

```

5  **      HBUG - HEATH/WINTER TERMINAL DEBUGGER.
6  *
7  *
8  *      J.G. LETWIN, 10/01/76, FOR *WINTER* CORPORATION
9  *
10 *      COPYRIGHT 10/76, WINTER CORPORATION,
11 *      LAFAYETTE, INDIANA.
12 *      G. Chandler,      78.10
13 *      79/12      ---.05---
14 *

```

```

16 **      ASSEMBLY CONFIGURATION.
17

```

```

19

```

```

21 **      MACHINE INSTRUCTIONS.
22
000.303 23 MI.JMP EQU 11000011B JMP
000.072 24 MI.LDA EQU 00111010B LDA
000.327 25 MI.BKP EQU 11010111B RST 2 (BREAKPOINT)
26
27 **      CHANNEL USED FOR LOAD/DUMP
28
000.005 29 CN.LD EQU 5 CHANNEL 5
30
31
000.000 32 XTEXT ASCII

```

```

34X **      ASCII CHARACTER EQUIVALENCES.
35X
000.015 36X CR EQU 13 CARRIAGE RETURN
000.012 37X LF EQU 10 LINE FEED
000.200 38X NULL EQU 200Q PAD CHARACTER
000.000 39X NUL2 EQU 0
000.007 40X BELL EQU 7 BELL CHARACTER
000.177 41X RUBOUT EQU 177Q
000.010 42X BKSP EQU 10Q CTL-H
000.026 43X C.SYN EQU 26Q SYNC
000.002 44X C.STX EQU 2 STX
000.047 45X QUOTE EQU 47Q
000.011 46X TAB EQU 11Q
000.033 47X ESC EQU 33Q
000.012 48X NL EQU 12Q NEW LINE (HDOS SYSTEMS)
000.212 49X ENL EQU NL+200Q NL + END-OF-LINE-FLAG
000.014 50X FF EQU 14Q FORM FEED
000.001 51X CTLA EQU 01Q CTL-A
000.002 52X CTLB EQU 02Q CTL-B

```

```
000.003      53X CTLC EQU 030      CTL-C
000.004      54X CTLD EQU 040      CTL-D
000.017      55X CTLO EQU 170      CTL-O
000.020      56X CTLP EQU 200      CTL-P
000.021      57X CTLQ EQU 210      CTL-Q
000.023      58X CTLS EQU 230      CTL-S
000.032      59X CTLZ EQU 320      CTL-Z
000.000      60      XTEXT HOSDEF
```

62X \*\* HOSDEF - DEFINE HOS PARAMETER.

63X \*

64X

65X

```
000.026      66X VERS EQU 1*16+6      VERSION 1.6
```

67X

```
000.377      68X SYSCALL EQU 3770      SYSCALL INSTRUCTION
```

69X

70X

```
000.000      71X      ORG 0
```

72X

73X \* RESIDENT FUNCTIONS

74X

```
000.000      75X .EXIT DS 1      EXIT (MUST BE FIRST)
```

```
000.001      76X .SCIN DS 1      SCIN
```

```
000.002      77X .SCOUT DS 1      SCOUT
```

```
000.003      78X .PRINT DS 1      PRINT
```

```
000.004      79X .READ DS 1      READ
```

```
000.005      80X .WRITE DS 1      WRITE
```

```
000.006      81X .CONSL DS 1      SET/CLEAR CONSOLE OPTIONS
```

```
000.007      82X .CLRCD DS 1      CLEAR CONSOLE BUFFER
```

```
000.010      83X .LOADO DS 1      LOAD AN OVERLAY
```

```
000.011      84X .VERS DS 1      RETURN HDOS VERSION NUMBER
```

```
000.012      85X .SYSRES DS 1      PRECEDING FUNCTIONS ARE RESIDENT
```

86X

87X

88X \* \*HDOSOVLO.SYS\* FUNCTIONS

89X

```
000.040      90X      ORG 40A
```

91X

```
000.040      92X .LINK DS 1      LINK (MUST BE FIRST)
```

```
000.041      93X .CTLC DS 1      CTL-C
```

```
000.042      94X .OPENR DS 1      OPENR
```

```
000.043      95X .OPENW DS 1      OPENW
```

```
000.044      96X .OPENU DS 1      OPENU
```

```
000.045      97X .OPENC DS 1      OPENC
```

```
000.046      98X .CLOSE DS 1      CLOSE
```

```
000.047      99X .POSIT DS 1      POSITION
```

```
000.050      100X .DELET DS 1      DELETE
```

```
000.051      101X .RENAM DS 1      RENAME
```

```
000.052      102X .SETTP DS 1      SETTOP
```

```
000.053      103X .DECODE DS 1      NAME DECODE
```

```
000.054      104X .NAME DS 1      GET FILE NAME FROM CHANNEL
```

```
000.055      105X .CLEAR DS 1      CLEAR CHAN
```

```
000.056      106X .CLEARA DS 1      CLEAR ALL CHANS
```

## ASSEMBLY CONSTANTS.

HOSDEF

15:28:48 16-MAY-80

000.057	107X	.ERROR	DS	1	LOOKUP ERROR
000.060	108X	.CHFLG	DS	1	CHANGE FLAGS
000.061	109X	.DISMT	DS	1	FLAG SYSTEM DISK DISMOUNTED
000.062	110X	.LOADD	DS	1	LOAD DEVICE DRIVER
	111X				
	112X				
	113X	*			*HDSOVL1.SYS* FUNCTIONS
	114X				
000.200	115X		DS	2000	
	116X				
000.200	117X	.MOUNT	DS	1	MOUNT (MUST BE FIRST)
000.201	118X	.DMOUN	DS	1	DISMOUNT
000.202	119X	.MONMS	DS	1	MOUNT/NO MESSAGE
000.203	120X	.DMNMS	DS	1	DISMOUNT/NO MESSAGE
000.204	121X	.RESET	DS	1	RESET = DISMOUNT/MOUNT OF UNIT
000.205	122		XTEXT	MTR	

125X \*\* MTR - PAM/8 EQUIVALENCES.

126X \*  
127X \* THIS DECK CONTAINS SYMBOLIC DEFINITIONS USED TO  
128X \* MAKE USE OF THE PAM/8 CODE AND CONTROL BYTES.

130X \*\* IO PORTS

	131X			
000.360	132X	IP.PAD	EQU	360Q PAD INPUT PORT
000.360	133X	OP.CTL	EQU	360Q CONTROL OUTPUT PORT
000.360	134X	OP.DIG	EQU	360Q DIGIT SELECT OUTPUT PORT
000.361	135X	OP.SEG	EQU	361Q SEGMENT SELECT OUTPUT PORT

137X \*\* FRONT PANEL CONTROL BITS.

	138X			
000.020	139X	CB.SSI	EQU	00010000B SINGLE STEP INTERRUPT
000.040	140X	CB.MTL	EQU	00100000B MONITOR LIGHT
000.100	141X	CB.CLI	EQU	01000000B CLOCK INTERRUPT ENABLE
000.200	142X	CB.SPK	EQU	10000000B SPEAKER ENABLE

144X \*\* MONITOR MODE FLAGS.

	145X			
000.000	146X	DM.MR	EQU	0 MEMORY READ
000.001	147X	DM.MW	EQU	1 MEMORY WRITE
000.002	148X	DM.RR	EQU	2 REGISTER READ
000.003	149X	DM.RW	EQU	3 REGISTER WRITE

151X \*\* USER OPTION BITS.

	152X	*		
	153X	*		THESE BITS ARE SET IN CELL .MFLAG.
	154X			
000.200	155X	UD.HLT	EQU	10000000B DISABLE HALT PROCESSING
000.100	156X	UD.NFR	EQU	CB.CLI NO REFRESH OF FRONT PANEL
000.002	157X	UD.DDU	EQU	00000010B DISABLE DISPLAY UPDATE
000.001	158X	UD.CLK	EQU	00000001B ALLOW PRIVATE INTERRUPT PROCESSING

160X \*\* MONITOR IDENTIFICATION FLAGS

	161X	*		
	162X	*		THESE BYTES IDENTIFY THE ROM MONITOR.
	163X	*		THEY ARE THE VARIOUS VALUES OF LOCATION .IDENT
	164X			
000.021	165X	M.PAMB	EQU	021Q 'LXI' INSTRUCTION AT 000.000 IN PAM-8
000.303	166X	M.FOX	EQU	303Q 'JMP' INSTRUCTION AT 000.000 IN FOX ROM

RAM/8 EQUIVALENCES.

ENTRY

15:28:52 16-MAY-80

## 168X \*\* ROUTINE ENTRY POINTS.

169X \*

170X

000.000	171X .IDENT	EQU	0000A	IDENTIFICATION LOCATION
000.053	172X .DLY	EQU	0053A	DELAY
001.267	173X .LOAD	EQU	1267A	TAPE LOAD
001.374	174X .DUMP	EQU	1374A	TAPE DUMP
002.136	175X .ALARM	EQU	2136A	ALARM ROUTINE
002.140	176X .HORN	EQU	2140A	HORN
002.172	177X .CTC	EQU	2172A	CHECK TAPE CHECKSUM
002.205	178X .TPERR	EQU	2205A	TAPE ERROR ROUTINE
002.264	179X .PCHL	EQU	2264A	PCHL INSTRUCTION
002.265	180X .SRS	EQU	2265A	SCAN RECORD START
002.325	181X .RNP	EQU	2325A	READ NEXT PAIR
002.331	182X .RNB	EQU	2331A	READ NEXT BYTE
002.347	183X .CRC	EQU	2347A	CRC-16 CALCULATOR
003.017	184X .WNP	EQU	3017A	WRITE NEXT PAIR
003.024	185X .WNB	EQU	3024A	WRITE NEXT BYTE
003.122	186X .DOD	EQU	3122A	DECODE FOR OCTAL DISPLAY
003.260	187X .RCK	EQU	3260A	READ CONSOLE KEYSET
003.356	188X .DODA	EQU	3356A	SEGMENT CODE TABLE

## 190X \*\* RAM CELLS USED BY H8MTR.

191X \*

192X

040.000	193X .START	EQU	40000A	START DUMP ADDRESS
040.002	194X .IOWRK	EQU	40002A	IN OR OUT INSTRUCTION
040.005	195X .REGI	EQU	40005A	DISPLAYED REGISTER INDEX
040.006	196X .DSPROT	EQU	40006A	PERIOD FLAG BYTE
040.007	197X .DSPMOD	EQU	40007A	DISPLAY MODE
040.010	198X .MFLAG	EQU	40010A	USER OPTION BYTE
040.011	199X .CTLFLG	EQU	40011A	PANEL CONTROL BYTE
040.013	200X .ALEDS	EQU	40013A	ABUSS LEDS
040.021	201X .DLEDS	EQU	40021A	DBUSS LEDS
040.024	202X .ABUSS	EQU	40024A	ABUSS REGISTER
040.027	203X .CRCSUM	EQU	40027A	CRCSUM WORD
040.031	204X .TPERRX	EQU	40031A	TAPE ERROR EXIT VECTOR
040.033	205X .TICCNT	EQU	40033A	CLOCK TICK COUNTER
040.035	206X .REGPTR	EQU	40035A	REGISTER POINTER
040.037	207X .UIVEC	EQU	40037A	USER INTERRUPT VECTORS
000.205	208	XTEXT	H0SEQU	

## 210X \*\* HDOS SYSTEM EQUIVALENCES.

211X \*

212X

024.000	213X S.GRT0	EQU	24000A	SYSTEM AREA FOR GRT0
025.000	214X S.GRT1	EQU	25000A	SYSTEM AREA FOR GRT1
026.000	215X S.GRT2	EQU	26000A	SYSTEM AREA FOR GRT2
	216X			
030.000	217X ROMBOOT	EQU	30000A	ROM BOOT ENTRY
	218X			

040.100	219X	ORG	40100A	FREE SPACE FROM PAM-8
	220X			
040.100	221X	DS	8	JUMP TO SYSTEM EXIT
040.110	222X	D.CON	DS	16
040.130	223X	SYDD	EQU	*
040.130	224X	D.VEC	DS	24*3
040.240	225X	D.RAM	DS	31
040.277	226X	S.VAL	DS	36
040.343	227X	S.INT	DS	115
041.126	228X	DS	16	
041.146	229X	S.SOVR	DS	2
041.150	230X	DS	42200A-*	STACK OVERFLOW WARNING
001.032	231X	STACKL	EQU	*-S.SOVR
	232X			STACK SIZE
042.200	233X	STACK	EQU	*
042.200	234X	USERFWA	EQU	*
042.200	235	XTEXT	ESVAL	LWA+1 SYSTEM STACK
				USER FWA

237X \*\* S.VAL - SYSTEM VALUE DEFINITIONS.

