET
        LDA A   OPR+1           ;GET CONTROL
2$:     BIT A   #,100           ;SCREEN SET OPERATION ?
        BEQ     5$              ;IF NOT - FINISHED
        BIT A   #,40            ;SCREEN ON ?
```

```
        BEQ     3$              ;IF NOT - TURN  OFF
        LDA A   #,377           ;SET FLAG
        BRA     4$
3$:     CLR A                   ;OFF
4$:     STA A   GRAPH           ;STORE FLAG
        INC     UPDFLG          ;UPDATE SCREEN
5$:     RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  READ/WRITE FUNCTION
        ;
READ:   LDA B   OPR             ;GET OPERATION
        TBA                     ;SAVE
        AND B   #,17            ;MASK HIGH BITS
        ORA B   HADD            ;OR IN HIGH BITS
        STA B   RDADD           ;SAVE ADDRESS
        LDA B   OPR+1           ;GET LOW BYTE OF ADDRESS
        STA B   RDADD+1         ;SAVE
        LDX     RDADD           ;GET ADDRESS
        AND A   #,20            ;CHECK READ/WRITE
        BEQ     1$              ;ON READ - SKIP
        LDA A   WTBYTE          ;GET DATA
        STA A   0,X             ;STORE BYTE
1$:     LDA A   1,X             ;READ DATA AND SEND TO CPU
        STA A   WDTBA
        LDA B   0,X
        STA B   WDTBB
        RTS                     ;FINISHED
        ;
```

```
        .PAGE
        .SBTTL  SPECIAL MACRO DEFINITIONS
        ;
        ;
        .MACRO  SETBIT  I,J
        LDA A   I
        ORA A   #,J
        STA A   I
        .ENDM   SETBIT
        ;
        ;
        .MACRO  CLRBIT  I,J
        LDA A   I
        AND A   #,377-J
        STA A   I
        .ENDM   CLRBIT
        ;
        ;
        .MACRO  BITTST  I,J
        LDA A   I
        BIT A   #,J
        .ENDM   BITTST
        ;
        .PAGE
        .SBTTL  LIST PROCESSOR
        ;
LSTENT: LDA B   RSTAT0          ;RUNNING ?
        BPL     LSTGRP          ;IF NOT - DON'T INHIBIT PRINTING
        LDA B   STPFLG          ;STRIPPING CHARACTERS ?
        BEQ     LSTGRP          ;IF NOT - SKIP
        INC     CHRFLG          ;ELSE INHIBIT PRINTING
        ;
LSTGRP: CLI                     ;THIS RUN IN BACKGROUND
        AND A   #,177           ;ONLY 7-BIT ASCII
        STA A   CHARG           ;SAVE THE CHARACTER
        BSR     LISTPR          ;NOW DO PROCESS
        SEI                     ;HOLD INTERRUPTS AGAIN
        RTS                     ;FINISHED
        ;
        ; 1.    IF IN SINGLE CHAR MODE, THEN:
        ;               CLEAR SINGLE CHAR MODE
        ;               SCAN FOR CONTROL CHARACTERS, IF FOUND:
        ;                       CLEAR GOSUB
        ;                       END
        ;               DO GOSUB (IF DEFINED)
        ;               IF CHARACTER USED - END
        ;               ELSE GO TO 2.
        ;
LISTPR: LDA A   LSTAT           ;GET STATUS
        BPL     LIST.D          ;NOT IN SINGLE - SKIP
        AND A   #,177           ;CLEAR SINGLE MODE
        STA A   LSTAT           ;SAVE STATUS
        LDA A   CHARG           ;GET CHARACTER
        CMP A   #,40            ; A CONTROL ?
        BCC     LIST.A          ;IF NOT - SKIP
        LDX     #,0             ;ELSE CLEAR GOSUB
        STX     GOSUB
        BRA     LIST.B
LIST.A: LDX     GOSUB           ;GET PROCESS ADDRESS
        BEQ     LIST.B          ;IF UNDEFINED - SKIP
        JSR     0,X             ;DO IT
        BCC     LIST.D          ;CHARACTER NOT USED
LIST.B: RTS                     ;FINISHED
        ;
        ; 2.    DO COMMAND SCANNER
        ;       IF COMMAND IS FOUND - FINISHED
        ;
LIST.D: JSR     CMDSCN          ;SCAN FOR COMMANDS
        BCS     LIST.E          ;COMMAND FOUND - DONE
        LDA A   RSTAT0          ;ARE WE RUNNING
        BMI     LIST.F          ;IF SO - SKIP AHEAD
LIST.E: RTS                     ;FINISHED
        ;
        ; 3.    CHECK SCAN MODE, IF SET THEN:
        ;               IF CHAR IS (+) THEN:
        ;                       CLEAR SCAN
```

```
        ;                       SET BUILD
        ;                       RESET LIST BUFFER
        ;                       END
        ;
LIST.F: BITTST  LSTAT,100       ;IN SCAN MODE ?
        BEQ     LIST.J          ;IF NOT - SKIP
        LDA B   CHARG           ;GET CHARACTER
        CMP B   #,'+            ;IS IT A (+)  ?
        BNE     LIST.G          ;IF NOT - SKIP OUT
        AND A   #,277           ;ELSE CLEAR SCAN MODE
        ORA A   #,40            ;SET BUILD
        STA A   LSTAT           ;SAVE NEW STATUS
        CLR     SCHFLG          ;CLEAR SEEN FLAG
        CLR     LSTCNT          ;RESET BUFFER
        LDX     #,LSBUFF
        STX     LPNTR
LIST.G: RTS                     ;FINISHED
        ;
        ; 4.    CHECK BUILD, IF NOT SET THEN:
        ;               RESET BUFFER
        ;               IF CHAR IS (0-9) OR (-)
        ;                  OR FREE FORMAT   THEN:
        ;                       SET BUILD
        ;                       PUT CHAR IN BUFFER
        ;                       GOTO LCHECK
        ;                  ELSE: SET SCAN
        ;                        GO SCAN
        ;
LIST.J: BITTST LSTAT,40         ;BUILDING ?
        BNE     LIST.O          ;IF SO - SKIP
        CLR     SCHFLG          ;CLEAR SEEN FLAG
        CLR     LSTCNT          ;RESET BUFFER
        LDX     #,LSBUFF
        STX     LPNTR
        LDA B   FORMAT          ;IN FREE FORMAT ?
        BEQ     LIST.K          ;IF SO - SET UP BUILD
        LDA B   CHARG           ;GET CHARACTER
        CMP B   #,'-            ;IS IT A (-) ?
        BEQ     LIST.K          ;IF SO - SKIP
        CMP B   #,'0            ;A BCD CHARACTER ?
        BCS     LIST.L          ;<0 - SKIP
        CMP B   #,'9            ;>9 ?
        BHI     LIST.L          ;NOT A NUMBER - SKIP
LIST.K: ORA A   #,40            ;SET BUILD
        STA A   LSTAT           ;SAVE STATUS
        BRA     LIST.O          ;NOW PROCESS THIS CHARACTER
LIST.L: ORA A   #,100           ;SET SCAN
        STA A   LSTAT           ;SAVE STATUS
        BRA     LIST.F          ;GO SCAN
        ;
        ; 5.    BUILD IS SET
        ;               PUT CHARACTER IN BUFFER
        ;               CHECK BUFFFER
        ;
LIST.O: LDA A   FORMAT          ;CHECK FOR FREE FORMAT
        BEQ     LIST.Q          ;IF SO - SKIP
        LDA A   CHARG           ;GET CHARACTER
        CMP A   #,40            ;ANY CONTROLS ?
        BCC     LIST.X          ;IF NOT  SKIP
LIST.P: CLR     LSTCNT          ;RESET BUFFER
        CLR     SCHFLG          ;SEEN CHARACTER FLAG
        LDX     #,LSBUFF
        STX     LPNTR
        LDA A   LSTAT           ;SET STATUS FOR SCAN
        AND A   #,17            ;SAVE LOWER FOR GRAPHICS
        ORA A   #,100           ;SCANNING
        STA A   LSTAT
        RTS                     ;FINISHED
        ;
LIST.Q: LDA A   SCHFLG          ;SEEN A CHAR ?
        BNE     LIST.U          ;IF SO - SKIP
        LDA A   CHARG           ;ELSE SCAN THIS CHARACTER
        CMP A   #,'-            ;A - SIGN ?
        BEQ     LIST.R          ;IF SO - SAVE IT
        CMP A   #,'+            ;A + SIGN ?
        BEQ     LIST.T          ;STRIP +'S
```

```
        CMP A   #,40            ;SPACE ?
        BEQ     LIST.T          ;STRIP SPACES
        CMP A   #,'0            ;ONLY WANT NUMERALS
        BCS     LIST.P
        CMP A   #,':
        BCC     LIST.P
        BSR     LIST.R          ;SAVE CHARACTER
LIST.S: INC     SCHFLG          ;CHARACTER SEEN
LIST.T: RTS                     ;FINISHED
LIST.R: LDX     LPNTR           ;GET POINTER
        STA A   0,X             ;SAVE CHARACTER
        INX                     ;UPDATE POINTER
        STX     LPNTR
        INC     LSTCNT          ;ONE MORE CHARACTER
        RTS                     ;AND RETURN
        ;
LIST.U: LDA A   CHARG           ;GET CHARACTER
        CMP A   #,'-            ;A - SIGN ?
        BEQ     LIST.W          ;IF SO - SKIP
        CMP A   #,'+            ;A + SIGN ?
        BEQ     LIST.W          ;IF SO - SKIP
        CMP A   #,'0            ;WANT ONLY NUMERALS
        BCS     LIST.W
        CMP A   #,':
        BCC     LIST.W
        LDX     LPNTR           ;GET POINTER
        STA A   0,X             ;SAVE CHARACTER
        CPX     #,LSBUFF+8.     ;AT END OF BUFFER ?
        BEQ     LIST.V          ;IF SO - SKIP UPDATE
        INX                     ;ELSE UPDATE POINTER
        STX     LPNTR
        INC     LSTCNT          ;UPDATE COUNTER
LIST.V: RTS                     ;FINSHED
LIST.W: BSR     LCHE.A          ;GO EVALUATE AND PROCESS
        CLR     SCHFLG          ;CLEAR SEEN FLAG
        BRA     LIST.Q          ;RESCAN LAST CHARACTER
        ;
LIST.X: LDX     LPNTR           ;GET POINTER
        STA A   0,X             ;SAVE CHARACTER
        INX                     ;UPDATE ADDRESS
        STX     LPNTR
        INC     LSTCNT          ;UPDATE COUNT
        ;
        ;
LCHECK: LDA A   LSTCNT          ;GET COUNT
        CMP A   FORMAT          ;ENOUGH CHARACTERS ?
        BNE     LCHE.B          ;IF NOT - SKIP
LCHE.A: JSR     EVALN           ;GO EVALUATE DATA
        CLR     LSTCNT          ;CLEAR BUFFER
        LDX     #,LSBUFF
        STX     LPNTR
        LDX     GOSUB           ;GET PROCESS
        BEQ     LCHE.B          ;IF UNDEFINED - SKIP
        JSR     0,X             ;ELSE DO PROCESS
LCHE.B: RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  COMMAND SCANNER
        ;
CMDSCN: LDA B   CHARG           ;GET CHARACTER
        LDA A   PCHAR           ;GET PREVIOUS CHARACTER
        STA B   PCHAR           ;SAVE NEW CHARACTER
        CMP A   #,'$            ;OLD A ($) ?
        BNE     CMDS.D          ;IF NOT - SKIP
        CLRBIT  LSTAT,350       ;ALL NEW LIST CONTROL
        CMP B   #,141           ;ALLOW LOWER CASE COMMANDS
        BCS     CMD.UC
        CMP B   #,173
        BCC     CMD.UC
        SUB B   #,40            ;MAKE LOWER CASE UPPER CASE
CMD.UC: LDX     #,CMDTBL        ;GET COMMAND TABLE
        CMP B   0,X             ;RUN COMMAND ?
        BEQ     CMDS.B          ;IF SO - SKIP
        BITTST  RSTAT0,200      ;RUNNING ?
        BEQ     CMDS.D          ;IF NOT - SKIP
CMDS.A: INX                     ;GET PAST ADDRESS
```

```
        INX
        INX
        LDA A   0,X             ;END OF COMMANDS ?
        BEQ     CMDS.C          ;IF SO - SKIP
        CBA                     ;A COMMAND ?
        BNE     CMDS.A          ;NO - CONTINUE SCAN
CMDS.B: INX                     ;GET TO ADDRESS
        LDX     0,X             ;GET JUMP ADDRESS
        JSR     0,X             ;DO COMMAND
CMDS.C: SEC
        RTS                     ;FINISHED
CMDS.D: CLC                     ;NO COMMAND
        RTS                     ;FINISHED
        ;
        ;COMMAND AND JUMP ADDRESS TABLE
        ;
CMDTBL: FCC     <R>
        FDB     .$R
        FCC     <O>
        FDB     .$O
        FCC     <M>
        FDB     .$M
        FCC     <L>
        FDB     .$L
        FCC     <D>
        FDB     .$D
        FCC     <U>
        FDB     .$U
        FCC     <F>
        FDB     .$F
        FCC     <C>
        FDB     .$C
        FCC     <X>
        FDB     .$X
        FCC     <Y>
        FDB     .$Y
        FCC     <G>
        FDB     .$G
        FCC     <V>
        FDB     .$V
        FCC     <P>
        FDB     .$P
        FCC     <H>
        FDB     .$H
        FCC     <K>
        FDB     .$K
        FCC     <W>
        FDB     .$W
        FCC     <S>
        FDB     .$S
        FCC     <A>
        FDB     .$A
        FCC     <B>
        FDB     .$B
        FCC     <I>
        FDB     .$I
        FCC     <J>
        FDB     .$J
        FCC     <Z>
        FDB     .$Z
        .BYTE   0               ;TABLE TERMINATOR
        ;
        .PAGE
        .SBTTL  EVALUATE NUMBER ROUTINE
        ;
EVALN:  LDX     #,NSIGN         ;POINT TO VARIABLES
        LDA A   #,11.           ;11. BYTES TO CLEAR
EVAL.A: CLR     0,X             ;CLEAR BYTE
        INX                     ;UPDATE ADDRESS
        DEC A                   ;MORE ?
        BGT     EVAL.A          ;LOOP UNTIL ALL CLEARED
        LDX     #,NUMBER+3      ;SET UP POINTER
        STX     NPNTR
EVAL.B: LDX     LPNTR           ;GET STRING POINTER
        DEX                     ;POINT TO CHARACTER
        STX     LPNTR           ;SAVE
```

```
        LDA B   0,X             ;GET CHARACTER
        CMP B   #,'-            ; A (-) ?
        BNE     EVAL.C          ;IF NOT - SKIP
        LDA A   #,377           ;SIGN IS NEGATIVE
        STA A   NSIGN
        BRA     EVAL.F          ;SKIP AHEAD
EVAL.C: LDA A   NCNTR           ;GET COUNT
        CMP A   #,4             ;GOT FOUR YET ?
        BCC     EVAL.G          ;IF SO - SKIP
        LDX     NPNTR           ;GET POINTER
        CMP B   #,40            ;A SPACE ?
        BEQ     EVAL.F          ;IF SO - SKIP
        SUB B   #,'0            ;MAKE BCD
        BMI     EVAL.D          ;IF NOT -  SKIP
        CMP B   #,9.            ;MUST BE A DIGIT
        BLS     EVAL.E          ;IF SO - SKIP
EVAL.D: BRA     EVAL.F
EVAL.E: STA B   0,X             ;SAVE CHARACTER
EVAL.F: DEX                     ;UPDATE POINTER
        STX     NPNTR
        INC     NCNTR           ;UPDATE COUNT
EVAL.G: DEC     LSTCNT          ;ANY MORE ?
        BGT     EVAL.B          ;LOOP UNTIL DONE
        ;
        ;
        .PAGE
        .SBTTL  4-DIGIT BCD TO BINARY CONVERSION
        ;
BCD4BN: LDA A   NUMBER+1        ;PACK 10 & 100'S DIGITS
        ASL A                   ;SHIFT INTO POSITION
        ASL A
        ASL A
        ASL A
        ORA A   NUMBER+2
        LDA B   NUMBER          ;PACK MSD'S NOW
        STA B   T1              ;SAVE HIGH ORDER
        STA A   T2              ;SAVE LOW ORDER
        LDX     #,BCDTBL        ;GET POINTER TO TABLE
        LDA A   #,12.           ;SET UP BIT COUNTER
        STA A   T0
        LDA A   NUMBER+3        ;INITIALIZE RESULT TO 1'S DIGIT
        CLR B
BCDLP:  ASR     T1
        ROR     T2
        BCC     NOADD           ;IF BIT CLEAR - NO UPDATE
        ADD A   0,X             ;ELSE UPDATE RESULT
        ADC B   14,X
NOADD:  INX                     ;UPDATE TABLE POINTER
        DEC     T0              ;ANY BITS LEFT ?
        BGT     BCDLP           ;LOOP UNTIL DONE
        TST     NSIGN           ;NEGATIVE ?
        BEQ     BCD4.A          ;IF POSITIVE - SKIP
        COM A
        COM B
        ADD A   #,1
        ADC B   #,0
BCD4.A: STA A   NUMBER+1        ;SAVE RESULT
        STA B   NUMBER
        RTS                     ;FINISHED
        ;
        ;BCD TO BINARY CONVERSION TABLE
        ;
        .NLIST  BIN
BCDTBL: .BYTE   12,24,50,120
        .BYTE   144,310,220,40
        .BYTE   350,320,240,100
        ;
        .BYTE   0,0,0,0
        .BYTE   0,0,1,3
        .BYTE   3,7,17,37
        .LIST   BIN
        ;
        .PAGE
        .SBTTL  $R COMMAND
        ;
.$R:    SETBIT  RSTAT0,200      ;SAY RUNNING
```

```
        SETBIT  LSTAT,200       ;CHARACTER MODE
        CLR     FORMAT          ;SET FREE FORMAT AS DEFAULT
        LDX     #,.$RB          ;POINTER TO PROCESS
        STX     GOSUB
        RTS                     ;FINISHED
        ;
.$RB:   LDA A   CHARG           ;GET CHARACTER
        SUB A   #,'0            ;MAKE BCD
        BCS     .$RC            ;IF NOT - SKIP
        CMP A   #,9.            ;BCD ?
        BHI     .$RC            ;IF NOT - SKIP
        STA A   FORMAT          ;SAVE NEW FORMAT
        SEC                     ;CHARACTER USED
        BRA     .$RD
.$RC:   CLC                     ;CHARACTER NOT USED
.$RD:   LDX     #,0             ;NO MORE HERE
        STX     GOSUB
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  $O COMMAND
        ;
.$O:    BRA     .$OC            ;SET UP FOR NEXT ENTRY
        ;
.$OA:   LDX     NUMBER          ;GET VALUE
        STX     XTABLE+22       ;SAVE XORIGIN
        LDX     #,.$OB          ;SET FOR NEXT ENTRY
        BRA     .$OD
        ;
.$OB:   LDX     NUMBER          ;GET VALUE
        STX     YTABLE+22       ;SAVE YORIGIN
.$OC:   LDX     #,.$OA          ;SET UP NEXT ENTRY
.$OD:   STX     GOSUB
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  $M COMMAND
        ;
.$M:    CLRBIT  RSTAT0,160      ;RELATIVE TO ORIGIN
        BRA     .$MC            ;GO SET UP FOR NEXT ENTRY
        ;
.$MA:   LDX     NUMBER          ;GET VALUE
        STX     XTABLE+4        ;SAVE AS X
        LDX     #,.$MB          ;SET UP FOR NEXT ENTRY
        BRA     .$MD
        ;
.$MB:   LDX     NUMBER          ;GET VALUE
        STX     YTABLE+4        ;SAVE AS Y
        JSR     PROC            ;GO PROCESS
.$MC:   LDX     #,.$MA          ;SET UP NEXT ENTRY POINT
.$MD:   STX     GOSUB
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  $L COMMAND
        ;
.$L:    LDA A   RSTAT0          ;GET STATUS
        AND A   #,217           ;CLEAR MODES
        ORA A   #,100           ;SET RELATIVE TO CURRENT POSITION
        STA A   RSTAT0          ;SAVE
        BRA     .$LC
        ;
.$LA:   LDX     NUMBER          ;GET VALUE
        STX     XTABLE+4        ;SAVE AS X
        LDX     #,.$LB          ;SET UP FOR NEXT ENTRY
        BRA     .$LD
        ;
.$LB:   LDX     NUMBER          ;GET VALUE
        STX     YTABLE+4        ;SAVE AS Y
        JSR     PROC            ;GO PROCESS
.$LC:   LDX     #,.$LA          ;SET UP FOR NEXT ENTRY
.$LD:   STX     GOSUB
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  $D, $U, $V, $P, $S, AND $K COMMANDS
```

```
        ;
.$D:    SETBIT  RSTAT0,2        ;INDICATE PEN DOWN
        LDA A   #,1             ;SET UP PLOT CONTROL
        STA A   PTABLE+11
        JSR     PLOTXY          ;PLOT THE POINT
        RTS                     ;FINISHED
        ;
        ;
.$U:    CLRBIT  RSTAT0,2        ;INDICATE PEN UP
        RTS                     ;FINISHED
        ;
        ;
.$V:    CLRBIT  RSTAT0,10       ;VECTOR MODE
        RTS                     ;FINISHED
        ;
        ;
.$P:    SETBIT  RSTAT0,10       ;POINT MODE
        RTS                     ;FINISHED
        ;
        ;
.$S:    CLRBIT  RSTAT0,200      ;NOT RUNNING
        RTS                     ;FINISHED
        ;
        ;
.$K     =GRPCLR                 ;CLEAR GRAPHICS DIPLAY
        ;
        .PAGE
        .SBTTL  $A, $B, $I, AND $J COMMANDS
        ;
.$A:    LDA A   #,377           ;TURN ON SCREEN
        STA A   GRAPH
        INC     UPDFLG          ;TELL SYSTEM
        RTS
        ;
        ;
.$B:    CLR     GRAPH           ;TURN OFF SCREEN
        INC     UPDFLG          ;TELL SYSTEM
        RTS
        ;
        ;
.$I:    LDA A   #,377           ;INHIBIT PRINTING
        STA A   STPFLG
        RTS
        ;
        ;
.$J:    CLR     STPFLG          ;ENABLE PRINTING
        RTS
        ;
        .PAGE
        .SBTTL  $H COMMAND
        ;
.$H:    SETBIT  RSTAT0,4        ;HOLD FLAG
        LDX     #,HLDINT        ;PLACE NEW INTERRUPT VECTOR
        STX     IRQ13
1$:     CMP B   CHARG           ;AN H ?
        BEQ     2$              ;IF SO - SKIP
        LDA B   CHARG           ;ELSE USE AN H
        BRA     3$
2$:     LDA B   #,40            ;OR A SPACE
3$:     STA B   CTLINE+64       ;FOR FLASHING
        LDX     #,10000.        ;LOOP COUNTER
4$:     BITTST  RSTAT0,4        ;CLEARED YET /
        BEQ     5$              ;IF SO - SKIP
        DEX                     ;CHECK COUNTER
        BNE     4$              ;LOOP FOR A WHILE
        BRA     1$
5$:     LDX     #,HEREIS        ;RESTORE INTERRUPT VECTOR
        STX     IRQ13
        LDA A   #,40            ;CLEAR FLASHING CHARACTER
        STA A   CTLINE+64
        RTS                     ;FINISHED
        ;
HLDINT: LDA A   NPRDTA          ;CLEAR INTERRUPT FLAG
        CLRBIT  RSTAT0,4        ;CLEAR HOLD FLAG
        RTI                     ;RETURN FROM INTERRUPT
        ;
```

```
        ;
.$W:
.$Z:    RTS                     ;DUMBY ROUTINE
        ;
        .PAGE
        .SBTTL  $X COMMAND
        ;
.$X:    LDX     #,.$XA          ;SET FOR NEXT ENTRY
        STX     GOSUB
        LDA A   RSTAT0          ;GET STATUS
        AND A   #,217           ;CLEAR OTHER MODES
        ORA A   #,40            ;SET XSTEP MODE
        STA A   RSTAT0          ;SAVE
        RTS                     ;FINISHED
        ;
        ;
.$XA:   LDX     NUMBER          ;GET VALUE
        STX     XTABLE+4        ;SAVE
        LDX     #,.$XB          ;SET FOR NEXT ENTRY
        STX     GOSUB
        RTS                     ;FINISHED
        ;
        ;
.$XB:   LDX     NUMBER          ;GET NUMBER
        STX     YTABLE+4        ;SAVE
        JSR     PROC            ;DO PROCESS
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  $Y COMMAND
        ;
.$Y:    LDX     #,.$YA          ;NEXT ENTRY POINT
        STX     GOSUB
        LDA A   RSTAT0          ;GET STATUS
        AND A   #,217           ;CLEAR ALL MODES
        ORA A   #,20            ;SET YSTEP MODE
        STA A   RSTAT0          ;SAVE
        RTS                     ;FINISHED
        ;
        ;
.$YA:   LDX     NUMBER          ;GET VALUE
        STX     YTABLE+4        ;SAVE
        LDX     #,.$YB          ;NEXT ENTRY POINT
        STX     GOSUB
        RTS                     ;FINISHED
        ;
        ;
.$YB:   LDX     NUMBER          ;GET VALUE
        STX     XTABLE+4        ;SAVE
        JSR     PROC            ;DO PROCESS
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  PROCESS ROUTINE
        ;
PROC:   BITTST  RSTAT0,40       ;AUTO X MODE ?
        BEQ     PROC.A          ;IF NOT - SKIP
        BSR     VAL.X           ;FXVAL=XVAL+XSTEP
        BSR     ORG.Y           ;FYVAL=YORG+YSTEP
        BRA     PROC.D
        ;
PROC.A: BIT A   #,20            ;AUTO Y MODE ?
        BEQ     PROC.B          ;IF NOT - SKIP
        BSR     ORG.X           ;FXVAL=XORG+XSTEP
        BSR     VAL.Y           ;FYVAL=YVAL+YSTEP
        BRA     PROC.D
        ;
PROC.B: BIT A   #,100           ;RELATIVE TO CURRENT ?
        BEQ     PROC.C          ;IF NOT - SKIP
        BSR     VAL.X           ;FXVAL=XVAL+XSTEP
        BSR     VAL.Y           ;FYVAL=YVAL+YSTEP
        BRA     PROC.D
        ;
PROC.C: BSR     ORG.X           ;FXVAL=XORG+XSTEP
        BSR     ORG.Y           ;FYVAL=YORG+YSTEP
        ;
```

```
PROC.D: LDA A   RSTAT0          ;USE PEN STATUS FOR INTENSITY
        AND A   #,2
        ASR A
        STA A   PTABLE+11
        LDA A   #,40            ;INHIBIT VECTOR UPDATING
        STA A   OPR
        BITTST  RSTAT0,10       ;IN VECTOR MODE ?
        BNE     PROC.E          ;IF NOT - SKIP VECTOR DRAWING
        JSR     VECTOR          ;GO DO VECTOR DRAWING
PROC.E: LDX     XTABLE+2        ;XVAL=FXVAL
        STX     XTABLE
        STX     XTABLE+10       ;VALX=FXVAL
        LDX     YTABLE+2        ;YVAL=FYVAL
        STX     YTABLE
        STX     YTABLE+10       ;VALY=FYVAL
        BITTST  RSTAT0,10       ;POINT MODE ?
        BEQ     PROC.F          ;IF NOT - FINISHED
        JSR     PLOTXY          ;PLOT A POINT
PROC.F: RTS                     ;FINISHED
        ;
        ;
VAL.X:  LDX     #,XTABLE
        BRA     VALSTP
VAL.Y:  LDX     #,YTABLE
VALSTP: LDA A   1,X
        LDA B   0,X
        BRA     ORGS.A
        ;
ORG.X:  LDX     #,XTABLE
        BRA     ORGSTP
ORG.Y:  LDX     #,YTABLE
ORGSTP: LDA A   23,X
        LDA B   22,X
ORGS.A: ADD A   5,X
        ADC B   4,X
        STA A   3,X
        STA B   2,X
        RTS
        ;
```

```
        .PAGE
        .SBTTL  $F COMMAND
        ;
.$F:    LDX     #,.$FB          ;NEXT ENTRY POINT
        STX     GOSUB
.$FA:   SETBIT  LSTAT,200       ;RE-ENABLE CHARACTER MODE
        SEC                     ;CHARACTER USED
        RTS                     ;FINISHED
        ;
.$FB:   LDA A   CHARG           ;GET CHARACTER
        CMP A   #,141           ;CONVERT LOWER CASE TO UPPER CASE
        BCS     1$
        CMP A   #,173
        BCC     1$
        SUB A   #,40
1$:     CMP A   #,'H            ;HORIZONTAL MODE ?
        BNE     .$FC            ;IF NOT - SKIP
        CLRBIT  LSTAT,1         ;HORIZONTAL MODE
        BRA     .$FA            ;LEAVE
        ;
.$FC:   CMP A   #,'V            ;VERTICAL MODE ?
        BNE     .$FD            ;IF NOT - SKIP
        SETBIT  LSTAT,1         ;VERTICAL MODE
        BRA     .$FA
        ;
.$FD:   CMP A   #,'L            ;LOWER CASE ?
        BNE     .$FE            ;IF NOT SKIP
        CLRBIT  LSTAT,4         ;SAY LOWER CASE
        BRA     .$FA
        ;
.$FE:   CMP A   #,'G            ;GREEK LETTERS ?
        BNE     1$              ;IF NOT - SKIP
        SETBIT  LSTAT,4         ;INDICATE GREEK
        BRA     .$FA
        ;
1$:     CMP A   #,'/            ;SLASHED ZERO ?
        BEQ     .$FA            ;NO-OP
        ;
        CMP A   #,'\            ;NON SLASHED ZERO ?
        BEQ     .$FA            ;NO-OP
        ;
.$FF:   JSR     XYSIZE          ;FIND SIZING
        BCC     .$FZ            ;BAD SIZING
        LDA B   LSTAT           ;GET STATUS
        BIT B   #,10            ;NEED X OR Y SIZE ?
        BNE     .$FG            ;IF Y - SKIP
        ORA B   #,10            ;NEXT IS Y
        STA B   LSTAT           ;SAVE
        STA A   XTABLE+21       ;SAVE NEW XSIZE
        BRA     .$FA            ;DONE
.$FG:   AND B   #,367           ;NEXT IS X
        STA B   LSTAT           ;SAVE
        STA A   YTABLE+21       ;SAVE NEW YSIZE
        BRA     .$FA            ;DONE
.$FZ:   LDX     #,0             ;DONE
        STX     GOSUB
        CLC                     ;CHARACTER NOT USED
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  $C COMMAND
        ;
.$C:    JSR     .$U             ;ALWAYS RAISE PEN
        LDX     #,CHRSCN        ;NEXT ENTRY POINT
        STX     GOSUB
        LDA A   LSTAT           ;GET STATUS
        ORA A   #,200           ;ENABLE CHARACTER MODE
        AND A   #,375           ;CHARACTER SCAN
        STA A   LSTAT
        LDA A   RSTAT0          ;GET MODE STATUS
        AND A   #,217           ;CLEAR CURRENT MODE
        ORA A   #,100           ;$L MODE
        STA A   RSTAT0          ;SAVE
        RTS                     ;FINISHED
        ;
CHRSCN: LDA B   CHARG           ;GET CHARACTER
```

```
        SETBIT  LSTAT,200       ;RE-ENABLE CHARACTER MODE
        BIT A   #,2             ;PLOTTING CHARACTERS ?
        BNE     CHRS.B          ;IF SO - SKIP
        CMP B   #,'+            ;A + ?
        BNE     CHRS.A          ;IF NOT - SKIP
        ORA A   #,2             ;SET PLOTTING FLAG
        STA A   LSTAT           ;SAVE
        SEC                     ;CHARACTER USED
        RTS                     ;FINISHED
CHRS.A: CLC                     ;CHARACTER NOT USED
        RTS                     ;FINISHED
        ;
CHRS.B: BITTST  LSTAT,4         ;LOWER/GREEK
        BEQ     CHRS.D          ;IF LOWER - SKIP
        CMP B   #,140           ;IN RANGE OF GREEKS ?
        BCS     CHRS.D          ;IF NOT - SKIP
        BNE     CHRS.C          ;IF NOT - SKIP
        LDA B   #,37            ;OUT OF PLACE CHARACTER
        BRA     CHRS.D
CHRS.C: SUB B   #,141           ;GET TO GREEKS
CHRS.D: BITTST  RSTAT0,10       ;VECTOR MODE ?
        BNE     CHRS.L          ;IF NOT - POINT MODE
CHRSCX: CLR A                   ;FIND ENTRY IN TABLE
        ASL B                   ;TWO BYTES PER POINTER
        ROL A
        ADD B   #,CHTBLE&377
        ADC A   #,CHTBLE&177400/400
        STA B   CHPNTR+1        ;SAVE POINTER
        STA A   CHPNTR
        LDX     CHPNTR          ;GET POINTER
        LDX     0,X             ;GET ADDRESS OF STRING
        STX     CHPNTR          ;SAVE ADDRESS
CHRS.E: LDA A   0,X             ;GET VECTOR
        CMP A   #,END           ;END ?
        BEQ     CHRS.I          ;END OF THIS CHARACTER
        CMP A   #,DWN           ;A PEN CONTROL ?
        BNE     CHRS.F          ;IF NOT SKIP
        JSR     .$D             ;MOVE PEN DOWN
        BRA     CHRS.H
CHRS.F: CMP A   #,UP            ;PEN CONTROL ?
        BNE     CHRS.G          ;IF NOT - SKIP
        JSR     .$U             ;LIFT PEN
        BRA     CHRS.H
CHRS.G: JSR     GETXST          ;GET X VECTOR
        JSR     GETYST          ;GET Y VECTOR
        JSR     PROC            ;GO PROCESS
CHRS.H: LDX     CHPNTR          ;UPDATE POINTER
        INX
        STX     CHPNTR
        BRA     CHRS.E          ;GET NEXT VECTOR
CHRS.I: SEC                     ;USED CHARACTER
        RTS                     ;FINISHED
        ;
        ;POINT MODE
        ;
CHRS.L: CLR A                   ;COMPUTE ADDRESS IN CHROM
        ASL B
        ROL A
        ASL B
        ROL A
        ASL B
        ROL A
        ASL B
        ROL A
        ADD B   #,CHROM2&377
        ADC A   #,CHROM2&177400/400
        ADD B   #,11.           ;11 BYTE OFFSET
        ADC A   #,0
        STA B   CHPNTR+1        ;SAVE POINTER
        STA A   CHPNTR
        SETBIT  RSTAT0,2        ;SAY PEN DOWN (PHYSICALLY ITS UP !)
        LDA A   #,10            ;PRESET PXCNT
        STA A   XTABLE+25
        LDA A   #,9.            ;PRESET PYCNT
        STA A   YTABLE+25
        LDA A   #,12.           ;PRESET YCNT
```

```
        STA A   YTABLE+24
        LDX     CHPNTR          ;GET POINTER TO POINTS
CHRS.M: LDA A   0,X             ;GET DOT PATTERN
        BEQ     CHRS.P          ;IF NO POINTS - SKIP
        STA A   PROW            ;SAVE PATTERN
        LDA A   #,10            ;DOT COUNT
        STA A   XTABLE+24
CHRS.N: ASL     PROW            ;DOT HERE ?
        BCC     CHRS.O          ;IF NOT SKIP
        JSR     PGTXST          ;GET XSTEP VALUE
        JSR     PGTYST          ;GET YSTEP VALUE
        JSR     PROC            ;GO PROCESS POINT
CHRS.O: DEC     XTABLE+24       ;DONE YET ?
        BGT     CHRS.N          ;LOOP UNTIL ROW DONE
        LDX     CHPNTR          ;UPDATE POINTER
CHRS.P: DEX
        STX     CHPNTR
        DEC     YTABLE+24       ;ALL ROWS DONE ?
        BGT     CHRS.M          ;LOOP UNTIL DONE
        JSR     .$U             ;LIFT PEN
        CLR     XTABLE+24       ;SET UP FOR FINAL MOVE
        JSR     PGTXST          ;SET XSTEP VALUE
        LDA A   #,9.            ;SET UP YCNT
        STA A   YTABLE+24
        JSR     PGTYST          ;SET YSTEP VALUE
        JSR     PROC            ;GO PROCESS
        SEC                     ;CHARACTER USED
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  XYSIZE ROUTINE
        ;
XYSIZE: LDA A   CHARG           ;GET CHARACTER
        SUB A   #,'0            ;MAKE INTO BCD
        BCS     XYSI.Z          ;ERROR - SKIP
        CMP A   #,9.
        BHI     XYSI.Z          ;ERROR - SKIP
        CLR B                   ;COMPUTE TABLE ADDRESS
        ADD A   #,XYTBL&377     ;LOW ORDER
        ADC B   #,XYTBL&177400/400
        STA A   T3              ;SAVE ADDRESS
        STA B   T2
        LDX     T2              ;GET ADDRESS
        LDA A   0,X             ;GET SIZE SPECIFICATION
        SEC                     ;GOOD RESULT
        RTS                     ;FINISHED
XYSI.Z: CLC                     ;BAD SPECIFIER
        RTS                     ;FINISHED
        ;
XYTBL:  .BYTE   1,2,3,4,5,6,7,8.,9.,10.
        ;
        .PAGE
        .SBTTL  X/Y GET ROUTINES
        ;
XGET:   LDA A   0,X             ;GET VECTOR
        ASR A                   ;SHIFT INTO POSITION
        ASR A
        ASR A
        ASR A
        BRA     XGET.A
YGET:   LDA A   0,X             ;GET VECTOR
XGET.A: CLR B                   ;HIGH ORDER = 0
        AND A   #,17            ;MASK JUNK
        BIT A   #,10            ;NEGATIVE VECTOR ?
        BEQ     XGET.Z          ;IF NOT - FINISHED
        AND A   #,7             ;SAVE ONLY MAGITUDE
        NEG A                   ;TWO'S COMPLEMENT
        LDA B   #,377           ;SIGN EXTEND
XGET.Z: RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  GENERATE X STEP ROUTINE
        ;
        ;POINT MODE
        ;
PGTXST: LDA A   XTABLE+21       ;SET UP FOR MULTIPLY
```

```
        STA A   T0
        LDA A   #,10
        STA A   T1
        LDA B   XTABLE+24       ;GET LOCATION TO PLOT
        LDA A   XTABLE+25       ;PREVIOUS POINT
        STA B   XTABLE+25       ;SAVE FOR NEXT TIME
        SBA                     ;PXCNT-XCNT
        CLR B                   ;SET UP <B,A>
        TST A
        BPL     GETX.X          ;IF PLUS - SKIP
        COM B                   ;ELSE SIGN EXTEND
        BRA     GETX.X          ;GO FINISH
        ;
        ;VECTOR MODE
        ;
GETXST: LDA A   XTABLE+21       ;GET X SIZE
        STA A   T0              ;SAVE IN TEMPORARY
        LDA A   #,10            ;BIT COUNT
        STA A   T1              ;SAVE
        BSR     XGET            ;GET X VECTOR
GETX.X: BSR     MULTP           ;GO MULTIPLY <B,A>*XSIZE
        ROR     LSTAT           ;HORIZONTAL ?
        BCS     GETX.A          ;IF NOT - SKIP
        ROL     LSTAT           ;RESTORE LSTAT
        STA A   XTABLE+5                ;SAVE RESULT
        STA B   XTABLE+4
        RTS
GETX.A: ROL     LSTAT           ;RESTORE LSTAT
        STA A   YTABLE+5                ;SAVE RESULT
        STA B   YTABLE+4
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  GENERATE Y STEP ROUTINE
        ;
        ;POINT MODE
        ;
PGTYST: LDA A   YTABLE+21       ;SET UP FOR MULTIPLY
        STA A   T0
        LDA A   #,10
        STA A   T1
        LDA B   YTABLE+24       ;CURRENT POSITION TO PLOT
        LDA A   YTABLE+25       ;PREVIOUS POSITION
        STA B   YTABLE+25       ;SAVE FOR NEXT TIME
        SBA                     ;FIND RELATIVE MOVE
        CLR B                   ;SET UP <B,A>
        TST A
        BPL     GETY.X          ;IF PLUS - SKIP
        COM B                   ;ELSE SIGN EXTEND
        BRA     GETY.X
        ;
        ;VECTOR MODE
        ;
GETYST: LDA A   YTABLE+21       ;GET Y SIZE
        STA A   T0              ;SAVE IN TEMPORARY
        LDA A   #,10            ;BIT COUNT
        STA A   T1              ;SAVE
        BSR     YGET            ;GET Y VECTOR
GETY.X: BSR     MULTP           ;GO MULTIPLY <B,A>*YSIZE
        ROR     LSTAT           ;HORIZONTAL ?
        BCS     GETY.A          ;IF NOT - SKIP
        ROL     LSTAT           ;RESTORE LSTAT
        STA A   YTABLE+5        ;SAVE RESULT
        STA B   YTABLE+4
        RTS                     ;FINISHED
GETY.A: ROL     LSTAT           ;RESTORE LSTAT
        COM A                   ;NEED NEGATIVE
        COM B
        ADD A   #,1
        ADC B   #,0
        STA A   XTABLE+5        ;SAVE RESULT
        STA B   XTABLE+4
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  MULTIPLY ROUTINE
```

```
        ;
MULTP:  STA A   T3              ;SAVE IN TEMPORARY
        STA B   T2
        CLR A                   ;INITIALIZE
        CLR B
MULT.A: ASL A                   ;SHIFT RESULT
        ROL B
        ASL T0                  ;BIT SET ?
        BCC     MULT.B          ;IF NOT - SKIP
        ADD A   T3              ;UPDATE RESULT
        ADC B   T2
MULT.B: DEC     T1              ;FINISHED ?
        BGT     MULT.A          ;LOOP UNTIL DONE
        RTS                     ;FINISHED - RESULT <B,A>
        ;
        .PAGE
        .SBTTL  $G COMMAND
        ;
.$G:    LDX     GOSUB           ;GET JUMP LOCATION
        STX     SVGSUB          ;SAVE FOR RESTORATION
        LDX     #,.$GA          ;NEXT ENTRY POINT
        STX     GOSUB
        SETBIT  LSTAT,200       ;ENABLE CHARACTER MODE
        RTS                     ;FINISHED
        ;
.$GA:   LDX     SVGSUB          ;RESTORE JUMP ADDRESS
        STX     GOSUB
        LDA B   CHARG           ;GET CHARACTER
        SUB B   #,'0            ;MAKE BCD
        BCS     .$GZ            ;BAD CHARACTER - DONE
        CMP B   #,9.
        BHI     .$GZ            ;BAD CHARACTER - DONE
        PSH B                   ;SAVE CHARACTER
        JSR     .$U             ;RAISE PEN
        LDX     XTABLE+4        ;SAVE XSTEP
        STX     XTABLE+26
        LDX     YTABLE+4        ;SAVE YSTEP
        STX     YTABLE+26
        LDA A   RSTAT0          ;GET STATUS
        STA A   SVSTAT          ;SAVE FOR RESTORATION
        AND A   #,207           ;MAKE VECTOR
        ORA A   #,100           ;AND RELATIVE ($L)
        STA A   RSTAT0          ;SAVE NEW STATUS
        PUL B                   ;GET CHARACTER
        ADD B   #,200           ;GET TO GCH'S
        JSR     CHRSCX          ;GO DO PLOTTING
        LDA A   SVSTAT          ;GET OLD STATUS
        STA A   RSTAT0          ;RESTORE MODE
        LDX     YTABLE+26       ;RESTORE YSTEP
        STX     YTABLE+4
        LDX     XTABLE+26       ;RESTORE XSTEP
        STX     XTABLE+4
        SEC                     ;CHARACTER USED
        RTS                     ;FINISHFED
.$GZ:   CLC                     ;CHARACTER NOT USED
        RTS                     ;FINISHED
        ;
        .PAGE
        .SBTTL  CHARACTER TABLES
        ;
        .=150000
        CHTBLE  =.              ;START OF CHARACTER VECTOR TABLE
        ;
        .IRP    X,<CH0,CH1,CH2,CH3,CH4,CH5,CH6,CH7,CH8,CH9>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH10,CH11,CH12,CH13,CH14,CH15,CH16,CH17,CH18,CH19>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH20,CH21,CH22,CH23,CH24,CH25,CH26,CH27,CH28,CH29>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH30,CH31,CH32,CH33,CH34,CH35,CH36,CH37,CH38,CH39>
```