238X \*

239X \* THESE VALUES ARE SET AND MAINTAINED BY THE SYSTEM.

240X \*

241X \* THE DECK HDOSEQU MUST BE MODIFIED WHEN THIS IS MODIFIED.

242X

243X

040.277	244X	ORG	S.VAL
---------	------	-----	-------

245X

040.277	246X	S.DATE	DS	9	SYSTEM DATE (IN ASCII)
---------	------	--------	----	---	------------------------

040.310	247X	S.DATC	DS	2	CODED DATE
---------	------	--------	----	---	------------

040.312	248X	S.TIME	DS	4	TIME FROM MIDNIGHT (IN TICS)
---------	------	--------	----	---	------------------------------

040.316	249X	S.HIMEM	DS	2	HARDWARE HIGH MEMORY ADDRESS+1
---------	------	---------	----	---	--------------------------------

250X

040.320	251X	S.SYSM	DS	2	FWA RESIDENT SYSTEM
---------	------	--------	----	---	---------------------

252X

040.322	253X	S.USRM	DS	2	LWA USER MEMORY
---------	------	--------	----	---	-----------------

254X

040.324	255X	S.OMAX	DS	2	MAX OVERLAY SIZE FOR SYSTEM
---------	------	--------	----	---	-----------------------------

256X

257X

258X \*\* THE FOLLOWING FIVE CELLS SHOULD BE MODIFIED/READ ONLY VIA THE .CONSL SYSCALL

259X

000.200	260X	CSL.ECH	EQU	10000000B	SUPPRESS ECHO
---------	------	---------	-----	-----------	---------------

000.002	261X	CSL.WRP	EQU	00000010B	WRAP LINES AT WIDTH
---------	------	---------	-----	-----------	---------------------

000.001	262X	CSL.CHR	EQU	00000001B	OPERATE IN CHARACTER MODE
---------	------	---------	-----	-----------	---------------------------

263X

000.000	264X	I.CSLMD	EQU	0	S.CSLMD IS FIRST BYTE
---------	------	---------	-----	---	-----------------------

040.326	265X	S.CSLMD	DS	1	CONSOLE MODE
---------	------	---------	----	---	--------------

266X

000.200	267X	CTP.BKS	EQU	10000000B	TERMINAL PROCESSES BACKSPACES
---------	------	---------	-----	-----------	-------------------------------

000.040	268X	CTP.MLI	EQU	00100000B	MAP LOWER CASE TO UPPER ON INPUT
---------	------	---------	-----	-----------	----------------------------------

000.020	269X	CTP.MLO	EQU	00010000B	MAP LOWER CASE TO UPPER ON OUTPUT
---------	------	---------	-----	-----------	-----------------------------------

000.010	270X	CTP.2SB	EQU	00001000B	TERMINAL NEEDS TWO STOP BITS
---------	------	---------	-----	-----------	------------------------------

000.002	271X	CTP.BKM	EQU	00000010B	MAP BKSP (UPON INPUT) TO RUBOUT
---------	------	---------	-----	-----------	---------------------------------

000.001	272X	CTP.TAB	EQU	00000001B	TERMINAL SUPPORTS TAB CHARACTERS
	273X				
000.001	274X	I.CONTY	EQU	1	S.CONTY IS 2ND BYTE
000.000	275X		ERRNZ	*-S.CSLMD-I.CONTY	
040.327	276X	S.CONTY	DS	1	CONSOLE TYPE FLAGS
000.002	277X	I.CUSOR	EQU	2	S.CUSOR IS 3RD BYTE
000.000	278X		ERRNZ	*-S.CSLMD-I.CUSOR	
040.330	279X	S.CUSOR	DS	1	CURRENT CURSOR POSITION
000.003	280X	I.CONWI	EQU	3	S.CONWI IS 4TH BYTE
000.000	281X		ERRNZ	*-S.CSLMD-I.CONWI	
040.331	282X	S.CONWI	DS	1	CONSOLE WIDTH
	283X				
000.001	284X	CO.FLG	EQU	00000001B	CTL-D FLAG
000.200	285X	CS.FLG	EQU	10000000B	CTL-S FLAG
	286X				
000.004	287X	I.CONFL	EQU	4	S.CONFL IS 5TH BYTE
000.000	288X		ERRNZ	*-S.CSLMD-I.CONFL	
040.332	289X	S.CONFL	DS	1	CONSOLE FLAGS
	290X				
040.333	291X	S.CAADR	DS	2	ADDRESS FOR ABORT PROCESSING (>256 IF VALID)
040.335	292X	S.CCTAB	DS	6	ADDR FOR CTL-A, CTL-B, CTL-C PROCESSING
040.343	293	XTEXT	ECDEF		

## 295X \*\* ERROR CODE DEFINITIONS.

	296X				
	297X		ORG	0	
000.000	298X		DS	1	NO ERROR #0
000.000	299X	EC.EOF	DS	1	END OF FILE
000.001	300X	EC.EOM	DS	1	END OF MEDIA
000.002	301X	EC.ILC	DS	1	ILLEGAL SYSCALL CODE
000.003	302X	EC.CNA	DS	1	CHANNEL NOT AVAILABLE
000.004	303X	EC.DNS	DS	1	DEVICE NOT SUITABLE
000.005	304X	EC.IDN	DS	1	ILLEGAL DEVICE NAME
000.006	305X	EC.IFN	DS	1	ILLEGAL FILE NAME
000.007	306X	EC.NRD	DS	1	NO ROOM FOR DEVICE DRIVER
000.010	307X	EC.FNO	DS	1	CHANNEL NOT OPEN
000.011	308X	EC.ILR	DS	1	ILLEGAL REQUEST
000.012	309X	EC.FUC	DS	1	FILE USAGE CONFLICT
000.013	310X	EC.FNF	DS	1	FILE NAME NOT FOUND
000.014	311X	EC.UND	DS	1	UNKNOWN DEVICE
000.015	312X	EC.ICN	DS	1	ILLEGAL CHANNEL NUMBER
000.016	313X	EC.DIF	DS	1	DIRECTORY FULL
000.017	314X	EC.IFC	DS	1	ILLEGAL FILE CONTENTS
000.020	315X	EC.NEM	DS	1	NOT ENOUGH MEMORY
000.021	316X	EC.RF	DS	1	READ FAILURE
000.022	317X	EC.WF	DS	1	WRITE FAILURE
000.023	318X	EC.WPV	DS	1	WRITE PROTECTION VIOLATION
000.024	319X	EC.WP	DS	1	DISK WRITE PROTECTED
000.025	320X	EC.FAP	DS	1	FILE ALREADY PRESENT
000.026	321X	EC.DDA	DS	1	DEVICE DRIVER ABORT
000.027	322X	EC.FL	DS	1	FILE LOCKED
000.030	323X	EC.FAD	DS	1	FILE ALREADY OPEN
000.031	324X	EC.IS	DS	1	ILLEGAL SWITCH
000.032	325X	EC.UUN	DS	1	UNKNOWN UNIT NUMBER
000.033					

000.034	326X	EC.FNR	DS	1	FILE NAME REQUIRED
000.035	327X	EC.DIW	DS	1	DEVICE IS NOT WRITABLE (OR WRITE LOCKED)
000.036	328X	EC.UNA	DS	1	UNIT NOT AVAILABLE
000.037	329X	EC.ILV	DS	1	ILLEGAL VALUE
000.040	330X	EC.ILO	DS	1	ILLEGAL OPTION
000.041	331X	EC.VPM	DS	1	VOLUME PRESENTLY MOUNTED ON DEVICE
000.042	332X	EC.NVM	DS	1	NO VOLUME PRESENTLY MOUNTED
000.043	333X	EC.FOD	DS	1	FILE OPEN ON DEVICE
000.044	334X	EC.NPM	DS	1	NO PROVISIONS MADE FOR REMOUNTING MORE DISKS
000.045	335X	EC.DNI	DS	1	DISK NOT INITIALIZED
000.046	336X	EC.DNR	DS	1	DISK IS NOT READABLE
000.047	337X	EC.DSC	DS	1	DISK STRUCTURE IS CORRUPT
000.050	338X	EC.NCV	DS	1	NOT CORRECT VERSION OF HDOS
000.051	339X	EC.NOS	DS	1	NO OPERATING SYSTEM MOUNTED
000.052	340X	EC.IOI	DS	1	ILLEGAL OVERLAY INDEX
000.053	341X	EC.OTL	DS	1	OVERLAY TOO LARGE
000.054	342		XTEXT	FBDEF	

344X \*\* FILE BLOCK DEFINITIONS.

	345X				
000.000	346X	ORG	0		
000.000	347X	FB.CHA	DS	1	CHANNEL NUMBER
000.001	348X	FB.FLG	DS	1	FLAGS
000.002	349X	FB.FWA	DS	2	BUFFER FWA
000.004	350X	FB.PTR	DS	2	BUFFER POINTER
000.006	351X	FB.LIM	DS	2	LIMIT OF DATA IN BUFFER (READ OPERATIONS)
000.010	352X	FB.LWA	DS	2	LWA OF BUFFER
000.012	353X	FB.NAM	DS	4+8+4+1	NAME OF FILE
000.021	354X	FB.NAML	ERU	*-FB.NAM	
000.033	355X	FBENL	EQU	*	ENTRY LENGTH
000.033	356		XTEXT	FILDEF	

358X \*\* FILDEF - FILE TYPE DEFINITIONS.