```
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH40,CH41,CH42,CH43,CH44,CH45,CH46,CH47,CH48,CH49>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH50,CH51,CH52,CH53,CH54,CH55,CH56,CH57,CH58,CH59>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH60,CH61,CH62,CH63,CH64,CH65,CH66,CH67,CH68,CH69>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH70,CH71,CH72,CH73,CH74,CH75,CH76,CH77,CH78,CH79>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH80,CH81,CH82,CH83,CH84,CH85,CH86,CH87,CH88,CH89>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH90,CH91,CH92,CH93,CH94,CH95,CH96,CH97,CH98,CH99>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH100,CH101,CH102,CH103,CH104,CH105,CH106,CH107,CH108,CH109>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH110,CH111,CH112,CH113,CH114,CH115,CH116,CH117,CH118,CH119>
        FDB     X
        .ENDM
        ;
        .IRP    X,<CH120,CH121,CH122,CH123,CH124,CH125,CH126,CH127>
        FDB     X
        .ENDM
        ;
        .IRP    X,<GCH0,GCH1,GCH2,GCH3,GCH4,GCH5,GCH6,GCH7,GCH8,GCH9>
        FDB     X
        .ENDM
        ;
        .PAGE
        DWN     =       10
        UP      =       200
        END     =       210
        ;
        .NLIST  BIN
GCH0:   .BYTE   4,DWN,254,54,44,244,UP,14,END
GCH1:   .BYTE   DWN,3,16,3,260,140,260,UP,END
GCH2:   .BYTE   273,DWN,140,6,340,16,UP,63,END
GCH3:   .BYTE   3,DWN,273,73,63,263,UP,13,END
GCH4:   .BYTE   261,DWN,42,40,52,12,252,240,242,2,UP,71,END
GCH5:   .BYTE   DWN,3,220,40,220,16,220,40,220,3,UP,END
GCH6:   .BYTE   60,DWN,343,16,143,UP,260,END
GCH7:   .BYTE   260,DWN,153,6,353,UP,60,END
GCH8:   .BYTE   13,DWN,66,340,76,UP,3,END
GCH9:   .BYTE   3,DWN,76,340,66,UP,13,END
CH0:    .BYTE   160,DWN,304,220,231,12,31,20,104,UP,34,END
CH1:    .BYTE   33,DWN,21,7,60,31,231,260,60,31,11,231,260,UP,140,END
CH2:    .BYTE   24,DWN,20,31,16,220,2,125,UP,34,END
CH3:    .BYTE   107,DWN,221,220,231,12,73,11,231,220,221,1,21,20,UP,133,END
CH4:    .BYTE   100,DWN,220,242,1,60,260,1,42,20,UP,116,END
CH5:    .BYTE   60,DWN,20,21,221,240,221,1,104,11,240,221,UP,137,11,END
CH6:    .BYTE   23,DWN,21,31,13,3,21,20,31,16,UP,43,END
CH7:    .BYTE   22,DWN,4,42,20,52,14,252,220,242,2,120,UP,54,END
CH8:    .BYTE   25,DWN,14,31,20,21,UP,111,END
CH9:    .BYTE   20,DWN,6,13,20,42,252,73,21,UP,51,END
CH10:   .BYTE   27,1,DWN,52,14,252,42,20,52,UP,40,END
CH11:   .BYTE   33,DWN,7,14,40,21,3,13,31,UP,60,END
CH12:   .BYTE   25,DWN,20,15,104,1,UP,55,END
CH13:   .BYTE   127,DWN,220,1,11,220,231,31,20,220,252
        .BYTE   31,60,31,231,220,UP,100,END
CH14:   .BYTE   22,DWN,1,42,20,52,11,252,220,242,UP,172,END
CH15:   .BYTE   24,DWN,21,20,15,5,40,15,5,40,UP,35,END
```

```
CH16:   .BYTE   33,DWN,6,42,20,52,11,252,220,242,UP,172,END
CH17:   .BYTE   105,DWN,52,11,252,220,242,1,42,100,UP,35,END
CH18:   .BYTE   24,DWN,21,40,15,5,60,UP,35,END
CH19:   .BYTE   25,DWN,20,14,31,20,21,3,21,UP,55,END
CH20:   .BYTE   60,DWN,6,20,31,12,231,240,221,2,21,20,2,UP,137,11,END
CH21:   .BYTE   24,DWN,20,114,20,UP,320,DWN,104,UP,54,END
CH22:   .BYTE   24,DWN,31,12,31,40,21,2,21,UP
        .BYTE   261,DWN,17,11,UP,103,END
CH23:   .BYTE   45,DWN,231,13,31,20,21,2,12,31,20,21,3,221,UP,55,END
CH24:   .BYTE   20,DWN,20,2,221,2,42,40,52,12,231,12,20,UP,20,END
CH25:   .BYTE   22,DWN,52,7,1,100,UP,37,11,END
CH26:   .BYTE   24,DWN,140,242,52,252,UP,72,END
CH27:   .BYTE   66,DWN,252,52,242,140,UP,34,END
CH28:   .BYTE   46,DWN,42,52,242,17,11,UP,100,END
CH29:   .BYTE   106,DWN,0,UP,272,DWN,140,UP,272,DWN,0,UP,112,END
CH30:   .BYTE   160,DWN,360,104,20,220,304,160,UP,37,31,END
CH31:   .BYTE   26,DWN,21,20,52,20,21,UP,353,DWN,21,20,52,20,21,UP,33,END
CH32:   .BYTE   160,20,END
CH33:   .BYTE   100,DWN,0,UP,3,DWN,5,UP,117,11,END
CH34:   .BYTE   66,DWN,2,UP,60,DWN,12,UP,56,END
CH35:   .BYTE   60,DWN,7,1,UP,40,DWN,17,11,UP,305
        .BYTE   DWN,140,UP,352,DWN,140,UP,33,END
CH36:   .BYTE   21,DWN,120,21,1,221,300,221,1,21,120
        .BYTE   260,1,17,11,UP,100,END
CH37:   .BYTE   27,DWN,21,31,231,221,UP,16,DWN,146,UP
        .BYTE   16,DWN,231,221,21,31,UP,31,END
CH38:   .BYTE   163,DWN,273,240,221,1,104,1,221,240,231,11,156,UP,20,END
CH39:   .BYTE   26,DWN,21,1,UP,157,11,END
CH40:   .BYTE   100,DWN,242,4,42,UP,117,11,END
CH41:   .BYTE   100,DWN,42,4,242,UP,117,11,END
CH42:   .BYTE   104,DWN,60,260,63,273,3,13,263,73
        .BYTE   260,60,273,63,13,3,73,UP,31,END
CH43:   .BYTE   24,DWN,140,UP,263,DWN,16,UP,111,END
CH44:   .BYTE   40,DWN,20,12,231,UP,143,END
CH45:   .BYTE   24,DWN,140,UP,34,END
CH46:   .BYTE   60,DWN,0,UP,120,END
CH47:   .BYTE   21,DWN,146,UP,37,END
CH48:   .BYTE   21,DWN,6,21,100,31,356,31,100,21,6,UP,37,END
CH49:   .BYTE   67,DWN,21,17,11,240,100,UP,40,END
CH50:   .BYTE   27,DWN,21,100,31,11,252,220,273,11,140,UP,20,END
CH51:   .BYTE   27,DWN,21,100,31,12,231,260,60,31,12,231,300,221,UP,171,END
CH52:   .BYTE   140,DWN,7,1,335,140,UP,33,END
CH53:   .BYTE   21,DWN,31,100,21,2,221,320,4,140,UP,37,11,END
CH54:   .BYTE   167,DWN,221,300,231,16,31,100,21,2,221,320,UP,174,END
CH55:   .BYTE   27,DWN,1,140,11,314,13,UP,120,END
CH56:   .BYTE   44,DWN,231,12,31,100,21,2,221,300,221
        .BYTE   2,21,100,31,12,231,UP,54,END
CH57:   .BYTE   21,DWN,31,100,21,6,221,300,231,12,31,120,UP,34,END
CH58:   .BYTE   46,DWN,0,UP,14,DWN,0,UP,152,END
CH59:   .BYTE   45,DWN,0,UP,15,DWN,20,12,231,UP,143,END
CH60:   .BYTE   120,DWN,304,104,UP,77,11,END
CH61:   .BYTE   25,DWN,140,UP,352,DWN,140,UP,33,END
CH62:   .BYTE   60,DWN,104,304,UP,137,11,END
CH63:   .BYTE   26,DWN,1,21,100,31,11,273,11,UP,12,DWN,0,UP,100,END
CH64:   .BYTE   140,DWN,300,221,6,21,100,31,13,231,260
        .BYTE   1,21,20,12,UP,73,END
CH65:   .BYTE   20,DWN,6,42,40,52,12,340,140,14,UP,20,END
CH66:   .BYTE   20,DWN,120,21,2,221,300,100,21,2,221
        .BYTE   320,20,17,11,UP,140,END
CH67:   .BYTE   167,DWN,221,260,252,14,52,60,21,UP,31,END
CH68:   .BYTE   20,DWN,120,21,6,221,320,20,17,11,UP,140,END
CH69:   .BYTE   104,64,DWN,340,14,60,260,14,140,UP,20,END
CH70:   .BYTE   104,64,DWN,340,14,60,260,14,UP,160,END
CH71:   .BYTE   167,DWN,221,260,252,14,52,60,21,2,240,UP,73,END
CH72:   .BYTE   20,DWN,4,4,14,140,4,14,14,UP,20,END
CH73:   .BYTE   107,DWN,1,240,100,240,17,11,240,100,UP,40,END
CH74:   .BYTE   21,DWN,31,40,21,7,240,100,UP,37,11,END
CH75:   .BYTE   20,DWN,4,40,240,4,UP,140,DWN,314,114,UP,20,END
CH76:   .BYTE   24,4,DWN,14,14,140,UP,20,END
CH77:   .BYTE   20,DWN,4,4,73,11,1,63,14,14,UP,20,END
CH78:   .BYTE   20,DWN,4,4,156,6,14,14,UP,20,END
CH79:   .BYTE   22,DWN,4,42,40,52,14,252,240,242,UP,172,END
CH80:   .BYTE   20,DWN,4,4,120,31,12,231,320,UP,174,END
CH81:   .BYTE   141,DWN,21,4,242,240,252,14,52,40,21,221,52,UP,20,END
CH82:   .BYTE   20,DWN,4,120,21,2,221,320,14,40,114,UP,20,END
CH83:   .BYTE   21,DWN,31,100,21,2,221,300,221,2,21,100,31,UP,37,END
```

```
CH84:   .BYTE   100,DWN,4,4,260,140,UP,34,14,END
CH85:   .BYTE   24,4,DWN,17,31,100,21,7,UP,34,14,END
CH86:   .BYTE   24,4,DWN,15,73,63,5,UP,34,14,END
CH87:   .BYTE   24,4,DWN,14,14,63,2,12,73,4,4,UP,34,14,END
CH88:   .BYTE   20,DWN,1,146,1,UP,340,DWN,11,156,11,UP,20,END
CH89:   .BYTE   100,DWN,4,263,1,UP,140,DWN,11,273,UP,114,END
CH90:   .BYTE   24,4,DWN,140,11,356,11,140,UP,20,END
CH91:   .BYTE   144,4,DWN,260,14,14,60,UP,40,END
CH92:   .BYTE   27,DWN,156,UP,31,END
CH93:   .BYTE   44,4,DWN,60,17,11,260,UP,140,END
CH94:   .BYTE   27,DWN,21,100,31,UP,37,END
CH95:   .BYTE   20,DWN,140,UP,20,END
CH96:   .BYTE   144,4,DWN,11,31,UP,36,END
CH97:   .BYTE   142,DWN,252,240,221,2,42,60,15,UP,40,END
CH98:   .BYTE   24,4,DWN,16,52,40,21,3,221,240,252,13,UP,160,END
CH99:   .BYTE   144,DWN,221,260,231,13,31,60,21,UP,51,END
CH100:  .BYTE   144,4,DWN,16,252,240,221,3,21,40,52,13,UP,40,END
CH101:  .BYTE   22,DWN,120,2,221,260,231,13,31,100,UP,40,END
CH102:  .BYTE   60,DWN,4,240,100,240,3,21,20,31,UP,57,END
CH103:  .BYTE   32,DWN,31,60,21,5,242,240,231,13,31,40,42,3,UP,55,END
CH104:  .BYTE   24,4,DWN,15,13,3,42,20,52,13,UP,40,END
CH105:  .BYTE   107,DWN,0,UP,232,DWN,20,15,UP,100,END
CH106:  .BYTE   32,DWN,31,60,21,7,UP,55,END
CH107:  .BYTE   20,DWN,3,40,240,5,UP,113,DWN,252,73,UP,40,END
CH108:  .BYTE   64,4,DWN,14,14,UP,120,END
CH109:  .BYTE   20,DWN,5,40,31,14,4,21,20,31,14,UP,20,END
CH110:  .BYTE   25,DWN,15,3,42,40,31,14,UP,40,END
CH111:  .BYTE   21,DWN,3,21,60,31,13,231,260,221,UP,171,END
CH112:  .BYTE   25,DWN,13,52,40,21,3,221,240,252,16,UP,163,END
CH113:  .BYTE   145,DWN,13,252,240,221,3,21,40,52,16,UP,43,END
CH114:  .BYTE   25,DWN,15,3,42,40,31,UP,54,END
CH115:  .BYTE   21,DWN,31,60,21,221,220,221,220,221,21,60,31,UP,54,END
CH116:  .BYTE   67,DWN,12,,240,100,240,14,31,20,21,UP,51,END
CH117:  .BYTE   25,DWN,14,31,60,21,4,UP,55,END
CH118:  .BYTE   25,DWN,13,52,42,3,UP,75,END
CH119:  .BYTE   25,DWN,14,31,20,21,3,13,31,20,21,4,UP,35,END
CH120:  .BYTE   20,DWN,125,UP,320,DWN,135,UP,40,END
CH121:  .BYTE   25,DWN,14,31,40,42,3,17,231,260,221,UP,162,END
CH122:  .BYTE   25,DWN,120,335,120,UP,40,END
CH123:  .BYTE   144,4,DWN,240,231,12,231,31,12,31,40,UP,40,END
CH124:  .BYTE   103,DWN,12,UP,7,DWN,12,UP,116,END
CH125:  .BYTE   44,4,DWN,40,31,12,31,231,12,231,240,UP,140,END
CH126:  .BYTE   27,DWN,21,20,52,20,21,UP,37,END
CH127:  .BYTE   24,4,DWN,140,UP,351,DWN,140,UP,351,DWN,140,UP
        .BYTE   351,DWN,140,UP,351,DWN,140,UP,351,DWN,140,UP
        .BYTE   351,DWN,140,UP,351,DWN,140,UP,351,DWN,140,UP,20,END
        .LIST BIN
        .END
```

```
;       GRAPHICS SYSTEM EXCERCIZER FILE
;
;$R4$K$P$O03200128$M00000000$D
;$A$M
;+  50   0  50  50   0  50- 50  50
;+- 50   0- 50- 50   0- 50  50- 50   0   0
;
;$H
;
;$V$L
;+   0  50  50   0   0- 50- 50   0
;+- 50   0   0  50  50   0   0- 50
;+  50   0   0- 50- 50   0   0  50
;+   0- 50- 50   0   0  50  50   0
;
;$H
;
;$R4
;$U$M00000000
;$U$V$L00050005$D0000-010-0100000000000100010000$U-005-005
;$U$M00500050
;$U$V$L00050005$D0000-010-0100000000000100010000$U-005-005
;$M-050-050
;$U$V$L00050005$D0000-010-0100000000000100010000$U-005-005
;$M-0500050
;$U$V$L00050005$D0000-010-0100000000000100010000$U-005-005
;$M0050-050
;$U$V$L00050005$D0000-010-0100000000000100010000$U-005-005
;
;$H
;
;$R6
;$P
;$K$U$O000100000050$M000000000000
;$D$X000002
;+   135   135   128   137   127   118   117   113   110   110
;+   114   113   117   114   113   118   111   123   126   131
;+   124   118   130   138   142   129   129   128   135   122
;+   128   120   118   114   119   117   113   110   117   112
;+   110   112   116   111   110   113   116   122   123   118
;+   123   128   125   120   124   121   124   114   115   113
;+   115   108   110   107   101   105    96    98    99    92
;+   104    93    98    93    96    93    91    90    90    93
;+    85    90    91    89    89    90    84    89    86    87
;+    86    85    88    85    90    84    89    80    83    86
;+    86    83    80    87    82    75    82    78    81    77
;+    78    74    70    67    73    67    64    65    63    65
;+    69    67    61    64    60    61    62    61    64    65
;+    70    64    71    70    73    76    84    89    96    99
;+   108   113   113   114   113   107   102    92    80    69
;+    65    52    45    37    29    32    30    39    45    61
;+    74    90   101   123   125   133   131   121   120   109
;+    86    80    64    46    30    25    18    10     7     5
;+     3     3     4     4     5     3     4     4     3     4
;+     4     3     5     3     4     5     4     4     3     4
;$U
;
;$H
;
;$R6
;$V
;$K$U$O000100000050$M000000000000
;$D$X000002
;+   135   135   128   137   127   118   117   113   110   110
;+   114   113   117   114   113   118   111   123   126   131
;+   124   118   130   138   142   129   129   128   135   122
;+   128   120   118   114   119   117   113   110   117   112
;+   110   112   116   111   110   113   116   122   123   118
;+   123   128   125   120   124   121   124   114   115   113
;+   115   108   110   107   101   105    96    98    99    92
;+   104    93    98    93    96    93    91    90    90    93
;+    85    90    91    89    89    90    84    89    86    87
;+    86    85    88    85    90    84    89    80    83    86
;+    86    83    80    87    82    75    82    78    81    77
;+    78    74    70    67    73    67    64    65    63    65
;+    69    67    61    64    60    61    62    61    64    65
```

```
;+    70    64    71    70    73    76    84    89    96    99
;+   108   113   113   114   113   107   102    92    80    69
;+    65    52    45    37    29    32    30    39    45    61
;+    74    90   101   123   125   133   131   121   120   109
;+    86    80    64    46    30    25    18    10     7     5
;+     3     3     4     4     5     3     4     4     3     4
;+     4     3     5     3     4     5     4     4     3     4
;$U
;
;$H
;
;$R4$K
;$FH11
;$O00500050$V$U$M00000000$D05000000$U00000200$D00000000
;$U$M
;+00700120$G3$D
;+01400150$G3$D
;+02100160$G3$D
;+02800145$G3$D
;+03500180$G3
;
;+00700030$G5$D
;+01400060$G5$D
;+02100070$G5$D
;+02800080$G5$D
;+03500075$G5
;
;+00700040$G7$D
;+01400030$G7$D
;+02100080$G7$D
;+02800100$G7$D
;+03500090$G7
;
;$H
;
;$R4$K$U$P$O02000000$M00000210
;$FHL11
;$C  + !"#$%&'()*+,-./
;$M00000170
;$C  +0123456789:;<=>?
;$M00000130
;$C  +@ABCDEFGHIJKLMNO
;$M00000090
;$C  +PQRSTUVWXYZ[\]^_
;$M00000050
;$C  +`abcdefghijklmno
;$M00000010
;$C  +pqrstuvwxyz{|
;
;$H
;
;$R4$K$M00000210
;$FHG11
;$C  +abcdefghijklmno
;$M00000170
;$C  +pqrstuvwxyz{|`
;
;$H
;
;$R4$K$U$V$O02000000$M00000210
;$FHL11
;$C  + !"#$%&'()*+,-./
;$M00000170
;$C  +0123456789:;<=>?
;$M00000130
;$C  +@ABCDEFGHIJKLMNO
;$M00000090
;$C  +PQRSTUVWXYZ[\]^_
;$M00000050
;$C  +`abcdefghijklmno
;$M00000010
;$C  +pqrstuvwxyz{|
;
;$H
;
;$R4$K$M00000210
```

```
;$FHG11
;$C  +abcdefghijklmno
;$M00000170
;$C  +pqrstuvwxyz{|`
;$FHL00
;$M00000130
;$C  + !"#$%&'()*+,-./0123456789:;<=>?
;$M00000090
;$C  +@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
;$M00000050
;$C  +`abcdefghijklmnopqrstuvwxyz{|
;$M00000010
;$FHG00
;$C  +abcdefghijklmnopqrstuvwxyz{|`
;
;$H
;
;$R4$K$M00500000
;$FVL00
;$C  +ABCDEFGH 0123
;$M01000000
;$FVL00
;$C  +01233456789 !"#$%&'()=*+
;
;$H
;
;$R6
;$V
;$K$U$O000100000050$M000000000000
;$D$M000400000000$U000000000200$D000000000000
;$X000002
;+   135   135   128   137   127   118   117   113   110   110
;+   114   113   117   114   113   118   111   123   126   131
;+   124   118   130   138   142   129   129   128   135   122
;+   128   120   118   114   119   117   113   110   117   112
;+   110   112   116   111   110   113   116   122   123   118
;+   123   128   125   120   124   121   124   114   115   113
;+   115   108   110   107   101   105    96    98    99    92
;+   104    93    98    93    96    93    91    90    90    93
;+    85    90    91    89    89    90    84    89    86    87
;+    86    85    88    85    90    84    89    80    83    86
;+    86    83    80    87    82    75    82    78    81    77
;+    78    74    70    67    73    67    64    65    63    65
;+    69    67    61    64    60    61    62    61    64    65
;+    70    64    71    70    73    76    84    89    96    99
;+   108   113   113   114   113   107   102    92    80    69
;+    65    52    45    37    29    32    30    39    45    61
;+    74    90   101   123   125   133   131   121   120   109
;+    86    80    64    46    30    25    18    10     7     5
;+     3     3     4     4     5     3     4     4     3     4
;+     4     3     5     3     4     5     4     4     3     4
;
;$U
;$R4$M-0300050
;$FVL11
;$C  +COUNTS
;$M0050-040
;$FH
;$C  +FILE  BE120D.DAT
;
;
;$R$H$K$B$S
;
```

```
        .PAGE
        .SBTTL  6571A ROM IMAGE
        ;
R6571A: FCB     000 000 000 000 061 112 104 112
        FCB     061 000 000 000 000 000 000 000
        FCB     000 000 000 074 042 074 042 042
        FCB     074 040 040 100 000 000 000 000
        FCB     000 000 000 000 141 022 024 030
        FCB     020 060 060 060 000 000 000 000
        FCB     060 110 100 100 040 060 110 110
        FCB     060 000 000 000 000 000 000 000
        FCB     000 000 030 040 100 170 100 040
        FCB     030 000 000 000 000 000 000 000
        FCB     026 016 020 040 100 100 070 004
        FCB     030 000 000 000 000 000 000 000
        FCB     000 000 000 000 054 122 022 022
        FCB     022 002 002 002 000 000 000 000
        FCB     030 044 102 102 176 102 102 044
        FCB     030 000 000 000 000 000 000 000
        FCB     000 000 000 100 100 100 100 110
        FCB     060 000 000 000 000 000 000 000
        FCB     000 000 100 110 120 140 120 112
        FCB     104 000 000 000 000 000 000 000
        FCB     100 040 020 020 020 020 030 044
        FCB     102 000 000 000 000 000 000 000
        FCB     000 000 000 000 110 110 110 110
        FCB     164 100 100 100 000 000 000 000
        FCB     000 000 000 142 042 044 050 060
        FCB     040 000 000 000 000 000 000 000
        FCB     010 034 040 030 040 100 074 002
        FCB     014 000 000 000 000 000 000 000
        FCB     000 000 000 030 044 102 102 044
        FCB     030 000 000 000 000 000 000 000
        FCB     000 000 000 077 124 024 024 024
        FCB     024 000 000 000 000 000 000 000
        FCB     000 000 000 030 044 102 102 144
        FCB     130 100 100 100 000 000 000 000
        FCB     000 000 000 037 044 102 102 044
        FCB     030 000 000 000 000 000 000 000
        FCB     000 000 000 077 110 010 010 010
        FCB     010 000 000 000 000 000 000 000
        FCB     000 000 000 142 044 044 044 044
        FCB     030 000 000 000 000 000 000 000
        FCB     020 020 070 124 124 124 070 020
        FCB     020 000 000 000 000 000 000 000
        FCB     000 000 000 000 142 024 010 024
        FCB     043 000 000 000 000 000 000 000
        FCB     000 000 000 010 111 052 052 052
        FCB     034 010 010 010 000 000 000 000
        FCB     000 000 000 042 101 111 111 111
        FCB     066 000 000 000 000 000 000 000
        FCB     000 034 042 101 101 101 042 042
        FCB     143 000 000 000 000 000 000 000
        FCB     037 020 020 020 020 020 120 060
        FCB     020 000 000 000 000 000 000 000
        FCB     000 000 004 002 177 002 004 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 020 040 177 040 020 000
        FCB     000 000 000 000 000 000 000 000
        FCB     010 034 052 010 010 010 010 010
        FCB     010 000 000 000 000 000 000 000
        FCB     000 000 010 000 177 000 010 000
        FCB     000 000 000 000 000 000 000 000
        FCB     177 040 020 010 006 010 020 040
        FCB     177 000 000 000 000 000 000 000
        FCB     000 060 111 006 060 111 006 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     010 010 010 010 010 010 000 000
        FCB     010 000 000 000 000 000 000 000
        FCB     022 022 022 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     024 024 024 177 024 177 024 024
        FCB     024 000 000 000 000 000 000 000
        FCB     010 077 110 110 076 011 011 176
```

```
        FCB     010 000 000 000 000 000 000 000
        FCB     040 121 042 004 010 020 042 105
        FCB     002 000 000 000 000 000 000 000
        FCB     070 104 104 050 020 051 106 106
        FCB     071 000 000 000 000 000 000 000
        FCB     040 040 100 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     010 020 040 040 040 040 040 020
        FCB     010 000 000 000 000 000 000 000
        FCB     010 004 002 002 002 002 002 004
        FCB     010 000 000 000 000 000 000 000
        FCB     010 111 052 034 177 034 052 111
        FCB     010 000 000 000 000 000 000 000
        FCB     000 010 010 010 177 010 010 010
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     060 020 020 040 000 000 000 000
        FCB     000 000 000 000 177 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     020 000 000 000 000 000 000 000
        FCB     000 001 002 004 010 020 040 100
        FCB     000 000 000 000 000 000 000 000
        FCB     076 101 103 105 111 121 141 101
        FCB     076 000 000 000 000 000 000 000
        FCB     010 030 010 010 010 010 010 010
        FCB     076 000 000 000 000 000 000 000
        FCB     076 101 001 002 014 020 040 100
        FCB     177 000 000 000 000 000 000 000
        FCB     076 101 001 001 036 001 001 101
        FCB     076 000 000 000 000 000 000 000
        FCB     002 006 012 022 042 177 002 002
        FCB     002 000 000 000 000 000 000 000
        FCB     177 100 100 100 176 001 001 101
        FCB     076 000 000 000 000 000 000 000
        FCB     076 101 100 100 176 101 101 101
        FCB     076 000 000 000 000 000 000 000
        FCB     177 101 002 004 010 020 020 020
        FCB     020 000 000 000 000 000 000 000
        FCB     076 101 101 101 076 101 101 101
        FCB     076 000 000 000 000 000 000 000
        FCB     076 101 101 101 077 001 001 101
        FCB     076 000 000 000 000 000 000 000
        FCB     000 000 040 000 000 000 040 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 040 000 000 000 000
        FCB     060 020 020 040 000 000 000 000
        FCB     004 010 020 040 100 040 020 010
        FCB     004 000 000 000 000 000 000 000
        FCB     000 000 000 177 000 177 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     020 010 004 002 001 002 004 010
        FCB     020 000 000 000 000 000 000 000
        FCB     076 101 101 002 004 010 010 000
        FCB     010 000 000 000 000 000 000 000
        FCB     076 101 101 115 125 136 100 101
        FCB     076 000 000 000 000 000 000 000
        FCB     034 042 101 101 177 101 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     176 041 041 041 076 041 041 041
        FCB     176 000 000 000 000 000 000 000
        FCB     036 041 100 100 100 100 100 041
        FCB     036 000 000 000 000 000 000 000
        FCB     176 041 041 041 041 041 041 041
        FCB     176 000 000 000 000 000 000 000
        FCB     177 100 100 100 170 100 100 100
        FCB     177 000 000 000 000 000 000 000
        FCB     177 100 100 100 170 100 100 100
        FCB     100 000 000 000 000 000 000 000
        FCB     036 041 100 100 100 107 101 041
        FCB     036 000 000 000 000 000 000 000
        FCB     101 101 101 101 177 101 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     076 010 010 010 010 010 010 010
        FCB     076 000 000 000 000 000 000 000
        FCB     037 004 004 004 004 004 004 104
```

```
        FCB     070 000 000 000 000 000 000 000
        FCB     101 102 104 110 160 110 104 102
        FCB     101 000 000 000 000 000 000 000
        FCB     100 100 100 100 100 100 100 100
        FCB     177 000 000 000 000 000 000 000
        FCB     101 143 125 111 111 101 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     101 141 121 111 105 103 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     034 042 101 101 101 101 101 042
        FCB     034 000 000 000 000 000 000 000
        FCB     176 101 101 101 176 100 100 100
        FCB     100 000 000 000 000 000 000 000
        FCB     034 042 101 101 101 101 105 042
        FCB     035 000 000 000 000 000 000 000
        FCB     176 101 101 101 176 110 104 102
        FCB     101 000 000 000 000 000 000 000
        FCB     076 101 100 100 076 001 001 101
        FCB     076 000 000 000 000 000 000 000
        FCB     177 010 010 010 010 010 010 010
        FCB     010 000 000 000 000 000 000 000
        FCB     101 101 101 101 101 101 101 101
        FCB     076 000 000 000 000 000 000 000
        FCB     101 101 101 101 101 101 042 024
        FCB     010 000 000 000 000 000 000 000
        FCB     101 101 101 111 111 111 125 143
        FCB     101 000 000 000 000 000 000 000
        FCB     101 101 042 024 010 024 042 101
        FCB     101 000 000 000 000 000 000 000
        FCB     101 101 042 024 010 010 010 010
        FCB     010 000 000 000 000 000 000 000
        FCB     177 001 002 004 010 020 040 100
        FCB     177 000 000 000 000 000 000 000
        FCB     036 020 020 020 020 020 020 020
        FCB     036 000 000 000 000 000 000 000
        FCB     000 100 040 020 010 004 002 001
        FCB     000 000 000 000 000 000 000 000
        FCB     074 004 004 004 004 004 004 004
        FCB     074 000 000 000 000 000 000 000
        FCB     076 101 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     177 000 000 000 000 000 000 000
        FCB     002 002 001 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 036 042 102 102 106
        FCB     072 000 000 000 000 000 000 000
        FCB     100 100 100 134 142 102 102 142
        FCB     134 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 100 100 102
        FCB     074 000 000 000 000 000 000 000
        FCB     002 002 002 072 106 102 102 106
        FCB     072 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 102 176 100
        FCB     076 000 000 000 000 000 000 000
        FCB     014 022 020 020 174 020 020 020
        FCB     020 000 000 000 000 000 000 000
        FCB     000 000 000 072 106 102 102 106
        FCB     072 002 102 074 000 000 000 000
        FCB     100 100 100 130 144 102 102 102
        FCB     102 000 000 000 000 000 000 000
        FCB     000 010 000 030 010 010 010 010
        FCB     010 000 000 000 000 000 000 000
        FCB     000 000 000 002 002 002 002 002
        FCB     002 002 102 074 000 000 000 000
        FCB     100 100 100 104 110 160 110 104
        FCB     102 000 000 000 000 000 000 000
        FCB     020 020 020 020 020 020 020 020
        FCB     020 000 000 000 000 000 000 000
        FCB     000 000 000 166 111 111 111 111
        FCB     111 000 000 000 000 000 000 000
        FCB     000 000 000 134 142 102 102 102
        FCB     102 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 102 102 102
        FCB     074 000 000 000 000 000 000 000
        FCB     000 000 000 134 142 102 102 142
```

```
        FCB     134 100 100 100 000 000 000 000
        FCB     000 000 000 072 106 102 102 106
        FCB     072 002 002 002 000 000 000 000
        FCB     000 000 000 134 142 100 100 100
        FCB     100 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 060 014 102
        FCB     074 000 000 000 000 000 000 000
        FCB     000 020 020 174 020 020 020 022
        FCB     014 000 000 000 000 000 000 000
        FCB     000 000 000 102 102 102 102 102
        FCB     074 000 000 000 000 000 000 000
        FCB     000 000 000 104 104 104 104 050
        FCB     020 000 000 000 000 000 000 000
        FCB     000 000 000 101 111 111 111 111
        FCB     066 000 000 000 000 000 000 000
        FCB     000 000 000 102 044 030 030 044
        FCB     102 000 000 000 000 000 000 000
        FCB     000 000 000 102 102 102 102 106
        FCB     072 002 102 074 000 000 000 000
        FCB     000 000 000 176 004 010 020 040
        FCB     176 000 000 000 000 000 000 000
        FCB     016 020 020 020 040 020 020 020
        FCB     016 000 000 000 000 000 000 000
        FCB     010 010 010 000 000 010 010 010
        FCB     000 000 000 000 000 000 000 000
        FCB     070 004 004 004 002 004 004 004
        FCB     070 000 000 000 000 000 000 000
        FCB     060 111 006 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     177 177 177 177 177 177 177 177
        FCB     177 000 000 000 000 000 000 000
        ;
        .END
```

```
        .PAGE
        .SBTTL  6575 ROM IMAGE
        ;
R6575:  FCB     104 144 124 114 125 021 021 021
        FCB     016 000 000 000 000 000 000 000
        FCB     070 100 060 010 161 021 037 021
        FCB     021 000 000 000 000 000 000 000
        FCB     070 100 060 010 161 012 004 012
        FCB     021 000 000 000 000 000 000 000
        FCB     170 100 160 100 171 012 004 012
        FCB     021 000 000 000 000 000 000 000
        FCB     170 100 160 100 170 037 004 004
        FCB     004 000 000 000 000 000 000 000
        FCB     170 100 160 100 166 011 011 013
        FCB     007 000 000 000 000 000 000 000
        FCB     060 110 170 110 111 012 014 012
        FCB     011 000 000 000 000 000 000 000
        FCB     160 110 160 110 160 010 010 010
        FCB     017 000 000 000 000 000 000 000
        FCB     160 110 160 110 167 010 006 001
        FCB     016 000 000 000 000 000 000 000
        FCB     110 110 170 110 110 037 004 004
        FCB     004 000 000 000 000 000 000 000
        FCB     100 100 100 170 017 010 016 010
        FCB     010 000 000 000 000 000 000 000
        FCB     104 104 104 050 020 037 004 004
        FCB     004 000 000 000 000 000 000 000
        FCB     170 100 160 100 117 010 016 010
        FCB     010 000 000 000 000 000 000 000
        FCB     070 100 100 100 076 011 016 012
        FCB     011 000 000 000 000 000 000 000
        FCB     070 100 060 010 166 011 011 011
        FCB     006 000 000 000 000 000 000 000
        FCB     070 100 060 010 167 002 002 002
        FCB     007 000 000 000 000 000 000 000
        FCB     160 110 110 110 160 010 010 010
        FCB     017 000 000 000 000 000 000 000
        FCB     160 110 110 110 162 006 002 002
        FCB     007 000 000 000 000 000 000 000
        FCB     160 110 110 110 166 011 002 004
        FCB     017 000 000 000 000 000 000 000
        FCB     160 110 110 110 166 001 006 001
        FCB     016 000 000 000 000 000 000 000
        FCB     160 110 110 110 162 006 012 037
        FCB     002 000 000 000 000 000 000 000
        FCB     110 150 130 110 111 012 014 012
        FCB     011 000 000 000 000 000 000 000
        FCB     070 100 060 010 161 012 004 004
        FCB     004 000 000 000 000 000 000 000
        FCB     170 100 160 100 176 011 016 011
        FCB     016 000 000 000 000 000 000 000
        FCB     070 100 100 100 070 011 015 013
        FCB     011 000 000 000 000 000 000 000
        FCB     170 100 160 100 170 021 033 025
        FCB     021 000 000 000 000 000 000 000
        FCB     070 100 060 010 176 011 016 011
        FCB     016 000 000 000 000 000 000 000
        FCB     170 100 160 100 170 007 010 010
        FCB     007 000 000 000 000 000 000 000
        FCB     170 100 160 100 107 010 006 001
        FCB     016 000 000 000 000 000 000 000
        FCB     070 100 130 110 067 010 006 001
        FCB     016 000 000 000 000 000 000 000
        FCB     160 110 160 110 107 010 006 001
        FCB     016 000 000 000 000 000 000 000
        FCB     110 110 110 110 067 010 006 001
        FCB     016 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     010 010 010 010 010 000 000 010
        FCB     010 000 000 000 000 000 000 000
        FCB     044 044 044 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     024 024 024 177 024 177 024 024
        FCB     024 000 000 000 000 000 000 000
        FCB     010 077 110 110 076 011 011 176
```

```
        FCB     010 000 000 000 000 000 000 000
        FCB     040 121 042 004 010 020 042 105
        FCB     002 000 000 000 000 000 000 000
        FCB     070 104 104 050 020 051 106 106
        FCB     071 000 000 000 000 000 000 000
        FCB     014 014 010 020 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     004 010 020 020 020 020 020 010
        FCB     004 000 000 000 000 000 000 000
        FCB     020 010 004 004 004 004 004 010
        FCB     020 000 000 000 000 000 000 000
        FCB     000 010 111 052 034 052 111 010
        FCB     000 000 000 000 000 000 000 000
        FCB     000 010 010 010 177 010 010 010
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 030
        FCB     030 020 040 000 000 000 000 000
        FCB     000 000 000 000 177 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 030
        FCB     030 000 000 000 000 000 000 000
        FCB     000 001 002 004 010 020 040 100
        FCB     000 000 000 000 000 000 000 000
        FCB     076 101 103 105 111 121 141 101
        FCB     076 000 000 000 000 000 000 000
        FCB     010 030 050 010 010 010 010 010
        FCB     076 000 000 000 000 000 000 000
        FCB     076 101 001 002 034 040 100 100
        FCB     177 000 000 000 000 000 000 000
        FCB     076 101 001 001 036 001 001 101
        FCB     076 000 000 000 000 000 000 000
        FCB     002 006 012 022 042 102 177 002
        FCB     002 000 000 000 000 000 000 000
        FCB     177 100 100 174 002 001 001 102
        FCB     074 000 000 000 000 000 000 000
        FCB     036 040 100 100 176 101 101 101
        FCB     076 000 000 000 000 000 000 000
        FCB     177 101 002 004 010 020 020 020
        FCB     020 000 000 000 000 000 000 000
        FCB     076 101 101 101 076 101 101 101
        FCB     076 000 000 000 000 000 000 000
        FCB     076 101 101 101 077 001 001 002
        FCB     074 000 000 000 000 000 000 000
        FCB     000 000 000 030 030 000 000 030
        FCB     030 000 000 000 000 000 000 000
        FCB     000 000 000 030 030 000 000 030
        FCB     030 020 040 000 000 000 000 000
        FCB     004 010 020 040 100 040 020 010
        FCB     004 000 000 000 000 000 000 000
        FCB     000 000 000 076 000 076 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     020 010 004 002 001 002 004 010
        FCB     020 000 000 000 000 000 000 000
        FCB     036 041 041 001 006 010 010 000
        FCB     010 000 000 000 000 000 000 000
        FCB     036 041 115 125 125 136 100 040
        FCB     036 000 000 000 000 000 000 000
        FCB     034 042 101 101 101 177 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     176 041 041 041 076 041 041 041
        FCB     176 000 000 000 000 000 000 000
        FCB     036 041 100 100 100 100 100 041
        FCB     036 000 000 000 000 000 000 000
        FCB     174 042 041 041 041 041 041 042
        FCB     174 000 000 000 000 000 000 000
        FCB     177 100 100 100 170 100 100 100
        FCB     177 000 000 000 000 000 000 000
        FCB     177 100 100 100 170 100 100 100
        FCB     100 000 000 000 000 000 000 000
        FCB     036 041 100 100 100 117 101 041
        FCB     036 000 000 000 000 000 000 000
        FCB     101 101 101 101 177 101 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     076 010 010 010 010 010 010 010
        FCB     076 000 000 000 000 000 000 000
        FCB     037 004 004 004 004 004 004 104
```

```
        FCB     070 000 000 000 000 000 000 000
        FCB     101 102 104 110 120 150 104 102
        FCB     101 000 000 000 000 000 000 000
        FCB     100 100 100 100 100 100 100 100
        FCB     177 000 000 000 000 000 000 000
        FCB     101 143 125 111 111 101 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     101 141 121 111 105 103 101 101
        FCB     101 000 000 000 000 000 000 000
        FCB     034 042 101 101 101 101 101 042
        FCB     034 000 000 000 000 000 000 000
        FCB     176 101 101 101 176 100 100 100
        FCB     100 000 000 000 000 000 000 000
        FCB     034 042 101 101 101 111 105 042
        FCB     035 000 000 000 000 000 000 000
        FCB     176 101 101 101 176 110 104 102
        FCB     101 000 000 000 000 000 000 000
        FCB     076 101 100 100 076 001 001 101
        FCB     076 000 000 000 000 000 000 000
        FCB     177 010 010 010 010 010 010 010
        FCB     010 000 000 000 000 000 000 000
        FCB     101 101 101 101 101 101 101 101
        FCB     076 000 000 000 000 000 000 000
        FCB     101 101 101 042 042 024 024 010
        FCB     010 000 000 000 000 000 000 000
        FCB     101 101 101 101 111 111 125 143
        FCB     101 000 000 000 000 000 000 000
        FCB     101 101 042 024 010 024 042 101
        FCB     101 000 000 000 000 000 000 000
        FCB     101 101 042 024 010 010 010 010
        FCB     010 000 000 000 000 000 000 000
        FCB     177 001 002 004 010 020 040 100
        FCB     177 000 000 000 000 000 000 000
        FCB     074 040 040 040 040 040 040 040
        FCB     074 000 000 000 000 000 000 000
        FCB     000 100 040 020 010 004 002 001
        FCB     000 000 000 000 000 000 000 000
        FCB     074 004 004 004 004 004 004 004
        FCB     074 000 000 000 000 000 000 000
        FCB     010 024 042 101 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     177 000 000 000 000 000 000 000
        FCB     030 030 010 004 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     000 000 000 074 002 076 102 102
        FCB     075 000 000 000 000 000 000 000
        FCB     100 100 100 134 142 102 102 142
        FCB     134 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 100 100 102
        FCB     074 000 000 000 000 000 000 000
        FCB     002 002 002 072 106 102 102 106
        FCB     072 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 176 100 100
        FCB     074 000 000 000 000 000 000 000
        FCB     014 022 020 020 174 020 020 020
        FCB     020 000 000 000 000 000 000 000
        FCB     000 000 000 072 106 102 106 072
        FCB     002 002 102 074 000 000 000 000
        FCB     100 100 100 134 142 102 102 102
        FCB     102 000 000 000 000 000 000 000
        FCB     000 010 000 030 010 010 010 010
        FCB     034 000 000 000 000 000 000 000
        FCB     000 000 000 006 002 002 002 002
        FCB     002 002 042 034 000 000 000 000
        FCB     100 100 100 104 110 120 150 104
        FCB     102 000 000 000 000 000 000 000
        FCB     030 010 010 010 010 010 010 010
        FCB     034 000 000 000 000 000 000 000
        FCB     000 000 000 166 111 111 111 111
        FCB     111 000 000 000 000 000 000 000
        FCB     000 000 000 134 142 102 102 102
        FCB     102 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 102 102 102
        FCB     074 000 000 000 000 000 000 000
        FCB     000 000 000 134 142 102 102 142
```

```
        FCB     134 100 100 100 000 000 000 000
        FCB     000 000 000 072 106 102 102 106
        FCB     072 002 002 002 000 000 000 000
        FCB     000 000 000 134 142 100 100 100
        FCB     100 000 000 000 000 000 000 000
        FCB     000 000 000 074 102 060 014 102
        FCB     074 000 000 000 000 000 000 000
        FCB     000 020 020 174 020 020 020 022
        FCB     014 000 000 000 000 000 000 000
        FCB     000 000 000 102 102 102 102 106
        FCB     072 000 000 000 000 000 000 000
        FCB     000 000 000 101 101 101 042 024
        FCB     010 000 000 000 000 000 000 000
        FCB     000 000 000 101 111 111 111 111
        FCB     066 000 000 000 000 000 000 000
        FCB     000 000 000 102 044 030 030 044
        FCB     102 000 000 000 000 000 000 000
        FCB     000 000 000 102 102 102 102 106
        FCB     072 002 102 074 000 000 000 000
        FCB     000 000 000 176 004 010 020 040
        FCB     176 000 000 000 000 000 000 000
        FCB     014 020 020 020 040 020 020 020
        FCB     014 000 000 000 000 000 000 000
        FCB     010 010 010 000 000 010 010 010
        FCB     000 000 000 000 000 000 000 000
        FCB     030 004 004 004 002 004 004 004
        FCB     030 000 000 000 000 000 000 000
        FCB     060 111 006 000 000 000 000 000
        FCB     000 000 000 000 000 000 000 000
        FCB     044 111 022 044 111 022 044 111
        FCB     022 000 000 000 000 000 000 000
        ;
        .END
```

```
R MACRO
KUS=KUSCFG,KUS
```

```
;       KEYBOARD UTILITY SYSTEM DEVICE CONFIGURATION

        M$CSR=172522    ;MAGTAPE DEFINED

        Q$CSR=177170    ;RX01 FLOPPY DEFINED

        Y$CSR=177160    ;RX02/FD211/FD411 FLOPPY DEFINED

        Z$CSR=177500    ;TA-11 CASSETTE DEFINED


;       PROCESSOR CONFIGURATION

        PDP$11  =0      ;FOR PDP-11'S

;       LSI$11  =0      ;FOR LSI-11'S
```

```
        .TITLE  KEYBOARD UTILITY SYSTEM

;KEYBOARD UTILITY SYSTEM-V05
;COPYRIGHT OCT, 1976
;TENNECOMP SYSTEMS, INC.
;795 OAK RIDGE TPK
;OAK RIDGE, TENN

;REWRITTEN
;BY ALAN R. BALDWIN
;KENT STATE UNIVERSITY
;OCTOBER 1981

;NOTATION
;     ALL (N)'S IN THE FOLLOWING EXPLANATION REFER TO NUMERICAL ARGUMENTS.
; NORMALLY THESE ARE OCTAL NUMBERS LESS THAN 177777 OR DECIMAL NUMBERS LESS
; THAN 65536.  NUMBERS PRECEDED BY A "+", CONTAINING AN "8" OR "9", OR
; FOLLOWED BY A "." ARE READ AS DECIMAL.  ALL OTHER NUMBERS WILL BE READ AS
; OCTAL.  OCTAL OR DECIMAL NUMBERS MAY BE PRECEDED BY A "-" SIGN TO MAKE
; NEGATIVE.
;     THE CHARACTER "*" IS USED IN THE FOLLOWING EXPLANATION TO REPRESENT
; AN ALPHAMERIC CHARACTER, WHERE MORE THAN ONE CHARACTER MAY BE USED.
;     <CR>, <LF>, <RUBOUT>, ETC., DENOTE THE PRESSING OF THAT KEY ON THE
; KEYBOARD.