	359X	*			
	360X	*	DB	377Q,FT.XXX	
	361X				
	362X				
000.000	363X	FT.ABS	ERU	0	ABSOLUTE BINARY
000.001	364X	FT.PIC	EQU	1	POSITION INDEPENDANT CODE
000.002	365X	FT.REL	ERU	2	RELOCATABLE CODE
000.003	366X	FT.BAC	EQU	3	COMPILED BASIC CODE
000.033	367		XTEXT	U8251	



## 8251 USART BIT DEFINITIONS.

15:29:08 16-MAY-80

```

370X **      8251 USART BIT DEFINITIONS.
371X *
372X
373X **      PORT ADDRESSES
374X
000.000      375X UDR      EQU      0      DATA REGISTER IS EVEN
000.001      376X USR      EQU      1      STATUS REGISTER IS NEXT
377X
000.372      378X SC.USART EQU      372Q      CONSOLE USART ADDRESS (IFF 8251)
379X
380X
381X **      MODE INSTRUCTION CONTROL BITS.
382X
000.100      383X UMI.1B EQU      01000000B      1 STOP BIT
000.200      384X UMI.HB EQU      10000000B      1 1/2 STOP BITS
000.300      385X UMI.2B EQU      11000000B      2 STOP BITS
000.040      386X UMI.PE EQU      00100000B      EVEN PARITY
000.020      387X UMI.PA EQU      00010000B      USE PARITY
000.000      388X UMI.L5 EQU      00000000B      5 BIT CHARACTERS
000.004      389X UMI.L6 EQU      00000100B      6 BIT CHARACTERS
000.010      390X UMI.L7 EQU      00001000B      7 BIT CHARACTERS
000.014      391X UMI.L8 EQU      00001100B      8 BIT CHARACTERS
000.001      392X UMI.1X EQU      00000001B      CLOCK X 1
000.002      393X UMI.16X EQU      00000010B      CLOCK X 16
000.003      394X UMI.64X EQU      00000011B      CLOCK X 64
395X
396X **      COMMAND INSTRUCTION BITS.
397X
000.100      398X UCI.1R EQU      01000000B      INTERNAL RESET
000.040      399X UCI.R0 EQU      00100000B      READER-ON CONTROL FLAG
000.020      400X UCI.ER EQU      00010000B      ERROR RESET
000.004      401X UCI.RE EQU      00000100B      RECEIVE ENABLE
000.002      402X UCI.IE EQU      00000010B      ENABLE INTERRUPTS FLAG
000.001      403X UCI.TE EQU      00000001B      TRANSMIT ENABLE
404X
405X **      STATUS READ COMMAND BITS.
406X
000.040      407X USR.FE EQU      00100000B      FRAMING ERROR
000.020      408X USR.OE EQU      00010000B      OVERRUN ERROR
000.010      409X USR.PE EQU      00001000B      PARITY ERROR
000.004      410X USR.TXE EQU      00000100B      TRANSMITTER EMPTY
000.002      411X USR.RXR EQU      00000010B      RECEIVER READY
000.001      412X USR.TXR EQU      00000001B      TRANSMITTER READY
000.033      413      XTEXT      ABSDEF

```

```

415X **      ABS FORMAT EQUIVALENCES.
416X
000.000      417X      ORG      0
418X
000.000      419X ABS.ID DS      1      377Q = BINARY FILE FLAG
000.001      420X      DS      1      FILE TYPE (FT.ABS)
000.002      421X ABS.LDA DS      2      LOAD ADDRESS
000.004      422X ABS.LEN DS      2      LENGTH OF ENTIRE RECORD
000.006      423X ABS.ENT DS      2      ENTRY POINT

```

```

000.010      424X
000.010      425X ABS.COD DS      0      CODE STARTS HERE
000.010      426      XTEXT      PICDEF

      428X **      PIC FORMAT EQUIVALENCES.
      429X
000.000      430X      ORG      0
      431X
000.000      432X PIC.ID DS      1      377Q = BINARY FILE FLAG
000.001      433X      DS      1      FILE TYPE (FT.PIC)
000.002      434X PIC.LEN DS      2      LENGTH OF ENTIRE RECORD
000.004      435X PIC.PTR DS      2      INDEX OF START OF PIC TABLE
      436X
000.006      437X PIC.COD DS      0      CODE STARTS HERE
000.006      438      XTEXT      DIRDEF

      440X **      DIRECTORY ENTRY FORMAT.
      441X
000.000      442X      ORG      0
      443X
      444X
000.377      445X DF.EMP EQU      377Q      FLAGS ENTRY EMPTY
000.376      446X DF.CLR EQU      376Q      FLAGS ENTRY EMPTY, REST OF DIR ALSO CLEAR
      447X
000.000      448X DIR.NAM DS      8      NAME
000.010      449X DIR.EXT DS      3      EXTENSION
000.013      450X DIR.PRO DS      1      PROJECT
000.014      451X DIR.VER DS      1      VERSION
000.015      452X DIR.IDL EQU      *      FILE IDENTIFICATION LENGTH
      453X
000.015      454X DIR.CLU DS      1      CLUSTER FACTOR
000.016      455X DIR.FLG DS      1      FLAGS
000.017      456X      DS      1      RESERVED
000.020      457X DIR.FGN DS      1      FIRST GROUP NUMBER
000.021      458X DIR.LGN DS      1      LAST GROUP NUMBER
000.022      459X DIR.LSI DS      1      LAST SECTOR INDEX (IN LAST GROUP)
000.023      460X DIR.CRD DS      2      CREATION DATE
000.025      461X DIR.ALD DS      2      LAST ALTERATION DATE
      462X
000.027      463X DIRELEN EQU      *      DIRECTORY ENTRY LENGTH
000.027      464      XTEXT      IOCDEF

      466X **      I/O CHANNEL DEFINITIONS.
      467X
000.000      468X      ORG      0
      469X
000.000      470X IOC.LNK DS      2      ADDRESS OF NEXT CHANNEL, =0 IF LAST
000.002      471X IOC.DDA DS      2      THREAD JUMP TO DEVICE DRIVER (VIA DEV TABLE)
      472X
000.004      473X IOC.FLG DS      1      FILE TYPE FLAGS

```

8251 USART BIT DEFINITIONS.

IOC

15:29:22 16-MAY-80

000.001	474X	FT.DD	EQU	00000001B	=1 IF DIRECTORY DEVICE
000.002	475X	FT.DR	EQU	00000010B	=1 IF OPEN FOR READ
000.004	476X	FT.OW	EQU	00000100B	=1 IF OPEN FOR WRITE
000.010	477X	FT.OU	EQU	00001000B	=1 IF OPEN FOR UPDATE
000.003	478X	IOC.SGL	EQU	*-IOC.DDA	LENGTH OF INFO FOR SEQUENTIAL FILE (FROM IOC)
	479X				
000.005	480X	IOC.GRT	DS	2	ADDRESS OF GROUP RESERVATION TABLE
000.007	481X	IOC.SPG	DS	1	SECTORS PER GROUP, THIS DEVICE
000.010	482X	IOC.CGN	DS	1	CURRENT GROUP NUMBER
000.011	483X	IOC.CSI	DS	1	CURRENT SECTOR INDEX (IN CURRENT GROUP)
000.012	484X	IOC.LGN	DS	1	LAST GROUP NUMBER
000.013	485X	IOC.LSI	DS	1	LAST SECTOR INDEX (IN LAST GROUP)
000.010	486X	IOC.DRL	EQU	*-IOC.FLG	LENGTH OF INFO NORMALLY COPIED BACK TO
	487X	*			THE CHANNEL TABLE
000.014	488X	IOC.DTA	DS	2	DEVICE TABLE ADDRESS FOR THIS DEVICE
000.016	489X	IOC.DES	DS	2	SECTOR NUMBER OF DIRECTORY ENTRY
000.020	490X	IOC.DEV	DS	2	DEVICE CODE
000.022	491X	IOC.UNI	DS	1	UNIT NUMBER (0-9)
000.021	492X	IOC.DIL	EQU	*-IOC.DDA	LENGTH OF INFO FOR DIRECTORY FILE (FROM IOC)
	493X				
000.023	494X	IOC.DIR	DS	DIRELEN	DIRECTORY ENTRY
	495X				
000.052	496X	IOCELEN	EQU	*	IOC ENTRY LENGTH
	497X				
000.001	498X	IOCCTD	EQU	1	INDEX OF USER CHANNEL #0 IN CHANTAB (FIRST = 0)

042.170		500	ORG	USERFWA-ABS.COD	
042.170	377 000	501	DB	377Q,FT.ABS	
042.172	200 042	502	DW	USERFWA	LOAD
042.174	206 015	503	DW	MEML-USERFWA	SIZE
042.176	360 057	504	DW	PRS	ENTRY
		505			

```

509 **      CMD - COMMAND COMPLETION PROCESSOR.
510 *
511 *      (H,L) = COMMAND STRING ADDRESS
512 *      (B,C) = CONTROL CARD ADDRESS
513
514
515 042.200    CCP      EQU      *          ENTRY
516
517 042.200    076 072    517      MVI      A,M1.LDA
518 042.202    062 217 044 518      STA      FICA          READ CHARACTERS FROM BUFFER
519 042.205    041 312 044 519      LXI      H,LINE
520 042.210    042 020 045 520      SHLD     LINPTR
521
522 *          INPUT 1 CHARACTER
523
524 042.213    315 150 053 524      CMD2    CALL     $INCHA          READ ONE CHARACTER
525 042.216    315 120 053 525      CALL     $MCU          MAP TO UPPER CASE
526 042.221    376 004      526      CPI      CTLD
527 042.223    312 001 046 527      JE       EXIT          IS EXIT
528
529 *          ADD TEMPORARILY TO LINE
530
531 042.226    052 020 045 531      CMD3    LHLD     LINPTR
532 042.231    167          532      MOV      M,A          RTORE IN LINE
533 042.232    043          533      INX      H
534 042.233    257          534      XRA      A
535 042.234    167          535      MOV      M,A          FOLLOW WITH 00
536
537 *          CLEAR NXTCHA, PATCNT
538
539 042.235    041 000 377 539      LXI      H,377000A
540 042.240    042 306 044 540      SHLD     NXTCHA
541
542 042.243    041 237 056 542      LXI      H,CMDTAB
543 042.246    042 310 044 543      SHLD     CMDADR
544
545 *          CHECK AGAINST NEXT COMMAND DESCRIPTION.
546
547 042.251    041 022 045 547      CMD4    LXI      H,CMD.8A
548 042.254    006 057      548      MVI      B,CMD.TL-CMD.8A (B) = BYTE COUNT
549 042.256    315 212 031 549      CALL     $ZERO          ZERO TABLES
550 042.261    041 307 044 550      LXI      H,PATCNT
551 042.264    064          551      INR      M
552 042.265    043          552      INX      H
553 042.266    136          553      MOV      E,M
554 042.267    043          554      INX      H
555 042.270    126          555      MOV      D,M          (D,E) = ADDRESS OF LAST COMMAND
556 042.271    315 275 044 556      CALL     SRC          SCAN FOR NEXT COMMAND
557 042.274    162          557      MOV      M,D
558 042.275    053          558      DCX      H
559 042.276    163          559      MOV      M,E          REPLACE CMDADR
560 042.277    001 312 044 560      LXI      B,LINE          (BC) = ADDRESS OF INPUT CHARACTER
561 042.302    032          561      LDAX     D
562 042.303    247          562      ANA      A
563 042.304    302 325 042 563      JNZ     CMD5          HAVE COMMAND ELEMENT
564