;GENERAL USAGE
;     KUS IS RELOCATABLE, WITH THE STARTING ADDRESS EQUAL TO THE LOADING
; ADDRESS.  KUS USES A PROCESSOR STACK IN THE 20 WORDS JUST BELOW KUS, AND
; THEREFORE MUST BE RELOCATED TO AT LEAST ADDRESS 500 OR AT LEAST 20 WORDS
; ABOVE NON-VOLATILE DATA OR PROGRAMS.  KUS REQUIRES 1.875 K OF CORE OR
; 7377 BYTES ABOVE THE LOADING ADDRESS.  THE RECOMMENDED LOADING ADDRESS IS
; --0000, WHERE "--" ARE THE SAME DIGITS AS ARE USED BY THE BOOTSTRAP.
; THIS USES FROM --0000 THROUGH --7377 FOR KUS, AND LEAVES --7400 THROUGH
; --7777 FOR SYSTEM LOADER AND BOOTSTRAP.
;     KUS ACCEPTS ONE OR MORE CHARACTER COMMANDS INTO A BUFFER, AND BEGINS
; EXECUTION OF THE COMMAND WHEN A TERMINATOR SUCH AS <CR> OR <BLANK> IS
; TYPED.  IF NO TERMINATOR HAS YET BEEN TYPED, MISTAKES MAY BE CORRECTED BY
; TYPING <RUBOUT>.  EACH TIME <RUBOUT> IS TYPED, A CHARACTER IS DELETED
; FROM THE END OF THE ENTRY.
;     ON COMMANDS WHICH USE NUMERICAL ARGUMENTS, THE ARGUMENTS MAY BE GIVEN,
; SEPARATED BY SPACES, AFTER THE COMMAND.  TYPING <CR> WHEN ARGUMENTS ARE
; EXPECTED ENTERS ZEROES OR ELSE CAUSES CONVENIENT DEFAULT VALUES TO BE
; ASSUMED.

;COMMANDS
;  GENERAL PURPOSE
;<CTRL P>       INTERRUPTS ANY OPERATION.
;O*             USED TO SELECT THE PRIMARY OR SECONDARY DEVICE FOR COMMAND
;               INPUT.  "*" IS "K" FOR THE PRIMARY DEVICE WHICH IS NORMALLY
;               THE TELETYPE, AND "C" FOR THE SECONDARY DEVICE.  THE CSR FOR
;               THE SECONDARY DEVICE MAY BE DEFINED IN THE LOADING ADDRESS
;               PLUS 10.
;S*             SET THE VECTOR SPECIFIED BY THE CHARACTERS REPRESENTED BY
;               "*".  "K" SETS THE TELETYPE INTERRUPT VECTOR, AND MAY BE
;               SET BEFORE ENTERING A PROGRAM WHICH DOES NOT USE THE KEYBOARD.
;               "O" SETS THE PROCESS TIMEOUT VECTOR FOR KUS ERROR 011.  "T"
;               SETS THE TRAP VECTOR FOR THE KUS TRAP HANDLER.  IF NO CHARACTERS
;               ARE SPECIFIED AFTER THE "S", ALL ABOVE VECTORS ARE SET.
;R (N)          CAUSES CPU CONTROL TO BE TRANSFERRED TO A PROGRAM IN LOCATION
;               (N).  IF NO LOCATION IS GIVEN, THE LAST SPECIFIED RUN ADDRESS
;               IS USED.
;I*             ENABLES KUS TO READ A PAPER TAPE CONTAINING KUS COMMANDS
;               AND EXECUTE THEM AS IT READS THEM. THIS ALLOWS COMMONLY
;               REPEATED OPERATIONS TO BE DIRECTED FROM A PAPER TAPE.
;               "*" REPRESENTS THE READING DEVICE IF DIFFERENT FROM THE
;               PRIMARY DEVICE'S READER.  "H" USES THE HIGH SPEED READER,
;               AND "C" USES THE SECONDARY DEVICE READER.
;CC (N) (N)     ALLOWS CORE FROM THE FIRST ARGUMENT UP TO THE SECOND ARGUMENT
;               TO BE ZEROED.  IF THE SECOND ARG IS OMITTED, CORE IS
;               CLEARED UP TO KUS.  IF THE FIRST IS OMITTED, CORE IS CLEARED
;               FROM LOCATION 0 UP TO KUS.

;  CORE INTERROGATION AND MODIFICATION
;P* (N) (N) (N) PRINTS CONTENTS OF WORDS IN CORE FROM THE LOCATION SPECIFIED
;               BY THE FIRST (N), TO THE LOCATION SPECIFIED BY THE SECOND (N),
;               WITH THE NUMBER OF VALUES PER LINE SPECIFIED BY THE THIRD (N).
```

```
;               OMITTING THE THIRD (N) SELECTS 8 VALUES PER LINE, AND OMITTING
;               THE SECOND (N) SPECIFIES ONE LOCATION ONLY TO BE PRINTED.
;               "*" REPRESENTS CHARACTERS SPECIFING THE DEVICE, IF DIFFERENT
;               FROM THE CURRENT COMMAND DEVICE, AND/OR THE CHARACTER "B"
;               WHICH ALLOWS CORE TO BE PRINTED IN BYTES INSTEAD OF WORDS.
;               DEVICES WHICH MAY BE SELECTED INCLUDE "H" FOR HIGH SPEED
;               PUNCH, "P" FOR PRINTER, "L" OR "K" FOR PRIMARY
;               DEVICE, AND "C" FOR SECONDARY DEVICE.
;
;A (N)          ENTERS A CORE MODIFICATION MODE IN WHICH CORE BEGINNING AT (N)
;               MAY BE EXAMINED OR MODIFIED.  TO EXAMINE A LOCATION TYPE "#".
;               THE CONTENTS OF THE WORD WILL BE PRINTED, AND THE ADDRESS COUNTER
;               INCREMENTED.  TO CHANGE A LOCATION, TYPE THE APPROPRIATE NUMBER,
;               FOLLOWED BY A SPACE, COMMA, OR <CR>.  PRESSING <CR>
;               CAUSES THE VALUE OF THE ADDRESS COUNTER TO BE PRINTED.  PRESSING
;               <LF> EXITS THIS SPECIAL MODE.  PRESSING <RUBOUT> EITHER DELETES
;               DIGITS WITHIN A NUMBER OR, IF NO DIGITS ARE PRESENT TO DELETE,
;               CAUSES THE ADDRESS COUNTER TO BE DECREMENTED AND PRINTED.
;B (N)          IS THE SAME AS THE "A" COMMAND, EXCEPT CORE IS EXAMINED OR
;               MODIFIED BYTE BY BYTE, INSTEAD OF WORD BY WORD.
;E              CAUSES THE ADDRESS COUNTER TO BE ROUNDED UP TO THE NEXT MULTIPLE
;               OF 2.
;.A (N) *DATA*  ALLOWS THE USER TO INPUT ASCII DIRECTLY INTO CORE.  "*" MAY
;               BE ANY CHARACTER NOT USED AS DATA, AND IS USED TO DELIMIT INPUT.
;
;       IF (N) IS OMITTED IN THE "A", "B", ".A", OR "P" COMMANDS, THE LAST VALUE OF
;       THE ADDRESS COUNTER (AS LEFT BY AN "A", "B", ".A", "E", OR "P" COMMAND) IS
;       ASSUMED.
;
;       NOTE THAT THE "I" COMMAND ALLOWS TAPES CONTAINING PROGRAMS WRITTEN IN OCTAL
;       TO BE PREPARED OFF LINE.  NOTE ALSO THAT THE OUTPUT OF "P"
;       COMMANDS, IF PUNCHED ON PAPER TAPE, MAY BE READ BACK IN USING "I".

;  NON-FILE DEVICE (PAPER TAPE, ETC.)
;  LOADING AND DUMPING
;L* (N)         LOADS BINARY INTO CORE FROM THE DEVICE SPECIFIED.  THE ARGUMENT
;               SPECIFIES THE RELOCATION ADDRESS, IF PROGRAM REQUIRES RELOCATION.
;D* (N) (N)     DUMPS BINARY FROM THE FIRST (N) TO THE SECOND (N) LOCATIONS.
;               IF THE FINAL ADDRESS (SECOND (N)) IS ONE LESS THAN THE INITIAL
;               ADDRESS (FIRST (N)), A START OR STOP BLOCK IS PRODUCED, TERMINATING
;               THE DUMP.  THE INITIAL ADDRESS SHOULD BE ODD TO PRODUCE A STOP BLOCK
;               AND EQUAL TO THE STARTING ADDRESS TO PRODUCE A START BLOCK.
;D* (N)         CAUSES (N) TIMES 20 NULLS OF LEADER TO BE PUNCHED ON PAPER TAPE.
;               THUS "DL 10" IS EQUIVALENT TO PRESSING "HERE IS" 10 TIMES ON
;               A TELETYPE.
;
;       "*" IS THE DEVICE TO BE USED, TYPICALLY "L" OR "K" FOR THE PRIMARY
;       DEVICE READER OR PUNCH, "C" FOR THE SECONDARY DEVICE, OR "H" FOR HIGH
;       SPEED PAPER TAPE.
;
;       IF THE COMMAND DEVICE IS THE SAME AS THE DUMPING DEVICE,
;       THE PROGRAM WILL WAIT FOR ANY KEY TO BE PRESSED BEFORE
;       PUNCHING ANY DATA, TO ALLOW FOR THE PUNCH TO BE TURNED ON.  THE PROGRAM
;       WAITS AGAIN WHEN IT IS DONE PUNCHING SO THE PUNCH MAY BE TURNED OFF.
;
;
;HC             CAUSES THE HIGH SPEED READER/PUNCH TO COPY A PAPER TAPE.

;  FILE DEVICE LOADING AND DUMPING
;       FILE DEVICES, SUCH AS A DISK OR MAGTAPE, ALLOW ANY NUMBER OF DUMPS TO
;       BE STORED, WITH A USER DEFINED IDENTIFICATION NUMBER FOR EACH DUMP.
;
;*L             LISTS THE ID NUMBER, INITIAL ADDRESS, AND FINAL ADDRESS OF EACH
;               BLOCK DUMPED ON THE MEDIUM.  AN ALTERNATE DEVICE MAY BE
;               SELECTED FOR OUTPUT AS IN THE "P" COMMAND BY APPENDING THE
;               APPROPRIATE CHARACTER TO THE LIST COMMAND.  FOR EXAMPLE, "*LP"
;               CAUSES THE LISTING TO BE MADE ON THE PRINTER.
;*ZERO          INITIALIZES THE MEDIUM ON THE SELECTED DEVICE FOR DUMPING.
;               ***THIS ERASES ANY BINARY PREVIOUSLY ON THE MEDIUM***
;*D             INITIATES THE DUMPING SEQUENCE.  THE USER ENTERS, AS PROMPTED,
;               THE FILE NUMBER, AND THE INITIAL AND FINAL ADDRESSES FOR EACH BLOCK.
;               DUMPS ARE TERMINATED BY SPECIFYING EITHER A START OR STOP BLOCK.
;               THESE ARE PRODUCED THE SAME AS FOR NON-FILE DEVICES. (SEE "D*")
;*I (N)         REQUESTS THAT A DUMP, THE ID OF WHICH IS REQUESTED, BE READ
;               INTO CORE.  THE ARGUMENT SPECIFIES THE RELOCATION ADDRESS,
;               IF REQUIRED.
```

```
;*T (N)          SELECTS THE UNIT NUMBER GIVEN BY (N) FOR THE DEVICE SPECIFIED.
;*M (N)          ALLOWS THE USER TO CORRECT ERRORS BY REPLACING ALL INFORMATION
;                BEGINNING WITH DUMP (N) WITH A NEW DUMP.  DUMP (N) AND ALL DUMPS
;                AFTER IT ARE LOST.  CONTINUE AS WITH THE "*D" COMMAND.
;*P (N)          POSITIONS A TAPE UNIT BEFORE THE FILE SPECIFIED FOR
;                COPYING PURPOSES, OR POSITIONS TAPE OR DISK DEVICES FOR
;                THE "*EOF" COMMAND.
;*EOF            WRITES AN END OF FILE OR TERMINATES DISK DUMPS AT THE PLACE
;                WHERE THE UNIT WAS LAST POSITIONED.  THIS EFFECTIVELY ERASES
;                THE REST OF THE DUMPS ON THAT MEDIUM.
;
;FCOPY           IS A SPECIAL BINARY COPY COMMAND WHICH MAY BE USED TO COPY
;                MAGNETIC TAPES OR PORTIONS OF MAGNETIC TAPES, COPY FROM DISK
;                TO DISK, OR TO BACK UP OR RECOVER A DISK IMAGE ON MAGNETIC
;                TAPE.  TO USE, PRECEED THE "FCOPY" COMMAND WITH TWO "*T (N)"
;                COMMANDS, THE FIRST OF WHICH SPECIFIES THE OUTPUT DEVICE, AND
;                THE SECOND OF WHICH SPECIFIES THE INPUT DEVICE. BE SURE TO
;                EXECUTE THE TRANSPORT SELECT COMMANDS BEFORE EACH "FCOPY".
;
;       "*" IS THE DEVICE TO BE USED, TYPICALLY "M" FOR MAGTAPE,
;       "Y" FOR RX02/FD211/FD411 FLOPPY DISK,
;       "Q" FOR RX01 FLOPPY DISK, AND "Z" FOR TA-11 TAPE UNIT.
;
```

```
;ERROR DIAGNOSTICS

; ?001  INVALID COMMAND.
; ?002  NUMERICAL OVERFLOW.  NUMERICAL ARG WAS GREATER THAN 177777 OR 65535.
;       ENTER NEW NUMBER.
; ?003  NUMERICAL ARGUMENT NOT UNDERSTOOD.  ENTER NEW NUMBER.
; ?004  COMMAND OR ARGUMENT TOO LARGE FOR BUFFER. RE-ENTER.
; ?005  LOAD UNSUCCESFUL DUE TO IMPROPER INPUT.
; ?006  DUPLICATE OR NONEXISTENT FILE SPECIFIED.
; ?007  NON EXISTENT DEVICE SPECIFIED FOR INPUT OR OUTPUT.
; ?010  LOAD UNSUCCESFUL DUE TO CHECKSUM ERROR.  CHECKSUM IS PRINTED.
; ?011  CPU TIMEOUT TRAP OCCURRED. PC AFTER TIMEOUT IS PRINTED.
;
; ?021  MAGTAPE DEVICE ERROR OCCURRED 8 TIMES ON SAME TRANSFER.
; ?022  TA-11 DEVICE ERROR OCCURRED 8 TIMES ON SAME TRANSFER.
; ?026  FLOPPY DEVICE ERROR OCCURED 8 TIMES ON SAME TRANSFER
;
; ?122  TA-11 END-OF-TAPE DETECTED.  KUS WAITS FOR ANY KEYBOARD KEY TO
;       BE PRESSED AND THEN CONTINUES.  THIS ALLOWS A DISK TO BE BACKED UP
;       ON A TA-11 CASSETTE BY MANUALLY REMOVING AND INSTALLING CASSETTES
;       EACH TIME ?122 IS TYPED.

;TRAP INSTRUCTIONS

;     ROUTINES WITHIN KUS MAY BE READILY ACCESSED WITH ONE WORD POSITION
; INDEPENDENT CALLS VIA THE TRAP INTERRUPT.  THE "ST" COMMAND MUST HAVE BEEN
; EXECUTED PREVIOUSLY TO SET UP THE VECTOR.  IN ALL OF THE FOLLOWING,
; THE DEVICE LAST SELECTED ("K" OR "C") IS USED FOR I/O.
; INSTRUCTION      PURPOSE
;  104400       TYPE CR-LF
;  104401       TYPE MESSAGE IMMEDIATE.  MESSAGE FOLLOWS, AND IS TERMINATED
;               BY A BYTE CONTAINING A ZERO OR MINUS VALUE.  EXECUTION RESUMES AT THE
;               NEXT EVEN ADDRESS.
;  104402       CONTENT OF R1 IS TYPED AS A 6 DIGIT OCTAL NUMBER, FOLLOWED
;               BY A SPACE.
;  104403       TYPE KUS STYLE ERROR MESSAGE.  ERROR NUMBER IS IN WORD FOLLOWING
;               CALL.
;  104404       OUTPUT BYTE IN R4 TO DEVICE DATA BUFFER.
;  104405       ASKS FOR A NUMBER TO BE KEYED IN.  R1= VALUE GIVEN. R0
;               EQUALS THE TERMINATING CHARACTER, OR 0 IF TERMINATOR WAS A
;               CR.  R3 AND R4 ARE ALSO DESTROYED.
;  104406       ENTER KUS COMMAND MODE.  DOES NOT RETURN.
;  104407       READ BYTE FROM DEVICE DATA BUFFER INTO R4.
;  104410       SIMULATE THE "P" COMMAND.  THE THREE ARGUMENTS FOR THE "P"
;               COMMAND MUST FOLLOW IN THE THREE WORDS FOLLOWING THE TRAP.
;               R0, R1, AND R2 ARE DESTROYED.
```

```
R0=%0
R1=%1
R2=%2
R3=%3
R4=%4
R5=%5
R6=%6
R7=%7
SP=%6
PC=%7

BLANK=40
RUBOUT=177
CR=15
LF=12
CRLF=5015
CTRLP=20
BELL=7

NOP=240
SR=177570

.ENABLE ABS
.=0
K=.
        MOV     (PC)+,R0
TKS:    177560          ;PRIMARY DEVICE CSR
        BR      START1
TKSINV: 60              ;PRIMARY DEVICE VECTOR
CCSR:   175610          ;SECONDARY DEVICE CSR (CRT ?)
LPS:    177514          ;PRINTER CSR
RCSR:   177550          ;READER/PUNCH CSR

        .IF     DF      M$CSR
        MTCSR:  M$CSR           ;MAGTAPE CSR
        .ENDC

        .IF     DF      Q$CSR
        FLPCSR: Q$CSR           ;RX01 FLOPPY DISK CSR
        .ENDC

        .IF     DF      Y$CSR
        Y.CSR:  Y$CSR           ;RX02/FD211/FD411 FLOPPY DISK CSR
        .ENDC

        .IF     DF      Z$CSR
        TACSR:  Z$CSR           ;TA-11 CSR
        .ENDC

START1: MOV     R0,INDEV        ;SELECT PRIMARY OR SECONDARY DEVICE
STARTX: CLR     (PC)+           ;EXIT "INPUT" MODE
INFLAG: 0
START:
        .IF     DF      PDP$11
        MOV     #300,@#-2       ;LOCK OUT INTS
        .ENDC

        .IF     DF      LSI$11
        MTPS    #300
        NOP                     ;KEEPS LOAD MAP THE SAME
        .ENDC

        MOV     PC,R0           ;CALCULATE ADDRESS OF STACK
        ADD     #0-.-2,R0
        MOV     R0,R6           ;SET STACK
        JSR     R5,RESETO       ;SET OUTPUT = COMMAND DEVICE

        CMP     R6,@R6          ;NEED WE IDENTIFY OURSELF ?
        BEQ     NOTALK
        MOV     R6,@R6          ;TURN OFF GREETING
        JSR     R5,TYPEM
        CRLF
        .ASCII  "ARB-KUS-V01-"

        .IF     DF      M$CSR
        .ASCII  "M"             ;MAGTAPE
```

```
        .IF     DF      Q$CSR
        .ASCII  "Q"             ;RX01 FLOPPY DISK
        .ENDC

        .IF     DF      Y$CSR
        .ASCII  "Y"             ;RX02/FD211/FD411 FLOPPY DISK
        .ENDC

        .IF     DF      Z$CSR
        .ASCII  "Z"             ;TA-11 CASSETTE
        .ENDC

        .BYTE   0
        .EVEN

NOTALK: TSTB    INFLAG          ;INPUT MODE ?
        BNE     TTY.TO.CORE     ;YES, CHECK FOR NUMBERS FIRST
        JSR     R5,RESETI       ;INPUT DEVICE = COMMAND DEVICE
        JSR     R5,TYPEM
        CRLF
        "*
        0

INPUT:  JSR     R5,INBUF        ;READ COMMAND
        TSTB    @R3             ;DID WE GET ANY CHARS ?
        BNE     .+10            ;WE HAVE ONE
        TST     R0              ;CR ?
        BEQ     START           ;YES
        BR      INPUT           ;NO, IGNORE

        JSR     R5,COMMAND      ;CHECK FOR COMMAND

HELP:   JSR     R5,ERROR        ;BAD COMMAND
        1
        BR      START
```

```
COMMAND=.;DECODE COMMAND
        MOV     R2,-(SP)
        MOV     PC,R2
        ADD     #ADDRESS-.,R2   ;R2 => ADDRESS FOR P,B,A,.A, + E CMDS
        CMPB    @R3,#'.         ;"." TYPE COMMAND ?
        BEQ     DOT
        CMPB    @R3,#'A         ;MAKE SURE ALPAH
        BLT     NOTALP
        CMPB    @R3,#'Z
        BGT     NOTALP
        MOVB    (R3)+,R4
        ASL     R4              ;DO TABLE LOOK UP FOR ROUTINE
        ADD     PC,R4
        ADD     #DOTAB-.-'A-'A,R4
        ADD     @R4,R4          ;FINALLY, R4 => ROUTINE
        MOV     TKS,LDEV        ;SET DEFAULT FOR "L" AND "D" CMDS
        MOV     #1404,MODPTX    ;DISABLE RUBOUT AS A TERMINATOR
        CLR     (PC)+           ;CLEAR
BFLAG:  .BYTE   0       ;BYTE/WORD MODE FLAG
SPECFG: .BYTE   0       ;NORMAL/SPECIAL OUTPUT DEVICE FLAG
        TSTB    @R3             ;SET CONDITION CODES ON NEXT CHAR
        JSR     R5,@R4          ;GO TO IT
        JSR     R5,TYPEM
        CR-400
STARTL=.
        BR      START           ;DO NEXT COMMAND
NOTALP: MOV     (SP)+,R2
        RTS     R5              ;COMMAND UNDEFINED
DOT=.   ;
        CMPB    1(R3),#'A               ;".A" ?
        BNE     NOTALP          ;IF NOT, REJECT
        CMPB    (R3)+,(R3)+
        INCB    BFLAG+1         ;SET IND FOR ASCII
        BR      BB              ;GO PICK UP ADDRESS, PRINT "BXXXXX"
```

```
E=.     ;
        INC     @R2             ;ROUND ADDRESS UP
        BIC     #1,@R2
        RTS     R5

P=.     ;
        JSR     R5,SPEC         ;GO LOOK FOR SPECIAL CHARS
        JSR     R5,GETARG       ;GET THE ADDRESS
        BNE     .+4
        MOV     @R2,R1          ;NONE GIVEN, R1 = ADDRESS
        MOV     R1,@R2          ;STORE ADDRESS
        JSR     R5,GETARG       ;GET THE END ADDRESS
        BNE     .+4
        MOV     @R2,R1          ;NO END GIVEN, USE ADDRESS
        MOV     R1,ENDADR
        JSR     R5,GETARG
        BNE     .+6
        MOV     #10,R1          ;VALUES PER LINE
        MOV     R1,PERLINE
        INCB    SPECFG          ;ENABLE SECONDARY OUTPUT DEV
PGO:    JSR     R5,ATYPE        ;TYPE "AXXXXXX"
        MOV     (PC)+,R0        ;START LINE CTR
PERLIN: 0
PAGAIN: JSR     R5,CTYPE        ;TYPE THE CONTENTS
        CMP     @R2,(PC)+       ;DONE ?
ENDADR: 0
        BHIS    RTS5A           ;YES
        JSR     R5,INCADR       ;NO, DO NEXT
        DEC     R0
        BGT     PAGAIN
        BR      PGO             ;BEGIN NEW LINE


A=.     ;
        JSR     R5,E            ;MAKE ADDRESS EVEN
        BR      .+6
B=.     ;
BB:     INCB    BFLAG           ;SET BYTE MODE
        JSR     R5,GETARG       ;GET THE ADDRESS
        BNE     .+4
        MOV     @R2,R1          ;UNLESS NONE IS GIVEN
        MOV     R1,@R2          ;STORE NEW ADDRESS
AADR:   TSTB    INFLAG          ;IF INPUT MODE, SKIP:
        BNE     .+6
        JSR     R5,ATYPE        ;TYPE "AXXXXXX"
        TSTB    BFLAG+1         ;IS THIS ASCII ?
        BNE     .A              ;YES
CNEXT:  TTY.TO.CORE:
        MOV     #1407,MODPTX    ;ENABLE RUBOUT AS VALID TERMINATOR
        JSR     R5,INBUF        ;GET A VALUE
        JSR     R5,COMMAND      ;CHECK FOR COMMAND FOR "INPUT" MODE
        JSR     R5,GETNUM       ;DECODE NUMBER
        BEQ     VOID            ;NONE GIVEN
        TSTB    BFLAG           ;THIS A BYTE OR WORD ??
        BNE     CBYTE           ;BYTE
        MOV     R1,@(PC)+       ;DEP WORD
ADDRES: 0
        BR      .+6
CBYTE:  MOVB    R1,@ADDRESS     ;BYTE
        JSR     R5,INCADR       ;INC ADDRESS
VOID:   TST     R0
        BEQ     AADR            ;BEGIN NEW LINE
        CMPB    R0,#LF          ;LF TO EXIT ?
        BEQ     STARTL          ;YES
        CMPB    R0,#RUBOUT      ;RUBOUT ?
        BNE     CNOTRU
        NEG     @R2
        JSR     R5,INCADR       ;DEC ADR BY -(-ADR+N)
        NEG     @R2
        BR      AADR            ;PRINT ADDRESS
CNOTRU: CMPB    R0,#'#          ;# TO PRINT OUT ?
        BNE     CNEXT           ;NO, GET NEXT VALUE
        JSR     R5,CTYPE        ;YES, TYPE OUT CONTENTS
        JSR     R5,INCADR       ;INC ADDRESS
        BR      CNEXT           ;GET NEXT
```

```
.A=.   ;
        JSR     R5,INCH
        JSR     R5,ECHO
        MOV     R4,R5           ;SAVE DELIMITER
.ALOOP: JSR     R5,INCH
        JSR     R5,ECHO         ;GET CHAR
        CMP     R4,R5           ;IF DELIMITER
        BEQ     STARTL          ;EXIT
        MOVB    R4,@ADDRESS     ;DEPOSIT
        INC     @R2
        BR      .ALOOP

INCADR=.;INC ADDRESS FOR A,B,+P
        INC     @R2
        TSTB    BFLAG
        BNE     .+4     ;IF BYTE, INC BY 1
        INC     @R2     ;IF WORD, INC BY 2
        RTS     R5

ATYPE=. ;TYPE "AXXXXXX"
        DECB    BFLAG           ;SET MODE = WORD
        BLT     .+14            ;SKIP IF IT WAS ALREDY
        JSR     R5,TYPEM
        CRLF
        'B
        BR      .+12            ;PRINT "A" OR "B"
        JSR     R5,TYPEM
        CRLF
        'A
        MOV     @R2,R1          ;R1 = ADDRESS
        JSR     R5,NTYPE        ;OUTPUT NUMBER
        INCB    BFLAG
RTS5A:  RTS     R5

CTYPE=. ;TYPE CONTENTS
        TSTB    BFLAG
        BNE     .+10            ;BYTE
        MOV     @ADDRESS,R1     ;WORD
        BR      .+6
        MOVB    @ADDRESS,R1
```

```
NTYPE=.         ;OUTPUT VALUE IN R1
        ;AS ASCII
        MOV     R4,-(SP)
        CLR     R4
        MOV     #6,-(SP)        ;DIGIT CTR
        TSTB    BFLAG           ;BYTE OR WORD ?
        BLE     NWORD           ;WORD
        ASR     @SP             ;BYTE, DO 3 DIGITS
        SWAB    R1              ;MSB'S ONLY
        BR      NBYTE

NNEXT:  ASL     R1              ;COPY 1 BIT INTO R4
        ROL     R4
NBYTE:  ASL     R1
        ROL     R4
NWORD:  ASL     R1
        ROL     R4
        BIS     #60,R4
        JSR     R5,OUTCH        ;OUTPUT DIGIT
        CLR     R4
        DEC     @SP             ;DONE ?
        BGT     NNEXT           ;NOT YET
        JSR     R5,TYPEM
        BLANK
        TST     (SP)+
        MOV     (SP)+,R4
        RTS     R5
```

```
GETARG=.;NUMERICAL ARG DECODER
        TSTB    @R3             ;ENTRY #1, READ IF NECESSARY
        BNE     GETNUM          ;NOT NECESSARY
        TST     R0              ;LAST ENTRY = CR ?
        BEQ     GETNUM          ;YES: RETURN VOID
GETNEW: JSR     R5,INBUF        ;NO: READ ENTRY
GETNUM=.        ;
        CLR     -(SP)
        CLR     (PC)+           ;RESET NFLAG, HENCE NUMBER IS + VOID
NFLAG:  0
        MOV     #10,(PC)+       ;ASSUME OCTAL
RADIX:  0
GETLED: CLR     R1              ;RESET VALUE
        MOV     R3,@SP          ;SAVE BUFFER PTR INCASE DECIMAL
        MOVB    (R3)+,R4                ;GET FIRST CHAR
        BEQ     NUMOUT          ;POSSIBLE VOID
        INCB    NFLAG           ;NO LONGER VOID
        CMPB    R4,#'+          ;LEADING "+" ?
        BNE     .+12
DECNUM: MOV     #10.,RADIX      ;TREAT AS DECIMAL
        BR      GETLED          ;GET ANOTHER LEADING DIGIT
        CMPB    R4,#'-          ;LEADING MINUS ?
        BNE     DODIG           ;NO, MUST BE DIGIT
        COMB    NFLAG+1         ;NUMBER IS NEGATIVE
        BR      GETLED

DODIG:  SUB     #'0,R4          ;NUMERICAL ?
        BLT     NUMERR          ;NO
        CMP     R4,RADIX        ;TOO BIT FOR RADIX ?
        BGE     NUMERR          ;YES
        MOV     R1,-(SP)        ;MULT R1 BY RADIX
        MOV     RADIX,-(SP)
        DEC     @SP
        BEQ     NEWDIG          ;DONE WITH MULT
        ADD     2(SP),R1
        BCC     .-10
NUMOFL: CMP     (SP)+,(SP)+     ;NUMERICAL OFLO
        JSR     R5,ERROR
        2
GETNWP: TST     (SP)+
        BR      GETNEW

NEWDIG: ADD     R4,R1           ;ADD IN NEW DIGIT
        BCS     NUMOFL          ;BAD
        CMP     (SP)+,(SP)+
        MOVB    (R3)+,R4        ;GET NEXT CHAR
        BNE     DODIG
NUMOUT: DEC     R3              ;POINT R3 TO NULL CHAR
        TST     (SP)+
        TST     NFLAG           ;EXIT TIME
        BPL     .+4
        NEG     R1              ;NEGATE IF MINUS SET
        TST     NFLAG           ;THIS ENTRY VOID ?
        RTS     R5              ; 'EZ' SET IF ENTRY VOID

NUMERR: CMPB    RADIX,#8.       ;TRY AGAIN IN DECIMAL ?
        BEQ     TRYDEC          ;YES
        CMP     R4,#'.-'0       ;DECIMAL
        BNE     .+6             ;NO, ERROR
        TSTB    (R3)            ;YES, MUST BE LAST CHAR
        BEQ     NUMOUT  ;OK
        JSR     R5,ERROR
        3
        BR      GETNWP          ;TRY AGAIN

TRYDEC: MOV     @SP,R3          ;RESTORE BUFFER PTR
        BR      DECNUM          ;SIMULATE LEADING "+"
```

```
INBUF=.         ;INPUT COMMAND OR NUMBER
        MOV     PC,-(SP)
        ADD     #INBUFR-.,@SP   ;@SP => BUFFER
INAGAI: MOV     @SP,R3
NEXTC:  JSR     R5,INCH         ;INPUT CHAR
        BEQ     NEXTC           ;IGNORE NULLS
        CMPB    R4,#'*          ;IGNORE ASTERICH
        BEQ     NEXTC
        BLT     INDONE          ;EXIT ON TERMINATOR
        CMPB    R4,#',
        BEQ     INDONE          ;"," IS ALSO TERMINATOR
        CMPB    R4,#177         ;RUBOUT ?
        BEQ     INRUB
        JSR     R5,ECHO         ;ECHO ALL ELSE
        MOVB    R4,(R3)+        ;PACK THIS CHAR
        TSTB    @R3             ;OUTA ROOM ?
        BGE     NEXTC           ;OK
        JSR     R5,ERROR
        4
        BR      INAGAIN

INRUB:  TSTB    INFLAG          ;IN INPUT MODE
        BNE     NEXTC           ;IGNORE RUBOUTS
        CMP     R3,@SP          ;TOO MANY RUBS
MODPTX: 1407    ;'BEQ INDONE+4' IN "A" MODE, ELSE  'BEQ INDONE-2'
        DEC     R3
        JSR     R5,TYPEM
        '\-400
        BR      NEXTC           ;ONE CHARACTER IS RUBBED
INDONE: JSR     R5,ECHO
        CMPB    R4,#BLANK       ;LEADING BLANK ?
        BNE     .+6             ;NO
        CMP     R3,@SP          ;MAYBE
        BEQ     NEXTC           ;YES
        CLRB    (R3)            ;TERMINATE BUFFER
        MOV     (SP)+,R3        ;SET OUTPUT PTR
        MOV     R4,R0           ;R0 = TERMINATOR
        CMP     R0,#CR
        BNE     .+4
        CLR     R0              ;IF CR, R0 = 0
        RTS     R5
```

```
S=.    ;
        BEQ     SETALL
        MOVB    @R3,R4
        BEQ     SRTS
        JSR     R5,SORTJ
        SETC-.
        SETJ-.
HELPL=.
        JMP     HELP
SETALL:
SETO:   MOV     PC,R1
        ADD     #TIMEOT-.,R1
        MOV     R1,@#4
        MOV     #340,@#6
        TSTB    @R3
        BNE     SNEXT
SETK:   MOV     PC,R1
        ADD     #TTYRT-.,R1
        MOV     TKSINV,R2
        MOV     R1,(R2)+        ;SET TTY VECTOR
        MOV     #200,(R2)+
        TSTB    @R3
        BNE     SNEXT
SETT:   MOV     PC,R1
        ADD     #TRAPI-.,R1
        MOV     R1,@#34
        MOV     #340,@#36
        TSTB    @R3
        BEQ     SRTS
SNEXT:  INC     R3
        BR      S+2

SETC:   .ASCII  "KOT"
        .BYTE   0
        .EVEN
SETJ:   SETK-.
        SETO-.
        SETT-.
```

```
TTYRT=.         ;TTY INT
        JSR     PC,TSTTTY
        RTI

I=.     ;
        MOV     TKS,NEWIN       ;"INPUT" NORMALLY FROM TTY
        INCB    INFLAG
        JSR     R5,SPEC         ;LOOK FOR "H"
        BNE     HELPL           ;UNIDENTIFIED CHARACTER
        MOV     NEWIN,ACTVIN    ;SET NEW INDEV
SRTS:   RTS     R5
```

```
;PROCESS SPECIAL CHARS
SPECP:  JSR     R5,TSTDEV       ;"P" SELECTS PRINTER
        LPS-.
        MOV     LPS,NEWOUT
        BR      SPEC1
SPECB:  INCB    BFLAG           ;"B" SELECTS BYTE MODE
        BR      SPEC1
SPECC:  MOV     CCSR,R4         ;"C" SELECTS CRT
        BR      SPECL+4
SPECH:  MOV     RCSR,R4         ;"H" SELECTS HSR/HSP
        BR      .+6
SPECK:                          ;"K" SELECTS TTY KBD
SPECL:  MOV     TKS,R4          ;"L" SELECTS TTY
        MOV     R4,NEWIN        ;DEP INDEV
        JSR     R5,TSTDEV       ;MAKE SURE HE EXISTS
        NEWIN-.
        MOV     R4,NEWOUT       ;DEP OUTDEV
        ADD     #4,NEWOUT
SPEC2:  MOV     R4,(PC)+        ;SAVE FOR LOAD + DUMP ROUTINES
LDEV:   0

SPEC1:  INC     R3
SPEC=.  ;ENTRY
        MOVB    @R3,R4
        BEQ     .+14            ;BUFFER EMPTY, EXIT
        JSR     R5,SORTJ        ;LOOK FOR SPECIAL CHAR
        SPECS-.
        SPECJ-.
        TSTB    @R3             ;EXIT, 'EZ' SET IF BUFFER EMPTY
        RTS     R5

SPECS:  .ASCII  "PBHLCK"
        .BYTE   0
        .EVEN
SPECJ:  SPECP-.
        SPECB-.
        SPECH-.
        SPECL-.
        SPECC-.
        SPECK-.

O=.     ;OPERATE COMMAND
        JSR     R5,SPEC         ;GET DEV
        BNE     HELPL
        MOV     NEWIN,R0        ;PICK IT UP
        JMP     START1          ;GO INTO C/I MODE
```

```
TSTIN=.         ;INPUT VERIFY
        MOV     R4,-(SP)
        CMPB    (R3)+,(R5)
        BNE     HELPL           ;BAD !!
        TSTB    (R5)+           ;DONE ?
        BNE     TSTIN+2
        DEC     R3
        BR      TYPEME          ;YES

SORTJ=. ;SORT + JUMP
        MOV     R0,-(SP)
        MOV     R5,R0
        ADD     @R5,R0          ;R0 => LIST ONE
        CMPB    R4,@R0          ;THIS ONE ?
        BEQ     MATCH
        TSTB    (R0)+           ;END LIST ?
        BNE     SORTJ+6         ;NO
        CMP     (R5)+,(R5)+
        BR      SORTEX          ;EXIT

MATCH:  SUB     R5,R0
        SUB     (R5)+,R0
        ASL     R0
        ADD     @R5,R0
        ADD     R0,R5           ;R5 => LIST TWO
        ADD     @R5,R5          ;R5 => DESIRED ROUTINE
SORTEX: MOV     (SP)+,R0
        RTS     R5

TYPEM=. ;MESSAGE OUTPUT ROUTINE
        MOV     R4,-(SP)
        MOVB    (R5)+,R4
        BLE     TYPEME
        JSR     R5,OUTCH
        BR      .-10
TYPEME: INC     R5
        BIC     #1,R5
        MOV     (SP)+,R4
        RTS     R5
```

```
TSTDEV=.;MAKE SURE DEVICE EXITS
        MOV     #4,R1
        MOV     @R1,-(SP)       ;IMPOSE OUR VADDRESS IN VECTOR
        MOV     PC,@R1
        ADD     #BADDEV-.,(R1)+
        MOV     @R1,-(SP)
        CLR     @R1
        MOV     R5,-(SP)
        ADD     (R5)+,@SP       ;@SP => CELL CONTAINING DEV ADDRESS
        MOV     @(SP)+,-(SP)    ;@SP => DEV ADDRESS
        TST     @(SP)+          ;HELLO ?
        MOV     (SP)+,@R1       ;RESTORE VECTOR
        MOV     (SP)+,-(R1)
        TST     R5              ;WAS DEV OK ?
        BNE     NEWIN+2         ;YES, RETURN
        JSR     R5,BERROR       ;NO
        7                       ;ILL DEV

BADDEV: CLR     R5              ;TIMEOUT ON DEV TEST
        RTI

;INPUT/OUTPUT DEV UPKEEP
RESETI=.;RESET INPUT DEV
        MOV     (PC)+,(PC)+
INDEV:  0                       ;MASTER DEVICE ADDRESS
ACTVIN: 0                       ;CURRENT INPUT DEVICE
        MOV     INDEV,(PC)+
NEWIN:  0                       ;DEVICE SELECTED FOR NEXT OUTPUT
        RTS     R5

RESETR=.;RESET CPU + PERIFERALS
        CLR     R0
        INC     R0
        BPL     .-2
        RESET
        INC     R0
        BMI     .-2
;       BR      RESETO          ;GO INSTATE COMMAND OUT DEVICE

RESETO=.;RESET OUTPUT DEV
        MOV     INDEV,R0
        CMP     (R0)+,(R0)+
        MOV     R0,(PC)+
ACTVOU: 0                       ;CURRENT OUTPUT DEVICE
        MOV     R0,(PC)+
NEWOUT: 0                       ;DEVICE SELECTED FOR NEXT INPUT
        RTS     R5
```

```
INCH=. ;CHARACTER INPUT
        MOV     ACTVIN,R4
        TSTB    INFLAG          ;INPUT MODE ?
        BEQ     INKEY           ;NO, KEYBOARD INPUT
        INC     @R4             ;ENABLE READER
        CLR     -(SP)           ;TIME
        TSTB    @R4             ;READY ?
        BMI     .+12            ;YES
        INC     @SP
        BPL     .-6             ;TIME OUT...EXIT INPUT MODE
XSTART: JMP     STARTX
        TST     (SP)+

INKEY:  TSTB    @R4             ;DEV READY ?
        BMI     CHIN            ;YES
        CMP     R4,TKS          ;THIS = MASTER DEVICE ?
        BEQ     INKEY           ;YES
        JSR     PC,TSTTTY       ;NO, DOES HE WANT US ?
        BR      INKEY           ;NO
CHIN:   MOV     2(R4),-(SP)     ;READ CHAR
        BIC     #-200,@SP       ;STRIP PARITY
        CMP     @SP,#CTRLP      ;^P ?
        BEQ     ZAP             ;YES
        MOV     (SP)+,R4        ;NO, EXIT WITH CHAR
        RTS     R5

ZAP:    MOV     R4,INDEV        ;THE GUY WHO HIT ^P = NEW MASTER
        JSR     R5,RESETR       ;RESET CPU
        JSR     R5,TYPEM
        CRLF
        "^P
        0
        BR      XSTART

TSTC=.  ;DO CHECK FOR ^P
        JSR     PC,TSTTTY       ;TTY HAVE ^P ?
TSTCX:  MOV     R4,-(SP)        ;NO
        MOV     INDEV,R4        ;COMMAND DEVICE HAVE ^P ?
TSTC1:  TSTB    @R4
        BPL     .+6
        JSR     R5,CHIN         ;?????
        MOV     (SP)+,R4        ;NO
        RTS     PC

TSTTTY: MOV     R4,-(SP)
        MOV     TKS,R4
        BR      TSTC1

OUTCH=. ;OUTPUT CHAR
ECHO=.  ;
        MOV     R4,-(SP)
        MOV     ACTVOUT,R4
        TSTB    SPECFG          ;DO WE USE INSTEAD SPECIAL DEVICE ?
        BLE     .+6             ;NO
        MOV     NEWOUT,R4       ;YES
        JSR     R5,WAITD        ;WHEN DEV READY
        MOV     (SP),2(R4)      ;OUTPUT CHAR
        CMP     R4,#175614      ;THIS THE DEC CRT
        BEQ     DUMB            ;DO DUMB DELAY
NORMAL: MOV     (SP)+,R4
        RTS     R5

DUMB:   BIT     #-40,@SP        ;DELAY NEEDED
        BNE     NORMAL          ;NO
        JSR     R5,TYPEM        ;YES
        77577
        77577
        00177
        BR      NORMAL

WAITD=. ;WAIT FOR DEVICE
        JSR     PC,TSTC         ;CHECK FOR ^P
        CMP     R4,TACSR        ;SPECIAL CHECK
        BNE     1$              ;IF NOT - SKIP
        BIT     #240,@R4        ;CHECK TR & READY !
        BEQ     WAITD           ;LOOP UNTIL READY
```

```
1$:     TSTB    @R4
        BPL     WAITD           ;AND DEV DONE
2$:     TST     @R4             ;SET ERR
        RTS     R5
```

```
TIMEOT=.;TIME-OUT
        MOV     @R6,R1
        MOV     PC,R6
        ADD     #0-.-2,R6       ;SET SP
        JSR     R5,MERROR
        11
TYPEAT: JSR     R5,TYPEM
        '@
        CLRB    BFLAG
        JSR     R5,NTYPE        ;GIVE OFFENDING ADDRESS
        JMP     START

BERROR=.;FATAL ERROR ENTRY
        INCB    INFLAG          ;INHIBIT RETURN

MERROR=.;RESET OUTDEV + ERR
        JSR     R5,RESETR       ;RESET DEV

ERROR=. ;ERROR OUTPUT SUBROUTINE
        DECB    SPECFG          ;DE-ACTIVATE SPECIAL OUTPUT DEV
        JSR     R5,TYPEM
        CRLF
        .BYTE   '?,BELL
        0
        INCB    BFLAG
        MOV     R1,-(SP)
        MOV     (R5)+,R1
        JSR     R5,NTYPE        ;TYPE #
        MOV     (SP)+,R1
        DECB    BFLAG
        INCB    SPECFG          ;RE-ACTIVATE SPECIAL OUTPUT DEV
        TSTB    INFLAG          ;IF INPUT MODE
        BNE     XSTART          ;TERMINATE
        RTS     R5
```

```
R=.    ;
        JSR     R5,SPEC         ;ALLOW "RP" ETC.
        JSR     R5,GETARG
        BNE     .+6
        MOV     RUNADR,R1       ;ADDRESS UNSPECIFIED, USE LAST
        MOV     R1,(PC)+
RUNADR: 0
        INCB    SPECFG          ;ACTIVATE SPECIFIED OUTPUT DEV
RUN=.   ;RUN AFTER SETTING TTY INT
        BIT     #1,R1           ;STOP BLOCK ??
        BNE     ZSTART          ;YES, ENTER C/I MODE
        MOV     PC,R2
        ADD     #TTYRT-.,R2
        CMP     R2,@TKSINV      ;IS VECTOR SET TO US
        BNE     .+10
        BIS     #100,@TKS       ;YES, ENABLE INTS
        CLR     -2              ;ENABLE KUS HELLO

        .IF     DF      PDP$11
        CLR     @#-2            ;RE-ALLOW INTS
        .ENDC

        .IF     DF      LSI$11
        MTPS    #0
        .ENDC

        MOV     R1,PC           ;BYE
ZSTART: JMP     START

C=.     ;CLEAR CORE
        JSR     R5,TSTIN        ;CHECK FOR SECOND "C"
        'C
        JSR     R5,GETARG       ;GET IA
        MOV     R1,R2
        JSR     R5,GETARG       ;AND FA
        BNE     .+4
        MOV     R6,R1           ;DEFAULT = CLEAR UP TO KUS
        CLR     (R2)+
        CMP     R2,R1
        BLOS    .-4             ;WHEEE !!!
        BR      ZSTART
```

```
;JUMP TABLE FOR MAIN
DOTAB:  A-.
        B-.
        C-.
        D-.
        E-.
        F-.
        G-.
        H-.
        I-.
        J-.
        K-.
        L-.
        M-.
        N-.
        O-.
        P-.
        Q-.
        R-.
        S-.
        T-.
        U-.
        V-.
        W-.
        X-.
        Y-.
        Z-.

A=HELP
B=HELP
C=HELP
D=HELP
E=HELP
F=HELP
G=HELP
H=HELP
I=HELP
J=HELP
K=HELP
L=HELP
M=HELP
N=HELP
O=HELP
P=HELP
Q=HELP
R=HELP
S=HELP
T=HELP
U=HELP
V=HELP
W=HELP
X=HELP
Y=HELP
Z=HELP
```

```
INBUFR:        0,0,0,0,-400    ;INPUT BUFFER FROM TTY FOR 8 CHARS

;TRAP HANDLER
TRAPI=.
        .IF     DF      PDP$11
        MOV     2(SP),@#-2      ;RUN AT CALLERS PRIORITY
        .ENDC

        .IF     DF      LSI$11
        MTPS    2(SP)
        NOP                     ;KEEP LOAD MAP THE SAME
        .ENDC

        MOV     R5,2(SP)        ;SAVE R5
        MOV     @SP,R5          ;FIND CALLING CODE
        MOVB    -2(R5),R5
        ASL     R5
        ADD     PC,R5
        ADD     #TTABLE-.,R5    ;R5 => TABLE
        ADD     @R5,R5          ;R5 => SUBROUTINE
        RTS     R5              ;SIMULATES JSR R5,-----

TTABLE: CRLFR-.         ;104400 ;OUTPUT CRLF, NO REGS CHANGED
        TYPEM-.         ;104401 ;OUTPUT IMMEDIATE MESSAGE, NO REGS CHANGED
        NTYPE-.         ;104402 ;OUTPUT WORD IN R1 AS OCTAL, R1 DESTROYED.
        ERROR-.         ;104403 ;OUTPUT FOLLOWING WORD AS KUS ERROR, NO REGS DESTROYED
        OUTCH-.         ;104404 ;OUTPUT CHAR IN R4, NO REGS CHANGED
        GETNEW-.        ;104405 ;ASK # FROM KBD. R1=RESULT, R0=TERMINATOR (0 IF CR)
                                ;REGS 3 AND 4 ARE ALSO DESTROYED
        STARTX-.        ;104406 ;ENTER KUS. DOES NOT RETURN.
        INCH-.          ;104407 ;READ CHAR INTO R4, OTHER REGS UNCHANGED.
        PRINTR-.        ;104410 ;SIMULATES P N1 N2 N3, ARGS FOLLOW IN NEXT
                                ;THREE WORDS. R0,R1,+ R2 ARE DESTROYED.

CRLFR:  JSR     R5,TYPEM
        CRLF
        0
        RTS     R5

PRINTR: MOV     PC,R2
        ADD     #ADDRESS-.,R2
        MOV     (R5)+,@R2
        MOV     (R5)+,ENDADR
        MOV     (R5)+,PERLIN
        JMP     PGO
```

```
;NON FILE DEVICES
;KUS HANDLERS
L=.     ;LOAD
        JSR     R5,SPEC         ;FIND DEVICE
        JSR     R5,GETARG       ;GET THE RELOCATION ADDR

;REGISTER USAGE:
;R5=PTR TO BYTE INPUT SUBROUTINE
;R4=DEV CSR (EXCEPT MINIDECK)
;R3=BYTE COUNT
;R2=MINIDECK CSR OR PAPERTAPE TIMER
;R1=CURRENT ADDRESS
;R0=DATA BYTE

        MOV     PC,R5           ;MAKE R5 = PTR
        ADD     #RDBYTE-.,R5
        BIC     #1,R1           ;RELOCATION ADDRESS MUST BE EVEN
        MOV     R1,-(SP)        ;@SP = RELOCATION
        MOV     LDEV,R4
LEADER: JSR     PC,@R5          ;READ BYTE
        DECB    R0              ;CHECK FOR 001
        BNE     LEADER          ;NO
        JSR     PC,@R5          ;NOW 000
        TST     R0
        BNE     LEADER+2        ;NO

LOADIN: MOV     #1,CKSUM        ;INIT CHECKSUM
        JSR     PC,RWORD
        MOV     R1,R3           ;SAVE BC
        JSR     PC,RWORD        ;GET ADDRESS
        ADD     @SP,R1          ;PLUS RELOCATION
        SUB     #6,R3           ;SUBTRACT HEADER FROM BC
        BNE     READIN          ;NOT A TRANSFER BLOCK

        JSR     PC,@R5          ;GET THE CHECKSUM
        BNE     ERR10           ;BAD CHECKSUM
        JMP     RUN             ;GO "RUN"

READIN: JSR     PC,@R5          ;READ BYTE
        MOVB    R0,(R1)+        ;DEP
        DEC     R3              ;DEC BC
        BNE     READIN
        JSR     PC,@R5          ;READ THE CHECKSUM
        BEQ     LEADER          ;GET THE NEXT BLOCK
ERR10:  JSR     R5,MERROR
        10
        MOV     CKSUM,R1        ;PRINT CKSUM
        JMP     TYPEAT

;BYTE INPUT SUBROUTINE
RDBYTE: INC     @R4             ;DEMAND CHARACTER
        CLR     R2              ;BEGIN TIMER
RDLOOP: CMP     R4,TKS          ;TEST FOR TTY ^P ?
        BEQ     .+6             ;NO
        JSR     PC,TSTTTY
        CMP     R4,INDEV        ;TEST FOR INDEV ^P ?
        BEQ     .+6
        JSR     PC,TSTCX        ;YES
        TSTB    @R4             ;TEST FOR DEV READY
        BMI     GETBYT
        INC     R2              ;AND FOR TIMEOUT
        BNE     RDLOOP
ERR5:   JSR     R5,BERROR
        5
GETBYT: MOV     2(R4),R0        ;GET THE BYTE
DOSUM:  ADD     R0,CKSUM        ;UPDATE CHECKSUM
        BIC     #-400,(PC)+     ;CLEAR EXTRA BITS, SET COND CODES
CKSUM:  0
        RTS     PC

;READ WORD SUBROUTINE
RWORD:  JSR     PC,@R5
        MOV     R0,R1
        JSR     PC,@R5
        SWAB    R0
        ADD     R0,R1
```

```
D=.    ;DUMP
        JSR     R5,SPEC         ;FIND DEVICE
        ADD     #4,LDEV         ;POINT TO OUTPUT DEV REGS
        JSR     R5,GETARG       ;GET IA
        MOV     R1,R2           ;SAVE THE IA
        JSR     R5,GETARG       ;GET THE FA
        BNE     GOTFA
        JSR     PC,BEGDMP       ;WAIT FOR LSP ON
        MOV     #20.,R1         ;OUTPUT TRAILER
        CLR     R0
        JSR     PC,PBYTE
        DEC     R1
        BGT     .-6
        DEC     R2
        BGT     .-20
        JSR     PC,BEGDMP       ;WAIT FOR LSP OFF
        RTS     R5
GOTFA:

;REGISTER USAGE:
;R5=RETURN
;R4=DEV ADDRESS
;R3=CHECKSUM
;R2=CURRENT ADDRESS
;R1=FINAL ADDRESS
;R0=DATA BYTE

DMPBLK: JSR     PC,BEGDMP       ;WAIT FOR LSP ON
        CLR     R3              ;CLEAR THE CHECKSUM
        MOV     #1,R0           ;OUTPUT 000001
        JSR     PC,PWORD
        MOV     R1,R0
        SUB     R2,R0           ;R0=BYTES TO OUTPUT-1
        MOV     R0,R1           ;SAVE THIS BC
        ADD     #7,R0           ;R0=BC
        JSR     PC,PWORD        ;OUTPUT BC
        MOV     R2,R0
        JSR     PC,PWORD        ;OUTPUT ADDRESS
        INC     R1              ;CHECK FOR BC = -1
DMPDAT: BEQ     DMPSUM          ;DONE WITH DATA, DO CHECK SUM
        MOVB    (R2)+,R0
        JSR     PC,PBYTE        ;OUTPUT
        DEC     R1
        BR      DMPDAT

DMPSUM: MOV     R3,R0
        JSR     PC,PBYTE        ;OUTPUT THE CHECKSUM
        JSR     PC,ENDDMP       ;FINISH DMP, ECT
        RTS     R5

;OUTPUT BYTE ROUTINE
PBYTE:  MOV     LDEV,R4         ;GET CSR ADDRESS
        JSR     R5,WAITD        ;WAIT FOR DEV
        MOV     R0,2(R4)        ;OUTPUT BYTE
PBYTSM: SUB     R0,R3           ;UPDATE CKSUM
        RTS     PC

;WORD OUTPUT
PWORD:  JSR     PC,PBYTE
        SWAB    R0
        JSR     PC,PBYTE
        RTS     PC

;END OF DUMP + BEG OF DUMP UPKEEP:
ENDDMP:
BEGDMP: CMP     LDEV,ACTVOUT    ;CMD AND DUMP DEVS THE SAME ?
        BNE     .+6             ;NO
        JSR     R5,INCH         ;INPUT CHAR W NO ECHO
        RTS     PC
```

```
;H.S. COPY
H=.     ;
        CMPB    @R3,#'C         ;COPY ?
        BEQ     .+6
        JMP     HELP
        JSR     R5,TSTDEV       ;HSR HERE
        RCSR-.                  ;?
        MOV     RCSR,R4

        INC     @R4
        JSR     PC,HINCH        ;PRINE 4 CH BUFFERING
        JSR     PC,HINCH
HLOOP:  JSR     PC,HINCH
        JSR     R5,WAITD
        TST     -4(R4)          ;END OF TAPE
        BMI     GOTFA-2         ;YES
        MOV     R2,2(R4)        ;OUTPUT CHAR
        BR      HLOOP

HINCH:  BIC     #6,R4
        JSR     PC,TSTC
        BIT     #100200,(R4)+   ;READY ?
        BEQ     HINCH           ;NO
        BMI     GOTFA-2         ;END OF TAPE
        MOV     R1,R2           ;ROTATE BUFFER
        MOV     R0,R1
        MOV     (R4)+,R0        ;READ CHAR
        INC     -4(R4)
        RTS     PC
```

```
;FILE LOAD/DUMP FRAMEWORK
;COMMAND PROCESSORS USING FOLLOWING DEV DEPENDENT SUBROUTINES:
;1)     TRAN    ;TRANSFER RECORD TO/FROM DEVICE
;2)     WEOF    ;TERMINATE DUMPS/WRITE END-OF-FILE
;3)     REW     ;REWIND (TAPE UNITS)
;4)     SKBK    ;SKIP DATA BLOCK (TAPE UNITS)
;5)     BACK    ;BACKSPACE TO BEFORE LAST ID

;ALL ROUTINES USE THE FOLLOWING STORAGE FOR READ/WRITE INFO:
PTR:    0       ;BEGINNING OF BLOCK FOR DISK LIKE DEVICES.
                ;POINTS TO NEXT ID BLOCK, =0 TO TERMINATE DUMPS.

                ;FOR TAPE DEVS, =0 WHEN AT EOF OR BEFORE ID BLOCK

ID:     0       ;BEGINNING OF BLOCK FOR TAPE DEVICES
                ;=ID NUMBER THIS DUMP
IA:     0       ;IA OF BLOCK
BC:     0       ;NEGATIVE BC OF BLOCK, READ AND WRITTEN /2 FOR WORD DEVS.
        .=ID+16 ;SCRATCH
TRBC:   0       ;-BC OF READ/WRITE
TRIA:   0       ;IA OF READ/WRITE

;REGISTER USAGE IS AS FOLLOWS:
;R0     =       DEV NUMBER
;R1     =       ID AND SCRATCH
;R2     =       POINTER TO PTR
;R3     =       CHARACTER OUTPUT PTR
;R4     =       ADDRESS OF DEVICE CSR
;R5     =       RETURN, RELOCATION ADDRESS, SCRATCH.

;COMMON ENTRY WITH R0=DEV NUMBER
COMMON: MOV     R0,-(SP)        ;PREPARE TO SAVE LAST UNIT FOR FCOPY
        JSR     R5,FUNIT        ;GET LAST UNIT
        MOV     R0,UNIT2        ;SAVE UNIT SELECT
        MOV     FCSR,FCSR2      ;SAVE HIS CSR
        MOV     DEV,DEV2        ;AND DEV NUMBER
        MOV     (SP)+,R0        ;READY FOR FCOPY XPORT #2

        MOV     R1,DEV          ;SAVE DEV
        MOV     R4,FCSR         ;AND CSR ADDRESS
        JSR     R5,TSTDEV
        FCSR-.
        MOV     PC,R2
        ADD     #PTR-.,R2       ;POINT R2 TO ID
        MOVB    (R3)+,R4
        BEQ     .+12
        JSR     R5,SORTJ
        FILECH-.
        FILEJP-.
HELPL=.
        JMP     HELP

FILECH: .ASCII  "IDWMLPZET"     ;COMMANDS
        .BYTE   0
        .EVEN
FILEJP: READ-.  ;ROUTINES
        DUMP-.
        DUMP-.
        MOD-.
        LIST-.
        PCMD-.
        ZERO-.
        EOFCMD-.
        TRANS-.

;"*T"
TRANS:  JSR     R5,GETARG       ;GET THE UNIT
PUNIT:  MOV     DEV,R0
        ADD     PC,R0
        ADD     #UNIT-.,R0
        MOVB    R1,@R0          ;STORE IT
        RTS     R5

;UNIT STORATE TABLE FOR 8 DEVS
UNIT=.-1
        0,0,0,0
```

```
;FCOPY
F=.     ;
        JSR     R5,TSTIN
        "CO
        "PY
        0
        MOV     PC,R2
        ADD     #PTR-.,R2
        CLR     ID              ;CLEAR EXIT INDICATOR
        CLR     @R2             ;CLEAR SECTOR CTR FOR DISKS
        CLR     TRIA            ;COPY LOOP.... IA = BOTTOM OF CORE
        MOV     R6,R0
        NEG     R0
        ADD     #20,R0
        MOV     R0,-(SP)                ;SET MAXIMUM BC
        CMP     DEV,#DISKS              ;DISK DEV ?
        BLE     FCOPY                   ;NO
        MOV     #-100000,R0
        ASR     R0
        CMP     @SP,R0                  ;FIND POWER OF 2 FOR BC
        BHI     .-6
        MOV     R0,@SP          ;USE THIS BC
FCOPY:  MOV     @R2,SECTOR
        MOV     @SP,TRBC                ;INIT BC
        JSR     R5,SORT         ;READ A RECORD
        TRAN-.
        0
FEXTRA: JSR     R5,SWAP         ;IMPOSE NEW XPORT
        JSR     R5,SORT         ;WRITE A RECORD
        TRAN-.
        2
        JSR     R5,SWAP         ;RESTORE PRIMARY XPORT
        JSR     R5,FNEXT        ;UPDATE SECTOR BASED ON BC
        TST     @R3             ;DONE ?
        BEQ     FCOPY           ;NO
        JMP     START

SWAP:   JSR     R5,FUNIT        ;GET CURRENT UNIT
        MOV     R0,-(SP)        ;AND SAVE
        MOV     DEV,-(SP)       ;SAVE DEV
        MOV     FCSR,-(SP)      ;SAVE DEV CSR
        MOV     FCSR2,FCSR      ;IMPOSE NEW ONES
        BEQ     FERR5
        MOV     DEV2,DEV
        MOV     UNIT2,R1
        JSR     R5,PUNIT
        MOV     (SP)+,(PC)+     ;SAVE OLD ONES
FCSR2:  0
        MOV     (SP)+,(PC)+
DEV2:   0
        MOV     (SP)+,(PC)+
UNIT2:  0
        RTS     R5
```

```
;"*I"
READ:   JSR     R5,GETARG       ;GET RELOCATION
        MOV     R1,R5
        JSR     R5,AFILE        ;ASK FILE #
        JSR     R5,POSIT        ;FIND FILE
        FERR6-.                 ;NOT THERE
        MOV     R1,-(SP)        ;SAVE FILE #
READL:  JSR     R5,IDTRAN
        0
        CMP     @SP,@R3         ;CORRECT ?
        BNE     FERR5
        INC     @R3
        ADD     R5,IA           ;ADD IN RELOCATION
        TST     BC              ;TRANSFER BLOCK ?
        BNE     NOTTRN
        JSR     R5,SORT
        REW-.                   ;SAVE TIME NEXT TIME
        MOV     IA,R1
        JMP     RUN             ;YES, TRANSFER.
NOTTRN: JSR     R5,TRTRAN
        0
        BR      READL

FERR5:  JSR     R5,BERROR
        5
```

```
;"*D"
DUMP:   JSR     R5,AFILE        ;ASK FILE #
        JSR     R5,POSIT        ;LOOK FOR IT
        DUMP1-.                 ;SHOULDN'T BE THERE
FERR6:  JSR     R5,BERROR
        6

;"*M"
MOD:    JSR     R5,GETARG
        JSR     R5,POSIT        ;POSITION AT OLD FILE
        FERR6-.
        JSR     R5,AFILE        ;ASK FILE # INCASE DIFFERENT
DUMP1:  MOV     R1,@R3          ;SAVE FILE #
        MOV     R3,-(SP)
        JSR     R5,TYPEM        ;GET IA
        CRLF
        "IA
        BLANK
        JSR     R5,GETNEW
        MOV     R1,IA
        JSR     R5,TYPEM
        CRLF
        "FA
        BLANK
        JSR     R5,GETNEW
        COM     R1
        ADD     IA,R1
        MOV     R1,BC
        MOV     (SP)+,R3
        JSR     R5,IDTRAN
        2
        JSR     R5,TRTRAN
        2
        TST     BC              ;DONE ?
        BNE     DUMP1+2         ;LOOP UNTIL HE GIVES XFER
        JSR     R5,SORT         ;TERMINATE DUMPS
        WEOF-.
EXITRW: JSR     R5,SORT
        REW-.                   ;REWIND
        RTS     R5              ;AND EXIT


;"*L"
LIST:   JSR     R5,SPEC         ;CHECK FOR SPECIAL OUTDEV
        INCB    SPECFG          ;ACTIVATE SPECIFIED OUTPUT DEV
        JSR     R5,SORT
        REW-.                   ;REWIND
LISTL:  JSR     R5,IDTRAN
        0
        TST     TRBC            ;EOF ?
        BEQ     EXITRW          ;YES, DONE.
        JSR     R5,SORT
        SKBK-.                  ;SKIP THE DATA BLOCK
        JSR     R5,TYPEM
        CRLF
        0
        MOV     @R3,R1
        JSR     R5,NTYPE        ;FILE #
        MOV     IA,R1
        JSR     R5,NTYPE        ;IA
        MOV     BC,R1
        COM     R1
        ADD     IA,R1
        JSR     R5,NTYPE        ;FA
        BR      LISTL
```

```
;"*P"
PCMD:   JSR     R5,GETARG
        JSR     R5,POSIT        ;SEARCH
        FERR6-.
        RTS     R5              ;AND RETURN

;"*XERO"
ZERO:   JSR     R5,TSTIN
        "ER
        'O
        JSR     R5,SORT
        REW-.
        BR      EOF1

;"*EOF"
EOFCMD: JSR     R5,TSTIN
        "OF
        0
EOF1:   JSR     R5,SORT
        WEOF-.
        RTS     R5
```

```
;-----------------------------------------------
;SUBROUTINES OF FILE LOAD/DUMP HANDLERS

;ID AND DATA BLOCK XFER ROUTINES FOR LOAD/DUMP
IDTRAN: MOV     R3,TRIA         ;FIRST ASSUME TAPE DEV
        CMP     DEV,#DISKS      ;DISK DEV ?
        BLE     IDTAPE          ;NO, SKIP:
        INC     @R2             ;SECTOR =
        MOV     @R2,(PC)+       ;PTR+1
SECTOR: 0
        MOV     BC,TRBC
        JSR     R5,FNEXT        ;POINT PTR TO NEXT USABLE SECTOR
IDTRAX: MOV     R2,TRIA
IDTAPE: MOV     #-16,TRBC
        BR      TRANX

TRTRAN: INC     SECTOR          ;DATA BEGINS IN SECTOR AFTER ID BLOCK
        MOV     IA,TRIA
        MOV     BC,TRBC
        BEQ     IDTAPE          ;WRITE DUMMY DATA ON XFER BLOCKS
TRANX:  CMP     DEV,#DISKS
        BLE     TRANXX          ;SKIP FOR TAPE:
        TST     BC
        BEQ     .+10
        SEC
        ROR     BC              ;MAKE BC INTO WC
TRANXX: MOV     (R5)+,.+12      ;COPY ARG + CALL DRIVER
        JSR     R5,SORT
        TRAN-.
        0
        CMP     DEV,#DISKS      ;DISK DEV
        BLE     EXTAPE          ;NO
        ASL     BC              ;CORRECT BC
        TST     @R2             ;THIS = TERMINATING ID ?
        BNE     .+6
        CLR     TRBC            ;RETURN BC=0 FOR EOF INDICATION
EXTAPE: TST     ERRS            ;IF ERROR OCCURED
        BNE     FDO-2
        JMP     START           ;ABORT XFER


;SORT CALL BASED ON DEV NUMBER
SORT:   MOV     (PC)+,R0        ;NORMAL ENTRY
DEV:    0
        MOV     (PC)+,R4        ;GET CSR ADDRESS
FCSR:   0
        MOV     R2,R3
        TST     (R3)+           ;POINT R3 TO ID
        MOV     #10,(PC)+
ERRS:   0                       ;INIT ERROR COUNTER
        TST     (R5)+
        MOV     R5,-(SP)

        ADD     -(R5),R5        ;POINT TO NEXT ROUTINE
        DEC     R0              ;THIS THE ONE ?
        BGT     .-4             ;NOT YET
        RTS     R5              ;GO

;ASK FILE
AFILE:  JSR     R5,TYPEM
        CRLF
        "FI
        "LE
        BLANK
        JSR     R5,GETNEW
        MOV     R2,R3
        TST     (R3)+           ;POINT R3 TO ID
        RTS     R5

;POSITON SUBROUTINE
;CALLING:       JSR     R5,POSIT
;               ADDRESS-. FOR RTN IF UNFOUND
;               RTN IF FOUND
POSIT:  TST     (R5)+
        JSR     R5,SORT
        REW-.
```

```
        0
        TST     TRBC            ;END-OF-FILE ?
        BNE     .+6             ;NO
        ADD     -(R5),R5        ;YES
        BR      POS02
        CMP     @R3,R1          ;THIS IT ?
        BEQ     POS02           ;YES
        JSR     R5,SORT         ;NO, SKIP DATA
        SKBK-.
        BR      POS01
POS02:  JSR     R5,SORT
        BACK-.
MDONE:  RTS     R5              ;RETURN AFTER BACKING UP

;FIND NEXT UNUSED BLOCK FOR DISK DEVICES
FNEXT:  MOV     #777,-(SP)
        SUB     TRBC,@SP        ;NEGATE, ADD 512, ROUND
        CLC
        ROR     @SP
        CLRB    @SP
        SWAB    @SP             ;DIVIDE BY 256
        ADD     (SP)+,@R2       ;ADD TO ID PTR
        RTS     R5

;RETURN INIT NUMBER IN R0
FUNIT:  MOV     DEV,R0
        ADD     PC,R0
        ADD     #UNIT-.,R0
        MOVB    @R0,R0
        CLC
        RTS     R5

;"DO" COMMAND FOR STANDARD DEC DEVICES
FDO:    JSR     R5,FUNIT        ;GET UNIT IN R0
        MOVB    R0,1(R4)        ;LOAD IT
        JSR     R5,WAITD        ;WAIT FOR READY
FDOX:   CLRB    @R4
        BIS     (R5)+,(R4)      ;BEGIN JOB
        JMP     WAITD           ;WAIT FOR COMPLETION

TDEV=0
DISKS=0
```

```
;************ DISK TYPE DRIVERS COME FIRST **************

;REWIND, BACKSPACE, SKIP DATA, AND WRITE END-OF-FILE FOR DISKS:


2
REW=.
        CLR     @R2
        RTS     R5

2
BACK=.
        MOV     SECTOR,@R2
        DEC     @R2
        RTS     R5

2
SKBK=.
        RTS     R5

2
WEOF=.
        INC     @R2
        MOV     @R2,SECTOR      ;IN PLACE OF PTR IN NEXT ID
        CLR     @R2             ;WRITE A 0
        JSR     R5,IDTRAX
        2
        RTS     R5

DSKTST: CMP     SECTOR,(R5)+    ;TEST FOR COPY TO/FROM DISK COMPLETE
        BGE     DSKFUL
        TST     TRBC            ;NOT YET, IS BC=0 ?
        BEQ     DSKFUL+2        ;YES, EXIT CALL WITH NO XFER
        SEC                     ;NORMAL XFER: MAKE BC INTO WC
        ROR     TRBC
        RTS     R5

DSKFUL: INC     @R3             ;INDICATE EXIT TIME
        CLR     TRBC            ;CAUSE EOF ON TAPE TO BE READ OR WRITTEN
        MOV     (SP)+,R5        ;POP THIS SUBROUTINE RTN
        TST     (R5)+           ;IGNORE IN/OUT CODE
        RTS     R5              ;RETURN WITH NO ACTION
```

```
       .IF     DF      Q$CSR

;RX11 FLOPPY DISK DRIVER
Q=.
        MOV FLPCSR,R4
        MOV #NODEV-TDEV,R1
        JMP COMMON
TDEV=TDEV+1
DISKS=TDEV
TRAN-.          ;READ/WRITE
TRAN=.
        MOV R3,-(SP)
        MOV R2,-(SP)
        MOV     R1,-(SP)
        JSR R5,DSKTST
        500.
        MOV #8.,ERRS
F.TRY:  MOV SECTOR,R0           ;PICK UP SECTOR
        ASL R0
        ASL R0                  ;TIMES 4 FOR FLOPPY LOGICAL SECTOR
        MOV TRBC,R1             ;RE=ACTUAL TRANSFER WC
        ASL R1                  ;MAKE IT A BC
        MOV TRIA,R2             ;R2=BUFFER ADDRESS
F.NBLK: MOV #1,R3               ;PUT FILL BUFFER COMMAND IN R3
        TST (R5)                ;READ OR WRITE?
        BNE F.SILO              ;WRITE, FILL SILO FIRST
        JSR R5,F.ISSU           ;READ, READ DATA INTO SILO
        .WORD 7
        MOV #3,R3               ;CHANGE COMMAND IN R3 TO READ BUFFER
F.SILO: JSR PC,F.LOAD           ;SET UNIT BIT IN R3, ISSUE COMMAND
        MOV #-128.,R3           ;EXACTLY 128 BYTES MUST BE TRANSFERRED
        SUB R3,R1               ;REMAINING BYTE COUNT < 128?
        BLE F.TRN               ;YES
        ADD R1,R3               ;NO, R3=-ACTUAL NUMBER OF BYTES TO TRANSFERR
F.TRN:  TST (R5)                ;READ OR WRITE?
        BNE F.WRB               ;WRITE
F.W0:   TSTB @R4                ;WAIT
        BEQ F.W0
        MOVB 2(R4),(R2)+        ;READ A BYTE
        INC R3                  ;END OF DATA?
        BLT F.W0                ;NO, READ ON
        BR F.SLUF               ;YES, SLUF REST OF SILO
F.WRB:  TSTB @R4                ;WRITE WAIT
        BEQ F.WRB
        MOVB (R2)+,2(R4)        ;WRITE BYTE
        INC R3                  ;DONE?
        BLT F.WRB               ;IF NOT, KEEP WRITING
F.SLUF: TSTB @R4                ;SILO NEED MORE TO REACH 128 BYTES?
        BEQ F.SLUF              ;DON'T KNOW YET
        BPL F.TRDN              ;IT ALREADY HAS 128 BYTES
        ADC 2(R4)               ;NEEDS MORE, READ OR WRITE TO FILL
        BR F.SLUF               ;EITHER READ OR WRITE
F.TRDN: JSR     R5,F.ETST               ;CHECK FOR ERROR
        TST (R5)                ;READ OR WRITE
        BEQ F.TSTD              ;READ - DONE WITH 128 BYTE BLOCK?
        JSR R5,F.ISSU           ;ISSUE WRITE TO TRANSFER SILO TO DISK
        .WORD 5
F.TSTD: INC R0                  ;ADVANCE TO NEXT SECTOR
        TST R1                  ;TRANSFER COMPLETE?
        BLT F.NBLK              ;NO, TRANSFER ANOTHER 128 BYTES
        BR      F.EXIT          ;YES, EXIT

;SUBROUTINE ISSUE - CONVERTS LOGICAL SECTOR TO PHYSICAL ADDRESS

;       -ISSUES READ OR WRITE COMMAND
;       -LOADS SECTOR & TRACK
;       -WAITS FOR COMPLETION AND TESTS FOR ERROR
F.ISSU: MOV R0,-(SP)            ;PRESERVE LOGICAL SECTOR
                                ;***TSI INTERLEAVE***
        CMP     #1000.,R0       ;IF LOGICAL SECTOR IS 0-249
        BHI     F.CV00          ;CARRY IS CLEAR
        ADD     #-1000.,R0      ;IF LOGICAL SECTOR IS 250-500
                                ;SUBTRACT 250. (CARRY IS SET)
F.CV00: ROL     R0              ;MULT BY 2, ADD CARRY
                                ;***END INTERLEAVE***
        MOV #8.,R3              ;PREPARE TO CONVERT LOGICAL SECTOR TO PHYSICAL
```

```
       .IF     DF      Y$CSR

;RX02/FD211/FD411 FLOPPY DISK DRIVER
Y=.
        JSR     PC,Y.STAT       ;CHECK OUT DISK
        MOV Y.CSR,R4
        MOV #NODEV-TDEV,R1
        JMP COMMON
Y.DNSD: .WORD   0               ;DENSITY AND SIDE
TDEV=TDEV+1
DISKS=TDEV
TRAN-.          ;READ/WRITE
TRAN=.
        MOV R3,-(SP)
        MOV R2,-(SP)
        MOV     R1,-(SP)
        JSR R5,DSKTST
Y.LEN:  500.                    ;MEDIA LENGTH
        MOV #8.,ERRS
Y.TRY:  MOV SECTOR,R0           ;PICK UP SECTOR
        BIT     #400,Y.DNSD     ;DOUBLE DENSITY ?
        BNE     1$              ;IF SO - SKIP
        ASL R0
1$:     ASL R0                  ;TIMES 2 OR 4 FOR FLOPPY LOGICAL SECTOR
        MOV TRBC,R1             ;RE=ACTUAL TRANSFER WC
        MOV TRIA,R2             ;R2=BUFFER ADDRESS
Y.NBLK: MOV #1,R3               ;PUT FILL BUFFER COMMAND IN R3
        TST (R5)                ;READ OR WRITE?
        BNE Y.SILO              ;WRITE, FILL SILO FIRST
        JSR R5,Y.ISSU           ;READ, READ DATA INTO SILO
        .WORD 7
        MOV #3,R3               ;CHANGE COMMAND IN R3 TO READ BUFFER
Y.SILO: JSR PC,Y.LOAD           ;SET UNIT BIT IN R3, ISSUE COMMAND
        MOV     #-128.,R3       ;128. WORDS
        BIT     #400,Y.DNSD     ;DOUBLE DENSITY ?
        BNE     1$              ;IF SO - SKIP
        ASR     R3              ;ELSE 64. WORDS FOR SINGLE DENSITY
1$:     SUB R3,R1               ;REMAINING WORD COUNT < NEEDED ?
        BLE Y.TRN               ;YES
        ADD R1,R3               ;NO, R3=-ACTUAL NUMBER OF WORDS TO TRANSFERR
Y.TRN:  TSTB @R4                ;WAIT
        BEQ Y.TRN
        NEG     R3              ;NEED + WC
        MOV     R3,2(R4)        ;LOAD WORD COUNT
1$:     TSTB    @R4             ;WAIT
        BEQ     1$
        MOV     R2,2(R4)        ;LOAD BUFFER ADDRESS
        ADD     R3,R2           ;UPDATE BUFFER POINTER
        ADD     R3,R2
Y.SLUF: TSTB @R4                ;WAIT
        BEQ Y.SLUF
Y.TRDN: JSR     R5,Y.ETST       ;CHECK FOR ERROR
        TST (R5)                ;READ OR WRITE
        BEQ Y.TSTD              ;READ - DONE WITH SECTOR
        JSR R5,Y.ISSU           ;ISSUE WRITE TO TRANSFER SILO TO DISK
        .WORD 5
Y.TSTD: INC R0                  ;ADVANCE TO NEXT SECTOR
        TST R1                  ;TRANSFER COMPLETE?
        BLT Y.NBLK              ;NO, TRANSFER ANOTHER SECTOR
        BR      Y.EXIT          ;YES, EXIT


;SUBROUTINE ISSUE - CONVERTS LOGICAL SECTOR TO PHYSICAL ADDRESS

;       -ISSUES READ OR WRITE COMMAND
;       -LOADS SECTOR & TRACK
;       -WAITS FOR COMPLETION AND TESTS FOR ERROR
Y.ISSU: MOV R0,-(SP)            ;PRESERVE LOGICAL SECTOR
        BIC     #1000,Y.DNSD    ;INIT TO SIDE 1
        CMP     #2000.,R0       ;ON SIDE 2 ?
        BHI     1$              ;IF NOT - SKIP
        BIS     #1000,Y.DNSD    ;SET SIDE
        SUB     #2000.,R0       ;USE SAME INTERLEAVE AS SIDE 1
                                ;***TSI INTERLEAVE***
1$:     CMP     #1000.,R0       ;IF LOGICAL SECTOR IS 0-249
        BHI     Y.CV00          ;CARRY IS CLEAR
```

```
       .IF     DF      M$CSR

;MAGTAPE MODULE
M=.     ;
        MOV     MTCSR,R4
        MOV     #NODEV-TDEV,R1
        JMP     COMMON
TDEV=TDEV+1

REW-.   ;REWIND
REW=.
        JSR     R5,FDO          ;LOAD REWIND
        60017
        RTS     R5

BACK-.  ;BACK UP 1 ID
BACK=.
        MOV     #-1,2(R4)
        JSR     R5,FDO
        60013
        RTS     R5

WEOF-.  ;WRITE EOF
WEOF=.
        JSR     R5,FDO
        60007
        RTS     R5

SKBK-.  ;SKIP BLOCK
SKBK=.
        MOV     #-1,2(R4)
        MOVB    #11,@R4         ;SKIP OVER DATA
        RTS     R5

TRAN-.  ;READ/WRITE
TRAN=.
        JSR     R5,WAITD
        MOV     #60003,R0       ;ASSEMBLE READ OR WRITE
        ADD     (R5)+,R0
        MOV     R0,MTCODE
MTTRY:  MOV     TRIA,4(R4)
        MOV     TRBC,2(R4)
        BEQ     WEOF
        JSR     R5,FDO
MTCODE: 0
        BPL     MTDON
        ROL     -2(R4)          ;EOF ?
        BMI     MTEOF
        DEC     ERRS
        BNE     MTAGAI
        JSR     R5,ERROR
        21
MTDON:  SUB     2(R4),TRBC
MTRTS:  RTS     R5
MTAGAI: JSR     R5,BACK
        BR      MTTRY
MTEOF:  CLR     TRBC
        RTS     R5

        .ENDC
```

```
       .IF     DF      Z$CSR

;TA-11 MODULE
Z=.     ;
        MOV     TACSR,R4
        MOV     #NODEV-TDEV,R1
        JMP     COMMON
TDEV=TDEV+1

REW-.   ;REWIND
REW=.
        JSR     R5,FDO
        17                      ;LOAD REWIND WITH GO
        MOV     #TA11Z,-(SP)    ;FROM 'ZERO' COMMAND ?
        ADD     PC,(SP)
TA11Z=ZERO+16-.
        CMP     R5,(SP)+
        BNE     TARWND          ;IF NOT - EXIT AFTER BLOCK SKIPPED
        JSR     R5,FDO          ;DO A SENTINAL WRITE
        3
        BISB    #20,(R4)        ;END WRITE
        JMP     WAITD

TARWND: JSR     R5,FDO          ;SKIP SENTINAL BLOCK
        15
        BISB    #20,(R4)        ;END SKIP
        RTS     R5

BACK-.  ;BACK UP
BACK=.
        MOV     (R4),-(SP)      ;SAVE STATUS
        JSR     R5,FDO
        11
        BIT     #4000,(SP)+     ;BACKUP FROM FILE GAP ?
        BEQ     .+10            ;NOPE - SKIP
        JSR     R5,FDO
        15
        RTS     R5

WEOF-.  ;WRITE END OF FILE
WEOF=.
        BIT     #10000,(R4)     ;WRITE PROTECT ON?
        BEQ     .+10
        JSR     R5,BERROR
        22
        JSR     R5,FDO
        1
        RTS     R5

SKBK-.  ;SKIP BLOCK
SKBK=.
        MOVB    #15,@R4
        RTS     R5

TRAN-.  ;READ/WRITE
TRAN=.
        MOV     #5,R0           ;ASSEMBLE READ OR WRITE
        SUB     (R5)+,R0
        MOV     R0,TACODE
TATRY:  MOV     TRIA,TAADR
        MOV     TRBC,TABC
        BEQ     WEOF
        JSR     R5,FDO
TACODE: 0
TALOOP: JSR     R5,WAITD+4
        CMPB    #204,(R4)       ;A READ ?
        BEQ     TAREAD
        CMPB    #202,(R4)       ;A WRTE ?
        BEQ     TAWRIT
        BIT     #4000,@R4       ;FILE GAP ?
        BNE     TAEOF
        TST     @R4
        BPL     TADON
        BIT     #20000,(R4)     ;END OF TAPE ?
        BEQ     1$              ;NO
        JSR     R5,ERROR        ;YES, DO END OF TAPE ERROR
```

```
;END OF FILE DRIVERS
NODEV=TDEV              ;NUMBER OF DEVICES
DISKS=NODEV-DISKS       ;DISK DEVS ARE DRIVERS GT "DISKS".