```

```

565 * NO MORE COMMANDS. HAVE:
566 *
567 * 1) NO MATCHES, OR
568 * 2) A UNIQUE NEXT CHARACTER
569
042.307 072 306 044 570 LDA NXTCHA
042.312 247 571 ANA A
042.313 302 226 042 572 JNZ CMD3 (A) = AUTO GENERATED CHARACTER
042.316 315 065 054 573 CALL $TYPCH
042.321 007 574 DB 7 BELL
042.322 303 213 042 575 JMP CMD2 READ FROM CONSOLE
576
577 * CHECK NEXT TABLE ELEMENT FOR MATCH
578
042.325 012 579 CMD5 LDAX B (A) = NEXT LINE CHARACTER
042.326 247 580 ANA A
042.327 302 376 042 581 JNZ CMD7 IF SOME
582
583 * NO MORE TEXT. SEE IF CAN ANTICIPATE NEXT CHARACTER
584
042.332 032 585 LDAX D (A) = COMMAND ELEMENT
042.333 376 300 586 CPI 0C0H
042.335 312 376 042 587 JE CMD7 PROCESS STRING RETURNS
042.340 315 061 044 588 CMD6 CALL AEC ACCEPT ENTERED COMMAND
042.343 376 012 589 CPI NL
042.345 312 213 042 590 JE CMD2 CANNOT COMPLETE CARRIAGE-RETURN
042.350 247 591 ANA A
042.351 310 592 RZ EXIT IF ENTIRE COMMAND MATCHED
042.352 372 213 042 593 JM CMD2 CANNOT COMPLETE
042.355 041 306 044 594 LXI H,NXTCHA
595
596 * SEE IF THIS IS THE FIRST COMPLETION CHARACTER,
597 * OR IF IT IS THE SAME CHARACTER AS PREVIOUSLY FOUND
598
599
042.360 276 600 CMP M
042.361 312 251 042 601 JE CMD4 SAME AS PREVIOUS, CAN COMPLETE
042.364 127 602 MOV D,A
042.365 206 603 ADD M
042.366 167 604 MOV M,A
042.367 272 605 CMP D SEE IF NXTCHA WAS 0
042.370 312 251 042 606 JE CMD4 CAN COMPLETE
042.373 303 213 042 607 JMP CMD2 CANNOT COMPLETE
608
609 * HAVE PATTERN AND TEXT. SEE IF MATCH.
610
042.376 325 611 CMD7 PUSH D
042.377 041 000 000 612 LXI H,0
043.002 071 613 DAD SP
043.003 042 304 044 614 SHLD STKPTR SAVE STACKPOINTER
043.006 041 053 043 615 LXI H,CMD.NG
043.011 345 616 PUSH H SET 'CMD.NG' AS RETURN ADDRESS
043.012 032 617 LDAX D
043.013 147 618 MOV H,A (H) = NEXT REQUIRED CHARACTER
043.014 007 619 RLC (A) = PATTERN ELEMENT
043.015 332 027 043 620 JC CMD8 IS COMPLEX ELEMENT

```

```

043.020 012 621 LDAX B (A) = NEXT TEXT ELEMENT
043.021 003 622 INX B ASSUME MATCH
043.022 274 623 CMP H
043.023 300 624 RNE TO CMD.NG IF BAD
043.024 303 045 043 625 JMP CMD.OK GOOD
626
627 * HAVE COMPLEX PATTERN ELEMENT
628
043.027 007 629 CMD8 RLC
043.030 007 630 RLC
043.031 007 631 RLC
043.032 346 007 632 ANI 7
043.034 315 076 031 633 CALL $TBRA BRANCH TO PROCESSOR
634
635 ** SPECIAL PATTERN ELEMENT TABLE.
636
043.037 036 637 DB CMD.8-* ENCLOSURE
043.040 106 638 DB CMD.9-* STRING CALL
043.041 132 639 DB CMD.A-* OCTAL ADDRESS
043.042 150 640 DB CMD.B-* FILE NAME
043.043 233 641 DB CMD.C-* STRING RETURN
043.044 241 642 DB CMD.D-* ADDRESS LIST
643
644 ** COMPLEX ROUTINES RETURN TO THESE THREE POINTS:
645 **
646
647
648 ** CMD.OK - NORMAL EXIT
649
043.045 023 650 CMD.OK INX D
043.046 341 651 CMD.OK POP H
043.047 341 652 POP H
043.050 303 325 042 653 JMP CMD5
654
655
656 ** CMD.NG - MATCH NO GOOD.
657
043.053 052 304 044 658 CMD.NG LHLD STKPTR
043.056 371 659 SPHL
043.057 321 660 POP D
043.060 303 251 042 661 JMP CMD4 TRY NEXT COMMAND
662
663
664 ** CMD.RA - RAN OUT OF TEXT WHILE MATCHING A COMPLEX
665 * ELEMENT.
666 *
667 * (A) = NEXT ELEMENT NEEDED
668
043.063 052 304 044 669 CMD.RA LHLD STKPTR
043.066 371 670 SPHL
043.067 341 671 POP H
043.070 076 200 672 MVI A,2000 DONT ALLOW ANY COMPLETION
043.072 303 340 042 673 JMP CMD6

```

```

677 **      CMD8 - PROCESS OPTION STRINGS.
678 *
679 *      1000      8-CODE
680 *      NNN      TARGET INDEX
681 *      F      FLAG
682 *
683 *      F = 0, MAY MATCH ONE
684 *      F = 1, MUST MATCH ONE
685
686
043.075      687 CMD.8 EQU *
043.075 012      688 LDAX B (A) = TEXT CHARACTER
043.076 147      689 MOV H,A (H) = TEXT CHARACTER
043.077 032      690 LDAX D
043.100 157      691 MOV L,A (L) = 8X FLAG
043.101 023      692 CMD.81 INX D
043.102 032      693 LDAX D (A) = NEXT PATTERN CHARACTER
043.103 274      694 CMP H
043.104 312 121 043 695 JE CMD.82 IF GOT A MATCH
043.107 007      696 RLC
043.110 322 101 043 697 JNC CMD.81 NOT FINISHED YET
698
699 *      NO MATCH
700
043.113 175      701 MOV A,L (A) = 8X CODE
043.114 017      702 RRC
043.115 330      703 RC
043.116 303 045 043 704 JMP CMD.OK REQUIRE MATCH - EXIT TO CMD.NG
705
706 *      HAVE MATCH
707
043.121 175      708 CMD.82 MOV A,L (A) = 8X CODE
043.122 017      709 RRC
043.123 346 007      710 ANI 7
043.125 306 022      711 ADI #CMD.8A
043.127 157      712 MOV L,A
043.130 174      713 MOV A,H (A) = TEXT CHARACTER
043.131 046 045      714 MVI H,CMD.8A/256
043.133 167      715 MOV M,A
043.134 003      716 INX B
717
718 *      SKIP REMAINDER OF OPTIONS
719
043.135 023      720 CMD.83 INX D
043.136 032      721 LDAX D CHECK TEXT PATTERN CHARACTER
043.137 007      722 RLC
043.140 322 135 043 723 JNC CMD.83 IF NOT TERMINATOR
043.143 303 045 043 724 JMP CMD.OK EXIT FOUND

```

```

726 **      CMD.9 - STRING CALL
727 *
728 *      1001          9 CODE
729 *      NNNN      STRING NUMBER
730
731
043.146 032      732 CMD.9 LDAX      D      (A) = 9X MODE
043.147 353      733          XCHG
043.150 042 026 045 734          SHLD      CMD.9A      SAVE RETURN ADDRESS
043.153 021 025 057 735          LXI      D,CMDXS      POINT TO EXTENSION STRING
043.156 346 017      736          ANI      170
043.160 157      737          MOV      L,A
043.161 315 275 044 738 CMD.91 CALL      SRC      SKIP REMAINDER OF COMMAND STRING
043.164 055      739          DCR      L
043.165 302 161 043 740          JNZ      CMD.91      IF MORE
043.170 303 046 043 741          JMP      CMD.OK.      DONE

```

```

743 **      CMD.A - OCTAL ADDRESS.
744 *
745 *      NO DEFAULTING IS ALLOWED.
746 *      THE ADDRESS MAY BE FOLLOWED BY A MODIFIER
747 *      AAAAAA(NNN)
748 *
749 *      1010          A CODE
750 *      NN      VALUE INDEX
751 *      F      =1 IF NO DEFAULT ALLOWED
752 *      F      =1 IF NO /LEN ALLOWED
753 *
754
755
043.173 032      756 CMD.A LDAX      D      (A) = FLAG
043.174 346 014      757          ANI      140
043.176 041 030 045 758          LXI      H,CMD.AA
043.201 315 072 030 759          CALL      $DADA      (HL) = ADDRESS OF STORE AREA
043.204 315 112 044 760          CALL      DAS      DECODE ADDRESS SPECIFICATION
043.207 303 045 043 761          JMP      CMD.OK      IS OK

```

```

763 **      CMD.B - FILE NAME
764 *
765 *      VALID HDOS FILE NAME
766 *
767 *      1011          B CODE
768 *      0000          NO SPECIFICATION
769
770
043.212 315 250 043 771 CMD.B CALL      CMD.B5      EXAMINE NEXT CHARACTER
043.215 330      772          RC      NOT GOOD 1ST CHARACTER
043.216 003      773          INX      B      ADVANCE POINTER
043.217 041 227 057 774          LXI      H,CMD.BA      (HL) = WORK AREA
043.222 167      775          MOV      M,A      STORE 1ST CHARACTER

```



043.223	043	776	INX	H	
043.224	315 250 043	777	CMD.B1	CALL	CMD.B5 GET NEXT CHARACTER
043.227	332 243 043	778		JC	CMD.B2 NOT PART OF FILE NAME
043.232	003	779		INX	B ADVANCE POINTER
043.233	167	780		MOV	M,A STORE
043.234	043	781		INX	H
043.235	076 250	782		MVI	A,#CMD.BA+FB.NAML
043.237	275	783		CMP	L
043.240	302 224 043	784		JNE	CMD.B1 NOT JUST LONG ENOUGH
		785			
		786	*		NAME GATHERED.
		787			
043.243	066 000	788	CMD.B2	MVI	M,0 FLAG END OF NAME
043.245	303 045 043	789		JMP	CMD.OK EXIT

791 \*\* CMD.B5 - EXAMINE NEXT CHARACTER FOR VALIDITY  
792 \*  
793 \* ENTRY (BC) = CHARACTER ADDRESS  
794 \* EXIT 'C' CLEAR IF CHARACTER VALID (0-9, A-Z, ; OR .)  
795 \* 'C' SET IF CHARACTER INVALID  
796 \* USES A,F

		797			
		798			
043.250	012	799	CMD.B5	LDAX	B
043.251	376 056	800		CPI	'.'
043.253	330	801		RC	TOO SMALL
043.254	310	802		RE	IS .
043.255	376 072	803		CPI	':'
043.257	310	804		RE	IS :
043.260	376 060	805		CPI	'0'
043.262	330	806		RC	NOT DIGIT
043.263	376 072	807		CPI	'9'+1
043.265	077	808		CMC	
043.266	320	809		RNC	IS DIGIT
043.267	376 101	810		CPI	'A'
043.271	330	811		RC	NOT ALPHA
043.272	376 133	812		CPI	'Z'+1
043.274	077	813		CMC	
043.275	311	814		RET	RETURN WITH VERDICT