.END
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35

Table of contents

```
42-   1      EVALUATE NUMBER ROUTINE
43-   1      4-DIGIT BCD TO BINARY CONVERSION
44-   1      $R COMMAND
45-   1      $O COMMAND
46-   1      $M COMMAND
47-   1      $L COMMAND
48-   1      $D, $U, $V, $P, $S, AND $K COMMANDS
49-   1      $A, $B, $I, AND $J COMMANDS
50-   1      $H COMMAND
51-   1      $X COMMAND
52-   1      $Y COMMAND
53-   1      PROCESS ROUTINE
55-   1      $F COMMAND
56-   1      $C COMMAND
57-   1      XYSIZE ROUTINE
58-   1      X/Y GET ROUTINES
59-   1      GENERATE X STEP ROUTINE
60-   1      GENERATE Y STEP ROUTINE
61-   1      MULTIPLY ROUTINE
62-   1      $G COMMAND
63-   1      CHARACTER TABLES
```

6800 TERMINAL SOFTWARE

```
                                .SBTTL  6800 TERMINAL SOFTWARE
                                ;
                                ; THE SOFTWARE CONTAINED IN THIS PROGRAM IS FOR THE
                                ;OPERATION OF THE 6800 BASED VIDEO BOARD USED
                                ;WITH THE ARB-11 PROCESSOR.
                                ; THE SOFTWARE SUPPORTS A STANDARD DL-11 TYPE
                                ;INTERFACE FORMAT PLUS A SEPERATE INTERFACE FOR
                                ;THE GRAPHICS PROCESSOR.
                                ;
                                ;       TERMINAL INTERFACE
                                ;               KEYBOARD CSR = 177560
                                ;                        BUF = 177562
                                ;                        VEC = 60
                                ;               PRINTER  CSR = 177564
                                ;                        BUF = 177566
                                ;                        VEC = 64
                                ;
                                ;       GRAPHICS INTERFACE
                                ;               RESULT   CSR = 176670
                                ;                        BUF = 176672
                                ;                        VEC = 170
                                ;               COMMAND  CSR = 176674
                                ;                        BUF = 176676
                                ;                        VEC = 174
                                ;
```

DEFINITION OF I/O REGISTERS

```
                                .SBTTL  DEFINITION OF I/O REGISTERS
                                ;
                                ; THE I/O REGISTERS ARE ALL 6820 PIA CIRCUITS
                                ;
                                ;       REGISTER ADDRESSING
                                ; ___XX
                                ;    00 I/O REGISTER A (DIRECTION OF A)
                                ;    01 I/O REGISTER B (DIRECTION OF B)
                                ;    10 STATUS REGISTER A
                                ;    11 STATUS REGISTER B
                                ;
                                ;       REFER TO MC6820/6821 DATA SHEETS FOR DETAILS
                                ;       OF THE CONTROL REGISTERS
                                ;
```

I/O REGISTER LOCATIONS

```
                                .SBTTL  I/O REGISTER LOCATIONS
                                ;
000000                          DARHA=0         ;DISPLAY ADDRESS REGISTER H (A-PORT)
000001                          DARLB=1         ;DISPLAY ADDRESS REGISTER L (B-PORT)
000002                          PCRA1=2         ;CSR FOR DARHA [INIT  IRQ1]
000003                          PCRB1=3         ;CSR FOR DARHB
                                ;
000004                          CARHA=4         ;CURSOR ADDRESS REGISTER H (A-PORT)
000005                          CARLB=5         ;CURSOR ADDRESS REGISTER L (B-PORT)
000006                          PCRA2=6         ;CSR FOR CARHA
000007                          PCRB2=7         ;CSR FOR CARLB [VERTICAL RETRACE  NMI2]
                                ;
000010                          CSL=10          ;CURRENT SCAN LINE
000011                          ISL=11          ;INTERRUPT SCAN LINE
000012                          PCRA3=12        ;CSR FOR CSL
000013                          PCRB3=13        ;CSR FOR ISL [SCAN LINE  NMI3]
                                ;
000014                          CHARPL=14       ;CHARACTERS PER LINE
000015                          DSPCTL=15       ;DISPLAY CONTROL
                                                ;BIT    0 - GO
                                                ;       3 - GRAPHIC ENABLE
                                                ;       4 - CONTINUOS SCAN
                                                ;       5 - VIDEO POLARITY
                                                ;       7 - ODD FRAME
                                                ;
000016                          PCRA4=16        ;CSR FOR CHARPL [NMI STROBE]
000017                          PCRB4=17        ;CSR FOR DSPCTL [ODD FRAME  NMI1]
                                ;
000020                          WDFBA=20        ;WORD DATA FROM BUS <15:08>
000021                          WDFBB=21        ;WORD DATA FROM BUS <07:00>
000022                          PCRA5=22        ;CSR FOR WDFBA [DATA READY  IRQ8]
000023                          PCRB5=23        ;CSR FOR WDFBB
                                ;
000024                          WDTBA=24        ;WORD DATA TO BUS <15:08>
000025                          WDTBB=25        ;WORD DATA TO BUS <07:00>
```

```
37          000027                      PCRB6=27        ;CSR FOR WDTBB [DATA READY  IRQ9]
38                                      ;
39          000030                      TDF11=30        ;TERMNAL DATA FROM ARB-11
40          000031                      TDT11=31        ;TERMINAL DATA TO ARB-11
41          000032                      PCRA7=32        ;CSR FOR TDF11 [IRQ6]
42          000033                      PCRB7=33        ;CSR FOR TDT11 [IRQ7]
43                                      ;
44          000034                      KASCII=34       ;KEYBOARD
45          000035                      KBUTN=35        ;KEYBOARD BUTTONS
46                                                      ;BIT    0 - REPEAT
47                                                      ;       1 - BREAK
48                                                      ;       2 - PAPER
49                                                      ;       3 - HEREIS
50                                                      ;       4 - TAPE>
51                                                      ;       5 - <TAPE
52                                                      ;
53          000036                      PCRA8=36        ;CSR FOR KASCII [IRQ5]
54          000037                      PCRB8=37        ;CSR FOR KBUTN [BELL CONTROL]
55                                      ;
56          000040                      HA=40           ;PRINTER CONTROL REGISTER
57          000041                      HB=41           ;PRINTER DATA REGISTER
```

I/O REGISTER LOCATIONS

```
     58         000042                          PCRA9=42        ;CSR FOR HA
     59         000043                          PCRB9=43        ;CSR FOR HB
     60                                         ;
     61         000044                          PHOTO=44        ;PHOTOREADER DATA
     62         000045                          NPRCSR=45       ;NPR CONTROL REGISTER
     63                                                         ;BIT    0 - NPR GO
     64                                                         ;       1 - BUS C0 CONTROL
     65                                                         ;       2 - BUS C1 CONTROL
     66                                                         ;       3 - ADDRESS BIT 16
     67                                                         ;       4 - ADDRESS BIT 17
     68                                                         ;       5 - NXM ERROR
     69                                                         ;       6 - BUS GRANT LATE ERROR
     70                                                         ;       7 - NPR DONE
     71                                                         ;
     72         000046                          PCRA10=46       ;CSR FOR PHOTO [IRQ3]
     73         000047                          PCRB10=47       ;CSR FOR NPR CONTROL [IRQ10]
     74                                         ;
     75         000050                          NPRADA=50       ;NPR ADDRESS <15:08>
     76         000051                          NPRADB=51       ;NPR ADDRESS <07:00>
     77         000052                          PCRA11=52       ;CSR FOR NPRADA [TAPE<  IRQ11]
     78         000053                          PCRB11=53       ;CSR FOR NPRADB [TAPE>  IRQ12]
     79                                         ;
     80         000054                          NPRDTA=54       ;NPR DATA TO BUS <15:08>
     81         000055                          NPRDTB=55       ;NPR DATA TO BUS <07:00>
     82         000056                          PCRA12=56       ;CSR FOR NPRDTA [HERE IS  IRQ13]
     83         000057                          PCRB12=57       ;CSR FOR NPRDTB [PAPER  IRQ14]
     84                                         ;
     85         000060                          NPRDFA=60       ;NPR DATA FROM BUS <15:08>
     86         000061                          NPRDFB=61       ;NPR DTAT FROM BUS <07:00>
     87         000062                          PCRA13=62       ;CSR FOR NPRDFA [BREAK  IRQ15]
     88         000063                          PCRB13=63       ;CSR FOR NPRDFB [REPEAT  IRQ16]
     89                                         ;
```

GRAPHICS VARIABLES

```
      1                                    .SBTTL  GRAPHICS VARIABLES
      2                                    ;
      3         001200                     .=1200  ;VARIABLE AREA
      4                                    ;
      5 001200                    YTABLE: .BLKB   30      ;Y AXIS TABLE
      6 001230                    XTABLE: .BLKB   30      ;X AXIS TABLE
      7 001260                    PTABLE: .BLKB   20      ;PLOTTING TABLE
      8                                    ;
      9 001300     000     000    OPR:    .BYTE   0,0     ;OPERATION CODE
     10 001302     000     000    CHK:    .BYTE   0,0     ;XY PLOT BOUNDS CHECK
     11 001304     000     000    SAVEX:  .BYTE   0,0     ;TEMPORARY
     12 001306     000     000    GTEMPA: .BYTE   0,0
     13 001310     000     000    GTEMPB: .BYTE   0,0
     14 001312     000     000    VCNTR:  .BYTE   0,0     ;VECTOR LENGTH COUNT
     15 001314     000             DIVCNT: .BYTE   0       ;DIVIDE COUNTER
     16 001315     000     000    DND:    .BYTE   0,0     ;DIVIDEND
     17 001317     000     000    DSR:    .BYTE   0,0     ;DIVISOR
     18 001321     000     000    QT:     .BYTE   0,0     ;QUOTIENT
     19 001323     000             GCSR:   .BYTE   0       ;CURRENT COMMAND CODE
     20 001324     000             BT:     .BYTE   0       ;BIT COUNTER
     21 001325     000             LT:     .BYTE   0       ;LINE COUNTER
     22 001326     000             CWORD:  .BYTE   0       ;BIT PATTERN FROM CHARACTER RAM
     23 001327     000             WTBYTE: .BYTE   0       ;DATA FOR WRITE FUNCTION
     24 001330     000     000    RADD:   .BYTE   0,0     ;CHARACTER ADDRESS IN ASCII SELECT
     25 001332     000             HADD:   .BYTE   0       ;HIGH BITS ADDRESS
     26 001333     000     000    RDADD:  .BYTE   0,0     ;BYTE ADDRESS
     27 001335     000     000    CADR:   .BYTE   0,0     ;COLUMN ADDRESS
     28 001337     000     000    ROWUPD: .BYTE   0,0     ;CHARACTER ROW VALUES
     29 001341     000     000    ROWADD: .BYTE   0,0
     30 001343     000     000    ROWRST: .BYTE   0,0
     31 001345     000     000    COLUPD: .BYTE   0,0     ;CHARACTER COLUMN VALUES
     32 001347     000     000    COLADD: .BYTE   0,0
     33 001351     000     000    COLRST: .BYTE   0,0
     34                                   ;
     35                                   ;       LIST PROCESSOR VARIABLES
```

```
     37 001353                          LSBUFF: .BLKB   9.              ;CHARACTER BUFFER
     38 001364     000                  LSTCNT: .BYTE   0               ;CHARS IN LSBUFF
     39 001365     000     000          LPNTR:  .BYTE   0,0             ;POINTER TO CHARS
     40 001367     000                  FORMAT: .BYTE   0               ;INTEGER FORMAT LENGTH
     41 001370     000     000          GOSUB:  .BYTE   0,0             ;ADDRESS OF PROCESS
     42 001372     000                  CHARG:  .BYTE   0               ;CURRENT CHARACTER
     43 001373     000                  PCHAR:  .BYTE   0               ;PREVIOUS CHARACTER
     44 001374     000                  SCHFLG: .BYTE   0               ;SEEN A CHARACTER FLAG
     45 001375     000                  STPFLG: .BYTE   0               ;STRIPPING FLAG
     46                                         ;
     47                                         ;       EVALN VARIABLES
     48                                         ;
     49 001376     000                  NSIGN:  .BYTE   0               ;SIGN OF NUMBER
     50 001377     000     000     000  NUMBER: .BYTE   0,0,0,0          ;16-BIT RESULT / 4 CHARACTER BUFF
ER
        001402     000
     51 001403     000                  NCNTR:  .BYTE   0               ;DIGITS IN NUMBER
     52 001404     000     000          NPNTR:  .BYTE   0,0             ;POINTER FOR DIGITS
     53 001406     000                  T0:     .BYTE   0               ;TEMPS
     54 001407     000                  T1:     .BYTE   0
     55 001410     000                  T2:     .BYTE   0
     56 001411     000                  T3:     .BYTE   0
```

GRAPHICS VARIABLES

```
     57                                     ;
     58 001412     000                PNTSKP: .BYTE   0               ;POINTS TO SKIP
     59 001413     000     000        CHPNTR: .BYTE   0,0             ;CHARACTER POINTER
     60 001415     000                PROW:   .BYTE   0               ;CHARACTER DOTS
     61 001416     000     000        SVGSUB: .BYTE   0,0             ;GOSUB SAVE
     62 001420     000                SVSTAT: .BYTE   0               ;RSTAT0 SAVE
     63                                     ;
```

RUN AND LIST STATUS DEFINITION

```
      1                                     .SBTTL  RUN AND LIST STATUS DEFINITION
      2                                     ;
      3 001421     000                RSTAT0: .BYTE   0               ;RUN STATUS
      4                                     ;
      5                                     ;       RSTAT0
      6                                     ;
      7                                     ;       7-      RUNNING (1)
      8                                     ;       6-      RELATIVE ORIGIN (0) / RELATIVE POSITION (
1)
      9                                     ;       5-      AUTO X STEP (1)
     10                                     ;       4-      AUTO Y STEP (1)
     11                                     ;       3-      VECTOR (0) / POINT (1)
     12                                     ;       2-      HOLD FLAG
     13                                     ;       1-      PEN UP (0) / PEN DOWN (1)
     14                                     ;
     15 001422     000                LSTAT:  .BYTE   0               ;LIST PROCESSOR STATUS
     16                                     ;
     17                                     ;       LSTAT
     18                                     ;
     19                                     ;       7- SINGLE CHARACTER MODE
     20                                     ;       6- LIST SCAN
     21                                     ;       5- BUILD MODE
     22                                     ;       3- SIZING FLAG
     23                                     ;       2- LOWER/GREEK
     24                                     ;       1- SCAN/PLOT
     25                                     ;       0- HORIZONTAL/VERTICAL
     26                                     ;
     27                                     ;
```

ABSOLUTE DEFINITIONS

```
                                .SBTTL  ABSOLUTE DEFINITIONS
                                ;
020000                          CTLINE=20000    ;CONTROL LINE LOCATION
                                ;
000100                          GRAPH=100       ;GLOBLE GRAPH ON/OFF FLAG
000105                          CHRFLG=105      ;GLOBLE INHIBIT PRINTING FLAG
000106                          UPDFLG=106      ;GLOBLE DISPLAY UPDATE FLAG
                                ;
                                ;
                                ; THESE MEMORY LOACATIONS CONTAIN THE
                                ;INTERRUPT VECTOR ADDRESSES AS INITIALIZED
                                ;BY THE RUNNING PROGRAM
                                ;
                                ; NON-MASKABLE INTERRUPT VECTORS
                                ;
007732                          NMI1    =7732           ;ODD FRAME
007734                          NMI2    =7734           ;VERTICAL RETRACE
007736                          NMI3    =7736           ;SCAN LINE
                                ;
                                ; NORMAL INTERRUPT VECTORS
                                ;
007740                          IRQ1    =7740           ;ARB-11 BUS INIT
007742                          IRQ2    =7742           ;60 HZ CLOCK
007744                          IRQ3    =7744           ;PHOTO READER DATA
007746                          IRQ4    =7746           ;PRINTER CONTROL
007750                          IRQ5    =7750           ;KEYBOARD DATA
007752                          IRQ6    =7752           ;DATA FROM ARB-11
007754                          IRQ7    =7754           ;DATA TAKEN BY ARB-11
007756                          IRQ8    =7756           ;DISPLAY DATA FROM ARB-11
007760                          IRQ9    =7760           ;DISPLAY DATA TO ARB-11 TAKEN
007762                          IRQ10   =7762           ;ARB-11 BUS NPR DONE
007764                          IRQ11   =7764           ;SCROLL  <TAPE
007766                          IRQ12   =7766           ;SCROLL  >TAPE
007770                          IRQ13   =7770           ;HERE IS BUTTON
007772                          IRQ14   =7772           ;PAPER BUTTON
```

```
 37         007776                          IRQ16   =7776           ;REPEAT BUTTON
 38                                         ;
 39         010000                  CHRAM   =10000          ;4096 BYTE CHARACTER DEFINITION BUFFER
 40                                                         ;CONTAINING 256 CHARACTER DEFINITIONS
 41                                         ;
 42         030000                  GPBOT   =30000          ;20480 BYTE GRAPHICS AREA
 43                                                         ;ALLOWS A 640(X) BY 256(Y) DISPLAY
 44         030000                  EVNSCN  =30000          ;SCAN ADDRESS FOR EVEN SCANS
 45         054000                  ODDSCN  =54000          ;SCAN ADDRESS FOR ODD SCANS
 46         077777                  GPTOP   =77777          ;LAST DISPLAY BYTE
 47                                         ;
 48         100000                          CHROM1  =100000 ;CHARACTER ROM #1 (2K BYTES)
 49         104000                          CHROM2  =104000 ;CHARACTER ROM #2 (2K BYTES)
 50                                         ;
```

SUMMARY OF GRAPHICS TABLES

```
      1                                    .SBTTL  SUMMARY OF GRAPHICS TABLES
      2                                    ;
      3                                    ;       THREE PARAMETER TABLES ARE USED BY
      4                                    ;       THE GRAPHICS PROCESSOR.
      5                                    ;
      6                                    ;       1       XTABLE
      7                                    ;       2       YTABLE
      8                                    ;       3       PLTBLE
      9                                    ;
     10                                    ;       THE X AND Y TABLES ARE ORGANIZED AS FOLLOWS:
     11                                    ;
     12                                    ;_VAL   0,X     PREVIOUS VALUE (16 BITS)
     13                                    ;F_VAL  2,X     NEW VALUE (16BITS)
     14                                    ;_STEP  4,X     RELATIVE REGISTER (16 BITS)
     15                                    ;VCTL_  6,X     VECTOR LENGTH (16 BITS)
     16                                    ;VAL_   10,X    RUNNING VALUE DURING VECTOR PLOT (16 BITS
)
     17                                    ;       12,X    RUNNING FRACTIONAL VALUE DURING VECTOR PL
OT (16 BITS)
     18                                    ;FRCT_  14,X    UPDATE VALUE DURING VECTOR PLOT (16 BITS)
     19                                    ;       16,X    FRACTIONAL UPDATE VALUE DURING VECTOR PLO
T (16 BITS)
     20                                    ;SING_  20,X    SIGN OF UPDATE
     21                                    ;_SIZE  21,X    CHARACTER SIZING
     22                                    ;_ORG   22,X    ORIGIN
     23                                    ;_CNT   24,X    POINT POSITION
     24                                    ;P_CNT  25,X    PREVIOUS POSITION
     25                                    ;SAV_   26,X    SAVE STEP SIZE
     26                                    ;
     27                                    ;
     28                                    ;       THE PLOTTING TABLE IS ORGANIZED AS FOLLOWS:
     29                                    ;
     30                                    ;       0,X     Y POSITION VALUE (16 BITS)
     31                                    ;       2,X     X POSITION VALUE (16 BITS)
     32                                    ;       4,X     HORIZONTAL LIN BASE ADDRESS
     33                                    ;       6,X     BYTE OFFSET IN LINE
     34                                    ;       10,X    EXPLICIT BIT POSITION
```

```
     35                                     ;       11,X    CONTROL STATUS DURING PLOT
     36                                     ;       12,X    ABSOLUTE LOCATION OF BYTE TO BE UPDATED
     37                                     ;       14,X    TEMPORARES (- 17,X)
     38                                     ;
```

SUMMARY OF PLOTTER COMMANDS

```
      1                                 .SBTTL  SUMMARY OF PLOTTER COMMANDS
      2                                 ;
      3                                 ;       $R_     INITIATE VIDEO GRAPHICS
      4                                 ;               _ IS AN OPTIONAL INTEGER FORMAT SPECIFIER
      5                                 ;               FROM 0 TO 9. 0 IS THE DEFAULT VALUE
      6                                 ;               AND ALLOWS FREE FORMAT INPUT
      7                                 ;
      8                                 ;       $O      DEFINE ORIGIN OF PLOT
      9                                 ;               DATA FOLLOWING THE COMMAND IS STORED AS
     10                                 ;               XORIGIN, YORIGIN. IF MORE DATA IS SPECIFI
ED
     11                                 ;               THEN DATA IS SEQUENCIALLY STORED IN
     12                                 ;               XO, YO, XO, YO,  ETC.
     13                                 ;
     14                                 ;       $M      MOVE TO POINT RELATIVE TO ORIGIN MODE.
     15                                 ;               DATA FORMAT IS X, Y, X, Y,  ETC.
     16                                 ;               POSITION IS (XO+X,YO+Y),  ETC.
     17                                 ;
     18                                 ;       $L      MOVE RELATIVE TO CURRENT POSITION
     19                                 ;               DATA FORMAT IS DX, DY, DX, DY,  ETC.
     20                                 ;               POSITION IS  (X+DX,Y+DY) THEN X=X+DX, Y=Y
+DY
     21                                 ;
     22                                 ;       $D      PEN DOWN COMMAND
     23                                 ;
     24                                 ;       $U      PEN UP COMMAND
     25                                 ;
     26                                 ;       $X      AUTO X STEP MODE
     27                                 ;               FIRST DATA IS X STEP VALUE
     28                                 ;               SUBSEQUENT DATA IS Y VALUE (RELATIVE TO O
RIGIN)
     29                                 ;               PLOTTED DATA IS THUS:
     30                                 ;                       X,YO+Y  THEN X=X+XSTEP
     31                                 ;                       X,YO+Y  THEN X=X+XSTEP  ETC.
     32                                 ;
     33                                 ;       $Y      AUTO Y STEP MODE
     34                                 ;               FIRST DATA IS Y STEP VALUE
```

```
     35                                     ;               SUBSEQUENT DATA IS X VALUE (RELATIVE TO O
RIGIN)
     36                                     ;               PLOTTED DATA IS THUS:
     37                                     ;                       XO+X,Y  THEN Y=Y+YSTEP
     38                                     ;                       XO+X,Y  THEN Y=Y+YSTEP  ETC.
     39                                     ;
```

SUMMARY OF PLOTTER COMMANDS

```
      1                                 ;
      2                                 ;       $F__... CHARACTER FORMAT SPECIFIER
      3                                 ;               THE $F COMMAND IS FOLLOWED (IN ANY ORDER)
      4                                 ;               BY CHARACTERS H (HORIZONTAL), V (VERTICAL
),
      5                                 ;               L (LOWER CASE), G (GREEK CHARACTERS),
      6                                 ;               OR DIGITS 0-9. THE FIRST DIGIT IS THE
      7                                 ;               CHARACTER HEIGHT SPECIFIER AND THE SECOND
      8                                 ;               IS THE CHARACTER WIDTH SPECIFIER.
      9                                 ;               ANY OTHER CHARACTER TERMINATES THIS COMMA
ND
     10                                 ;
     11                                 ;       $C      CHARACTER MODE PLOTTING
     12                                 ;               THE $C COMMAND INITIATES THE CHARACTER
     13                                 ;               DRAWING SEQUENCE.
     14                                 ;                       1. THE PEN IS LIFTED
     15                                 ;                       2. SEARCH MODE IS INITIATED FOR A
 +
     16                                 ;                       3. ALL CHARACTERS AFTER THE + ARE
 DRAWN
     17                                 ;                       4. ANY CONTROL CHARACTER <CR>,<LF
>
     18                                 ;                               TERMINATES THE $C COMMAND
     19                                 ;
     20                                 ;       $G_     GRAPHICAL PLOT COMMAND
     21                                 ;               THIS COMMAND PROVIDES 10 GRAPHICAL
     22                                 ;               SYMBOLS WHICH MAY BE PLOTTED AS
     23                                 ;               DATA MARKERS. THE COMMAND DOES NOT
     24                                 ;               CHANGE THE CURRENT PLOTTING
     25                                 ;               MODE DURRING EXECUTION. HOWEVER, THE
     26                                 ;               COMMAND PLOTS IN VECTOR MODE AND ALWAYS L
IFTS
     27                                 ;               THE PEN. _ IS A DIGIT 0-9 SPECIFYING THE
     28                                 ;               SYMBOL TO BE PLOTTED.
     29                                 ;
```

SUMMARY OF PLOTTER COMMANDS

```
      1                                 ;       $V      VECTOR PLOTTING
      2                                 ;               DRAW LINE BETWEEN END POINTS
      3                                 ;
      4                                 ;       $P      POINT PLOTTING
      5                                 ;               PLOT ONLY END POINT
      6                                 ;
      7                                 ;       $H      HOLD COMMAND
      8                                 ;               WAIT FOR OPERATOR TO PUSH 'HERE IS' BUTTO
N
      9                                 ;
     10                                 ;       $K      CLEAR VIDEO COMMAND
     11                                 ;
     12                                 ;       $W      INCLUDED FOR COMPATABILITY ONLY
     13                                 ;
     14                                 ;       $Z      SET PLOTTING INTENSITY
     15                                 ;
     16                                 ;       $S      STOP PLOTTING OPERATIONS
     17                                 ;
     18                                 ;       $A      TURNS DISPLAY SCREEN ON
     19                                 ;
     20                                 ;       $B      TURNS DISPLAY SCREEN OFF
     21                                 ;
     22                                 ;       $I      INHIBITS PRINTING OF INPUT DATA FROM $R-$
S
     23                                 ;
     24                                 ;       $J      ENABLES PRINTING OF INPUT DATA (WHEN USIN
G TERMINAL)
     25                                 ;
```

INPUT DATA FORMAT

```
      1                                       .SBTTL  INPUT DATA FORMAT
      2                                       ;
      3                                       ;         THE NUMERICAL DATA REQUIRED BY THE PLOTTING
      4                                       ;       COMMANDS MAY BE SPECIFIED IN ONE OF TWO FORMATS:
      5                                       ;
      6                                       ;       1.   IMMEDIATELY FOLLOWING A COMMAND, IF THE
      7                                       ;       FIRST CHARACTER IS A  -  OR A  DIGIT
      8                                       ;       DATA IS ASSUMED TO FOLLOW IN THE SPECIFIED INTEGE
R
      9                                       ;       FORMAT UNTIL A <CR>,<LF> OR OTHER CONTROL CHARACT
ER
     10                                       ;       IS ENCOUNTERED. (THIS MODE DOES NOT APPLY TO $R,
$F, OR $C)
     11                                       ;            IN FREE FORMAT DATA MAY FOLLOW WITHOUT
     12                                       ;       THE ABOVE RESTRICTIONS
     13                                       ;
     14                                       ;       2.   IF THE FIRST CHARACTER FOLLOWING A COMMAND I
S NOT
     15                                       ;       ONE OF THE ABOVE, THE GRAPHICS CONTROL ENTERS
     16                                       ;       THE SEARCH MODE. THIS MODE SEARCHES FOR THE
     17                                       ;       CHARACTER + . ALL CHARACTERS FOLLOWING THE DELIMI
TER +
     18                                       ;       ARE ASSUMED TO BE IN THE SPECIFIED INTEGER FORMAT
.
     19                                       ;       THE TERMINATION OF THE DATA BY A <CR>,<LF> OR
     20                                       ;       OTHER CONTROL CHARACTER CAUSES THE CONTROLLER TO
     21                                       ;       AGAIN ENTER THE SEARCH MODE.
     22                                       ;          A NEW PLOTTER COMMAND IS ALWAYS RECOGNIZED
     23                                       ;       WITH THE CONTROLLER IN SEARCH MODE.
     24                                       ;
```

GLOBLE ENTRY POINT

```
      2                                      .SBTTL  GLOBLE ENTRY POINT
      3                                      ;
      4         140000                       .=140000                ;LOCATION OF GRAPHICS PROGRAMS
      5                                      ;
      6 140000                       GPINIT: JMP     GPSET           ;INITIALIZE
        140000     176     300     006
      7 140003                       GRPLOT: JMP     LSTENT          ;PROCESS DATA
        140003     176     305     331
      8 140006                       GPSET:  LDX     #,0             ;CLEAR GOSUB POINTER
        140006     316     000     000
      9 140011                               STX     GOSUB
        140011     377     002     370
     10 140014                               LDX     #,HEREIS        ;SET UP 'HERE IS' BUTTON
        140014     316     300     104
     11 140017                               STX     IRQ13
        140017     377     017     370
     12 140022                               LDA A   #,7             ;ENABLE INTERRUPT
        140022     206     007
     13 140024                               STA A   PCRA12
        140024     227     056
     14 140026                               LDA A   NPRDTA          ;CLEAR PENDING INTERRUPT
        140026     226     054
     15                                      ;
     16 140030                               LDX     #,0             ;CLEAR CONTROL REGISTERS
        140030     316     000     000
     17 140033                               STX     PCRA5           ;GRAPHIC 'IN' PIA
        140033     337     022
     18 140035                               STX     WDFBA           ;DIRECTION - IN
        140035     337     020
     19 140037                               STX     PCRA6           ;GRAPHIC 'OUT' PIA
        140037     337     026
     20 140041                               LDX     #,377,377       ;DIRECTION - OUT
        140041     316     377     377
     21 140044                               STX     WDTBA
        140044     337     024
```

```
        140046     316     300     115
    23  140051                             STX     IRQ8
        140051     377     017     356
    24  140054                             LDX     #,GRPOUT        ;SETUP OUT INTERRUPT
        140054     316     300     115
    25  140057                             STX     IRQ9
        140057     377     017     360
    26  140062                             LDX     #,55,4          ;ENABLE IN INTERRUPT
        140062     316     055     004
    27  140065                             STX     PCRA5
        140065     337     022
    28  140067                             LDX     #,5,4           ;DISABLE OUT INTERRUPT
        140067     316     005     004
    29  140072                             STX     PCRA6           ;REENABLE CLOCK
        140072     337     026
    30  140074                             LDA A   WDFBA           ;CLEAR PENDING INTERRUPT
        140074     226     020
    31  140076                             LDA A   WDTBB           ;CLEAR PENDING INTERRUPT
        140076     226     025
    32  140100                             CLR     STPFLG          ;DISABLE STRIPPING
        140100     177     002     375
```

GLOBLE ENTRY POINT

```
     33 140103                                RTS                     ;RETURN TO MAIN LINE
        140103     071
     34                                       ;
```

HERE IS INTERRUPT HANDLER

```
      1                                    .SBTTL  HERE IS INTERRUPT HANDLER
      2                                    ;
      3 140104                       HEREIS: COM     GRAPH           ;FLIP-FLOP GRAPHIC
        140104     163     000     100
      4 140107                               INC     UPDFLG          ;DISPLAY UPDATE
        140107     174     000     106
      5 140112                               LDA A   NPRDTA          ;CLEAR INTERRUPT FLAG
        140112     226     054
      6 140114                               RTI                     ;FINISHED
        140114     073
      7                                    ;
```

GRAPHICS DISPATCHER

```
      1                                     .SBTTL  GRAPHICS DISPATCHER
      2                                     ;
      3 140115                        GRPIN:
      4 140115                        GRPOUT: LDA A   #,4             ;DISABLE FURTHER INTERRUPTS
        140115     206     004
      5 140117                                STA A   PCRA5
        140117     227     022
      6 140121                                CLI                     ;ENABLE ALL OTHERS
        140121     016
      7 140122                                BSR     1$              ;PLACE RETURN POINT ON STACK
        140122     215     010
      8 140124                                SEI                     ;INHIBIT INTERRUPTS
        140124     017
      9 140125                                LDA A   #,55            ;ENABLE GRAPHICS INTERRUPT
        140125     206     055
     10 140127                                STA A   PCRA5
        140127     227     022
     11 140131                                LDA A   WDFBA           ;SET READY FLAG
        140131     226     020
     12 140133                                RTI                     ;RETURN FROM INTERRUPT
        140133     073
     13 140134                        1$:     LDA A   WDFBB           ;GET CODED INSTRUCTION
        140134     226     021
     14 140136                                LDA B   WDFBA
        140136     326     020
     15 140140                                STA A   WDTBB           ;STORE AT OUTPUT PORT
        140140     227     025
     16 140142                                STA B   WDTBA
        140142     327     024
     17 140144                                STA A   OPR+1           ;SAVE FOR OPERATIONS
        140144     267     002     301
     18 140147                                STA B   OPR
        140147     367     002     300
     19 140152                                AND B   #,340           ;XY FUNCTION ?
        140152     304     340
```

```
        140154     053     003
     21 140156                              JMP     XYFNCT
        140156     176     302     212
     22 140161                          2$: CMP B   #,200           ;CHARACTER FUNCTION ?
        140161     301     200
     23 140163                              BNE     3$              ;IF NOT - SKIP
        140163     046     003
     24 140165                              JMP     CWRITE
        140165     176     303     337
     25 140170                          3$: CMP B   #,240           ;SELECT FUNCTION ?
        140170     301     240
     26 140172                              BNE     4$              ;IF NOT - SKIP
        140172     046     003
     27 140174                              JMP     SELECT
        140174     176     305     071
     28 140177                          4$: CMP B   #,300           ;CONTROL FUNCTION ?
        140177     301     300
     29 140201                              BNE     5$              ;IF NOT - SKIP
        140201     046     003
     30 140203                              JMP     CNTROL
        140203     176     305     171
```

GRAPHICS DISPATCHER

```
     31 140206                          5$:     JMP     READ            ;READ/WRITE FUNCTION
        140206     176     305     262
     32                                         ;
```

X-COORDINATE CALCULATION

```
                                        .SBTTL  X-COORDINATE CALCULATION
                                        ;
                                        ;       ENTER WITH [X] = TABLE ADDRESS
                                        ;
140211                          NCXVAL: CLR     CHK+1           ;XPLOT OK
140211  177     002     303
140214                                  LDA B   2,X             ;CHECK HIGH ORDER OF XVALUE
140214  346     002
140216                                  BMI     1$              ;IF <0 - DON'T PLOT
140216  053     010
140220                                  BSR     BXADC           ;COMPUTE BIT POSITION
140220  215     012
140222                                  BNE     1$              ;IF /=0 - DON'T PLOT
140222  046     004
140224                                  SUB A   #,80.           ;WITHIN LINE ?
140224  200     120
140226                                  BCS     2$              ;IF SO - PLOT
140226  045     003
140230                          1$:     COM     CHK+1           ;ELSE INHIBIT PLOT
140230  163     002     303
140233                          2$:     RTS                     ;FINISHED
140233  071
                                        ;
140234                          BXADC:  LDA A   3,X             ;GET LOW BYTE OF XV
140234  246     003
140236                                  CLR B
140236  137
140237                                  AND A   #,7             ;SAVE LOW THREE BITS
140237  204     007
140241                                  ADD A   #,BPTBLE&377    ;COMPUTE LOCATION IN TABLE
140241  213     305
140243                                  ADC B   #,BPTBLE&177400/400
140243  311     300
140245                                  STA A   15,X            ;SAVE IN TEMPORARY
140245  247     015
```

```
        140247     347     014
    22  140251                             STX     SAVEX           ;SAVE POINTER
        140251     377     002     304
    23  140254                             LDX     14,X            ;GET TABLE ADRDRESS
        140254     356     014
    24  140256                             LDA A   0,X             ;GET BIT PATTERN
        140256     246     000
    25  140260                             LDX     SAVEX           ;GET POINTER BACK
        140260     376     002     304
    26  140263                             STA A   10,X            ;SAVE BIT PATTERN
        140263     247     010
    27  140265                             LDA A   3,X             ;GET XV TO COMPUTE BYTE ADDRESS
        140265     246     003
    28  140267                             LDA B   2,X
        140267     346     002
    29  140271                             CLC
        140271     014
    30  140272                             ROR B                   ;DIVIDE BY 8
        140272     126
    31  140273                             ROR A
        140273     106
```

X-COORDINATE CALCULATION

```
     32 140274                                ASR B
        140274     127
     33 140275                                ROR A
        140275     106
     34 140276                                ASR B
        140276     127
     35 140277                                ROR A
        140277     106
     36 140300                                STA A   7,X             ;SAVE RESULT IN XVAL
        140300     247     007
     37 140302                                STA B   6,X
        140302     347     006
     38 140304                                RTS                     ;FINISHED
        140304     071
     39                                       ;
     40 140305     200     100     040  BPTBLE: .BYTE   200,100,40,20,10,4,2,1
        140310     020     010     004
        140313     002     001
     41                                       ;
```

Y-COORDINATE CALCULATION

```
      1                                      .SBTTL  Y-COORDINATE CALCULATION
      2                                      ;
      3                                      ;       ENTER WITH [X] = TABLE ADDRESS
      4                                      ;
      5 140315                         NCYVAL: CLR     CHK             ;YPLOT OK
        140315     177     002     302
      6 140320                                 LDA B   0,X             ;MUST = 0
        140320     346     000
      7 140322                                 BEQ     1$              ;IF SO - SKIP
        140322     047     004
      8 140324                                 COM     CHK             ;ELSE INHIBIT PLOT
        140324     163     002     302
      9 140327                                 RTS                     ;FINISHED
        140327     071
     10 140330                         1$:     LDA B   1,X             ;GET LOW ORDER VALUE
        140330     346     001
     11 140332                                 CLC
        140332     014
     12 140333                                 ROR B                   ;DIVIDE BY 2
        140333     126
     13 140334                                 LDA A   #,177           ;COMPUTE <B,A>=127.-YV/2
        140334     206     177
     14 140336                                 SBA
        140336     020
     15 140337                                 CLR B
        140337     137
     16 140340                                 ASL A                   ;COMPUTE 16*<B,A>
        140340     110
     17 140341                                 ROL B
        140341     131
     18 140342                                 ASL A
        140342     110
     19 140343                                 ROL B
        140343     131
     20 140344                                 ASL A
```

```
    21 140345                           ROL B
       140345     131
    22 140346                           ASL A
       140346     110
    23 140347                           ROL B
       140347     131
    24 140350                           STA A   5,X             ;SAVE TEMPORARILY
       140350     247     005
    25 140352                           STA B   4,X
       140352     347     004
    26 140354                           ASL A                   ;COMPUTE 64*<B,A>
       140354     110
    27 140355                           ROL B
       140355     131
    28 140356                           ASL A
       140356     110
    29 140357                           ROL B
       140357     131
    30 140360                           ADD A   5,X             ;COMPUTE 80*<B,A>
       140360     253     005
    31 140362                           ADC B   4,X
```

Y-COORDINATE CALCULATION

```
                140362  351  004
     32 140364                          ROR     1,X             ;CHECK EVEN/ODD SCAN
                140364  146  001
     33 140366                          BCC     2$              ;IF EVEN - SKIP
                140366  044  010
     34 140370                          ROL     1,X             ;RESTORE VALUE
                140370  151  001
     35 140372                          ADD A   #,ODDSCN&377    ;COMPUTE ABSOLUTE ADDRESS
                140372  213  000
     36 140374                          ADC B   #,ODDSCN&177400/400
                140374  311  130
     37 140376                          BRA     3$
                140376  040  006
     38 140400                    2$:   ROL     1,X             ;RESTORE VALUE
                140400  151  001
     39 140402                          ADD A   #,EVNSCN&377    ;COMPUTE ABSOLUTE ADDRESS
                140402  213  000
     40 140404                          ADC B   #,EVNSCN&177400/400
                140404  311  060
     41 140406                    3$:   STA A   5,X             ;SAVE VALUE IN YVAL
                140406  247  005
     42 140410                          STA B   4,X
                140410  347  004
     43 140412                          RTS                     ;FINISHED
                140412  071
     44                                 ;
```

PLOTXY POINT ROUTINE

```
      1                                     .SBTTL  PLOTXY POINT ROUTINE
      2                                     ;
      3 140413                      PLOTXY: LDX     YTABLE+2        ;NEW Y VALUE
        140413     376     002     202
      4 140416                              STX     PTABLE
        140416     377     002     260
      5 140421                              LDX     XTABLE+2        ;NEW X VALUE
        140421     376     002     232
      6 140424                              STX     PTABLE+2
        140424     377     002     262
      7 140427                      VECTXY: LDX     #,PTABLE        ;TABLE POINTER
        140427     316     002     260
      8 140432                              JSR     NCXVAL          ;COMPUTE XVAL AND BIT POSITION
        140432     275     300     211
      9 140435                              JSR     NCYVAL          ;COMPUTE YVAL
        140435     275     300     315
     10 140440                      NXYPLT: LDX     CHK             ;OK TO PLOT ?
        140440     376     002     302
     11 140443                              BNE     1$              ;IF NOT - SKIP
        140443     046     005
     12 140445                              LDX     #,PTABLE        ;TABLE POINTER
        140445     316     002     260
     13 140450                              BSR     XYBIS           ;DO A POINT
        140450     215     001
     14 140452                      1$:     RTS                     ;FINISHED
        140452     071
     15                                     ;
```

XY BIT SET ROUTINE

```
      1                                    .SBTTL  XY BIT SET ROUTINE
      2                                    ;
      3 140453                     XYBIS:  LDA A   5,X             ;GET YVAL
        140453     246     005
      4 140455                             LDA B   4,X
        140455     346     004
      5 140457                             ADD A   7,X             ;ADD XVAL
        140457     253     007
      6 140461                             ADC B   6,X
        140461     351     006
      7 140463                             STA A   13,X            ;SAVE XYVAL
        140463     247     013
      8 140465                             STA B   12,X
        140465     347     012
      9 140467                             LDA A   10,X            ;GET BIT POSITION
        140467     246     010
     10 140471                             LDA B   11,X            ;GET CONTROL
        140471     346     011
     11 140473                             ROR B                   ;CHECK ENABLE BIT
        140473     126
     12 140474                             BCC     3$              ;IF =0 NO PLOTTING
        140474     044     016
     13 140476                             LDX     12,X            ;GET XYVAL AS ADDRESS
        140476     356     012
     14 140500                             ROL B                   ;CHECK FOR ADD/DELETE
        140500     131
     15 140501                             BMI     1$              ;IF DELETE - SKIP
        140501     053     004
     16 140503                             ORA A   0,X             ;ADD BY ORING
        140503     252     000
     17 140505                             BRA     2$
        140505     040     003
     18 140507                     1$:     COM A                   ;BIT TO CLEAR IS '0'
        140507     103
     19 140510                             AND A   0,X             ;DELETE BY ANDING
```

```
    20 140512                          2$:     STA A   0,X             ;MOVE INTO DISPLAY AREA
       140512     247     000
    21 140514                          3$:     RTS                     ;FINISHED
       140514     071
    22                                         ;
```

AUTO-INCREMENT ROUTINE

```
      1                                    .SBTTL  AUTO-INCREMENT ROUTINE
      2                                    ;
      3                                    ;       ENTER WITH [X] = TABLE ADDRESS
      4                                    ;
      5 140515                       AI:   LDA A   3,X             ;P_ = N_
        140515     246     003
      6 140517                             LDA B   2,X
        140517     346     002
      7 140521                             STA A   1,X
        140521     247     001
      8 140523                             STA B   0,X
        140523     347     000
      9 140525                             ADD A   5,X             ;N_ = N_ + DEL
        140525     253     005
     10 140527                             ADC B   4,X
        140527     351     004
     11 140531                             STA A   3,X
        140531     247     003
     12 140533                             STA B   2,X
        140533     347     002
     13 140535                             RTS                     ;FINISHED
        140535     071
     14                                    ;
```

LOAD NEW VALUE ROUTINE

```
      1                                      .SBTTL  LOAD NEW VALUE ROUTINE
      2                                      ;
      3                                      ;       ENTER WITH [X] = TABLE ADDRESS
      4                                      ;
      5 140536                         LOADV:  LDA B   OPR             ;CHECK OPERATION CODE
        140536     366     002     300
      6 140541                                 BIT B   #,40            ;DEL MODE ?
        140541     305     040
      7 140543                                 BNE     1$              ;IF SO - SKIP
        140543     046     022
      8 140545                                 LDA A   3,X             ;P_ = N_
        140545     246     003
      9 140547                                 STA A   1,X
        140547     247     001
     10 140551                                 LDA A   2,X
        140551     246     002
     11 140553                                 STA A   0,X
        140553     247     000
     12 140555                                 LDA A   OPR+1           ;GET HIGH ORDER
        140555     266     002     301
     13 140560                                 AND B   #,3             ;MASK VALUE
        140560     304     003
     14 140562                                 STA A   3,X             ;SAVE NEW VALUE
        140562     247     003
     15 140564                                 STA B   2,X
        140564     347     002
     16 140566                                 RTS                     ;FINISHED
        140566     071
     17                                        ;
     18 140567                         1$:     LDA A   OPR+1           ;GET DEL VALUE
        140567     266     002     301
     19 140572                                 AND B   #,3
        140572     304     003
     20 140574                                 BIT B   #,2             ;A NEGATIVE DEL ?
        140574     305     002
```

```
   140576     047     002
22 140600                          ORA B   #,374           ;SIGN EXTEND VALUE
   140600     312     374
23 140602                    2$:   STA A   5,X             ;NEW DEL VALUE
   140602     247     005
24 140604                          STA B   4,X
   140604     347     004
25 140606                          RTS                     ;FINISHED
   140606     071
26                                 ;
```

ABSOLUTE VECTOR LENGTH CALCULATION

```
                                        .SBTTL  ABSOLUTE VECTOR LENGTH CALCULATION
                                        ;
                                        ;       ENTER WITH [X] = TABLE ADDDRESS
                                        ;
140607                          ABSD:   CLR     20,X            ;SET SIGN (+)
  140607    157     020
140611                                  LDA A   3,X             ;COMPUTE N_ - P_
  140611    246     003
140613                                  LDA B   2,X
  140613    346     002
140615                                  SUB A   1,X
  140615    240     001
140617                                  SBC B   0,X
  140617    342     000
140621                                  BPL     1$              ;IF (+) - SKIP
  140621    052     010
140623                                  COM     20,X            ;ELSE SIGN IS (-)
  140623    143     020
140625                                  COM A                   ;COMPUTE TWO'S COMPLEMENT
  140625    103
140626                                  COM B
  140626    123
140627                                  ADD A   #,1
  140627    213     001
140631                                  ADC B   #,0
  140631    311     000
140633                          1$:     STA A   7,X             ;SAVE ABS(VECT)
  140633    247     007
140635                                  STA B   6,X
  140635    347     006
140637                                  RTS                     ;FINISHED
  140637    071
                                        ;
```

VECTOR SET CONSTANT ROUTINE

```
                                        .SBTTL  VECTOR SET CONSTANT ROUTINE
                                        ;
                                        ;       ENTER WITH [X] = TABLE ADDRESS
                                        ;
140640                          VSETC:  CLR B
140640  137
140641                                  STA B   14,X            ;SET STEP VALUE SIZE
140641  347     014
140643                                  STA A   15,X
140643  247     015
140645                                  STA B   16,X
140645  347     016
140647                                  STA B   17,X
140647  347     017
140651                                  STA B   13,X            ;SET FRACTION = 1/2
140651  347     013
140653                                  LDA B   #,200
140653  306     200
140655                                  STA B   12,X
140655  347     012
140657                                  LDA A   1,X             ;MOV START VALUE TO CP_
140657  246     001
140661                                  LDA B   0,X
140661  346     000
140663                                  STA A   11,X
140663  247     011
140665                                  STA B   10,X
140665  347     010
140667                                  RTS                     ;FINISHED
140667  071
                                        ;
```

SET SIGN OF INCREMENT VALUE ROUTINE

```
      1                                    .SBTTL  SET SIGN OF INCREMENT VALUE ROUTINE
      2                                    ;
      3                                    ;       ENTER WITH [X] = TABLE ADDRESS
      4                                    ;
      5 140670                      SETSGN: TST     20,X            ;CHECK SIGN
        140670     155     020
      6 140672                              BEQ     1$              ;IF (+) - SKIP
        140672     047     026
      7 140674                              COM     17,X            ;ELSE COMPUTE 32 BIT 2'S COMPLEME
NT
        140674     143     017
      8 140676                              COM     16,X
        140676     143     016
      9 140700                              COM     15,X
        140700     143     015
     10 140702                              COM     14,X
        140702     143     014
     11 140704                              INC     17,X
        140704     154     017
     12 140706                              BNE     1$
        140706     046     012
     13 140710                              INC     16,X
        140710     154     016
     14 140712                              BNE     1$
        140712     046     006
     15 140714                              INC     15,X
        140714     154     015
     16 140716                              BNE     1$
        140716     046     002
     17 140720                              INC     14,X
        140720     154     014
     18 140722                      1$:     RTS                     ;FINISHED
        140722     071
     19                                     ;
```

UPDATE VALUE ROUTINE

```
      1                                    .SBTTL  UPDATE VALUE ROUTINE
      2                                    ;
      3                                    ;       ENTER WITH [X] = TABLE ADDRESS
      4                                    ;
      5 140723                       UPDATE: LDA A   13,X            ;32 BIT ADDITION
        140723     246     013
      6 140725                               LDA B   12,X
        140725     346     012
      7 140727                               ADD A   17,X
        140727     253     017
      8 140731                               ADC B   16,X
        140731     351     016
      9 140733                               STA A   13,X
        140733     247     013
     10 140735                               STA B   12,X
        140735     347     012
     11 140737                               LDA A   11,X
        140737     246     011
     12 140741                               LDA B   10,X
        140741     346     010
     13 140743                               ADC A   15,X
        140743     251     015
     14 140745                               ADC B   14,X
        140745     351     014
     15 140747                               STA A   11,X
        140747     247     011
     16 140751                               STA B   10,X
        140751     347     010
     17 140753                               LDX     10,X            ;GET RESULT
        140753     356     010
     18 140755                               RTS                     ;FINISHED
        140755     071
     19                                     ;
```

CLEAR GRAPHIC MEMORY ROUTINE

```
      1                                    .SBTTL  CLEAR GRAPHIC MEMORY ROUTINE
      2                                    ;
      3 140756                      GRPCLR: LDA A   #,GPBOT&377     ;BASE ADDRESS
        140756     206     000
      4 140760                              LDA B   #,GPBOT&177400/400
        140760     306     060
      5 140762                      1$:     STA B   GTEMPA          ;SAVE
        140762     367     002     306
      6 140765                      2$:     STA A   GTEMPA+1
        140765     267     002     307
      7 140770                              LDX     GTEMPA          ;GET INDEX ADDRESS
        140770     376     002     306
      8 140773                              CLR     0,X             ;CLEAR 8 BYTES QUICKLY
        140773     157     000
      9 140775                              CLR     1,X
        140775     157     001
     10 140777                              CLR     2,X
        140777     157     002
     11 141001                              CLR     3,X
        141001     157     003
     12 141003                              CLR     4,X
        141003     157     004
     13 141005                              CLR     5,X
        141005     157     005
     14 141007                              CLR     6,X
        141007     157     006
     15 141011                              CLR     7,X
        141011     157     007
     16 141013                              ADD A   #,10            ;UPDATE ADDRESS
        141013     213     010
     17 141015                              BCC     2$              ;LOOP
        141015     044     346
     18 141017                              ADC B   #,0
        141017     311     000
     19 141021                              BPL     1$              ;END OF GRAPHIC IS 100000 (-)
```

```
    20 141023                           RTS                 ;FINISHED
       141023     071
    21                                  ;
```

PLOT VECTOR HANDLER

```
      1                                        .SBTTL  PLOT VECTOR HANDLER
      2                                        ;
      3 141024                          PLOTVC: LDX     XTABLE+10       ;LOAD RUNNING VALUES
        141024     376     002     240
      4 141027                                  STX     PTABLE+2
        141027     377     002     262
      5 141032                                  LDX     YTABLE+10
        141032     376     002     210
      6 141035                                  STX     PTABLE
        141035     377     002     260
      7 141040                                  JSR     VECTXY          ;PLOT FIRST POINT
        141040     275     301     027
      8 141043                          1$:     LDX     VCNTR           ;GET COUNT
        141043     376     002     312
      9 141046                                  BEQ     4$              ;IF 0 - FINISHED
        141046     047     061
     10 141050                                  DEX                     ;UPDATE COUNT
        141050     011
     11 141051                                  STX     VCNTR
        141051     377     002     312
     12 141054                                  LDX     #,XTABLE        ;DO X UPDATES
        141054     316     002     230
     13 141057                                  JSR     UPDATE
        141057     275     301     323
     14 141062                                  CPX     PTABLE+2        ;HAS VALUE CHANGED ?
        141062     274     002     262
     15 141065                                  BEQ     2$              ;IF NOT - SKIP CALCULATION
        141065     047     011
     16 141067                                  STX     PTABLE+2        ;ELSE SAVE NEW VALUE
        141067     377     002     262
     17 141072                                  LDX     #,PTABLE        ;COMPUTE NEW XVAL
        141072     316     002     260
     18 141075                                  JSR     NCXVAL
        141075     275     300     211
     19 141100                          2$:     LDX     #,YTABLE        ;DO Y UPDATES
```

```
    20 141103                              JSR     UPDATE
       141103     275     301     323
    21 141106                              CPX     PTABLE          ;HAS VALUE CHANGED ?
       141106     274     002     260
    22 141111                              BEQ     3$              ;IF NOT - SKIP CALCULATION
       141111     047     011
    23 141113                              STX     PTABLE          ;ELSE SAVE NEW VALUE
       141113     377     002     260
    24 141116                              LDX     #,PTABLE        ;COMPUTE YVAL
       141116     316     002     260
    25 141121                              JSR     NCYVAL
       141121     275     300     315
    26 141124                         3$:  JSR     NXYPLT          ;PLOT POINT
       141124     275     301     040
    27 141127                              BRA     1$              ;LOOP AGAIN
       141127     040     312
    28 141131                         4$:  RTS                     ;FINISHED
       141131     071
    29                                     ;
```

DIVISION ROUTINE

```
      1                                    .SBTTL  DIVISION ROUTINE
      2                                    ;
      3 141132                      DIVIDE: LDA A   #,16.           ;16 BIT DIVIDE
        141132     206     020
      4 141134                              STA A   DIVCNT          ;SAVE COUNTER
        141134     267     002     314
      5 141137                              LDA A   DND+1           ;GET DIVIDEND
        141137     266     002     316
      6 141142                              LDA B   DND
        141142     366     002     315
      7 141145                              CLC                     ;DO INITIAL SHIFT
        141145     014
      8 141146                              ROL A
        141146     111
      9 141147                              ROL B
        141147     131
     10 141150                           1$:     SUB A   DSR+1          ;DO INITIAL DIVISION
        141150     260     002     320
     11 141153                              SBC B   DSR
        141153     362     002     317
     12 141156                              BCC     2$              ;IF DND>DSR - BRANCH
        141156     044     006
     13 141160                              ADD A   DSR+1           ;ELSE RESTORE
        141160     273     002     320
     14 141163                              ADC B   DSR
        141163     371     002     317
     15 141166                           2$:     ROL     QT+1            ;SHIFT IN COMPUTED BIT
        141166     171     002     322
     16 141171                              ROL     QT
        141171     171     002     321
     17 141174                              ASL A                   ;SHIFT DIVIDEND
        141174     110
     18 141175                              ROL B
        141175     131
     19 141176                              DEC     DIVCNT          ;16 BITS YET ?
```

```
    20 141201                          BNE     1$              ;LOOP UNTIL FINISHED
       141201     046     345
    21 141203                          COM     QT+1            ;GET TRUE RESULT
       141203     163     002     322
    22 141206                          COM     QT
       141206     163     002     321
    23 141211                          RTS                     ;FINISHED
       141211     071
    24                                 ;
```

XY FUNCTION SCAN ROUTINE

```
      1                                      .SBTTL  XY FUNCTION SCAN ROUTINE
      2                                      ;
      3 141212                       XYFNCT: LDA A   GCSR            ;USE CURRENT STATUS INFO
        141212     266     002     323
      4 141215                               STA A   PTABLE+11       ;SAVE IN PTABLE
        141215     267     002     271
      5 141220                               LDA A   OPR             ;CHECK FOR X OPERATION
        141220     266     002     300
      6 141223                               BIT A   #,100
        141223     205     100
      7 141225                               BNE     2$              ;IF NOT - SKIP
        141225     046     022
      8 141227                               BIT A   #,20            ;AUTO-INCREMENT ?
        141227     205     020
      9 141231                               BEQ     1$              ;IF NOT - SKIP
        141231     047     006
     10 141233                               LDX     #,YTABLE        ;UPDATE VALUES
        141233     316     002     200
     11 141236                               JSR     AI
        141236     275     301     115
     12 141241                       1$:     LDX     #,XTABLE        ;LOAD VALUES
        141241     316     002     230
     13 141244                               JSR     LOADV
        141244     275     301     136
     14 141247                               BRA     4$
        141247     040     020
     15 141251                       2$:     BIT A   #,20            ;AUTO-INCREMENT ?
        141251     205     020
     16 141253                               BEQ     3$              ;IF NOT - SKIP
        141253     047     006
     17 141255                               LDX     #,XTABLE        ;UPDATE VALUES
        141255     316     002     230
     18 141260                               JSR     AI
        141260     275     301     115
     19 141263                       3$:     LDX     #,YTABLE        ;LOAD NEW VALUES
```

```
    20 141266                                  JSR     LOADV
       141266     275     301     136
    21                                         ;
    22 141271                          4$:     LDA A   OPR             ;GET OPERATION
       141271     266     002     300
    23 141274                                  BIT A   #,14            ;PL OR V SET ?
       141274     205     014
    24 141276                                  BEQ     6$              ;IF NOT - FINISHED
       141276     047     032
    25 141300                                  BIT A   #,40            ;RELATIVE MODE ?
       141300     205     040
    26 141302                                  BEQ     5$              ;IF NOT - SKIP
       141302     047     017
    27 141304                                  LDX     #,XTABLE        ;P_=N_ ; N_=N_+DEL
       141304     316     002     230
    28 141307                                  JSR     AI
       141307     275     301     115
    29 141312                                  LDX     #,YTABLE        ;P_=N_ ; N_=N_+DEL
       141312     316     002     200
    30 141315                                  JSR     AI
       141315     275     301     115
```

XY FUNCTION SCAN ROUTINE

```
 31 141320                                  LDA A   OPR             ;GET OPERATION
    141320     266     002     300
 32 141323                          5$:     BIT A   #,10            ;VECTOR ?
    141323     205     010
 33 141325                                  BNE     VECTOR          ;IF SO - SKIP
    141325     046     004
 34 141327                                  JSR     PLOTXY          ;ELSE PLOT POINT
    141327     275     301     013
 35 141332                          6$:     RTS                     ;FINISHED
    141332     071
 36                                         ;
```

VECTOR DRAWER

```
      1                                      .SBTTL  VECTOR DRAWER
      2                                      ;
      3 141333                        VECTOR: LDX     #,XTABLE        ;COMPUTE /X/
        141333     316     002     230
      4 141336                                JSR     ABSD
        141336     275     301     207
      5 141341                        CHEK1:  LDX     #,YTABLE        ;COMPUTE /Y/
        141341     316     002     200
      6 141344                                JSR     ABSD
        141344     275     301     207
      7 141347                                LDX     XTABLE+6        ;DX=DY ?
        141347     376     002     236
      8 141352                                CPX     YTABLE+6
        141352     274     002     206
      9 141355                                BNE     CHEK2           ;IF NOT - SKIP
        141355     046     060
     10 141357                                STX     VCNTR           ;COUNT
        141357     377     002     312
     11 141362                                LDX     #,XTABLE        ;UPDATE COUNT = 1
        141362     316     002     230
     12 141365                                LDA A   #,1
        141365     206     001
     13 141367                                JSR     VSETC
        141367     275     301     240
     14 141372                                JSR     SETSGN          ;SET SIGN
        141372     275     301     270
     15 141375                                LDX     #,YTABLE        ;UPDATE COUNT = 1
        141375     316     002     200
     16 141400                                LDA A   #,1
        141400     206     001
     17 141402                                JSR     VSETC
        141402     275     301     240
     18 141405                                JSR     SETSGN          ;SET SIGN
        141405     275     301     270
     19 141410                                JSR     PLOTVC          ;PLOT THE VECTOR
```

```
    20                                      ;
    21 141413                       VECTDN: LDA A   OPR             ;CHECK FOR RELATIIVE OPERATION
       141413     266     002     300
    22 141416                               BIT A   #,40
       141416     205     040
    23 141420                               BNE     1$              ;IF NOT - SKIP
       141420     046     014
    24 141422                               LDX     YTABLE+2
       141422     376     002     202
    25 141425                               STX     YTABLE
       141425     377     002     200
    26 141430                               LDX     XTABLE+2
       141430     376     002     232
    27 141433                               STX     XTABLE
       141433     377     002     230
    28 141436                       1$:     RTS                     ;FINISHED
       141436     071
    29                                      ;
    30 141437                       CHEK2:  LDX     XTABLE+6        ;DX=0 ?
       141437     376     002     236
    31 141442                               BNE     CHEK3           ;IF NOT - SKIP AHEAD
```

VECTOR DRAWER

```
        141442     046     036
     32 141444                              LDX     YTABLE+6        ;DY IS COUNT
        141444     376     002     206
     33 141447                              STX     VCNTR
        141447     377     002     312
     34 141452                              LDX     #,XTABLE        ;UPDATE VALUE = 0
        141452     316     002     230
     35 141455                              CLR A
        141455     117
     36 141456                              JSR     VSETC
        141456     275     301     240
     37 141461                              LDX     #,YTABLE        ;UPDATE OF 1
        141461     316     002     200
     38 141464                              LDA A   #,1
        141464     206     001
     39 141466                              JSR     VSETC
        141466     275     301     240
     40 141471                              JSR     SETSGN          ;SET SIGN
        141471     275     301     270
     41 141474                              JSR     PLOTVC          ;GO PLOT VECTOR
        141474     275     302     024
     42 141477                              JMP     VECTDN          ;FINISH UP
        141477     176     303     013
     43                                     ;
     44 141502                         CHEK3:  LDX     YTABLE+6        ;DY = 0 ?
        141502     376     002     206
     45 141505                              BNE     CHEK4           ;IF NOT - SKIP
        141505     046     036
     46 141507                              LDX     XTABLE+6        ;DX IS COUNT
        141507     376     002     236
     47 141512                              STX     VCNTR
        141512     377     002     312
     48 141515                              LDX     #,XTABLE        ;UPDATE = 1
        141515     316     002     230
     49 141520                              LDA A   #,1
```

```
 50 141522                              JSR     VSETC
    141522     275     301     240
 51 141525                              JSR     SETSGN          ;SET SIGN
    141525     275     301     270
 52 141530                              LDX     #,YTABLE        ;UPDATE = 0
    141530     316     002     200
 53 141533                              CLR A
    141533     117
 54 141534                              JSR     VSETC
    141534     275     301     240
 55 141537                              JSR     PLOTVC          ;PLOT VECTOR
    141537     275     302     024
 56 141542                              JMP     VECTDN          ;FINISH UP
    141542     176     303     013
 57                                     ;
 58 141545                      CHEK4:  LDA A   XTABLE+7        ;WHICH IS LARGER ?
    141545     266     002     237
 59 141550                              LDA B   XTABLE+6
    141550     366     002     236
 60 141553                              SUB A   YTABLE+7
    141553     260     002     207
```

```
     61 141556                              SBC B   YTABLE+6
        141556     362     002     206
     62 141561                              BCS     CHEK5           ;IF DY>DX BRANCH
        141561     045     066
     63 141563                              LDX     XTABLE+6        ;DX IS COUNT
        141563     376     002     236
     64 141566                              STX     VCNTR
        141566     377     002     312
     65 141571                              STX     DSR             ;DO DIVISION
        141571     377     002     317
     66 141574                              LDX     YTABLE+6
        141574     376     002     206
     67 141577                              STX     DND
        141577     377     002     315
     68 141602                              JSR     DIVIDE
        141602     275     302     132
     69 141605                              LDX     #,XTABLE        ;UPDATE = 1
        141605     316     002     230
     70 141610                              LDA A   #,1
        141610     206     001
     71 141612                              JSR     VSETC
        141612     275     301     240
     72 141615                              JSR     SETSGN          ;SET SIGN
        141615     275     301     270
     73 141620                              LDX     #,YTABLE        ;UPDATE = 0
        141620     316     002     200
     74 141623                              CLR A
        141623     117
     75 141624                              JSR     VSETC
        141624     275     301     240
     76 141627                              LDX     QT              ;SET FRACTIONAL PART
        141627     376     002     321
     77 141632                              STX     YTABLE+16
        141632     377     002     216
     78 141635                              LDX     #,YTABLE
```

```
 79 141640                                JSR     SETSGN          ;SET SIGN
    141640     275     301     270
 80 141643                                JSR     PLOTVC          ;PLOT VECTOR
    141643     275     302     024
 81 141646                                JMP     VECTDN          ;FINISH UP
    141646     176     303     013
 82                                       ;
 83 141651                          CHEK5: LDX     YTABLE+6        ;DY IS COUNT
    141651     376     002     206
 84 141654                                STX     VCNTR
    141654     377     002     312
 85 141657                                STX     DSR             ;DO DIVISION
    141657     377     002     317
 86 141662                                LDX     XTABLE+6
    141662     376     002     236
 87 141665                                STX     DND
    141665     377     002     315
 88 141670                                JSR     DIVIDE
    141670     275     302     132
 89 141673                                LDX     #,XTABLE        ;UPDATE = 0
    141673     316     002     230
```

```
     90 141676                              CLR A
        141676     117
     91 141677                              JSR     VSETC
        141677     275     301     240
     92 141702                              LDX     QT              ;FRACTIONAL VALUE
        141702     376     002     321
     93 141705                              STX     XTABLE+16
        141705     377     002     246
     94 141710                              LDX     #,XTABLE
        141710     316     002     230
     95 141713                              JSR     SETSGN          ;SET SIGN
        141713     275     301     270
     96 141716                              LDX     #,YTABLE        ;UPDATE = 1
        141716     316     002     200
     97 141721                              LDA A   #,1
        141721     206     001
     98 141723                              JSR     VSETC
        141723     275     301     240
     99 141726                              JSR     SETSGN          ;SET SIGN
        141726     275     301     270
    100 141731                              JSR     PLOTVC          ;PLOT VECTOR
        141731     275     302     024
    101 141734                              JMP     VECTDN          ;FINISH UP
        141734     176     303     013
    102                                     ;
```

CHARACTER WRITING HANDLER

```
      1                                      .SBTTL  CHARACTER WRITING HANDLER
      2                                      ;
      3 141737                         CWRITE: LDA A   OPR             ;GET CODE
        141737     266     002     300
      4 141742                                 TAB
        141742     026
      5 141743                                 AND A   #,4             ;CHACK REVERSE BIT
        141743     204     004
      6 141745                                 BNE     5$              ;IF SO - SKIP
        141745     046     073
      7 141747                                 LDX     #,PTABLE+2      ;ROW ALONG X
        141747     316     002     262
      8 141752                                 STX     ROWADD
        141752     377     002     341
      9 141755                                 LDX     XTABLE+2        ;SAVE ROW POSITION
        141755     376     002     232
     10 141760                                 STX     ROWRST
        141760     377     002     343
     11 141763                                 LDX     #,PTABLE        ;COLUMN ALONG Y
        141763     316     002     260
     12 141766                                 STX     COLADD
        141766     377     002     347
     13 141771                                 LDX     YTABLE+2        ;SAVE RCOLUMN POSITION
        141771     376     002     202
     14 141774                                 STX     COLRST
        141774     377     002     351
     15 141777                                 TBA                     ;GET CODE
        141777     027
     16 142000                                 AND A   #,2             ;CHECK ROW DIRECTION
        142000     204     002
     17 142002                                 BNE     1$              ;IF REVERSED - SKIP
        142002     046     005
     18 142004                                 LDX     #,1             ;(+)
        142004     316     000     001
     19 142007                                 BRA     2$
```

```
 20 142011                          1$:     LDX     #,-1            ;(-)
    142011     316     377     377
 21 142014                          2$:     STX     ROWUPD          ;SAVE UPDATING VALUE
    142014     377     002     337
 22 142017                                  TBA                     ;GET CODE
    142017     027
 23 142020                                  AND A   #,1             ;COLUMN REVERSED ?
    142020     204     001
 24 142022                                  BNE     3$              ;IF SO - SKIP
    142022     046     005
 25 142024                                  LDX     #,1             ;(+)
    142024     316     000     001
 26 142027                                  BRA     4$
    142027     040     003
 27 142031                          3$:     LDX     #,-1            ;(-)
    142031     316     377     377
 28 142034                          4$:     STX     COLUPD          ;SAVE UPDATE VALUE
    142034     377     002     345
 29 142037                                  JMP     CXWRT           ;DO CHARACTER
    142037     176     304     132
 30                                         ;
```

CHARACTER WRITING HANDLER

```
     31 142042                          5$:     LDX     #,PTABLE+2      ;COLUMN ALONG X
        142042     316     002     262
     32 142045                                  STX     COLADD
        142045     377     002     347
     33 142050                                  LDX     XTABLE+2        ;COLUMN RESET VALUE
        142050     376     002     232
     34 142053                                  STX     COLRST
        142053     377     002     351
     35 142056                                  LDX     #,PTABLE        ;ROW ALONG Y
        142056     316     002     260
     36 142061                                  STX     ROWADD
        142061     377     002     341
     37 142064                                  LDX     YTABLE+2        ;ROW RESET VALUE
        142064     376     002     202
     38 142067                                  STX     ROWRST
        142067     377     002     343
     39 142072                                  TBA                     ;GET CODE
        142072     027
     40 142073                                  AND A   #,2             ;REVERSED DIRECTION ?
        142073     204     002
     41 142075                                  BNE     6$              ;IF SO - SKIP
        142075     046     005
     42 142077                                  LDX     #,1             ;(+)
        142077     316     000     001
     43 142102                                  BRA     7$
        142102     040     003
     44 142104                          6$:     LDX     #,-1            ;(-)
        142104     316     377     377
     45 142107                          7$:     STX     COLUPD          ;SAVE UPDATING VALUE
        142107     377     002     345
     46 142112                                  TBA                     ;GET CODE
        142112     027
     47 142113                                  AND A   #,1             ;REVERSED ROW ?
        142113     204     001
     48 142115                                  BNE     8$              ;IF SO - SKIP
```

```
    49 142117                              LDX     #,1             ;(+)
       142117     316     000     001
    50 142122                              BRA     9$
       142122     040     003
    51 142124                         8$:  LDX     #,-1            ;(-)
       142124     316     377     377
    52 142127                         9$:  STX     ROWUPD          ;SAVE UPDATE VALUE
       142127     377     002     337
    53                                                             ;FALL THROUGH TO CXWRT
    54                                     ;
```

CHARACTER WRITING HANDLER

```
      1                                      ;
      2 142132                         CXWRT:  LDA A   OPR+1           ;GET ASCII CODE
        142132     266     002     301
      3 142135                                 CLR B
        142135     137
      4 142136                                 ASL A                   ;16.*ASCII CODE
        142136     110
      5 142137                                 ROL B
        142137     131
      6 142140                                 ASL A
        142140     110
      7 142141                                 ROL B
        142141     131
      8 142142                                 ASL A
        142142     110
      9 142143                                 ROL B
        142143     131
     10 142144                                 ASL A
        142144     110
     11 142145                                 ROL B
        142145     131
     12 142146                                 ORA A   #,17            ;FILL IN REST IF ADDRESS
        142146     212     017
     13 142150                                 ORA B   #,CHRAM&177400/400
        142150     312     020
     14 142152                                 STA A   CADR+1          ;SAVE ADDRESS
        142152     267     002     336
     15 142155                                 STA B   CADR
        142155     367     002     335
     16 142160                                 LDA A   GCSR            ;SET CURRENT STATUS IN PTABLE
        142160     266     002     323
     17 142163                                 STA A   PTABLE+11
        142163     267     002     271
     18 142166                                 LDX     COLADD          ;SET UP COLUMN PARAMETER
        142166     376     002     347
```

```
        142171     366     002     351
     20 142174                             STA B   0,X
        142174     347     000
     21 142176                             LDA A   COLRST+1
        142176     266     002     352
     22 142201                             STA A   1,X
        142201     247     001
     23 142203                             LDA A   #,16.           ;16 ROWS
        142203     206     020
     24 142205                             STA A   LT
        142205     267     002     325
     25 142210                          1$:     LDX     CADR            ;GET CHARACTER ADDRESS
        142210     376     002     335
     26 142213                             LDA A   0,X             ;GET ROW OF DOTS
        142213     246     000
     27 142215                             STA A   CWORD           ;AND SAVE
        142215     267     002     326
     28 142220                             LDX     ROWADD          ;SET UP ROW PARAMETER
        142220     376     002     341
     29 142223                             LDA B   ROWRST
        142223     366     002     343
```

CHARACTER WRITING HANDLER

```
     30 142226                              STA B   0,X
        142226     347     000
     31 142230                              LDA A   ROWRST+1
        142230     266     002     344
     32 142233                              STA A   1,X
        142233     247     001
     33 142235                              LDA A   #,8.            ;8. BITS PER ROW
        142235     206     010
     34 142237                              STA A   BT
        142237     267     002     324
     35 142242                         2$:  ROL     CWORD           ;IS BIT SET ?
        142242     171     002     326
     36 142245                              BCC     3$              ;IF NOT - SKIP
        142245     044     003
     37 142247                              JSR     VECTXY          ;ELSE PLOT POINT
        142247     275     301     027
     38 142252                         3$:  LDX     ROWADD          ;UPDATE ROW POSITION
        142252     376     002     341
     39 142255                              LDA A   1,X
        142255     246     001
     40 142257                              ADD A   ROWUPD+1
        142257     273     002     340
     41 142262                              STA A   1,X
        142262     247     001
     42 142264                              LDA B   0,X
        142264     346     000
     43 142266                              ADC B   ROWUPD
        142266     371     002     337
     44 142271                              STA B   0,X
        142271     347     000
     45 142273                              DEC     BT              ;ONE LESS BIT
        142273     172     002     324
     46 142276                              BNE     2$              ;LOOP FOR ALL 8 BITS
        142276     046     342
     47 142300                              LDX     COLADD          ;UPDATE COLUMN POSITION
```

```
    48 142303                          LDA A   1,X
       142303     246     001
    49 142305                          ADD A   COLUPD+1
       142305     273     002     346
    50 142310                          STA A   1,X
       142310     247     001
    51 142312                          LDA B   0,X
       142312     346     000
    52 142314                          ADC B   COLUPD
       142314     371     002     345
    53 142317                          STA B   0,X
       142317     347     000
    54 142321                          LDX     CADR            ;UPDATE CHARACTER ROW ADDRESS
       142321     376     002     335
    55 142324                          DEX
       142324     011
    56 142325                          STX     CADR
       142325     377     002     335
    57 142330                          DEC     LT              ;MORE ROWS ?
       142330     172     002     325
    58 142333                          BNE     1$              ;LOOP FOR ALL LINES
```

CHARACTER WRITING HANDLER

```
   142333     046     253
59                                      ;
60 142335                               LDA A   OPR             ;UPDATE ALONG X AXIS ?
   142335     266     002     300
61 142340                               AND A   #,20
   142340     204     020
62 142342                               BEQ     4$              ;IF NOT - SKIP
   142342     047     014
63 142344                               LDX     XTABLE+2        ;PX=NX
   142344     376     002     232
64 142347                               STX     XTABLE
   142347     377     002     230
65 142352                               LDX     PTABLE+2        ;NX=NX+DELX
   142352     376     002     262
66 142355                               STX     XTABLE+2
   142355     377     002     232
67 142360                          4$:  LDA A   OPR             ;UPDATE ALONG Y AXIS ?
   142360     266     002     300
68 142363                               AND A   #,10
   142363     204     010
69 142365                               BEQ     5$              ;IF NOT - KIP
   142365     047     014
70 142367                               LDX     YTABLE+2        ;PY=NY
   142367     376     002     202
71 142372                               STX     YTABLE
   142372     377     002     200
72 142375                               LDX     PTABLE          ;NY=NY+DELY
   142375     376     002     260
73 142400                               STX     YTABLE+2
   142400     377     002     202
74 142403                          5$:  RTS                     ;FINISHED
   142403     071
75                                      ;
```

CHARACTER RAM SET UP

```
      1                                      .SBTTL  CHARACTER RAM SET UP
      2                                      ;
      3 142404                        RSTRAM: LDX     #,CHRAM         ;RAM AREA
        142404     316     020     000
      4 142407                                CLR     0,X             ;CLEAR FIRST 4
        142407     157     000
      5 142411                                CLR     1,X
        142411     157     001
      6 142413                                CLR     2,X
        142413     157     002
      7 142415                                CLR     3,X
        142415     157     003
      8 142417                                LDX     #,CHRAM+4       ;FILL ADDRESS
        142417     316     020     004
      9 142422                                STX     GTEMPB
        142422     377     002     310
     10 142425                                LDX     #,CHROM1        ;ROM AREA
        142425     316     200     000
     11 142430                                STX     GTEMPA
        142430     377     002     306
     12 142433                           1$:  LDX     GTEMPA          ;UPDATE ADDRESS
        142433     376     002     306
     13 142436                                LDA A   0,X             ;GET BYTE
        142436     246     000
     14 142440                                LDA B   1,X             ;NEXT BYTE
        142440     346     001
     15 142442                                INX
        142442     010
     16 142443                                INX
        142443     010
     17 142444                                STX     GTEMPA
        142444     377     002     306
     18 142447                                LDX     GTEMPB          ;UPDATE ADDRESS
        142447     376     002     310
     19 142452                                STA A   0,X             ;MOVE BYTE
```

```
 20 142454                              STA B   1,X             ;AND NEXT BYTE
    142454     347     001
 21 142456                              INX
    142456     010
 22 142457                              INX
    142457     010
 23 142460                              STX     GTEMPB
    142460     377     002     310
 24 142463                              CPX     #,CHRAM+10000   ;END OF RAM AREA ?
    142463     214     040     000
 25 142466                              BNE     1$              ;IF NOT - LOOP
    142466     046     343
 26 142470                              RTS                     ;FINISHED
    142470     071
 27                                     ;
```

SELECT ROUTINE

```
      1                                        .SBTTL  SELECT ROUTINE
      2                                        ;
      3 142471                          SELECT: LDA A   OPR             ;GET OPERATION
        142471     266     002     300
      4 142474                                  BIT A   #,20            ;REPLACING ?
        142474     205     020
      5 142476                                  BEQ     1$              ;IF SO - SKIP
        142476     047     036
      6 142500                                  AND A   #,17            ;CHECK FUNCTION BITS
        142500     204     017
      7 142502                                  BNE     2$              ;IF NOT WORD SELECT - SKIP
        142502     046     056
      8 142504                                  LDA A   OPR+1           ;ELSE SPECIFYING A WORD
        142504     266     002     301
      9 142507                                  STA A   WTBYTE          ;SAVE FOR WRITE FUNCTION
        142507     267     002     327
     10 142512                                  CLR B                   ;COMPUTE ADDRESS IN CHRAM
        142512     137
     11 142513                                  ASL A
        142513     110
     12 142514                                  ROL B
        142514     131
     13 142515                                  ASL A
        142515     110
     14 142516                                  ROL B
        142516     131
     15 142517                                  ASL A
        142517     110
     16 142520                                  ROL B
        142520     131
     17 142521                                  ASL A
        142521     110
     18 142522                                  ROL B
        142522     131
     19 142523                                  ADD A   #,CHRAM&377
```

```
 20 142525                              ADC B   #,CHRAM&177400/400
    142525     311     020
 21 142527                              STA A   RADD+1          ;SAVE CHAR ADDRESS
    142527     267     002     331
 22 142532                              STA B   RADD
    142532     367     002     330
 23 142535                              RTS                     ;FINISHED
    142535     071
 24                                     ;
 25 142536                         1$:  AND A   #,17            ;MASK LINE SELECT
    142536     204     017
 26 142540                              LDA B   RADD+1          ;GET LOW BYTE ADDRESS
    142540     366     002     331
 27 142543                              AND B   #,360           ;MASK LINE SELECT
    142543     304     360
 28 142545                              ABA                     ;MASK IN NEW SELECT
    142545     033
 29 142546                              STA A   RADD+1
    142546     267     002     331
 30 142551                              LDX     RADD            ;GET COMPLETE ADDRESS
    142551     376     002     330
```

SELECT ROUTINE

```
     31 142554                                LDA A   OPR+1           ;GET DATA
        142554     266     002     301
     32 142557                                STA A   0,X             ;STORE IN PLACE
        142557     247     000
     33 142561                                RTS                     ;FINISHED
        142561     071
     34                                       ;
     35 142562                          2$:   LDA A   OPR+1           ;THIS IS CHARACTER
        142562     266     002     301
     36 142565                                JSR     LSTGRP          ;PROCESS IT
        142565     275     305     346
     37 142570                                RTS                     ;FINISHED
        142570     071
     38                                       ;
```

CONTROL ROUTINE

```
      1                                        .SBTTL  CONTROL ROUTINE
      2                                        ;
      3 142571                          CNTROL: LDA A   OPR             ;GET OPERATION
        142571     266     002     300
      4 142574                                  BIT A   #,20            ;CLEAR SCREEN ?
        142574     205     020
      5 142576                                  BEQ     1$              ;IF NOT - SKIP
        142576     047     006
      6 142600                                  JSR     GRPCLR          ;ELSE CLEAR SCREEN
        142600     275     301     356
      7 142603                                  LDA A   OPR
        142603     266     002     300
      8 142606                          1$:     AND A   #,17            ;MASK HIGH BITS
        142606     204     017
      9 142610                                  ASL A
        142610     110
     10 142611                                  ASL A
        142611     110
     11 142612                                  ASL A
        142612     110
     12 142613                                  ASL A
        142613     110
     13 142614                                  STA A   HADD            ;SAVE BITS FOR R/W FUNCTION
        142614     267     002     332
     14 142617                                  LDA A   OPR+1           ;SAVE NEW CONTROL
        142617     266     002     301
     15 142622                                  STA A   GCSR
        142622     267     002     323
     16 142625                                  BIT A   #,20            ;RESTORE  CHARACTERS ?
        142625     205     020
     17 142627                                  BEQ     2$              ;IF NOT - SKIP
        142627     047     006
     18 142631                                  JSR     RSTRAM          ;RESTORE CHARACTER SET
        142631     275     305     004
     19 142634                                  LDA A   OPR+1           ;GET CONTROL
```

```
 20 142637                          2$:     BIT A   #,100          ;SCREEN SET OPERATION ?
    142637     205     100
 21 142641                                  BEQ     5$             ;IF NOT - FINISHED
    142641     047     016
 22 142643                                  BIT A   #,40           ;SCREEN ON ?
    142643     205     040
 23 142645                                  BEQ     3$             ;IF NOT - TURN  OFF
    142645     047     004
 24 142647                                  LDA A   #,377          ;SET FLAG
    142647     206     377
 25 142651                                  BRA     4$
    142651     040     001
 26 142653                          3$:     CLR A                  ;OFF
    142653     117
 27 142654                          4$:     STA A   GRAPH          ;STORE FLAG
    142654     227     100
 28 142656                                  INC     UPDFLG         ;UPDATE SCREEN
    142656     174     000     106
 29 142661                          5$:     RTS                    ;FINISHED
    142661     071
 30                                         ;
```

READ/WRITE FUNCTION

```
      1                                      .SBTTL  READ/WRITE FUNCTION
      2                                      ;
      3 142662                         READ: LDA B   OPR             ;GET OPERATION
        142662     366     002     300
      4 142665                               TBA                    ;SAVE
        142665     027
      5 142666                               AND B   #,17           ;MASK HIGH BITS
        142666     304     017
      6 142670                               ORA B   HADD           ;OR IN HIGH BITS
        142670     372     002     332
      7 142673                               STA B   RDADD          ;SAVE ADDRESS
        142673     367     002     333
      8 142676                               LDA B   OPR+1          ;GET LOW BYTE OF ADDRESS
        142676     366     002     301
      9 142701                               STA B   RDADD+1        ;SAVE
        142701     367     002     334
     10 142704                               LDX     RDADD          ;GET ADDRESS
        142704     376     002     333
     11 142707                               AND A   #,20           ;CHECK READ/WRITE
        142707     204     020
     12 142711                               BEQ     1$             ;ON READ - SKIP
        142711     047     005
     13 142713                               LDA A   WTBYTE         ;GET DATA
        142713     266     002     327
     14 142716                               STA A   0,X            ;STORE BYTE
        142716     247     000
     15 142720                         1$:   LDA A   1,X            ;READ DATA AND SEND TO CPU
        142720     246     001
     16 142722                               STA A   WDTBA
        142722     227     024
     17 142724                               LDA B   0,X
        142724     346     000
     18 142726                               STA B   WDTBB
        142726     327     025
     19 142730                               RTS                    ;FINISHED
```

```
     20                                        ;
```