816 \*\* CMD.C - STRING RETURN  
817 \*  
818 \* 1100 C FLAG  
819 \* 0000

		820			
		821			
043.276	052 026 045	822	CMD.C	LHLD	CMD.9A
043.301	353	823		XCHG	
043.302	303 045 043	824		JMP	CMD.OK EXIT

```

826 **      CMD.D - ADDRESS LIST
827 *
828 *      ADDR[CNT]],...,ADDR[CNT]]
829 *
830 *      NONE MAY BE NULL.
831
832
043.305 041 040 045 833 CMD.D LXI      H,CMD.DA
043.310 325          834 CMD.D1 PUSH     D
043.311 021 352 043 835          LXI      D,CMD.DB      POINT TO FLAG CHARACTER
043.314 315 112 044 836          CALL     DAS
837
838 *      WAS OK. SEE IF MORE TEXT FOLLOWS.
839 *
840 *      IF ',', TAKE IT AND PROCESS NEXT ADDRESS
841 *      IF NL, EXIT WITH MATCH
842 *      IF NULL, REQUIRE A ','
843 *      ELSE ERROR
844
043.317 321          845          POP      D
043.320 043          846          INX      H
043.321 076 100       847          MVI      A,#CMD.DA2
043.323 275          848          CMP      L      'Z' SET IF ENOUGH VALUES READ
043.324 012          849          LDAX    B
043.325 003          850          INX      B
043.326 312 336 043 851          JE       CMD.D2      IF ALREADY READ ENOUGH VALUES
043.331 376 054       852          CPI      ','
043.333 312 310 043 853          JE       CMD.D1      DECODE NEXT ADDRESS
043.336 066 003       854 CMD.D2 MVI      H,3      SET DEFAULT FLAG FOR LAST+1 VALUE
043.340 376 012       855          CPI      NL
043.342 312 045 043 856          JE       CMD.OK      COMMAND COMPLETE. ACCEPT
043.345 247          857          ANA      A
043.346 300          858          RNZ
043.347 303 063 043 859          JMP      CMD.RA      IS NOT NULL: ILLEGAL
860          RUN OUT
043.352 242          861 CMD.DB DB      0A2H

```

```

865 **      ACN - ACCUMULATE NUMBER.
866 *
867 *      ACN ACCUMULATES A N-DIGIT NUMBER
868 *
869 *      ENTRY (B,C) = TEXT ADDRESS
870 *            (A) = NUMBER OF DIGITS
871 *            (D) = BASE
872 *      EXIT (D,E) = VALUE
873 *            'Z' FLAG SET OF 0 DIGITS
874 *            (A) = NZ IF OVERFLOW
875
876
043.353 345 877 ACN    PUSH    H            SAVE (H,L)
043.354 365 878        PUSH    PSW
043.355 041 000 000 879        LXI    H,0        (H,L) = ACCUMULATOR
043.360 134 880        MOV     E,H        (E) = OVERFLOW FLAG
043.361 365 881 ACN1   PUSH    PSW
043.362 315 217 044 882        CALL   FIC
043.365 326 060 883        SUI     '0'
043.367 332 027 044 884        JC     ACN2        NOT DIGIT
043.372 272 885        CMP     D
043.373 322 027 044 886        JNC    ACN2        TOO LARGE
043.376 365 887        PUSH    PSW        SAVE DIGIT VALUE
043.377 325 888        PUSH    D        SAVE BASE AND OVERFLOW FLAG
044.000 172 889        MOV     A,D        (A) = BASE
044.001 353 890        XCHG        (DE) = ACCUMULATOR
044.002 315 007 031 891        CALL   $MUB6    (HL) = ACCUMULATOR*BASE
044.005 321 892        POP     D        RESTORE (DE)
044.006 203 893        ADD     E        ACCUMULATE OVERFLOWS
044.007 137 894        MOV     E,A        (E) = OVERFLOW INDICATOR
044.010 361 895        POP     PSW
044.011 315 072 030 896        CALL   $DADA    (HL) = ACCUMULATOR*BASE+DIGIT
044.014 173 897        MOV     A,E
044.015 316 000 898        ACI     0
044.017 137 899        MOV     E,A        ACCUMULATE OVERFLOWS
044.020 361 900        POP     PSW        (A) = COUNT
044.021 075 901        DCR     A
044.022 302 361 043 902        JNZ    ACN1        IF MORE TO GO
044.025 365 903        PUSH    PSW
044.026 003 904        INX     B
905
906 *      GOT ALL DIGITS
907
044.027 013 908 ACN2   DCX     B
909
910 *      IF BASE = 8, SHIFT TOP HALF RIGHT TO MAKE UP
911 *      FOR DIGIT 2, WHICH CONTAINS ONLY 2 DIGITS.
912
044.030 076 010 913        MVI     A,8
044.032 272 914        CMP     D
044.033 302 051 044 915        JNE    ACN3        NOT OCTAL
044.036 173 916        MOV     A,E        (A) = OVERFLOW
044.037 037 917        RAR
044.040 137 918        MOV     E,A        (E) = BITS 1-7 OF OVERFLOW
044.041 174 919        MOV     A,H
044.042 037 920        RAR

```

044.043	147	921	MOV	H,A	
044.044	076 000	922	MVI	A,0	
044.046	213	923	ADC	E	ADD OVERFLOW FROM SHIFT
044.047	213	924	ADC	E	
044.050	137	925	MOV	E,A	
044.051	361	926	ACN3 POP	PSW	(A) = ORIGINAL DIGIT COUNT
044.052	127	927	MOV	D,A	(D) = COUNT
044.053	361	928	POP	PSW	
044.054	272	929	CMP	D	
044.055	173	930	MOV	A,E	(A) = CARRY FLAG
044.056	353	931	XCHG		(DE) = RESULT
044.057	341	932	POP	H	
044.060	311	933	RET		RETURN

		935	**	AEC - ACCEPT ECHOED CHARACTER.	
		936	*		
		937	*	AEC ACCEPTS AND ECHOS THE ENTERED CHARACTER.	
		938			
		939			
044.061	365	940	AEC	PUSH	PSW
044.062	052 020 045	941		LHLD	LINPTR
044.065	176	942		MOV	A,M
044.066	247	943		ANA	A
044.067	312 110 044	944		JZ	AEC1
044.072	315 071 054	945		CALL	\$TYPC.
044.075	043	946		INX	H
044.076	066 000	947		MVI	M,0
044.100	042 020 045	948		SHLD	LINPTR
044.103	376 012	949		CPI	NL
000.000		950		ERRNZ	LF-NL
		951	*	MVI	A,LF
044.105	314 071 054	952		CE	\$TYPC.
044.110	361	953	AEC1	POP	PSW
044.111	311	954		RET	

IF ALREADY TYPED  
TYPE IT

TWO CHARACTER MATCH  
ASSUME CR  
IF CR, ECHO CRLF

EXIT

		956	**	DAS - DECODE ADDRESS SPECIFICATION.	
		957	*		
		958	*	ENTRY ((HL)) = VALUE BLOCK	
		959	*	((DE)) = PATTERN CODE	
		960	*	EXIT TO CMD.NG IF BAD	
		961	*	RETURNS IF OK	
		962			
		963			
044.112	325	964	DAS	PUSH	D
044.113	032	965		LDAX	D
044.114	365	966		PUSH	PSW
044.115	076 006	967		MVI	A,6
044.117	026 010	968		MVI	D,8
044.121	315 353 043	969		CALL	ACN
044.124	066 000	970		MVI	M,0

SAVE CODE  
(A) = MAX DIGITS  
(D) = BASE  
ACCUMULATE NUMBER

DAS

```

044.126 302 151 044 971      JNZ      DAS1      NOT DEFAULTED
044.131 361          972      POP      PSW      (A) = OPTION FLAG
044.132 365          973      PUSH     PSW
044.133 017          974      RRC
044.134 017          975      RRC
044.135 332 053 043 976      JC       CMD.NG      DEFAULT NOT ALLOWED
977
978 *      HAVE NON-NUMERIC. IS EITHER DEFAULT (NULL) OR #
979
044.140 064          980      INR      M      ASSUME NULL
044.141 012          981      LDAX     B
044.142 326 043      982      SUI      '#/'
044.144 302 155 044 983      JNE      DAS2      NOT #, IS NULL
044.147 003          984      INX      B
044.150 064          985      INR      M
044.151 247          986      DAS1     ANA      A      CHECK CARRY
044.152 302 053 043 987      JNZ      CMD.NG      OVERFLOW
044.155 043          988      DAS2     INX      H
044.156 163          989      MOV      M,E
044.157 043          990      INX      H
044.160 162          991      MOV      M,D
044.161 043          992      INX      H      (HL) = ADDRESS OF COUNT FIELD
044.162 066 001      993      MVI      M,1      ASSUME 1
044.164 361          994      POP      PSW
044.165 321          995      POP      D
044.166 017          996      RRC
044.167 330          997      RC          IF COUNT NOT ALLOWED
998
999 *      SEE IF /CNT FOLLOWS
1000
044.170 012          1001     LDAX     B
044.171 376 057      1002     CPI      '/'
044.173 300          1003     RNE          IF NONE
044.174 325          1004     PUSH     D
044.175 003          1005     INX      B
044.176 076 003      1006     MVI      A,3
044.200 026 012      1007     MVI      D,10
044.202 315 353 043 1008     CALL     ACN      ACCUMULATE DECIMAL NUMBER
044.205 312 053 043 1009     JZ       CMD.NG      IF NONE
044.210 262          1010     ORA      D
044.211 302 053 043 1011     JNZ      CMD.NG      IF OVERFLOW
044.214 163          1012     MOV      M,E      SAVE VALUE
044.215 321          1013     POP      D
044.216 311          1014     RET          IS OK ELEMENT

```

```

1016 **      FIC - FETCH INPUT CHARACTER.
1017 *
1018 *      FIC IS CALLED TO GET THE NEXT INPUT CHARACTER.
1019 *
1020 *      ENTRY (B,C) = INPUT POINTER
1021
1022
044.217      1023     FIC      EQU      *

```

```

044.217      1024 FICA EQU *      TOGGLE FLAG
044.217 303 235 044 1025 JMP FIC2 NO-OP'ED IF TO READ FROM MEMORY
044.222 012      1026 LDAX B
044.223 247      1027 ANA A
044.224 312 063 043 1028 JZ CMD.RA IF NONE
044.227 003      1029 INX B
044.230 311      1030 RET
                1031
                1032 *      READ FROM TERMINAL
                1033
044.231 315 065 054 1034 FIC1 CALL $TYPCH REFUSE ENTRY
044.234 007      1035 DB 7 BELL
044.235 315 131 053 1036 FIC2 CALL $RCHAR INPUT A CHARACTER
044.240 376 004      1037 CPI CTLD
044.242 312 001 046 1038 JE EXIT CTL-D
044.245 062 214 057 1039 FIC2.5 STA $LSTIN
044.250 376 012      1040 CPI NL
044.252 310      1041 RE
044.253 376 040      1042 CPI
044.255 312 071 054 1043 JE $TYPC. ACCEPT WITH ECHO
044.260 376 060      1044 CPI '0'
044.262 332 231 044 1045 JC FIC1 NOT DIGIT
044.265 376 072      1046 CPI '9'+1
044.267 332 071 054 1047 JC $TYPC. ACCEPT DIGIT WITH ECHO
044.272 303 231 044 1048 JMP FIC1 REFUSE