SPECIAL MACRO DEFINITIONS

```
      2                                     .SBTTL  SPECIAL MACRO DEFINITIONS
      3                                     ;
      4                                     ;
      5                                     .MACRO  SETBIT  I,J
      6                                     LDA A   I
      7                                     ORA A   #,J
      8                                     STA A   I
      9                                     .ENDM   SETBIT
     10                                     ;
     11                                     ;
     12                                     .MACRO  CLRBIT  I,J
     13                                     LDA A   I
     14                                     AND A   #,377-J
     15                                     STA A   I
     16                                     .ENDM   CLRBIT
     17                                     ;
     18                                     ;
     19                                     .MACRO  BITTST  I,J
     20                                     LDA A   I
     21                                     BIT A   #,J
     22                                     .ENDM   BITTST
     23                                     ;
```

LIST PROCESSOR

```
                                        .SBTTL  LIST PROCESSOR
                                        ;
142731                          LSTENT: LDA B   RSTAT0          ;RUNNING ?
  142731     366     003     021
142734                                  BPL     LSTGRP          ;IF NOT - DON'T INHIBIT PRINTING
  142734     052     010
142736                                  LDA B   STPFLG          ;STRIPPING CHARACTERS ?
  142736     366     002     375
142741                                  BEQ     LSTGRP          ;IF NOT - SKIP
  142741     047     003
142743                                  INC     CHRFLG          ;ELSE INHIBIT PRINTING
  142743     174     000     105
                                        ;
142746                          LSTGRP: CLI                     ;THIS RUN IN BACKGROUND
  142746     016
142747                                  AND A   #,177           ;ONLY 7-BIT ASCII
  142747     204     177
142751                                  STA A   CHARG           ;SAVE THE CHARACTER
  142751     267     002     372
142754                                  BSR     LISTPR          ;NOW DO PROCESS
  142754     215     002
142756                                  SEI                     ;HOLD INTERRUPTS AGAIN
  142756     017
142757                                  RTS                     ;FINISHED
  142757     071
                                        ;
                                        ; 1.    IF IN SINGLE CHAR MODE, THEN:
                                        ;               CLEAR SINGLE CHAR MODE
                                        ;               SCAN FOR CONTROL CHARACTERS, IF FOUND:
                                        ;                       CLEAR GOSUB
                                        ;                       END
                                        ;               DO GOSUB (IF DEFINED)
                                        ;               IF CHARACTER USED - END
                                        ;               ELSE GO TO 2.
                                        ;
```

```
        142760     266     003     022
    26  142763                             BPL     LIST.D          ;NOT IN SINGLE - SKIP
        142763     052     036
    27  142765                             AND A   #,177           ;CLEAR SINGLE MODE
        142765     204     177
    28  142767                             STA A   LSTAT           ;SAVE STATUS
        142767     267     003     022
    29  142772                             LDA A   CHARG           ;GET CHARACTER
        142772     266     002     372
    30  142775                             CMP A   #,40            ; A CONTROL ?
        142775     201     040
    31  142777                             BCC     LIST.A          ;IF NOT - SKIP
        142777     044     010
    32  143001                             LDX     #,0             ;ELSE CLEAR GOSUB
        143001     316     000     000
    33  143004                             STX     GOSUB
        143004     377     002     370
    34  143007                             BRA     LIST.B
        143007     040     011
    35  143011                       LIST.A: LDX     GOSUB           ;GET PROCESS ADDRESS
        143011     376     002     370
```

LIST PROCESSOR

```
     36 143014                                BEQ     LIST.B          ;IF UNDEFINED - SKIP
        143014     047     004
     37 143016                                JSR     0,X             ;DO IT
        143016     255     000
     38 143020                                BCC     LIST.D          ;CHARACTER NOT USED
        143020     044     001
     39 143022                          LIST.B: RTS                     ;FINISHED
        143022     071
     40                                       ;
     41                                       ; 2.    DO COMMAND SCANNER
     42                                       ;       IF COMMAND IS FOUND - FINISHED
     43                                       ;
     44 143023                          LIST.D: JSR     CMDSCN          ;SCAN FOR COMMANDS
        143023     275     307     033
     45 143026                                BCS     LIST.E          ;COMMAND FOUND - DONE
        143026     045     005
     46 143030                                LDA A   RSTAT0          ;ARE WE RUNNING
        143030     266     003     021
     47 143033                                BMI     LIST.F          ;IF SO - SKIP AHEAD
        143033     053     001
     48 143035                          LIST.E: RTS                     ;FINISHED
        143035     071
     49                                       ;
     50                                       ; 3.    CHECK SCAN MODE, IF SET THEN:
     51                                       ;               IF CHAR IS (+) THEN:
     52                                       ;                       CLEAR SCAN
     53                                       ;                       SET BUILD
     54                                       ;                       RESET LIST BUFFER
     55                                       ;                       END
     56                                       ;
     57 143036                          LIST.F: BITTST  LSTAT,100       ;IN SCAN MODE ?
        143036     266     003     022
        143041     205     100
     58 143043                                BEQ     LIST.J          ;IF NOT - SKIP
        143043     047     033
```

```
   143045     366     002     372
60 143050                              CMP B   #,'+            ;IS IT A (+)  ?
   143050     301     053
61 143052                              BNE     LIST.G          ;IF NOT - SKIP OUT
   143052     046     023
62 143054                              AND A   #,277           ;ELSE CLEAR SCAN MODE
   143054     204     277
63 143056                              ORA A   #,40            ;SET BUILD
   143056     212     040
64 143060                              STA A   LSTAT           ;SAVE NEW STATUS
   143060     267     003     022
65 143063                              CLR     SCHFLG          ;CLEAR SEEN FLAG
   143063     177     002     374
66 143066                              CLR     LSTCNT          ;RESET BUFFER
   143066     177     002     364
67 143071                              LDX     #,LSBUFF
   143071     316     002     353
68 143074                              STX     LPNTR
   143074     377     002     365
69 143077                         LIST.G: RTS                   ;FINISHED
   143077     071
```

```
     70                                     ;
     71                                     ; 4.    CHECK BUILD, IF NOT SET THEN:
     72                                     ;               RESET BUFFER
     73                                     ;               IF CHAR IS (0-9) OR (-)
     74                                     ;                  OR FREE FORMAT   THEN:
     75                                     ;                       SET BUILD
     76                                     ;                       PUT CHAR IN BUFFER
     77                                     ;                       GOTO LCHECK
     78                                     ;                  ELSE: SET SCAN
     79                                     ;                        GO SCAN
     80                                     ;
     81 143100                        LIST.J: BITTST LSTAT,40         ;BUILDING ?
        143100     266     003     022
        143103     205     040
     82 143105                                BNE     LIST.O          ;IF SO - SKIP
        143105     046     056
     83 143107                                CLR     SCHFLG          ;CLEAR SEEN FLAG
        143107     177     002     374
     84 143112                                CLR     LSTCNT          ;RESET BUFFER
        143112     177     002     364
     85 143115                                LDX     #,LSBUFF
        143115     316     002     353
     86 143120                                STX     LPNTR
        143120     377     002     365
     87 143123                                LDA B   FORMAT          ;IN FREE FORMAT ?
        143123     366     002     367
     88 143126                                BEQ     LIST.K          ;IF SO - SET UP BUILD
        143126     047     017
     89 143130                                LDA B   CHARG           ;GET CHARACTER
        143130     366     002     372
     90 143133                                CMP B   #,'-            ;IS IT A (-) ?
        143133     301     055
     91 143135                                BEQ     LIST.K          ;IF SO - SKIP
        143135     047     010
     92 143137                                CMP B   #,'0            ;A BCD CHARACTER ?
```

```
 93 143141                              BCS     LIST.L          ;<0 - SKIP
    143141     045     013
 94 143143                              CMP B   #,'9            ;>9 ?
    143143     301     071
 95 143145                              BHI     LIST.L          ;NOT A NUMBER - SKIP
    143145     042     007
 96 143147                      LIST.K: ORA A   #,40            ;SET BUILD
    143147     212     040
 97 143151                              STA A   LSTAT           ;SAVE STATUS
    143151     267     003     022
 98 143154                              BRA     LIST.O          ;NOW PROCESS THIS CHARACTER
    143154     040     007
 99 143156                      LIST.L: ORA A   #,100           ;SET SCAN
    143156     212     100
100 143160                              STA A   LSTAT           ;SAVE STATUS
    143160     267     003     022
101 143163                              BRA     LIST.F          ;GO SCAN
    143163     040     251
102                                     ;
103                                     ; 5.    BUILD IS SET
104                                     ;               PUT CHARACTER IN BUFFER
```

```
105                                     ;               CHECK BUFFFER
106                                     ;
107 143165                      LIST.O: LDA A   FORMAT          ;CHECK FOR FREE FORMAT
    143165     266     002     367
108 143170                              BEQ     LIST.Q          ;IF SO - SKIP
    143170     047     036
109 143172                              LDA A   CHARG           ;GET CHARACTER
    143172     266     002     372
110 143175                              CMP A   #,40            ;ANY CONTROLS ?
    143175     201     040
111 143177                              BCC     LIST.X          ;IF NOT  SKIP
    143177     044     162
112 143201                      LIST.P: CLR     LSTCNT          ;RESET BUFFER
    143201     177     002     364
113 143204                              CLR     SCHFLG          ;SEEN CHARACTER FLAG
    143204     177     002     374
114 143207                              LDX     #,LSBUFF
    143207     316     002     353
115 143212                              STX     LPNTR
    143212     377     002     365
116 143215                              LDA A   LSTAT           ;SET STATUS FOR SCAN
    143215     266     003     022
117 143220                              AND A   #,17            ;SAVE LOWER FOR GRAPHICS
    143220     204     017
118 143222                              ORA A   #,100           ;SCANNING
    143222     212     100
119 143224                              STA A   LSTAT
    143224     267     003     022
120 143227                              RTS                     ;FINISHED
    143227     071
121                                     ;
122 143230                      LIST.Q: LDA A   SCHFLG          ;SEEN A CHAR ?
    143230     266     002     374
123 143233                              BNE     LIST.U          ;IF SO - SKIP
    143233     046     052
```

```
        143235     266     002     372
    125 143240                             CMP A   #,'-            ;A - SIGN ?
        143240     201     055
    126 143242                             BEQ     LIST.R          ;IF SO - SAVE IT
        143242     047     026
    127 143244                             CMP A   #,'+            ;A + SIGN ?
        143244     201     053
    128 143246                             BEQ     LIST.T          ;STRIP +'S
        143246     047     021
    129 143250                             CMP A   #,40            ;SPACE ?
        143250     201     040
    130 143252                             BEQ     LIST.T          ;STRIP SPACES
        143252     047     015
    131 143254                             CMP A   #,'0            ;ONLY WANT NUMERALS
        143254     201     060
    132 143256                             BCS     LIST.P
        143256     045     321
    133 143260                             CMP A   #,':
        143260     201     072
    134 143262                             BCC     LIST.P
        143262     044     315
```

```
  135 143264                                BSR     LIST.R          ;SAVE CHARACTER
      143264     215     004
  136 143266                        LIST.S: INC     SCHFLG          ;CHARACTER SEEN
      143266     174     002     374
  137 143271                        LIST.T: RTS                     ;FINISHED
      143271     071
  138 143272                        LIST.R: LDX     LPNTR           ;GET POINTER
      143272     376     002     365
  139 143275                                STA A   0,X             ;SAVE CHARACTER
      143275     247     000
  140 143277                                INX                     ;UPDATE POINTER
      143277     010
  141 143300                                STX     LPNTR
      143300     377     002     365
  142 143303                                INC     LSTCNT          ;ONE MORE CHARACTER
      143303     174     002     364
  143 143306                                RTS                     ;AND RETURN
      143306     071
  144                                       ;
  145 143307                        LIST.U: LDA A   CHARG           ;GET CHARACTER
      143307     266     002     372
  146 143312                                CMP A   #,'-            ;A - SIGN ?
      143312     201     055
  147 143314                                BEQ     LIST.W          ;IF SO - SKIP
      143314     047     036
  148 143316                                CMP A   #,'+            ;A + SIGN ?
      143316     201     053
  149 143320                                BEQ     LIST.W          ;IF SO - SKIP
      143320     047     032
  150 143322                                CMP A   #,'0            ;WANT ONLY NUMERALS
      143322     201     060
  151 143324                                BCS     LIST.W
      143324     045     026
  152 143326                                CMP A   #,':
      143326     201     072
```

```
        143330     044     022
    154 143332                             LDX     LPNTR           ;GET POINTER
        143332     376     002     365
    155 143335                             STA A   0,X             ;SAVE CHARACTER
        143335     247     000
    156 143337                             CPX     #,LSBUFF+8.     ;AT END OF BUFFER ?
        143337     214     002     363
    157 143342                             BEQ     LIST.V          ;IF SO - SKIP UPDATE
        143342     047     007
    158 143344                             INX                     ;ELSE UPDATE POINTER
        143344     010
    159 143345                             STX     LPNTR
        143345     377     002     365
    160 143350                             INC     LSTCNT          ;UPDATE COUNTER
        143350     174     002     364
    161 143353                  LIST.V:    RTS                     ;FINSHED
        143353     071
    162 143354                  LIST.W:    BSR     LCHE.A          ;GO EVALUATE AND PROCESS
        143354     215     031
    163 143356                             CLR     SCHFLG          ;CLEAR SEEN FLAG
        143356     177     002     374
```

```
   164 143361                                BRA     LIST.Q          ;RESCAN LAST CHARACTER
       143361     040     245
   165                                       ;
   166 143363                          LIST.X: LDX     LPNTR           ;GET POINTER
       143363     376     002     365
   167 143366                                STA A   0,X             ;SAVE CHARACTER
       143366     247     000
   168 143370                                INX                     ;UPDATE ADDRESS
       143370     010
   169 143371                                STX     LPNTR
       143371     377     002     365
   170 143374                                INC     LSTCNT          ;UPDATE COUNT
       143374     174     002     364
   171                                       ;
   172                                       ;
   173 143377                          LCHECK: LDA A   LSTCNT          ;GET COUNT
       143377     266     002     364
   174 143402                                CMP A   FORMAT          ;ENOUGH CHARACTERS ?
       143402     261     002     367
   175 143405                                BNE     LCHE.B          ;IF NOT - SKIP
       143405     046     023
   176 143407                          LCHE.A: JSR     EVALN           ;GO EVALUATE DATA
       143407     275     307     236
   177 143412                                CLR     LSTCNT          ;CLEAR BUFFER
       143412     177     002     364
   178 143415                                LDX     #,LSBUFF
       143415     316     002     353
   179 143420                                STX     LPNTR
       143420     377     002     365
   180 143423                                LDX     GOSUB           ;GET PROCESS
       143423     376     002     370
   181 143426                                BEQ     LCHE.B          ;IF UNDEFINED - SKIP
       143426     047     002
   182 143430                                JSR     0,X             ;ELSE DO PROCESS
       143430     255     000
```

```
        143432     071
    184                                     ;
```

COMMAND SCANNER

```
      1                                      .SBTTL  COMMAND SCANNER
      2                                      ;
      3 143433                         CMDSCN: LDA B   CHARG           ;GET CHARACTER
        143433     366     002     372
      4 143436                                 LDA A   PCHAR           ;GET PREVIOUS CHARACTER
        143436     266     002     373
      5 143441                                 STA B   PCHAR           ;SAVE NEW CHARACTER
        143441     367     002     373
      6 143444                                 CMP A   #,'$            ;OLD A ($) ?
        143444     201     044
      7 143446                                 BNE     CMDS.D          ;IF NOT - SKIP
        143446     046     061
      8 143450                                 CLRBIT  LSTAT,350       ;ALL NEW LIST CONTROL
        143450     266     003     022
        143453     204     027
        143455     267     003     022
      9 143460                                 CMP B   #,141           ;ALLOW LOWER CASE COMMANDS
        143460     301     141
     10 143462                                 BCS     CMD.UC
        143462     045     006
     11 143464                                 CMP B   #,173
        143464     301     173
     12 143466                                 BCC     CMD.UC
        143466     044     002
     13 143470                                 SUB B   #,40            ;MAKE LOWER CASE UPPER CASE
        143470     300     040
     14 143472                         CMD.UC: LDX     #,CMDTBL        ;GET COMMAND TABLE
        143472     316     307     133
     15 143475                                 CMP B   0,X             ;RUN COMMAND ?
        143475     341     000
     16 143477                                 BEQ     CMDS.B          ;IF SO - SKIP
        143477     047     021
     17 143501                                 BITTST  RSTAT0,200      ;RUNNING ?
        143501     266     003     021
        143504     205     200
```

```
   143506     047     021
19 143510                          CMDS.A: INX                     ;GET PAST ADDRESS
   143510     010
20 143511                                  INX
   143511     010
21 143512                                  INX
   143512     010
22 143513                                  LDA A   0,X             ;END OF COMMANDS ?
   143513     246     000
23 143515                                  BEQ     CMDS.C          ;IF SO - SKIP
   143515     047     010
24 143517                                  CBA                     ;A COMMAND ?
   143517     021
25 143520                                  BNE     CMDS.A          ;NO - CONTINUE SCAN
   143520     046     366
26 143522                          CMDS.B: INX                     ;GET TO ADDRESS
   143522     010
27 143523                                  LDX     0,X             ;GET JUMP ADDRESS
   143523     356     000
28 143525                                  JSR     0,X             ;DO COMMAND
   143525     255     000
```

COMMAND SCANNER

```
     29 143527                          CMDS.C: SEC
        143527     015
     30 143530                                  RTS                     ;FINISHED
        143530     071
     31 143531                          CMDS.D: CLC                     ;NO COMMAND
        143531     014
     32 143532                                  RTS                     ;FINISHED
        143532     071
     33                                         ;
     34                                         ;COMMAND AND JUMP ADDRESS TABLE
     35                                         ;
     36 143533                          CMDTBL: FCC     <R>
        143533     122
     37 143534                                  FDB     .$R
        143534     310     104
     38 143536                                  FCC     <O>
        143536     117
     39 143537                                  FDB     .$O
        143537     310     167
     40 143541                                  FCC     <M>
        143541     115
     41 143542                                  FDB     .$M
        143542     310     221
     42 143544                                  FCC     <L>
        143544     114
     43 143545                                  FDB     .$L
        143545     310     266
     44 143547                                  FCC     <D>
        143547     104
     45 143550                                  FDB     .$D
        143550     310     335
     46 143552                                  FCC     <U>
        143552     125
     47 143553                                  FDB     .$U
        143553     310     356
```

```
        143555     106
     49 143556                              FDB     .$F
        143556     312     074
     50 143560                              FCC     <C>
        143560     103
     51 143561                              FDB     .$C
        143561     312     301
     52 143563                              FCC     <X>
        143563     130
     53 143564                              FDB     .$X
        143564     311     157
     54 143566                              FCC     <Y>
        143566     131
     55 143567                              FDB     .$Y
        143567     311     227
     56 143571                              FCC     <G>
        143571     107
     57 143572                              FDB     .$G
        143572     314     261
     58 143574                              FCC     <V>
        143574     126
```

COMMAND SCANNER

```
     59 143575                              FDB     .$V
        143575     310     367
     60 143577                              FCC     <P>
        143577     120
     61 143600                              FDB     .$P
        143600     311     000
     62 143602                              FCC     <H>
        143602     110
     63 143603                              FDB     .$H
        143603     311     053
     64 143605                              FCC     <K>
        143605     113
     65 143606                              FDB     .$K
        143606     301     356
     66 143610                              FCC     <W>
        143610     127
     67 143611                              FDB     .$W
        143611     311     156
     68 143613                              FCC     <S>
        143613     123
     69 143614                              FDB     .$S
        143614     311     011
     70 143616                              FCC     <A>
        143616     101
     71 143617                              FDB     .$A
        143617     311     022
     72 143621                              FCC     <B>
        143621     102
     73 143622                              FDB     .$B
        143622     311     032
     74 143624                              FCC     <I>
        143624     111
     75 143625                              FDB     .$I
        143625     311     041
     76 143627                              FCC     <J>
```

```
    77 143630                              FDB     .$J
       143630     311     047
    78 143632                              FCC     <Z>
       143632     132
    79 143633                              FDB     .$Z
       143633     311     156
    80 143635     000                      .BYTE   0               ;TABLE TERMINATOR
    81                                     ;
```

EVALUATE NUMBER ROUTINE

```
                                        .SBTTL  EVALUATE NUMBER ROUTINE
                                        ;
143636                          EVALN:  LDX     #,NSIGN         ;POINT TO VARIABLES
  143636    316     002     376
143641                                  LDA A   #,11.           ;11. BYTES TO CLEAR
  143641    206     013
143643                          EVAL.A: CLR     0,X             ;CLEAR BYTE
  143643    157     000
143645                                  INX                     ;UPDATE ADDRESS
  143645    010
143646                                  DEC A                   ;MORE ?
  143646    112
143647                                  BGT     EVAL.A          ;LOOP UNTIL ALL CLEARED
  143647    056     372
143651                                  LDX     #,NUMBER+3      ;SET UP POINTER
  143651    316     003     002
143654                                  STX     NPNTR
  143654    377     003     004
143657                          EVAL.B: LDX     LPNTR           ;GET STRING POINTER
  143657    376     002     365
143662                                  DEX                     ;POINT TO CHARACTER
  143662    011
143663                                  STX     LPNTR           ;SAVE
  143663    377     002     365
143666                                  LDA B   0,X             ;GET CHARACTER
  143666    346     000
143670                                  CMP B   #,'-            ; A (-) ?
  143670    301     055
143672                                  BNE     EVAL.C          ;IF NOT - SKIP
  143672    046     007
143674                                  LDA A   #,377           ;SIGN IS NEGATIVE
  143674    206     377
143676                                  STA A   NSIGN
  143676    267     002     376
143701                                  BRA     EVAL.F          ;SKIP AHEAD
```

```
 20 143703                          EVAL.C: LDA A   NCNTR           ;GET COUNT
    143703     266     003     003
 21 143706                                  CMP A   #,4             ;GOT FOUR YET ?
    143706     201     004
 22 143710                                  BCC     EVAL.G          ;IF SO - SKIP
    143710     044     032
 23 143712                                  LDX     NPNTR           ;GET POINTER
    143712     376     003     004
 24 143715                                  CMP B   #,40            ;A SPACE ?
    143715     301     040
 25 143717                                  BEQ     EVAL.F          ;IF SO - SKIP
    143717     047     014
 26 143721                                  SUB B   #,'0            ;MAKE BCD
    143721     300     060
 27 143723                                  BMI     EVAL.D          ;IF NOT -  SKIP
    143723     053     004
 28 143725                                  CMP B   #,9.            ;MUST BE A DIGIT
    143725     301     011
 29 143727                                  BLS     EVAL.E          ;IF SO - SKIP
    143727     043     002
 30 143731                          EVAL.D: BRA     EVAL.F
```

EVALUATE NUMBER ROUTINE

```
           143731     040     002
    31     143733                         EVAL.E: STA B   0,X             ;SAVE CHARACTER
           143733     347     000
    32     143735                         EVAL.F: DEX                     ;UPDATE POINTER
           143735     011
    33     143736                                 STX     NPNTR
           143736     377     003     004
    34     143741                                 INC     NCNTR           ;UPDATE COUNT
           143741     174     003     003
    35     143744                         EVAL.G: DEC     LSTCNT          ;ANY MORE ?
           143744     172     002     364
    36     143747                                 BGT     EVAL.B          ;LOOP UNTIL DONE
           143747     056     306
    37                                            ;
    38                                            ;
```

4-DIGIT BCD TO BINARY CONVERSION

```
      1                                    .SBTTL  4-DIGIT BCD TO BINARY CONVERSION
      2                                    ;
      3 143751                      BCD4BN: LDA A   NUMBER+1        ;PACK 10 & 100'S DIGITS
        143751     266     003     000
      4 143754                              ASL A                   ;SHIFT INTO POSITION
        143754     110
      5 143755                              ASL A
        143755     110
      6 143756                              ASL A
        143756     110
      7 143757                              ASL A
        143757     110
      8 143760                              ORA A   NUMBER+2
        143760     272     003     001
      9 143763                              LDA B   NUMBER          ;PACK MSD'S NOW
        143763     366     002     377
     10 143766                              STA B   T1              ;SAVE HIGH ORDER
        143766     367     003     007
     11 143771                              STA A   T2              ;SAVE LOW ORDER
        143771     267     003     010
     12 143774                              LDX     #,BCDTBL        ;GET POINTER TO TABLE
        143774     316     310     054
     13 143777                              LDA A   #,12.           ;SET UP BIT COUNTER
        143777     206     014
     14 144001                              STA A   T0
        144001     267     003     006
     15 144004                              LDA A   NUMBER+3        ;INITIALIZE RESULT TO 1'S DIGIT
        144004     266     003     002
     16 144007                              CLR B
        144007     137
     17 144010                      BCDLP:  ASR     T1
        144010     167     003     007
     18 144013                              ROR     T2
        144013     166     003     010
     19 144016                              BCC     NOADD           ;IF BIT CLEAR - NO UPDATE
```

```
 20 144020                                ADD A   0,X             ;ELSE UPDATE RESULT
    144020     253     000
 21 144022                                ADC B   14,X
    144022     351     014
 22 144024                          NOADD: INX                    ;UPDATE TABLE POINTER
    144024     010
 23 144025                                DEC     T0              ;ANY BITS LEFT ?
    144025     172     003     006
 24 144030                                BGT     BCDLP           ;LOOP UNTIL DONE
    144030     056     356
 25 144032                                TST     NSIGN           ;NEGATIVE ?
    144032     175     002     376
 26 144035                                BEQ     BCD4.A          ;IF POSITIVE - SKIP
    144035     047     006
 27 144037                                COM A
    144037     103
 28 144040                                COM B
    144040     123
 29 144041                                ADD A   #,1
    144041     213     001
 30 144043                                ADC B   #,0
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35 Page 41-1

4-DIGIT BCD TO BINARY CONVERSION

```
   144043     311     000
31 144045                          BCD4.A: STA A   NUMBER+1        ;SAVE RESULT
   144045     267     003     000
32 144050                                  STA B   NUMBER
   144050     367     002     377
33 144053                                  RTS                     ;FINISHED
   144053     071
34                                         ;
35                                         ;BCD TO BINARY CONVERSION TABLE
36                                         ;
37                 .NLIST  BIN
38 144054  BCDTBL: .BYTE   12,24,50,120
39 144060          .BYTE   144,310,220,40
40 144064          .BYTE   350,320,240,100
41                 ;
42 144070          .BYTE   0,0,0,0
43 144074          .BYTE   0,0,1,3
44 144100          .BYTE   3,7,17,37
45                                         .LIST   BIN
46                                         ;
```

$R COMMAND

```
                                        .SBTTL  $R COMMAND
                                        ;
144104                          .$R:    SETBIT  RSTAT0,200      ;SAY RUNNING
  144104     266     003     021
  144107     212     200
  144111     267     003     021
144114                                  SETBIT  LSTAT,200       ;CHARACTER MODE
  144114     266     003     022
  144117     212     200
  144121     267     003     022
144124                                  CLR     FORMAT          ;SET FREE FORMAT AS DEFAULT
  144124     177     002     367
144127                                  LDX     #,.$RB          ;POINTER TO PROCESS
  144127     316     310     136
144132                                  STX     GOSUB
  144132     377     002     370
144135                                  RTS                     ;FINISHED
  144135     071
                                        ;
144136                          .$RB:   LDA A   CHARG           ;GET CHARACTER
  144136     266     002     372
144141                                  SUB A   #,'0            ;MAKE BCD
  144141     200     060
144143                                  BCS     .$RC            ;IF NOT - SKIP
  144143     045     012
144145                                  CMP A   #,9.            ;BCD ?
  144145     201     011
144147                                  BHI     .$RC            ;IF NOT - SKIP
  144147     042     006
144151                                  STA A   FORMAT          ;SAVE NEW FORMAT
  144151     267     002     367
144154                                  SEC                     ;CHARACTER USED
  144154     015
144155                                  BRA     .$RD
  144155     040     001
```

```
   144157     014
19 144160                          .$RD:   LDX     #,0             ;NO MORE HERE
   144160     316     000     000
20 144163                                  STX     GOSUB
   144163     377     002     370
21 144166                                  RTS                     ;FINISHED
   144166     071
22                                         ;
```

$O COMMAND

```
      1                                      .SBTTL  $O COMMAND
      2                                      ;
      3 144167                        .$O:   BRA     .$OC            ;SET UP FOR NEXT ENTRY
        144167     040     021
      4                                      ;
      5 144171                        .$OA:  LDX     NUMBER          ;GET VALUE
        144171     376     002     377
      6 144174                               STX     XTABLE+22       ;SAVE XORIGIN
        144174     377     002     252
      7 144177                               LDX     #,.$OB          ;SET FOR NEXT ENTRY
        144177     316     310     204
      8 144202                               BRA     .$OD
        144202     040     011
      9                                      ;
     10 144204                        .$OB:  LDX     NUMBER          ;GET VALUE
        144204     376     002     377
     11 144207                               STX     YTABLE+22       ;SAVE YORIGIN
        144207     377     002     222
     12 144212                        .$OC:  LDX     #,.$OA          ;SET UP NEXT ENTRY
        144212     316     310     171
     13 144215                        .$OD:  STX     GOSUB
        144215     377     002     370
     14 144220                               RTS                     ;FINISHED
        144220     071
     15                                      ;
```

$M COMMAND

```
      1                                        .SBTTL  $M COMMAND
      2                                        ;
      3 144221                          .$M:   CLRBIT  RSTAT0,160      ;RELATIVE TO ORIGIN
        144221     266     003     021
        144224     204     217
        144226     267     003     021
      4 144231                                 BRA     .$MC            ;GO SET UP FOR NEXT ENTRY
        144231     040     024
      5                                        ;
      6 144233                          .$MA:  LDX     NUMBER          ;GET VALUE
        144233     376     002     377
      7 144236                                 STX     XTABLE+4        ;SAVE AS X
        144236     377     002     234
      8 144241                                 LDX     #,.$MB          ;SET UP FOR NEXT ENTRY
        144241     316     310     246
      9 144244                                 BRA     .$MD
        144244     040     014
     10                                        ;
     11 144246                          .$MB:  LDX     NUMBER          ;GET VALUE
        144246     376     002     377
     12 144251                                 STX     YTABLE+4        ;SAVE AS Y
        144251     377     002     204
     13 144254                                 JSR     PROC            ;GO PROCESS
        144254     275     311     277
     14 144257                          .$MC:  LDX     #,.$MA          ;SET UP NEXT ENTRY POINT
        144257     316     310     233
     15 144262                          .$MD:  STX     GOSUB
        144262     377     002     370
     16 144265                                 RTS                     ;FINISHED
        144265     071
     17                                        ;
```

$L COMMAND

```
      1                                       .SBTTL  $L COMMAND
      2                                       ;
      3 144266                          .$L:  LDA A   RSTAT0          ;GET STATUS
        144266     266     003     021
      4 144271                                AND A   #,217           ;CLEAR MODES
        144271     204     217
      5 144273                                ORA A   #,100           ;SET RELATIVE TO CURRENT POSITION
        144273     212     100
      6 144275                                STA A   RSTAT0          ;SAVE
        144275     267     003     021
      7 144300                                BRA     .$LC
        144300     040     024
      8                                       ;
      9 144302                         .$LA:  LDX     NUMBER          ;GET VALUE
        144302     376     002     377
     10 144305                                STX     XTABLE+4        ;SAVE AS X
        144305     377     002     234
     11 144310                                LDX     #,.$LB          ;SET UP FOR NEXT ENTRY
        144310     316     310     315
     12 144313                                BRA     .$LD
        144313     040     014
     13                                       ;
     14 144315                         .$LB:  LDX     NUMBER          ;GET VALUE
        144315     376     002     377
     15 144320                                STX     YTABLE+4        ;SAVE AS Y
        144320     377     002     204
     16 144323                                JSR     PROC            ;GO PROCESS
        144323     275     311     277
     17 144326                         .$LC:  LDX     #,.$LA          ;SET UP FOR NEXT ENTRY
        144326     316     310     302
     18 144331                         .$LD:  STX     GOSUB
        144331     377     002     370
     19 144334                                RTS                     ;FINISHED
        144334     071
     20                                       ;
```

$D, $U, $V, $P, $S, AND $K COMMANDS

```
                                        .SBTTL  $D, $U, $V, $P, $S, AND $K COMMANDS
                                        ;
144335                          .$D:    SETBIT  RSTAT0,2        ;INDICATE PEN DOWN
  144335     266     003     021
  144340     212     002
  144342     267     003     021
144345                                  LDA A   #,1             ;SET UP PLOT CONTROL
  144345     206     001
144347                                  STA A   PTABLE+11
  144347     267     002     271
144352                                  JSR     PLOTXY          ;PLOT THE POINT
  144352     275     301     013
144355                                  RTS                     ;FINISHED
  144355     071
                                        ;
                                        ;
144356                          .$U:    CLRBIT  RSTAT0,2        ;INDICATE PEN UP
  144356     266     003     021
  144361     204     375
  144363     267     003     021
144366                                  RTS                     ;FINISHED
  144366     071
                                        ;
                                        ;
144367                          .$V:    CLRBIT  RSTAT0,10       ;VECTOR MODE
  144367     266     003     021
  144372     204     367
  144374     267     003     021
144377                                  RTS                     ;FINISHED
  144377     071
                                        ;
                                        ;
144400                          .$P:    SETBIT  RSTAT0,10       ;POINT MODE
  144400     266     003     021
  144403     212     010
```

```
    19 144410                                RTS                     ;FINISHED
       144410     071
    20                                       ;
    21                                       ;
    22 144411                          .$S:  CLRBIT  RSTAT0,200      ;NOT RUNNING
       144411     266     003     021
       144414     204     177
       144416     267     003     021
    23 144421                                RTS                     ;FINISHED
       144421     071
    24                                       ;
    25                                       ;
    26         140756                  .$K   =GRPCLR                 ;CLEAR GRAPHICS DIPLAY
    27                                       ;
```

$A, $B, $I, AND $J COMMANDS

```
      1                                       .SBTTL  $A, $B, $I, AND $J COMMANDS
      2                                       ;
      3 144422                          .$A:  LDA A   #,377           ;TURN ON SCREEN
        144422     206     377
      4 144424                                STA A   GRAPH
        144424     227     100
      5 144426                                INC     UPDFLG          ;TELL SYSTEM
        144426     174     000     106
      6 144431                                RTS
        144431     071
      7                                       ;
      8                                       ;
      9 144432                          .$B:  CLR     GRAPH           ;TURN OFF SCREEN
        144432     177     000     100
     10 144435                                INC     UPDFLG          ;TELL SYSTEM
        144435     174     000     106
     11 144440                                RTS
        144440     071
     12                                       ;
     13                                       ;
     14 144441                          .$I:  LDA A   #,377           ;INHIBIT PRINTING
        144441     206     377
     15 144443                                STA A   STPFLG
        144443     267     002     375
     16 144446                                RTS
        144446     071
     17                                       ;
     18                                       ;
     19 144447                          .$J:  CLR     STPFLG          ;ENABLE PRINTING
        144447     177     002     375
     20 144452                                RTS
        144452     071
     21                                       ;
```

$H COMMAND

```
      1                                          .SBTTL  $H COMMAND
      2                                          ;
      3 144453                           .$H:    SETBIT  RSTAT0,4        ;HOLD FLAG
        144453     266     003     021
        144456     212     004
        144460     267     003     021
      4 144463                                   LDX     #,HLDINT        ;PLACE NEW INTERRUPT VECTOR
        144463     316     311     143
      5 144466                                   STX     IRQ13
        144466     377     017     370
      6 144471                           1$:     CMP B   CHARG           ;AN H ?
        144471     361     002     372
      7 144474                                   BEQ     2$              ;IF SO - SKIP
        144474     047     005
      8 144476                                   LDA B   CHARG           ;ELSE USE AN H
        144476     366     002     372
      9 144501                                   BRA     3$
        144501     040     002
     10 144503                           2$:     LDA B   #,40            ;OR A SPACE
        144503     306     040
     11 144505                           3$:     STA B   CTLINE+64       ;FOR FLASHING
        144505     367     040     064
     12 144510                                   LDX     #,10000.        ;LOOP COUNTER
        144510     316     047     020
     13 144513                           4$:     BITTST  RSTAT0,4        ;CLEARED YET /
        144513     266     003     021
        144516     205     004
     14 144520                                   BEQ     5$              ;IF SO - SKIP
        144520     047     005
     15 144522                                   DEX                     ;CHECK COUNTER
        144522     011
     16 144523                                   BNE     4$              ;LOOP FOR A WHILE
        144523     046     366
     17 144525                                   BRA     1$
        144525     040     342
```

```
   144527     316     300     104
19 144532                              STX     IRQ13
   144532     377     017     370
20 144535                              LDA A   #,40            ;CLEAR FLASHING CHARACTER
   144535     206     040
21 144537                              STA A   CTLINE+64
   144537     267     040     064
22 144542                              RTS                     ;FINISHED
   144542     071
23                                     ;
24 144543                         HLDINT: LDA A   NPRDTA          ;CLEAR INTERRUPT FLAG
   144543     226     054
25 144545                              CLRBIT  RSTAT0,4        ;CLEAR HOLD FLAG
   144545     266     003     021
   144550     204     373
   144552     267     003     021
26 144555                              RTI                     ;RETURN FROM INTERRUPT
   144555     073
27                                     ;
28                                     ;
29 144556                         .$W:
```

M6800 CROSS-ASSEMBLER MACRO V05.06 Monday 15-Oct-07 06:35 Page 48-1

```
$H COMMAND


     30 144556                          .$Z:    RTS                     ;DUMBY ROUTINE
        144556     071
     31                                         ;
```

$X COMMAND

```
      1                                      .SBTTL  $X COMMAND
      2                                      ;
      3 144557                        .$X:   LDX     #,.$XA          ;SET FOR NEXT ENTRY
        144557     316     311     200
      4 144562                               STX     GOSUB
        144562     377     002     370
      5 144565                               LDA A   RSTAT0          ;GET STATUS
        144565     266     003     021
      6 144570                               AND A   #,217           ;CLEAR OTHER MODES
        144570     204     217
      7 144572                               ORA A   #,40            ;SET XSTEP MODE
        144572     212     040
      8 144574                               STA A   RSTAT0          ;SAVE
        144574     267     003     021
      9 144577                               RTS                     ;FINISHED
        144577     071
     10                                      ;
     11                                      ;
     12 144600                       .$XA:  LDX     NUMBER          ;GET VALUE
        144600     376     002     377
     13 144603                              STX     XTABLE+4        ;SAVE
        144603     377     002     234
     14 144606                              LDX     #,.$XB          ;SET FOR NEXT ENTRY
        144606     316     311     215
     15 144611                              STX     GOSUB
        144611     377     002     370
     16 144614                              RTS                     ;FINISHED
        144614     071
     17                                      ;
     18                                      ;
     19 144615                       .$XB:  LDX     NUMBER          ;GET NUMBER
        144615     376     002     377
     20 144620                              STX     YTABLE+4        ;SAVE
        144620     377     002     204
     21 144623                              JSR     PROC            ;DO PROCESS
```

```
    22 144626                           RTS                 ;FINISHED
       144626     071
    23                                  ;
```

$Y COMMAND

```
      1                                          .SBTTL  $Y COMMAND
      2                                          ;
      3 144627                           .$Y:    LDX     #,.$YA          ;NEXT ENTRY POINT
        144627     316     311     250
      4 144632                                   STX     GOSUB
        144632     377     002     370
      5 144635                                   LDA A   RSTAT0          ;GET STATUS
        144635     266     003     021
      6 144640                                   AND A   #,217           ;CLEAR ALL MODES
        144640     204     217
      7 144642                                   ORA A   #,20            ;SET YSTEP MODE
        144642     212     020
      8 144644                                   STA A   RSTAT0          ;SAVE
        144644     267     003     021
      9 144647                                   RTS                     ;FINISHED
        144647     071
     10                                          ;
     11                                          ;
     12 144650                           .$YA:   LDX     NUMBER          ;GET VALUE
        144650     376     002     377
     13 144653                                   STX     YTABLE+4        ;SAVE
        144653     377     002     204
     14 144656                                   LDX     #,.$YB          ;NEXT ENTRY POINT
        144656     316     311     265
     15 144661                                   STX     GOSUB
        144661     377     002     370
     16 144664                                   RTS                     ;FINISHED
        144664     071
     17                                          ;
     18                                          ;
     19 144665                           .$YB:   LDX     NUMBER          ;GET VALUE
        144665     376     002     377
     20 144670                                   STX     XTABLE+4        ;SAVE
        144670     377     002     234
     21 144673                                   JSR     PROC            ;DO PROCESS
```

```
    22 144676                           RTS                 ;FINISHED
       144676     071
    23                                  ;
```

PROCESS ROUTINE

```
      1                                        .SBTTL  PROCESS ROUTINE
      2                                        ;
      3 144677                         PROC:   BITTST  RSTAT0,40       ;AUTO X MODE ?
        144677     266     003     021
        144702     205     040
      4 144704                                 BEQ     PROC.A          ;IF NOT - SKIP
        144704     047     006
      5 144706                                 BSR     VAL.X           ;FXVAL=XVAL+XSTEP
        144706     215     121
      6 144710                                 BSR     ORG.Y           ;FYVAL=YORG+YSTEP
        144710     215     142
      7 144712                                 BRA     PROC.D
        144712     040     030
      8                                        ;
      9 144714                         PROC.A: BIT A   #,20            ;AUTO Y MODE ?
        144714     205     020
     10 144716                                 BEQ     PROC.B          ;IF NOT - SKIP
        144716     047     006
     11 144720                                 BSR     ORG.X           ;FXVAL=XORG+XSTEP
        144720     215     125
     12 144722                                 BSR     VAL.Y           ;FYVAL=YVAL+YSTEP
        144722     215     112
     13 144724                                 BRA     PROC.D
        144724     040     016
     14                                        ;
     15 144726                         PROC.B: BIT A   #,100           ;RELATIVE TO CURRENT ?
        144726     205     100
     16 144730                                 BEQ     PROC.C          ;IF NOT - SKIP
        144730     047     006
     17 144732                                 BSR     VAL.X           ;FXVAL=XVAL+XSTEP
        144732     215     075
     18 144734                                 BSR     VAL.Y           ;FYVAL=YVAL+YSTEP
        144734     215     100
     19 144736                                 BRA     PROC.D
        144736     040     004
```

```
 21 144740                          PROC.C: BSR     ORG.X           ;FXVAL=XORG+XSTEP
    144740     215     105
 22 144742                                  BSR     ORG.Y           ;FYVAL=YORG+YSTEP
    144742     215     110
 23                                         ;
 24 144744                          PROC.D: LDA A   RSTAT0          ;USE PEN STATUS FOR INTENSITY
    144744     266     003     021
 25 144747                                  AND A   #,2
    144747     204     002
 26 144751                                  ASR A
    144751     107
 27 144752                                  STA A   PTABLE+11
    144752     267     002     271
 28 144755                                  LDA A   #,40            ;INHIBIT VECTOR UPDATING
    144755     206     040
 29 144757                                  STA A   OPR
    144757     267     002     300
 30 144762                                  BITTST  RSTAT0,10       ;IN VECTOR MODE ?
    144762     266     003     021
    144765     205     010
 31 144767                                  BNE     PROC.E          ;IF NOT - SKIP VECTOR DRAWING
```

PROCESS ROUTINE

```
   144767     046     003
32 144771                              JSR     VECTOR          ;GO DO VECTOR DRAWING
   144771     275     302     333
33 144774                      PROC.E: LDX     XTABLE+2        ;XVAL=FXVAL
   144774     376     002     232
34 144777                              STX     XTABLE
   144777     377     002     230
35 145002                              STX     XTABLE+10       ;VALX=FXVAL
   145002     377     002     240
36 145005                              LDX     YTABLE+2        ;YVAL=FYVAL
   145005     376     002     202
37 145010                              STX     YTABLE
   145010     377     002     200
38 145013                              STX     YTABLE+10       ;VALY=FYVAL
   145013     377     002     210
39 145016                              BITTST  RSTAT0,10       ;POINT MODE ?
   145016     266     003     021
   145021     205     010
40 145023                              BEQ     PROC.F          ;IF NOT - FINISHED
   145023     047     003
41 145025                              JSR     PLOTXY          ;PLOT A POINT
   145025     275     301     013
42 145030                      PROC.F: RTS                     ;FINISHED
   145030     071
43                                     ;
44                                     ;
45 145031                      VAL.X:  LDX     #,XTABLE
   145031     316     002     230
46 145034                              BRA     VALSTP
   145034     040     003
47 145036                      VAL.Y:  LDX     #,YTABLE
   145036     316     002     200
48 145041                      VALSTP: LDA A   1,X
   145041     246     001
49 145043                              LDA B   0,X
```

```
 50 145045                                BRA     ORGS.A
    145045     040     014
 51                                       ;
 52 145047                        ORG.X:  LDX     #,XTABLE
    145047     316     002     230
 53 145052                                BRA     ORGSTP
    145052     040     003
 54 145054                        ORG.Y:  LDX     #,YTABLE
    145054     316     002     200
 55 145057                        ORGSTP: LDA A   23,X
    145057     246     023
 56 145061                                LDA B   22,X
    145061     346     022
 57 145063                        ORGS.A: ADD A   5,X
    145063     253     005
 58 145065                                ADC B   4,X
    145065     351     004
 59 145067                                STA A   3,X
    145067     247     003
 60 145071                                STA B   2,X
    145071     347     002
```

PROCESS ROUTINE

```
     61 145073                                  RTS
        145073     071
     62                                         ;
```

$F COMMAND

```
      2                                       .SBTTL  $F COMMAND
      3                                       ;
      4 145074                          .$F:  LDX     #,.$FB          ;NEXT ENTRY POINT
        145074     316     312     114
      5 145077                                STX     GOSUB
        145077     377     002     370
      6 145102                          .$FA: SETBIT  LSTAT,200       ;RE-ENABLE CHARACTER MODE
        145102     266     003     022
        145105     212     200
        145107     267     003     022
      7 145112                                SEC                     ;CHARACTER USED
        145112     015
      8 145113                                RTS                     ;FINISHED
        145113     071
      9                                       ;
     10 145114                          .$FB: LDA A   CHARG           ;GET CHARACTER
        145114     266     002     372
     11 145117                                CMP A   #,141           ;CONVERT LOWER CASE TO UPPER CASE
        145117     201     141
     12 145121                                BCS     1$
        145121     045     006
     13 145123                                CMP A   #,173
        145123     201     173
     14 145125                                BCC     1$
        145125     044     002
     15 145127                                SUB A   #,40
        145127     200     040
     16 145131                          1$:   CMP A   #,'H            ;HORIZONTAL MODE ?
        145131     201     110
     17 145133                                BNE     .$FC            ;IF NOT - SKIP
        145133     046     012
     18 145135                                CLRBIT  LSTAT,1         ;HORIZONTAL MODE
        145135     266     003     022
        145140     204     376
        145142     267     003     022
```

```
   145145     040     333
20                                      ;
21 145147                       .$FC:   CMP A   #,'V            ;VERTICAL MODE ?
   145147     201     126
22 145151                               BNE     .$FD            ;IF NOT - SKIP
   145151     046     012
23 145153                               SETBIT  LSTAT,1         ;VERTICAL MODE
   145153     266     003     022
   145156     212     001
   145160     267     003     022
24 145163                               BRA     .$FA
   145163     040     315
25                                      ;
26 145165                       .$FD:   CMP A   #,'L            ;LOWER CASE ?
   145165     201     114
27 145167                               BNE     .$FE            ;IF NOT SKIP
   145167     046     012
28 145171                               CLRBIT  LSTAT,4         ;SAY LOWER CASE
   145171     266     003     022
   145174     204     373
   145176     267     003     022
```

$F COMMAND

```
    29 145201                                BRA     .$FA
       145201     040     277
    30                                       ;
    31 145203                          .$FE: CMP A   #,'G            ;GREEK LETTERS ?
       145203     201     107
    32 145205                                BNE     1$             ;IF NOT - SKIP
       145205     046     012
    33 145207                                SETBIT  LSTAT,4        ;INDICATE GREEK
       145207     266     003     022
       145212     212     004
       145214     267     003     022
    34 145217                                BRA     .$FA
       145217     040     261
    35                                       ;
    36 145221                          1$:   CMP A   #,'/            ;SLASHED ZERO ?
       145221     201     057
    37 145223                                BEQ     .$FA           ;NO-OP
       145223     047     255
    38                                       ;
    39 145225                                CMP A   #,'\           ;NON SLASHED ZERO ?
       145225     201     134
    40 145227                                BEQ     .$FA           ;NO-OP
       145227     047     251
    41                                       ;
    42 145231                          .$FF: JSR     XYSIZE         ;FIND SIZING
       145231     275     313     310
    43 145234                                BCC     .$FZ           ;BAD SIZING
       145234     044     033
    44 145236                                LDA B   LSTAT          ;GET STATUS
       145236     366     003     022
    45 145241                                BIT B   #,10           ;NEED X OR Y SIZE ?
       145241     305     010
    46 145243                                BNE     .$FG           ;IF Y - SKIP
       145243     046     012
    47 145245                                ORA B   #,10           ;NEXT IS Y
```

```
$F COMMAND


        145300     071

     59                                      ;
```

$C COMMAND

```
                                        .SBTTL  $C COMMAND
                                        ;
145301                          .$C:    JSR     .$U             ;ALWAYS RAISE PEN
  145301     275     310     356
145304                                  LDX     #,CHRSCN        ;NEXT ENTRY POINT
  145304     316     312     337
145307                                  STX     GOSUB
  145307     377     002     370
145312                                  LDA A   LSTAT           ;GET STATUS
  145312     266     003     022
145315                                  ORA A   #,200           ;ENABLE CHARACTER MODE
  145315     212     200
145317                                  AND A   #,375           ;CHARACTER SCAN
  145317     204     375
145321                                  STA A   LSTAT
  145321     267     003     022
145324                                  LDA A   RSTAT0          ;GET MODE STATUS
  145324     266     003     021
145327                                  AND A   #,217           ;CLEAR CURRENT MODE
  145327     204     217
145331                                  ORA A   #,100           ;$L MODE
  145331     212     100
145333                                  STA A   RSTAT0          ;SAVE
  145333     267     003     021
145336                                  RTS                     ;FINISHED
  145336     071
                                        ;
145337                          CHRSCN: LDA B   CHARG           ;GET CHARACTER
  145337     366     002     372
145342                                  SETBIT  LSTAT,200       ;RE-ENABLE CHARACTER MODE
  145342     266     003     022
  145345     212     200
  145347     267     003     022
145352                                  BIT A   #,2             ;PLOTTING CHARACTERS ?
  145352     205     002
```

$C COMMAND

```
     30 145400                                BEQ     CHRS.D          ;IF LOWER - SKIP
        145400     047     014
     31 145402                                CMP B   #,140           ;IN RANGE OF GREEKS ?
        145402     301     140
     32 145404                                BCS     CHRS.D          ;IF NOT - SKIP
        145404     045     010
     33 145406                                BNE     CHRS.C          ;IF NOT - SKIP
        145406     046     004
     34 145410                                LDA B   #,37            ;OUT OF PLACE CHARACTER
        145410     306     037
     35 145412                                BRA     CHRS.D
        145412     040     002
     36 145414                      CHRS.C: SUB B   #,141           ;GET TO GREEKS
        145414     300     141
     37 145416                      CHRS.D: BITTST  RSTAT0,10       ;VECTOR MODE ?
        145416     266     003     021
        145421     205     010
     38 145423                              BNE     CHRS.L          ;IF NOT - POINT MODE
        145423     046     101
     39 145425                      CHRSCX: CLR A                   ;FIND ENTRY IN TABLE
        145425     117
     40 145426                              ASL B                   ;TWO BYTES PER POINTER
        145426     130
     41 145427                              ROL A
        145427     111
     42 145430                              ADD B   #,CHTBLE&377
        145430     313     000
     43 145432                              ADC A   #,CHTBLE&177400/400
        145432     211     320
     44 145434                              STA B   CHPNTR+1        ;SAVE POINTER
        145434     367     003     014
     45 145437                              STA A   CHPNTR
        145437     267     003     013
     46 145442                              LDX     CHPNTR          ;GET POINTER
        145442     376     003     013
```

$C COMMAND

```
    58 145475                              JSR     .$U             ;LIFT PEN
       145475     275     310     356
    59 145500                              BRA     CHRS.H
       145500     040     011
    60 145502                   CHRS.G:    JSR     GETXST          ;GET X VECTOR
       145502     275     314     044
    61 145505                              JSR     GETYST          ;GET Y VECTOR
       145505     275     314     150
    62 145510                              JSR     PROC            ;GO PROCESS
       145510     275     311     277
    63 145513                   CHRS.H:    LDX     CHPNTR          ;UPDATE POINTER
       145513     376     003     013
    64 145516                              INX
       145516     010
    65 145517                              STX     CHPNTR
       145517     377     003     013
    66 145522                              BRA     CHRS.E          ;GET NEXT VECTOR
       145522     040     326
    67 145524                   CHRS.I:    SEC                     ;USED CHARACTER
       145524     015
    68 145525                              RTS                     ;FINISHED
       145525     071
    69                                     ;
    70                                     ;POINT MODE
    71                                     ;
    72 145526                   CHRS.L:    CLR A                   ;COMPUTE ADDRESS IN CHROM
       145526     117
    73 145527                              ASL B
       145527     130
    74 145530                              ROL A
       145530     111
    75 145531                              ASL B
       145531     130
    76 145532                              ROL A
       145532     111
```

$C COMMAND

```
        145560     212     002
        145562     267     003     021
     88 145565                             LDA A   #,10            ;PRESET PXCNT
        145565     206     010
     89 145567                             STA A   XTABLE+25
        145567     267     002     255
     90 145572                             LDA A   #,9.            ;PRESET PYCNT
        145572     206     011
     91 145574                             STA A   YTABLE+25
        145574     267     002     225
     92 145577                             LDA A   #,12.           ;PRESET YCNT
        145577     206     014
     93 145601                             STA A   YTABLE+24
        145601     267     002     224
     94 145604                             LDX     CHPNTR          ;GET POINTER TO POINTS
        145604     376     003     013
     95 145607                        CHRS.M: LDA A   0,X             ;GET DOT PATTERN
        145607     246     000
     96 145611                             BEQ     CHRS.P          ;IF NO POINTS - SKIP
        145611     047     036
     97 145613                             STA A   PROW            ;SAVE PATTERN
        145613     267     003     015
     98 145616                             LDA A   #,10            ;DOT COUNT
        145616     206     010
     99 145620                             STA A   XTABLE+24
        145620     267     002     254
    100 145623                        CHRS.N: ASL     PROW            ;DOT HERE ?
        145623     170     003     015
    101 145626                             BCC     CHRS.O          ;IF NOT SKIP
        145626     044     011
    102 145630                             JSR     PGTXST          ;GET XSTEP VALUE
        145630     275     314     010
    103 145633                             JSR     PGTYST          ;GET YSTEP VALUE
        145633     275     314     114
    104 145636                             JSR     PROC            ;GO PROCESS POINT
```

$C COMMAND

```
        145673     206     011
    116 145675                             STA A   YTABLE+24
        145675     267     002     224
    117 145700                             JSR     PGTYST          ;SET YSTEP VALUE
        145700     275     314     114
    118 145703                             JSR     PROC            ;GO PROCESS
        145703     275     311     277
    119 145706                             SEC                     ;CHARACTER USED
        145706     015
    120 145707                             RTS                     ;FINISHED
        145707     071
    121                                    ;
```

XYSIZE ROUTINE

```
      1                                       .SBTTL  XYSIZE ROUTINE
      2                                       ;
      3 145710                          XYSIZE: LDA A   CHARG           ;GET CHARACTER
        145710     266     002     372
      4 145713                                  SUB A   #,'0            ;MAKE INTO BCD
        145713     200     060
      5 145715                                  BCS     XYSI.Z          ;ERROR - SKIP
        145715     045     026
      6 145717                                  CMP A   #,9.
        145717     201     011
      7 145721                                  BHI     XYSI.Z          ;ERROR - SKIP
        145721     042     022
      8 145723                                  CLR B                   ;COMPUTE TABLE ADDRESS
        145723     137
      9 145724                                  ADD A   #,XYTBL&377     ;LOW ORDER
        145724     213     347
     10 145726                                  ADC B   #,XYTBL&177400/400
        145726     311     313
     11 145730                                  STA A   T3              ;SAVE ADDRESS
        145730     267     003     011
     12 145733                                  STA B   T2
        145733     367     003     010
     13 145736                                  LDX     T2              ;GET ADDRESS
        145736     376     003     010
     14 145741                                  LDA A   0,X             ;GET SIZE SPECIFICATION
        145741     246     000
     15 145743                                  SEC                     ;GOOD RESULT
        145743     015
     16 145744                                  RTS                     ;FINISHED
        145744     071
     17 145745                          XYSI.Z: CLC                     ;BAD SPECIFIER
        145745     014
     18 145746                                  RTS                     ;FINISHED
        145746     071
     19                                         ;
```

X/Y GET ROUTINES

```
      1                                     .SBTTL  X/Y GET ROUTINES
      2                                     ;
      3 145761                      XGET:   LDA A   0,X             ;GET VECTOR
        145761     246     000
      4 145763                              ASR A                   ;SHIFT INTO POSITION
        145763     107
      5 145764                              ASR A
        145764     107
      6 145765                              ASR A
        145765     107
      7 145766                              ASR A
        145766     107
      8 145767                              BRA     XGET.A
        145767     040     002
      9 145771                      YGET:   LDA A   0,X             ;GET VECTOR
        145771     246     000
     10 145773                      XGET.A: CLR B                   ;HIGH ORDER = 0
        145773     137
     11 145774                              AND A   #,17            ;MASK JUNK
        145774     204     017
     12 145776                              BIT A   #,10            ;NEGATIVE VECTOR ?
        145776     205     010
     13 146000                              BEQ     XGET.Z          ;IF NOT - FINISHED
        146000     047     005
     14 146002                              AND A   #,7             ;SAVE ONLY MAGITUDE
        146002     204     007
     15 146004                              NEG A                   ;TWO'S COMPLEMENT
        146004     100
     16 146005                              LDA B   #,377           ;SIGN EXTEND
        146005     306     377
     17 146007                      XGET.Z: RTS                     ;FINISHED
        146007     071
     18                                     ;
```

GENERATE X STEP ROUTINE

```
                                        .SBTTL  GENERATE X STEP ROUTINE
                                        ;
                                        ;POINT MODE
                                        ;
146010                          PGTXST: LDA A   XTABLE+21       ;SET UP FOR MULTIPLY
  146010     266     002     251
146013                                  STA A   T0
  146013     267     003     006
146016                                  LDA A   #,10
  146016     206     010
146020                                  STA A   T1
  146020     267     003     007
146023                                  LDA B   XTABLE+24       ;GET LOCATION TO PLOT
  146023     366     002     254
146026                                  LDA A   XTABLE+25       ;PREVIOUS POINT
  146026     266     002     255
146031                                  STA B   XTABLE+25       ;SAVE FOR NEXT TIME
  146031     367     002     255
146034                                  SBA                     ;PXCNT-XCNT
  146034     020
146035                                  CLR B                   ;SET UP <B,A>
  146035     137
146036                                  TST A
  146036     115
146037                                  BPL     GETX.X          ;IF PLUS - SKIP
  146037     052     020
146041                                  COM B                   ;ELSE SIGN EXTEND
  146041     123
146042                                  BRA     GETX.X          ;GO FINISH
  146042     040     015
                                        ;
                                        ;VECTOR MODE
                                        ;
146044                          GETXST: LDA A   XTABLE+21       ;GET X SIZE
  146044     266     002     251
```

GENERATE X STEP ROUTINE

```
     33 146102                          GETX.A: ROL     LSTAT           ;RESTORE LSTAT
        146102     171     003     022
     34 146105                                  STA A   YTABLE+5                ;SAVE RESULT
        146105     267     002     205
     35 146110                                  STA B   YTABLE+4
        146110     367     002     204
     36 146113                                  RTS                     ;FINISHED
        146113     071
     37                                         ;
```

GENERATE Y STEP ROUTINE

```
      1                                       .SBTTL  GENERATE Y STEP ROUTINE
      2                                       ;
      3                                       ;POINT MODE
      4                                       ;
      5 146114                          PGTYST: LDA A   YTABLE+21       ;SET UP FOR MULTIPLY
        146114     266     002     221
      6 146117                                  STA A   T0
        146117     267     003     006
      7 146122                                  LDA A   #,10
        146122     206     010
      8 146124                                  STA A   T1
        146124     267     003     007
      9 146127                                  LDA B   YTABLE+24       ;CURRENT POSITION TO PLOT
        146127     366     002     224
     10 146132                                  LDA A   YTABLE+25       ;PREVIOUS POSITION
        146132     266     002     225
     11 146135                                  STA B   YTABLE+25       ;SAVE FOR NEXT TIME
        146135     367     002     225
     12 146140                                  SBA                     ;FIND RELATIVE MOVE
        146140     020
     13 146141                                  CLR B                   ;SET UP <B,A>
        146141     137
     14 146142                                  TST A
        146142     115
     15 146143                                  BPL     GETY.X          ;IF PLUS - SKIP
        146143     052     020
     16 146145                                  COM B                   ;ELSE SIGN EXTEND
        146145     123
     17 146146                                  BRA     GETY.X
        146146     040     015
     18                                       ;
     19                                       ;VECTOR MODE
     20                                       ;
     21 146150                          GETYST: LDA A   YTABLE+21       ;GET Y SIZE
        146150     266     002     221
```

GENERATE Y STEP ROUTINE

```
     33 146206                          GETY.A: ROL     LSTAT           ;RESTORE LSTAT
        146206     171     003     022
     34 146211                                  COM A                   ;NEED NEGATIVE
        146211     103
     35 146212                                  COM B
        146212     123
     36 146213                                  ADD A   #,1
        146213     213     001
     37 146215                                  ADC B   #,0
        146215     311     000
     38 146217                                  STA A   XTABLE+5        ;SAVE RESULT
        146217     267     002     235
     39 146222                                  STA B   XTABLE+4
        146222     367     002     234
     40 146225                                  RTS                     ;FINISHED
        146225     071
     41                                         ;
```

MULTIPLY ROUTINE

```
      1                                     .SBTTL  MULTIPLY ROUTINE
      2                                     ;
      3 146226                        MULTP:  STA A   T3              ;SAVE IN TEMPORARY
        146226     267     003     011
      4 146231                                STA B   T2
        146231     367     003     010
      5 146234                                CLR A                   ;INITIALIZE
        146234     117
      6 146235                                CLR B
        146235     137
      7 146236                        MULT.A: ASL A                   ;SHIFT RESULT
        146236     110
      8 146237                                ROL B
        146237     131
      9 146240                                ASL T0                  ;BIT SET ?
        146240     170     003     006
     10 146243                                BCC     MULT.B          ;IF NOT - SKIP
        146243     044     006
     11 146245                                ADD A   T3              ;UPDATE RESULT
        146245     273     003     011
     12 146250                                ADC B   T2
        146250     371     003     010
     13 146253                        MULT.B: DEC     T1              ;FINISHED ?
        146253     172     003     007
     14 146256                                BGT     MULT.A          ;LOOP UNTIL DONE
        146256     056     356
     15 146260                                RTS                     ;FINISHED - RESULT <B,A>
        146260     071
     16                                     ;
```

$G COMMAND

```
      1                                        .SBTTL  $G COMMAND
      2                                        ;
      3 146261                           .$G:  LDX     GOSUB           ;GET JUMP LOCATION
        146261     376     002     370
      4 146264                                 STX     SVGSUB          ;SAVE FOR RESTORATION
        146264     377     003     016
      5 146267                                 LDX     #,.$GA          ;NEXT ENTRY POINT
        146267     316     314     306
      6 146272                                 STX     GOSUB
        146272     377     002     370
      7 146275                                 SETBIT  LSTAT,200       ;ENABLE CHARACTER MODE
        146275     266     003     022
        146300     212     200
        146302     267     003     022
      8 146305                                 RTS                     ;FINISHED
        146305     071
      9                                        ;
     10 146306                           .$GA: LDX     SVGSUB          ;RESTORE JUMP ADDRESS
        146306     376     003     016
     11 146311                                 STX     GOSUB
        146311     377     002     370
     12 146314                                 LDA B   CHARG           ;GET CHARACTER
        146314     366     002     372
     13 146317                                 SUB B   #,'0            ;MAKE BCD
        146317     300     060
     14 146321                                 BCS     .$GZ            ;BAD CHARACTER - DONE
        146321     045     073
     15 146323                                 CMP B   #,9.
        146323     301     011
     16 146325                                 BHI     .$GZ            ;BAD CHARACTER - DONE
        146325     042     067
     17 146327                                 PSH B                   ;SAVE CHARACTER
        146327     067
     18 146330                                 JSR     .$U             ;RAISE PEN
        146330     275     310     356
```

$G COMMAND

```
     30 146367                                JSR     CHRSCX          ;GO DO PLOTTING
        146367     275     313     025
     31 146372                                LDA A   SVSTAT          ;GET OLD STATUS
        146372     266     003     020
     32 146375                                STA A   RSTAT0          ;RESTORE MODE
        146375     267     003     021
     33 146400                                LDX     YTABLE+26       ;RESTORE YSTEP
        146400     376     002     226
     34 146403                                STX     YTABLE+4
        146403     377     002     204
     35 146406                                LDX     XTABLE+26       ;RESTORE XSTEP
        146406     376     002     256
     36 146411                                STX     XTABLE+4
        146411     377     002     234
     37 146414                                SEC                     ;CHARACTER USED
        146414     015
     38 146415                                RTS                     ;FINISHFED
        146415     071
     39 146416                          .$GZ: CLC                     ;CHARACTER NOT USED
        146416     014
     40 146417                                RTS                     ;FINISHED
        146417     071
     41                                       ;
```

CHARACTER TABLES

```
      1                                  .SBTTL  CHARACTER TABLES
      2                                  ;
      3         150000                   .=150000
      4         150000                   CHTBLE  =.              ;START OF CHARACTER VECTOR TABLE
      5                                  ;
      6                                  .IRP    X,<CH0,CH1,CH2,CH3,CH4,CH5,CH6,CH7,CH8,CH9>
      7                                  FDB     X
      8                                  .ENDM
        150000     321     161
        150002     321     175
        150004     321     215
        150006     321     230
        150010     321     251
        150012     321     266
        150014     321     306
        150016     321     323
        150020     321     342
        150022     321     353
      9                                  ;
     10                                  .IRP    X,<CH10,CH11,CH12,CH13,CH14,CH15,CH16,CH17,CH18,C
H19>
     11                                  FDB     X
     12                                  .ENDM
        150024     321     367
        150026     322     003
        150030     322     017
        150032     322     030
        150034     322     053
        150036     322     070
        150040     322     105
        150042     322     122
        150044     322     137
        150046     322     151
     13                                  ;
     14                                  .IRP    X,<CH20,CH21,CH22,CH23,CH24,CH25,CH26,CH27,CH28,C
H29>
```

CHARACTER TABLES

```
         150112     323     124
         150114     323     150
         150116     323     170
     21                                  ;
     22                                  .IRP    X,<CH40,CH41,CH42,CH43,CH44,CH45,CH46,CH47,CH48,C
H49>
     23                                  FDB     X
     24                                  .ENDM
         150120     323     200
         150122     323     211
         150124     323     222
         150126     323     246
         150130     323     260
         150132     323     270
         150134     323     276
         150136     323     304
         150140     323     312
         150142     323     330
     25                                  ;
     26                                  .IRP    X,<CH50,CH51,CH52,CH53,CH54,CH55,CH56,CH57,CH58,C
H59>
     27                                  FDB     X
     28                                  .ENDM
         150144     323     342
         150146     323     360
         150150     324     001
         150152     324     012
         150154     324     030
         150156     324     047
         150160     324     061
         150162     324     105
         150164     324     124
         150166     324     136
     29                                  ;
     30                                  .IRP    X,<CH60,CH61,CH62,CH63,CH64,CH65,CH66,CH67,CH68,C
H69>
     31                                  FDB     X
```

CHARACTER TABLES

```
        150234     325     114
        150236     325     127
     37                                 ;
     38                                 .IRP    X,<CH80,CH81,CH82,CH83,CH84,CH85,CH86,CH87,CH88,C
H89>
     39                                 FDB     X
     40                                 .ENDM
        150240     325     144
        150242     325     160
        150244     325     200
        150246     325     216
        150250     325     236
        150252     325     250
        150254     325     264
        150256     325     277
        150260     325     316
        150262     325     334
     41                                 ;
     42                                 .IRP    X,<CH90,CH91,CH92,CH93,CH94,CH95,CH96,CH97,CH98,C
H99>
     43                                 FDB     X
     44                                 .ENDM
        150264     325     351
        150266     325     364
        150270     325     376
        150272     326     004
        150274     326     016
        150276     326     026
        150300     326     034
        150302     326     044
        150304     326     060
        150306     326     077
     45                                 ;
     46                                 .IRP    X,<CH100,CH101,CH102,CH103,CH104,CH105,CH106,CH10
7,CH108,CH109>
     47                                 FDB     X
     48                                 .ENDM
```

CHARACTER TABLES

```
         150356     327     071
     53                                 ;
     54                                 .IRP    X,<CH120,CH121,CH122,CH123,CH124,CH125,CH126,CH12
7>
     55                                 FDB     X
     56                                 .ENDM
         150360     327     110
         150362     327     122
         150364     327     140
         150366     327     150
         150370     327     166
         150372     327     200
         150374     327     216
         150376     327     230
     57                                 ;
     58                                 .IRP    X,<GCH0,GCH1,GCH2,GCH3,GCH4,GCH5,GCH6,GCH7,GCH8,G
CH9>
     59                                 FDB     X
     60                                 .ENDM
         150400     321     024
         150402     321     035
         150404     321     046
         150406     321     057
         150410     321     070
         150412     321     105
         150414     321     121
         150416     321     131
         150420     321     141
         150422     321     151
     61                                 ;
```

CHARACTER TABLES

```
      1          000010                          DWN     =       10
      2          000200                          UP      =       200
      3          000210                          END     =       210
      4                                          ;
      5                  .NLIST  BIN
      6 150424  GCH0:    .BYTE   4,DWN,254,54,44,244,UP,14,END
      7 150435  GCH1:    .BYTE   DWN,3,16,3,260,140,260,UP,END
      8 150446  GCH2:    .BYTE   273,DWN,140,6,340,16,UP,63,END
      9 150457  GCH3:    .BYTE   3,DWN,273,73,63,263,UP,13,END
     10 150470  GCH4:    .BYTE   261,DWN,42,40,52,12,252,240,242,2,UP,71,END
     11 150505  GCH5:    .BYTE   DWN,3,220,40,220,16,220,40,220,3,UP,END
     12 150521  GCH6:    .BYTE   60,DWN,343,16,143,UP,260,END
     13 150531  GCH7:    .BYTE   260,DWN,153,6,353,UP,60,END
     14 150541  GCH8:    .BYTE   13,DWN,66,340,76,UP,3,END
     15 150551  GCH9:    .BYTE   3,DWN,76,340,66,UP,13,END
     16 150561  CH0:     .BYTE   160,DWN,304,220,231,12,31,20,104,UP,34,END
     17 150575  CH1:     .BYTE   33,DWN,21,7,60,31,231,260,60,31,11,231,260,UP,140,END
     18 150615  CH2:     .BYTE   24,DWN,20,31,16,220,2,125,UP,34,END
     19 150630  CH3:     .BYTE   107,DWN,221,220,231,12,73,11,231,220,221,1,21,20,UP,133,END
     20 150651  CH4:     .BYTE   100,DWN,220,242,1,60,260,1,42,20,UP,116,END
     21 150666  CH5:     .BYTE   60,DWN,20,21,221,240,221,1,104,11,240,221,UP,137,11,END
     22 150706  CH6:     .BYTE   23,DWN,21,31,13,3,21,20,31,16,UP,43,END
     23 150723  CH7:     .BYTE   22,DWN,4,42,20,52,14,252,220,242,2,120,UP,54,END
     24 150742  CH8:     .BYTE   25,DWN,14,31,20,21,UP,111,END
     25 150753  CH9:     .BYTE   20,DWN,6,13,20,42,252,73,21,UP,51,END
     26 150767  CH10:    .BYTE   27,1,DWN,52,14,252,42,20,52,UP,40,END
     27 151003  CH11:    .BYTE   33,DWN,7,14,40,21,3,13,31,UP,60,END
     28 151017  CH12:    .BYTE   25,DWN,20,15,104,1,UP,55,END
     29 151030  CH13:    .BYTE   127,DWN,220,1,11,220,231,31,20,220,252
     30 151043          .BYTE   31,60,31,231,220,UP,100,END
     31 151053  CH14:    .BYTE   22,DWN,1,42,20,52,11,252,220,242,UP,172,END
     32 151070  CH15:    .BYTE   24,DWN,21,20,15,5,40,15,5,40,UP,35,END
     33 151105  CH16:    .BYTE   33,DWN,6,42,20,52,11,252,220,242,UP,172,END
     34 151122  CH17:    .BYTE   105,DWN,52,11,252,220,242,1,42,100,UP,35,END
     35 151137  CH18:    .BYTE   24,DWN,21,40,15,5,60,UP,35,END
```

CHARACTER TABLES

```
    58 151537          .BYTE   16,DWN,231,221,21,31,UP,31,END
    59 151550  CH38:    .BYTE   163,DWN,273,240,221,1,104,1,221,240,231,11,156,UP,20,END
    60 151570  CH39:    .BYTE   26,DWN,21,1,UP,157,11,END
    61 151600  CH40:    .BYTE   100,DWN,242,4,42,UP,117,11,END
    62 151611  CH41:    .BYTE   100,DWN,42,4,242,UP,117,11,END
    63 151622  CH42:    .BYTE   104,DWN,60,260,63,273,3,13,263,73
    64 151634          .BYTE   260,60,273,63,13,3,73,UP,31,END
    65 151646  CH43:    .BYTE   24,DWN,140,UP,263,DWN,16,UP,111,END
    66 151660  CH44:    .BYTE   40,DWN,20,12,231,UP,143,END
    67 151670  CH45:    .BYTE   24,DWN,140,UP,34,END
    68 151676  CH46:    .BYTE   60,DWN,0,UP,120,END
    69 151704  CH47:    .BYTE   21,DWN,146,UP,37,END
    70 151712  CH48:    .BYTE   21,DWN,6,21,100,31,356,31,100,21,6,UP,37,END
    71 151730  CH49:    .BYTE   67,DWN,21,17,11,240,100,UP,40,END
    72 151742  CH50:    .BYTE   27,DWN,21,100,31,11,252,220,273,11,140,UP,20,END
    73 151760  CH51:    .BYTE   27,DWN,21,100,31,12,231,260,60,31,12,231,300,221,UP,171,END
    74 152001  CH52:    .BYTE   140,DWN,7,1,335,140,UP,33,END
    75 152012  CH53:    .BYTE   21,DWN,31,100,21,2,221,320,4,140,UP,37,11,END
    76 152030  CH54:    .BYTE   167,DWN,221,300,231,16,31,100,21,2,221,320,UP,174,END
    77 152047  CH55:    .BYTE   27,DWN,1,140,11,314,13,UP,120,END
    78 152061  CH56:    .BYTE   44,DWN,231,12,31,100,21,2,221,300,221
    79 152074          .BYTE   2,21,100,31,12,231,UP,54,END
    80 152105  CH57:    .BYTE   21,DWN,31,100,21,6,221,300,231,12,31,120,UP,34,END
    81 152124  CH58:    .BYTE   46,DWN,0,UP,14,DWN,0,UP,152,END
    82 152136  CH59:    .BYTE   45,DWN,0,UP,15,DWN,20,12,231,UP,143,END
    83 152152  CH60:    .BYTE   120,DWN,304,104,UP,77,11,END
    84 152162  CH61:    .BYTE   25,DWN,140,UP,352,DWN,140,UP,33,END
    85 152174  CH62:    .BYTE   60,DWN,104,304,UP,137,11,END
    86 152204  CH63:    .BYTE   26,DWN,1,21,100,31,11,273,11,UP,12,DWN,0,UP,100,END
    87 152224  CH64:    .BYTE   140,DWN,300,221,6,21,100,31,13,231,260
    88 152237          .BYTE   1,21,20,12,UP,73,END
    89 152246  CH65:    .BYTE   20,DWN,6,42,40,52,12,340,140,14,UP,20,END
    90 152263  CH66:    .BYTE   20,DWN,120,21,2,221,300,100,21,2,221
    91 152276          .BYTE   320,20,17,11,UP,140,END
    92 152305  CH67:    .BYTE   167,DWN,221,260,252,14,52,60,21,UP,31,END
```

CHARACTER TABLES

```
   115 152751  CH90:   .BYTE   24,4,DWN,140,11,356,11,140,UP,20,END
   116 152764  CH91:   .BYTE   144,4,DWN,260,14,14,60,UP,40,END
   117 152776  CH92:   .BYTE   27,DWN,156,UP,31,END
   118 153004  CH93:   .BYTE   44,4,DWN,60,17,11,260,UP,140,END
   119 153016  CH94:   .BYTE   27,DWN,21,100,31,UP,37,END
   120 153026  CH95:   .BYTE   20,DWN,140,UP,20,END
   121 153034  CH96:   .BYTE   144,4,DWN,11,31,UP,36,END
   122 153044  CH97:   .BYTE   142,DWN,252,240,221,2,42,60,15,UP,40,END
   123 153060  CH98:   .BYTE   24,4,DWN,16,52,40,21,3,221,240,252,13,UP,160,END
   124 153077  CH99:   .BYTE   144,DWN,221,260,231,13,31,60,21,UP,51,END
   125 153113  CH100:  .BYTE   144,4,DWN,16,252,240,221,3,21,40,52,13,UP,40,END
   126 153132  CH101:  .BYTE   22,DWN,120,2,221,260,231,13,31,100,UP,40,END
   127 153147  CH102:  .BYTE   60,DWN,4,240,100,240,3,21,20,31,UP,57,END
   128 153164  CH103:  .BYTE   32,DWN,31,60,21,5,242,240,231,13,31,40,42,3,UP,55,END
   129 153205  CH104:  .BYTE   24,4,DWN,15,13,3,42,20,52,13,UP,40,END
   130 153222  CH105:  .BYTE   107,DWN,0,UP,232,DWN,20,15,UP,100,END
   131 153235  CH106:  .BYTE   32,DWN,31,60,21,7,UP,55,END
   132 153246  CH107:  .BYTE   20,DWN,3,40,240,5,UP,113,DWN,252,73,UP,40,END
   133 153264  CH108:  .BYTE   64,4,DWN,14,14,UP,120,END
   134 153274  CH109:  .BYTE   20,DWN,5,40,31,14,4,21,20,31,14,UP,20,END
   135 153312  CH110:  .BYTE   25,DWN,15,3,42,40,31,14,UP,40,END
   136 153325  CH111:  .BYTE   21,DWN,3,21,60,31,13,231,260,221,UP,171,END
   137 153342  CH112:  .BYTE   25,DWN,13,52,40,21,3,221,240,252,16,UP,163,END
   138 153360  CH113:  .BYTE   145,DWN,13,252,240,221,3,21,40,52,16,UP,43,END
   139 153376  CH114:  .BYTE   25,DWN,15,3,42,40,31,UP,54,END
   140 153410  CH115:  .BYTE   21,DWN,31,60,21,221,220,221,220,221,21,60,31,UP,54,END
   141 153430  CH116:  .BYTE   67,DWN,12,,240,100,240,14,31,20,21,UP,51,END
   142 153446  CH117:  .BYTE   25,DWN,14,31,60,21,4,UP,55,END
   143 153460  CH118:  .BYTE   25,DWN,13,52,42,3,UP,75,END
   144 153471  CH119:  .BYTE   25,DWN,14,31,20,21,3,13,31,20,21,4,UP,35,END
   145 153510  CH120:  .BYTE   20,DWN,125,UP,320,DWN,135,UP,40,END
   146 153522  CH121:  .BYTE   25,DWN,14,31,40,42,3,17,231,260,221,UP,162,END
   147 153540  CH122:  .BYTE   25,DWN,120,335,120,UP,40,END
   148 153550  CH123:  .BYTE   144,4,DWN,240,231,12,231,31,12,31,40,UP,40,END
   149 153566  CH124:  .BYTE   103,DWN,12,UP,7,DWN,12,UP,116,END
```

Symbol table

```
ABSD    140607          CH111   153325          CH49    151730          CMDS.A  143510          GRPCLR  1
40756

AI      140515          CH112   153342          CH5     150666          CMDS.B  143522          GRPIN   1
40115

BCDLP   144010          CH113   153360          CH50    151742          CMDS.C  143527          GRPLOT  1
40003

BCDTBL  144054          CH114   153376          CH51    151760          CMDS.D  143531          GRPOUT  1
40115

BCD4BN  143751          CH115   153410          CH52    152001          CMDTBL  143533          GTEMPA  0
01306

BCD4.A  144045          CH116   153430          CH53    152012          CMD.UC  143472          GTEMPB  0
01310

BPTBLE  140305          CH117   153446          CH54    152030          CNTROL  142571          HA    = 0
00040

BT      001324          CH118   153460          CH55    152047          COLADD  001347          HADD    0
01332

BXADC   140234          CH119   153471          CH56    152061          COLRST  001351          HB    = 0
00041

CADR    001335          CH12    151017          CH57    152105          COLUPD  001345          HEREIS  1
40104

CARHA = 000004          CH120   153510          CH58    152124          CSL   = 000010          HLDINT  1
44543

CARLB = 000005          CH121   153522          CH59    152136          CTLINE= 020000          IRQ1  = 0
07740

CHARG   001372          CH122   153540          CH6     150706          CWORD   001326          IRQ10 = 0
07762

CHARPL= 000014          CH123   153550          CH60    152152          CWRITE  141737          IRQ11 = 0
07764

CHEK1   141341          CH124   153566          CH61    152162          CXWRT   142132          IRQ12 = 0
07766

CHEK2   141437          CH125   153600          CH62    152174          DARHA = 000000          IRQ13 = 0
07770

CHEK3   141502          CH126   153616          CH63    152204          DARLB = 000001          IRQ14 = 0
07772

CHEK4   141545          CH127   153630          CH64    152224          DIVCNT  001314          IRQ15 = 0
07774

CHEK5   141651          CH13    151030          CH65    152246          DIVIDE  141132          IRQ16 = 0
07776

CHK     001302          CH14    151053          CH66    152263          DND     001315          IRQ2  = 0
07742

CHPNTR  001413          CH15    151070          CH67    152305          DSPCTL= 000015          IRQ3  = 0
07744

CHRAM = 010000          CH16    151105          CH68    152321          DSR     001317          IRQ4  = 0
07746

CHRFLG= 000105          CH17    151122          CH69    152336          DWN   = 000010          IRQ5  = 0
07750
```

Symbol table

```
LSTCNT  001364          PCRA12= 000056          PROC.F  145030          VSETC   140640          .$J     1
44447

LSTENT  142731          PCRA13= 000062          PROW    001415          WDFBA = 000020          .$K   = 1
40756

LSTGRP  142746          PCRA2 = 000006          PTABLE  001260          WDFBB = 000021          .$L     1
44266

LT      001325          PCRA3 = 000012          QT      001321          WDTBA = 000024          .$LA    1
44302

MULTP   146226          PCRA4 = 000016          RADD    001330          WDTBB = 000025          .$LB    1
44315

MULT.A  146236          PCRA5 = 000022          RDADD   001333          WTBYTE  001327          .$LC    1
44326

MULT.B  146253          PCRA6 = 000026          READ    142662          XGET    145761          .$LD    1
44331

NCNTR   001403          PCRA7 = 000032          ROWADD  001341          XGET.A  145773          .$M     1
44221

NCXVAL  140211          PCRA8 = 000036          ROWRST  001343          XGET.Z  146007          .$MA    1
44233

NCYVAL  140315          PCRA9 = 000042          ROWUPD  001337          XTABLE  001230          .$MB    1
44246

NMI1  = 007732          PCRB1 = 000003          RSTAT0  001421          XYBIS   140453          .$MC    1
44257

NMI2  = 007734          PCRB10= 000047          RSTRAM  142404          XYFNCT  141212          .$MD    1
44262

NMI3  = 007736          PCRB11= 000053          SAVEX   001304          XYSIZE  145710          .$O     1
44167

NOADD   144024          PCRB12= 000057          SCHFLG  001374          XYSI.Z  145745          .$OA    1
44171

NPNTR   001404          PCRB13= 000063          SELECT  142471          XYTBL   145747          .$OB    1
44204

NPRADA= 000050          PCRB2 = 000007          SETSGN  140670          YGET    145771          .$OC    1
44212

NPRADB= 000051          PCRB3 = 000013          STPFLG  001375          YTABLE  001200          .$OD    1
44215

NPRCSR= 000045          PCRB4 = 000017          SVGSUB  001416          .$A     144422          .$P     1
44400

NPRDFA= 000060          PCRB5 = 000023          SVSTAT  001420          .$B     144432          .$R     1
44104

NPRDFB= 000061          PCRB6 = 000027          TDF11 = 000030          .$C     145301          .$RB    1
44136

NPRDTA= 000054          PCRB7 = 000033          TDT11 = 000031          .$D     144335          .$RC    1
44157

NPRDTB= 000055          PCRB8 = 000037          T0      001406          .$F     145074          .$RD    1
44160

NSIGN   001376          PCRB9 = 000043          T1      001407          .$FA    145102          .$S     1
44411
```

```
NXYPLT  140440          PGTYST  146114          T3      001411          .$FC    145147          .$V     1
44367

ODDSCN= 054000          PHOTO = 000044          UP    = 000200          .$FD    145165          .$W     1
44556

OPR     001300          PLOTVC  141024          UPDATE  140723          .$FE    145203          .$X     1
44557

ORGSTP  145057          PLOTXY  140413          UPDFLG= 000106          .$FF    145231          .$XA    1
44600

ORGS.A  145063          PNTSKP  001412          VALSTP  145041          .$FG    145257          .$XB    1
44615

ORG.X   145047          PROC    144677          VAL.X   145031          .$FZ    145271          .$Y     1
44627

ORG.Y   145054          PROC.A  144714          VAL.Y   145036          .$G     146261          .$YA    1
44650

PCHAR   001373          PROC.B  144726          VCNTR   001312          .$GA    146306          .$YB    1
44665

PCRA1 = 000002          PROC.C  144740          VECTDN  141413          .$GZ    146416          .$Z     1
44556

PCRA10= 000046          PROC.D  144744          VECTOR  141333          .$H     144453          ...A  = 0
00000

PCRA11= 000052          PROC.E  144774          VECTXY  140427          .$I     144441


. ABS.  153677    000   (RW,I,GBL,ABS,OVR)

        000000    001   (RW,I,LCL,REL,CON)

Errors detected:  0


*** Assembler statistics



Work  file  reads: 0

Work  file writes: 0

Size of work file: 14952 Words  ( 59 Pages)

Size of core pool: 18176 Words  ( 71 Pages)

Operating  system: RT-11


Elapsed time: 00:00:05.08

GRP[40],GRP=M6800,GRAPH1,GRAPH2,GRAPH3,GRAPH4
```