```

```

                1050 **      SRC - SKIP REMAINDER OF COMMAND PATTERN.
                1051 *
                1052 *      SRC SCANS A STRING UNTIL A BYTE IS FOUND.
                1053 *
                1054 *      ENTRY (D,E) = STRING ADDRESS
                1055 *      EXIT (D,E) UPDATED
                1056
                1057
044.275 032      1058 SRC LDAX D
044.276 247      1059 ANA A
044.277 023      1060 INX D
044.300 302 275 044 1061 JNZ SRC MORE TO GO
044.303 311      1062 RET

```

044.304	000 000	1065	STKPTR	DW	0	STACK POINTER
044.306	000	1066	NXTCHA	DB	0	NEXT CHAR
044.307	000	1067	PATCNT	DB	0	INDEX OF CURRENT PATTERN
044.310	000 000	1068	CMDADR	DW	0	ADDRESS OF CURRENT COMMAND DESCRIPTOR
044.312		1069	LINE	DS	70	
044.312		1070	FNRA	EQU	LINE	FNR WORK AREA
045.020		1071	LINPTR	DS	2	LINE POINTER
		1072				
045.022		1073	CMD.8A	DS	4	4 KEY VALUES
		1074				
045.026		1075	CMD.9A	DS	2	RETURN ADDRESS
		1076				
		1077	**			ADDRESS BLOCK FORMAT.
		1078	*			
		1079	*			EACH ADDRESS BLOCK CONSISTS OF 4 BYTES:
		1080	*			
		1081	*			0 - FLAG BITS.
		1082	*			1-2 - ADDRESS VALUE (IF EXPLICIT)
		1083	*			3 - LENGTH MODIFIER
		1084				
045.030		1085	CMD.AA	DS	2*4	TWO ADDRESSES
		1086				
045.040		1087	CMD.IA	DS	4*8	8 ADDRESSES
045.100		1088	CMD.DA2	DS	1	HOLDS END OF STRING FLAG IF 8 ENTRIES
		1089				
045.101		1090	CMD.TL	DS	0	END OF TABLE

045.101		1094	HBUG	EQU	*	MAIN ENTRY POINT
		1095				
045.101		1096	START	EQU	*	
		1097				
045.101	061 200 042	1098		LXI	SP,STACK	SET STACK VALUE
045.104	315 054 031	1099		CALL	\$SAVALL	SAVE ENTRY REGISTERS
045.107	315 136 031	1100		CALL	\$TYPTX	
045.112	012 012 110	1101		DB	NL,NL,'HDOS DEBUG # 102.05.00.'	
045.142	040 040 040	1102	ISSUEA	DB	'',NL,ENL	
045.150	076 001	1103		MVI	A,'A'-'@'	
045.152	041 332 045	1104		LXI	H,INTRPT	
045.155	377 041	1105		DB	SYSCALL,CTLC	
045.157	315 057 053	1106		CALL	SDC	SET UP DEBUG CONSOLE /79.12.GC/
		1107				
		1108	*			PRESET REGISTERS ON STACK
		1109				
045.162	361	1110		POP	PSW	RESTORE ENTRY REGISTERS
045.163	301	1111		POP	B	
045.164	321	1112		POP	D	
045.165	041 200 042	1113		LXI	H,USERFWA	
045.170	343	1114		XTHL		SET HBUG AS P-REG VALUE
045.171	345	1115	HBUG1	PUSH	H	SAVE H
045.172	325	1116		PUSH	D	
045.173	305	1117		PUSH	B	
045.174	365	1118		PUSH	PSW	
045.175	041 012 000	1119		LXI	H,10	
045.200	071	1120		DAD	SP	
045.201	345	1121		PUSH	H	SAVE SP
045.202	041 000 000	1122		LXI	H,0	
045.205	071	1123		DAD	SP	
045.206	042 226 045	1124		SHLD	REGPTR	SAVE REGISTER POINTER
		1126	**			TBGX - TERMINAL DEBUGGER EXIT.
		1127	*			
		1128	*			COMMAND PROCESSORS RETURN HERE.
		1129	*			
		1130				
		1131				
045.211		1132	RESTART	EQU	*	
045.211	076 005	1133		MVI	A,CN,LD	
045.213	377 055	1134		DB	SYSCALL,.CLEAR	CLEAR I/O CHANNEL
		1135				
		1136	*			CLEAR LOAD/DUMP CHANNEL
		1137				
045.215	076 005	1138		MVI	A,CN,LD	
045.217	377 055	1139		DB	SYSCALL,.CLEAR	CLEAR CHANNEL
045.221	257	1140		XRA	A	
045.222	062 216 057	1141		STA	MEMFB+FB,FLG	CLEAR OPEN/CLOSE FLAGS
		1142				
045.225		1143	TBGX	EQU	*	
045.225	061 000 000	1144		LXI	SP,0	(SP) = REGPTR
045.226		1145	REGPTR	EQU	*-2	FWA OF REGISTERS ON STACK
		1146	*	CALL	SDC	SET DEBUGGER CONSOLE INVIRONMENT /79.12.GC/
045.230	315 253 052	1147		CALL	RBM	REMOVE BREAKPOINTS FROM MEMORY



```

045.233 072 330 040 1148 LDA S.CUSOR
045.236 247 1149 ANA A
045.237 304 142 053 1150 CNZ $CRLF IF LF NEEDED
045.242 315 136 031 1151 CALL $TYPTX TYPE PROMPT
045.245 072 102 272 1152 DB 'B','+200Q
1153
1154 * GET ANOTHER COMMAND.
1155
045.250 315 200 042 1156 CALL CCP CALL COMMAND COMPLETION PROCESSOR
045.253 072 307 044 1157 LDA PATCNT (A) = COMMAND INDEX
045.256 041 225 045 1158 LXI H,TBGX
045.261 345 1159 PUSH H SET RETURN ADDRESS
045.262 041 030 045 1160 LXI H,CMD,AA
045.265 315 061 031 1161 CALL $TJMP BRANCH THROUGH TABLE
1162
045.270 052 046 1163 HBUGA DW TB.DVS DISPLAY VALUES, SINGLE ADDRESS
045.272 052 046 1164 DW TB.DVP DISPLAY VALUES, PAIR ADDRESS
045.274 070 046 1165 DW TB.CMS CHANGE MEMORY, SINGLE ADDRESS
045.276 070 046 1166 DW TB.CMP CHANGE MEMORY, PAIR ADDRESS
000.004 1167 TB.DARI EQU *-HBUGA/2 TB.DAR INDEX
045.300 077 046 1168 DW TB.DAR DISPLAY ALL REGISTERS
045.302 120 046 1169 DW TB.DSR DISPLAY SINGLE REGISTER
045.304 126 046 1170 DW TB.CSR CHANGE SINGLE REGISTER
045.306 152 046 1171 DW TB.EXE EXEC COMMAND
045.310 162 046 1172 DW TB.STP STEP COMMAND
045.312 236 046 1173 DW TB.SBL SET BREAKPOINT LIST COMMAND
045.314 241 046 1174 DW TB.DBL DISPLAY BREAKPOINT LIST
045.316 321 046 1175 DW TB.CBL CLEAR BREAKPOINT LIST
045.320 353 046 1176 DW TB.CAB CLEAR ALL BREAKPOINTS
045.322 223 047 1177 DW TB.DMP DUMP
045.324 013 050 1178 DW TB.LOA LOAD
045.326 155 050 1179 DW TB.LOA, LOAD PIC
045.330 363 046 1180 DW TB.GO GO

```

```

1182 ** INTRPT - CTL-C INTERRUPT PROCESSING.
1183 *
1184 * DECIDE IF WE WERE IN HBUG MODE OR IN USER MODE.
1185 * IF HBUG MODE, JUST POP THROUGH.
1186
1187

```

```

045.332 315 136 031 1188 INTRPT CALL $TYPTX
045.335 136 301 1189 DB 'B','+200Q
045.337 076 000 1190 MVI A,0 (A) = USER MODE FLAG
045.340 1191 USERMD EQU *-1
045.341 247 1192 ANA A
045.342 312 225 045 1193 JZ TBGX IS JUST IN HBUG
045.345 257 1194 XRA A
045.346 062 340 045 1195 STA USERMD SET DEBUG MODE
045.351 315 057 053 1196 CALL SDC SET UP DEBUG CONSOLE /79,12,GC/
045.354 361 1197 POP PSW DISCARD HDOS RETURN ADDRESS
045.355 361 1198 POP PSW (PSW) = USER PSW VALUES
045.356 345 1199 PUSH H RE-SAVE USER REGISTERS
045.357 325 1200 PUSH D RE-SAVE USER REGISTERS

```

045.360	305	1201	PUSH	B	
045.361	345	1202	PUSH	PSW	
045.362	041 012 000	1203	LXI	H,10	
045.365	071	1204	DAD	SP	
045.366	345	1205	PUSH	H	SAVE SP VALUE ON STACK
045.367	041 000 000	1206	LXI	H,0	
045.372	071	1207	DAD	SP	
045.373	042 226 045	1208	SHLD	REGPTR	SET NEW REGISTER POINTER
045.376	303 130 047	1209	JMP	REX	TREAT AS BREAKPOINT

1211 \*\* EXIT - PROCESS CTL-D (END OF FILE ON CONSOLE INPUT)  
1212 \*  
1213 \* IF HE IS SURE, EXIT TO O/S

046.001	315 136 031	1216	EXIT	CALL	\$TYPTX	
046.004	136 104 012	1217		DB	'D',NL,BELL,'Are You SURE?',', '+2000	
046.026	315 131 053	1218		CALL	\$RCHAR	
046.031	376 004	1219		CPI	CTLD	
046.033	312 046 046	1220		JE	EXIT1	STILL EOF
046.036	315 120 053	1221		CALL	\$MCU	
046.041	376 131	1222		CPI	'Y'	
046.043	302 225 045	1223		JNE	TBGX	SAVED AT THE BRINK OF DEATH!
046.046	076 001	1224	EXIT1	MVI	A,1	FLAG ABORT EXIT
046.050	377 000	1225		DB	SYSCALL,.EXIT	/79.12.6C/

```

1229 ** TB.DVS - DISPLAY VALUE, SINGLE ADDRESS SPECIFIED.
1230 *
1231 * ADDR{<LEN>}[<OPT>]
1232
1233
046.052 1234 TB.DVS EQU *
```

```

1236 ** TB.DVP - DISPLAY VALUE, PAIRED ADDRESS SPECIFIED.
1237 *
1238 * ADDR-ADDR[OPT]
1239
1240
046.052 1241 TB.DVP EQU *
046.052 037 1242 RAR (A) = COMMAND INDEX
046.053 315 175 052 1243 CALL RAS RESOLVE ADDRESS SPECIFICATION
046.056 315 000 052 1244 DVP2 CALL DVB DISPLAY VALUE WITH BLANK
046.061 315 312 051 1245 CALL CUB SEE IF DONE /80.02.GC/
046.064 330 1246 RC DONE /80.02.GC/
046.065 303 056 046 1247 JMP DVP2 /80.02.GC/
```

```
1251 **      TB.CMS - CHANGE MEMORY, SINGLE ADDRESS SPECIFIED.
1252 *
1253 *      ADDR(LEN)=OPT3VALUES
1254
1255
046.070      1256 TB.CMS EQU      *
```

```
1258 **      TB.CMP - CHANGE MEMORY ADDRESS PAIR.
1259 *
1260 *      ADDR-ADDR=OPT3VALUelist
1261
1262
046.070      1263 TB.CMP EQU      *
046.070 037      1264      RAR
046.071 315 175 052 1265      CALL      RAS      (A) = COMMAND INDEX
046.074 303 376 050 1266      JMP      ANV      RESOLVE ADDRESS SPECIFICATION
                                ACCEPT NEW VALUES
```

```

1269 ** TB.DAR - DISPLAY ALL REGISTERS.
1270 *
1271 * A=XXX, B=XXX, C=XXX; ... , ETC.
1272
1273
046.077 021 116 057 1274 TB.DAR LXI D,DARA
046.102 006 013 1275 MOV B,DARAL (B) = ENTRY COUNT
046.104 315 142 053 1276 CALL $CRLF NEW LINE
046.107 315 360 051 1277 TB.DAR1 CALL DRV DISPLAY REGISTER VALUE
046.112 005 1278 DCR B
046.113 023 1279 INX D
046.114 302 107 046 1280 JNZ TB.DAR1
046.117 311 1281 RET EXIT

```

```

1283 ** TB.DSR - DISPLAY SINGLE REGISTER
1284 *
1285
1286
046.120 315 344 051 1287 TB.DSR CALL DRI DETERMINE REGISTER INDEX
046.123 303 364 051 1288 JMP DRV. DISPLAY REGISTER VALUE

```

```

1290 ** TB.CSR - CHANGE SINGLE REGISTER
1291 *
1292
1293
046.126 1294 TB.CSR EQU *
046.126 315 344 051 1295 CALL DRI DETERMINE REGISTER INDEX
046.131 315 327 051 1296 CALL DRA DETERMINE REGISTER ADDRESS
046.134 124 1297 MOV D,H
046.135 135 1298 MOV E,L
046.136 362 142 046 1299 JP CSR1 IF SINGLE
046.141 023 1300 INX D
046.142 346 200 1301 CSR1 ANI 200H
046.144 062 023 045 1302 STA CMD.8A+1
046.147 303 376 050 1303 JMP ANV ACCEPT NEW VALUE AND EXIT

```

```

1307 **      TB.EXE - PROCESS EXEC COMMAND.
1308 *
1309 *      EXEC ADDR-ADDR(CNT)],...,ADDR(CNT)]
1310
1311
046.152      1312 TB.EXE EQU      *
046.152 345      1313      PUSH  H      SAVE START ADDRESS POINTER
046.153 315 333 052 1314      CALL SBL    SET BREAKPOINT LIST
046.156 341      1315      POP   H      (HL) = ADDRESS OF START BLOCK
046.157 303 363 046 1316      JMP   TB.GO  PROCESS AS *GO*

```

```

1320 **      TB.STP - PROCESS SINGLE STEP COMMAND.
1321 *
1322 *      STEP          SINGLE STEP AT *P*
1323 *      STEP (CNT)    STEP CNT TIMES FROM *P*
1324 *      STEP ADDR     STEP ONCE AT *ADDR*
1325 *      STEP ADDR(CNT) STEP CNT TIMES FROM *ADDR*
1326
1327
046.162      1328 TB.STP EQU      *
046.162 315 101 053 1329      CALL    SSA          SET STARTING ADDRESS
046.165 072 033 045 1330      LDA      CMD,AA+3      (A) = COUNT
046.170 062 201 046 1331 STP1     STA      STPA          SAVE
046.173 041 202 046 1332      LXI      H,STPRTN
046.176 303 165 047 1333      JMP      BKP2          PROCESS AS BKPT
1334
046.201 000      1335 STPA     DB      0
1336
1337 **      SINGLE STEP RETURNS HERE
1338
046.202 257      1339 STPRTN  XRA      A
046.203 062 340 045 1340      STA      USERMD
046.206 315 057 053 1341      CALL    SDC          SET UP DEBUG CONSOLE /79.12.GC/
046.211 041 000 000 1342      LXI      H,0
046.214 071      1343      DAD      SP          (HL) = REGPTR VALUE
046.215 042 226 045 1344      SHLD    REGPTR
046.220 315 306 052 1345      CALL    RFD          RESTORE FRONT PANEL DISPLAY
046.223 072 201 046 1346      LDA      STPA
046.226 373      1347      EI
046.227 075      1348      DCR      A
046.230 302 170 046 1349      JNZ      STP1
046.233 303 130 047 1350      JMF      REX          RETURN FROM EXECUTION

```

TB.SBL - SET BREAKPOINT LIST.

TB.SBL

15:29:44 16-MAY-80

1354 \*\* TB.SBL - SET BREAKPOINT LIST.

1355 \*

1356 \* BKPT A1,...,AN

1357

1358

046.236

1359 TB.SBL EQU \*

046.236 303 333 052

1360 JMP SBL

SET BREAKPOINT LIST



```

1364 ** TB.DBL - DISPLAY BREAKPOINT LIST.
1365 *
1366 * TYPE OUT LIST OF ALL BREAKPOINTS, WITH THEIR REPEAT COUNTS.
1367 *
1368 * ADDR/RPT
1369
1370
1371 TB.DBL EQU *
1372 LXI H,BKPTAB
1373 MVI B,BKPTBL
1374
1375 * TYPE NON-NULL ENTRYs
1376
1377 DBL1 XCHG
1378 LXI H,'F'*256 FULL WORD OCTAL
1379 SHLD CMD,8A SET OPTION
1380 XCHG
1381 MOV A,M
1382 INX H
1383 ORA M
1384 JZ DBL2 IF NULL
1385 DCX H
1386 CALL FVD FORMAT VALUE FOR DISPLAY
1387 CALL $TYPCH
1388 DB '/'
1389 XCHG
1390 LXI H,'D'
1391 SHLD CMD,8A SET DECIMAL BYTE
1392 XCHG
1393 CALL DVB DISPLAY VALUE WITH BLANK
1394 DCX H
1395 DCX H
1396
1397 * ENTRY PROCESSED. CHECK NEXT.
1398
1399 DBL2 INX H
1400 INX H
1401 INX H
1402 DCR B
1403 JNZ DBL1
1404 RET DONE, EXIT

```

```

1408 **      TB,CBL - CLEAR BREAKPOINT LIST.
1409 *
1410 *      CLEAR A1,...,AN
1411 *
1412
046.321      1413 TB,CBL EQU      *
046.321 056 040 1414 MVI      L,*CMD.DA
1415
1416 *      EXAMINE NEXT ADDRESS SUPPLIED.
1417
046.323 176   1418 CBL1      MOV      A,M
046.324 017   1419          RRC
046.325 330   1420          RC          END OF LIST
046.326 043   1421          INX      H
1422
1423 *      FIND SPECIFIED BREAKPOINT
1424
046.327 116   1425          MOV      C,M
046.330 043   1426          INX      H
046.331 106   1427          MOV      B,M      (BC) = SPECIFIED ADDRESS
046.332 315 010 052 1428          CALL     FBT      FIND BREAKPOINT IN TABLE
046.335 302 346 046 1429          JNE      CBL3      IF NOT FOUND
1430
1431 *      FOUND IT. (DE) = ADDRESS
1432
046.340 257   1433 CBL2      XRA      A
046.341 022   1434          STAX     D
046.342 023   1435          INX      D
046.343 022   1436          STAX     D
046.344 023   1437          INX      D
046.345 022   1438          STAX     D
1439
1440 *      LOOK AT NEXT ADDRESS
1441
046.346 043   1442 CBL3      INX      H
046.347 043   1443          INX      H
046.350 303 323 046 1444          JMP      CBL1

```

1448 \*\* TB.CAB - CLEAR ALL BREAKPOINTS.

1449 \*

1450 \*

1451 \*

1452

046.353 041 145 057

1453

TB.CAB

LXI

H,BKPTAB

046.356 006 040

1454

MVI

B,BKPTBL\*4

(D) = LENGTH

046.360 303 212 031

1455

JMP

\$ZERO

ZERO MEMORY

```

1459 **      TB.G0 - PROCESS *G0* COMMAND.
1460 *
1461
1462
046.363      1463 TB.G0 EQU *
046.363      1464 CALL SSA      SET START ADDRESS
046.366      1465 G00 CALL SRM      SET BREAKPOINTS IN MEMORY
046.371      1466 LXI H,.BKP.
046.374      1467 G02 SHLD .UIVEC+4
046.377      1468 MVI A,M1.JMP
047.001      1469 STA .UIVEC+3      SETUP VECTOR
047.004      1470 G0 LHLD REGPTR
047.007      1471 SPHL      RESET STACK
047.010      1472 DI
047.011      1473 LXI H,USERMD
047.014      1474 INR M      SET USER MODE
047.015      1475 POP H      (HL) = STACKPOINTER VALUE
047.016      1476 SHLD GOA      SAVE FOR STACK
047.021      1477 CALL RUC      RESTORE USER CONSOLE INVIORNMENT
047.024      1478 POP PSW
047.025      1479 POP B
047.026      1480 POP D
047.027      1481 POP H
047.030      1482 SHLD GOB      SAVE (HL) FOR LATER PICKUP
047.033      1483 POP H      (HL) = RETURN ADDRESS
047.034      1484 SHLD GOC      SET RETURN ADDRESS
047.037      1485 LXI H,0      (HL) = STACKPOINTER
047.040      1486 G0A EQU *-2
047.042      1487 SPHL      SET STACK
047.043      1488 LXI H,0      (HL) = (HL)
047.044      1489 GOB EQU *-2
047.046      1490 EI
047.047      1491 JMP 0
047.050      1492 GOC EQU *-2      ADDRESS OF ENTRY TO USER PROGRAM
1493
1494
1495 **      CONTROL IS PASSED HERE WHEN BREAKPOINT IS HIT.
1496
047.052      1497 .BKP. EQU *
047.052      1498 XRA A
047.053      1499 STA USERMD      CLEAR USER MODE
047.056      1500 CALL SDC      SET UP DEBUG CONSOLE      /79.12.GC/
047.061      1501 LXI H,0
047.064      1502 DAD SP
047.065      1503 SHLD REGPTR      SAVE REGISTER POINTER
047.070      1504 CALL RBM      REMOVE BREAKPOINTS FROM MEMORY
047.073      1505 LXI H,10
047.076      1506 DAD SP
047.077      1507 MOV C,M
047.100      1508 INX H
047.101      1509 MOV B,M      (BC) = ADDRESS OF INSTRUCTION HIT
047.102      1510 DCX B
047.103      1511 MOV M,B      STORE DECREMENTED PC
047.104      1512 DCX H
047.105      1513 MOV M,C
047.106      1514 CALL FBT      FIND BREAKPOINT

```

```

047.111 302 130 047 1515 JNZ REX IF NOT FOUND
047.114 023 1516 INX D
047.115 023 1517 INX D
047.116 353 1518 XCHG
047.117 065 1519 DCR M
047.120 302 142 047 1520 JNZ BKP1 IF MORE ITERATIONS BEFORE ACKNOWLEDGING
1521
1522 * BREAKPOINT COUNT EXHAUSTED. ACKNOWLEDGE.
1523
047.123 257 1524 XRA A
047.124 053 1525 DCX H
047.125 167 1526 MOV M,A
047.126 053 1527 DCX H
047.127 167 1528 MOV M,A CLEAR TABLE ENTRY

```

```

1530 ** REX - RETURN FROM EXECUTION
1531 *
1532 * PRINT -P=NNNNNN-
1533
047.130 1534 REX EQU *
047.130 315 136 031 1535 CALL $TYPTX
047.133 055 120 275 1536 DB '-P', '=+2000
047.136 041 000 106 1537 LXI H,'F'*256
047.141 042 022 045 1538 SHLD CMD,8A DOUBLE OCTAL VALUE
047.144 315 324 051 1539 CALL DRA. DETERMINE REGISTER ADDRESS
047.147 315 045 052 1540 CALL FVD. FORMAT VALUE
047.152 315 136 031 1541 CALL $TYPTX
047.155 055 212 1542 DB '-','ENL
047.157 303 225 045 1543 JMP TBGX ENTER CONTROL LOOP
1544
1545 * MORE HITS ON THIS BREAKPOINT
1546
047.162 041 206 047 1547 BKP1 LXI H,G03
047.165 363 1548 BKP2 DI
047.166 072 011 040 1549 LDA .CTLFLG
047.171 062 307 052 1550 STA RFD. SAVE FOR *RFD*
047.174 346 257 1551 ANI 3770-CB,SSI-CB,CLI ENABLE STEP, CLEAR CLOCK
047.176 062 011 040 1552 STA .CTLFLG
047.201 323 360 1553 OUT OF,CTL
047.203 303 374 046 1554 JMP G02 SINGLE STEP OVER SITE OF BREAKPOINT
1555
1556 ** RETURN FROM SINGLE STEPPING OVER BREAKPOINTED INSTRUCTION
1557
047.206 041 000 000 1558 G03 LXI H,0
047.211 071 1559 DAD SP
047.212 042 226 045 1560 SHLD REGPTR
047.215 315 306 052 1561 CALL RFD RESTORE FRONT PANEL DISPLAY
047.220 303 366 046 1562 JMP G00

```

```

1566 *** TB.DMP - PROCESS *DUMP* COMMAND.
1567 *
1568 * DUMP FNAME ADDR1-ADDR2
1569 *
1570 * DUMP IN ABS FORMAT.
1571 *
1572 *
047.223 1573 TB.DMP EQU *
1574 *
1575 * COMPUTE DUMP FWA
1576 *
047.223 072 030 045 1577 LDA CMD,AA
047.226 037 1578 RAR
047.227 332 240 047 1579 JC DMP0 DEFAULT FWA
047.232 052 031 045 1580 LHLD CMD,AA+1
047.235 042 252 057 1581 SHLD BFILHDR+ABS,LDA SET FWA
1582 *
1583 * COMPUTE LEN
1584 *
047.240 072 034 045 1585 DMP0 LDA CMD,AA+4
047.243 037 1586 RAR
047.244 332 317 047 1587 JC DMP2 LWA DEFAULTS
047.247 052 035 045 1588 LHLD CMD,AA+5
047.252 353 1589 XCHG
047.253 052 252 057 1590 LHLD BFILHDR+ABS,LDA
047.256 053 1591 DCX H /78.10.8C/
047.257 173 1592 MOV A,E
047.260 225 1593 SUB L
047.261 157 1594 MOV L,A
047.262 172 1595 MOV A,D
047.263 234 1596 SBB H
047.264 147 1597 MOV H,A (HL) = COUNT
047.265 332 276 047 1598 JC DMP1 LWA < FWA
047.270 042 254 057 1599 SHLD BFILHDR+ABS,LEN SET LENGTH
047.273 303 317 047 1600 JMP DMP2 OPEN FILE
1601 *
1602 * LWA < FWA
1603 *
047.276 315 136 031 1604 DMP1 CALL $TYPTX
047.301 007 114 127 1605 DB BELL,'LWA < FWA',ENL
047.314 303 225 045 1606 JMP TBGX EXIT
1607 *
1608 * OPEN DUMP FILE
1609 *
047.317 021 005 050 1610 DMP2 LXI D,DMPA USE 'SYOABS' AS DEFAULTS
047.322 041 215 057 1611 LXI H,MEMFB
047.325 315 103 054 1612 CALL $FOFEW
1613 *
1614 * WRITE HEADER INFO
1615 *
047.330 315 324 051 1616 CALL DRA, LOCATE PC
047.333 315 211 030 1617 CALL $HLIHL (HL) = (PC)
047.336 042 256 057 1618 SHLD BFILHDR+ABS,ENT SET ENTRY
047.341 041 377 000 1619 LXI H,FT,ABS*256+377Q
047.344 042 250 057 1620 SHLD BFILHDR SET BINARY ABS HEADER
047.347 001 010 000 1621 LXI B,ABS.COD

```

```

047.352 021 250 057 1622 LXI D,BFILHDR
047.355 041 215 057 1623 LXI H,MEMFB
047.360 315 005 055 1624 CALL $FWRIB WRITE HEADER BYTES TO FILE
047.363 052 254 057 1625 LHLD BFILHDR+ABS.LEN
047.366 104 1626 MOV B,H
047.367 115 1627 MOV C,L (BC) = COUNT
047.370 052 252 057 1628 LHLD BFILHDR+ABS.LDA
047.373 353 1629 XCHG (DE) = ADDRESS
047.374 041 215 057 1630 LXI H,MEMFB
047.377 315 005 055 1631 CALL $FWRIB WRITE BINARY
050.002 303 273 055 1632 JMP $FCLO CLOSE FILE
1633
050.005 123 131 060 1634 DMFA DB 'SYOABS' DEFAULTS FOR DUMP

```

```

1638 *** TB.LOAD - PROCESS *LOAD* COMMAND.
1639 *
1640 * LOAD FNAME
1641 *
1642 * LOAD ABS FILE INTO MEMORY.
1643
1644
050.013 1645 TB.LOA EQU *
050.013 021 147 050 1646 LXI D,LOAA DEFAULT TO 'SYOABS'
050.016 041 215 057 1647 LXI H,MEMFB
050.021 315 074 054 1648 CALL $FOPER OPEN FOR READ
050.024 001 010 000 1649 LXI B,ABS.COD
050.027 021 250 057 1650 LXI D,BFILHDR
050.032 315 234 054 1651 CALL $FREAB READ HEADER
050.035 332 114 050 1652 JC LOA2 PREMATURE EOF
050.040 052 250 057 1653 LHLD BFILHDR
050.043 054 1654 INR L
050.044 302 114 050 1655 JNZ LOA2 NOT BINARY FILE
000.000 1656 ERRNZ FT.ABS
050.047 174 1657 MOV A,H
050.050 247 1658 ANA A
050.051 302 114 050 1659 JNZ LOA2 NOT BINARY FILE
050.054 052 256 057 1660 LHLD BFILHDR+ABS.ENT (HL) = ENTRY POINT
050.057 345 1661 PUSH H
050.060 315 324 051 1662 CALL DRA. (HL) = ADDRESS OF USER PC
050.063 321 1663 POP D (DE) = NEW PC
050.064 163 1664 MOV M,E
050.065 043 1665 INX H
050.066 162 1666 MOV M,D
1667
1668 * SETUP LOAD FWA AND COUNT
1669
050.067 052 254 057 1670 LHLD BFILHDR+ABS.LEN
050.072 104 1671 MOV B,H
050.073 115 1672 MOV C,L (BC) = COUNT
050.074 052 252 057 1673 LHLD BFILHDR+ABS.LDA
050.077 124 1674 MOV D,H
050.100 135 1675 MOV E,L (DE) = FWA
050.101 011 1676 DAD B (HL) = LWA+1
050.102 345 1677 PUSH H SAVE FOR LATER
050.103 315 223 051 1678 CALL CLR CHECK LOAD RANGE
050.106 315 234 054 1679 CALL $FREAB READ DATA
050.111 322 321 050 1680 JNC LOA.2 CLOSE AND END, IF NO ERRORS
1681
1682 * FILE FORMAT ERROR
1683
050.114 315 136 031 1684 LOA2 CALL $TYPTX
050.117 007 106 117 1685 DB BELL,'FORMAT ERROR IN FIL','E'+2000
050.144 303 225 045 1686 JMP TBGX EXIT
1687
050.147 123 131 060 1688 LOAA DB 'SYOABS' DEFAULT LOAD

```



```

1690 *** TB.LOA. - PROCESS *LOAD* COMMAND.
1691 *
1692 * LOAD PIC FNAME ADDR
1693 *
1694 * LOAD PIC FILE INTO MEMORY AT LOCATION
1695
1696
050.155 1697 TB.LOA. EQU *
050.155 021 370 050 1698 LXI D,LOADB DEFAULTS OF 'SYOPIC'
050.160 041 215 057 1699 LXI H,MEMFB
050.163 315 074 054 1700 CALL $FOPER OPEN FILE
050.166 001 006 000 1701 LXI B,PIC.COD
050.171 021 250 057 1702 LXI D,BFILHDR
050.174 315 234 054 1703 CALL $FREAB READ HEADER
050.177 332 114 050 1704 JC LOA2 PREMATURE EOF
050.202 052 250 057 1705 LHLD BFILHDR
050.205 054 1706 INR L
050.206 302 114 050 1707 JNZ LOA2 NOT BINARY
000.000 1708 ERRNZ FT.PIC-1
050.211 045 1709 DCR H
050.212 302 114 050 1710 JNZ LOA2 NOT PIC
1711
1712 * LOAD CODE BEFORE RELOCATION
1713
050.215 052 254 057 1714 LHLD BFILHDR+PIC.PTR
050.220 001 372 377 1715 LXI B,-PIC.COD
050.223 011 1716 DAD B (HL) = BYTES TO READ
050.224 104 1717 MOV B,H
050.225 115 1718 MOV C,L
050.226 052 031 045 1719 LHLD CMD.AA+1
050.231 353 1720 XCHG (DE) = LOAD ADDRESS
050.232 315 223 051 1721 CALL CLR CHECK LOAD RANGE
050.235 315 234 054 1722 CALL $FREAB READ BYTES
050.240 332 114 050 1723 JC LOA2 FORMAT ERROR
1724
1725 * RELOCATE CODE
1726
050.243 325 1727 PUSH D SAVE NEXT FREE ADDRESS
050.244 052 031 045 1728 LHLD CMD.AA+1 (HL) = LOAD ADDRESS
050.247 001 372 377 1729 LXI B,-PIC.COD
050.252 011 1730 DAD B (HL) = RELOCATION FACTOR
050.253 104 1731 MOV B,H
050.254 115 1732 MOV C,L
050.255 041 215 057 1733 LXI H,MEMFB
050.260 305 1734 PUSH B SAVE RELOCATION FACTOR
050.261 001 002 000 1735 LXI B,2
050.264 021 312 044 1736 LXI D,LINE
050.267 315 234 054 1737 CALL $FREAB READ RELOCATION BYTES
050.272 301 1738 POP B RESTORE RELOCATION FACTOR
050.273 332 114 050 1739 JC LOA2 FORMAT ERROR
050.276 052 312 044 1740 LHLD LINE (HL) = REL ADDRESS OF WORD TO RELOCATE
050.301 174 1741 MOV A,H
050.302 265 1742 ORA L
050.303 312 321 050 1743 JZ LOA.2 ALL DONE
050.306 011 1744 DAD B (HL) = ABS ADDRESS OF WORD TO RELOCATE
050.307 176 1745 MOV A,M

```

TB.LOA.

050.310	201	1746	ADD	C	
050.311	167	1747	MOV	M,A	
050.312	043	1748	INX	H	
050.313	176	1749	MOV	A,M	
050.314	210	1750	ADC	B	
050.315	167	1751	MOV	M,A	RELOCATE WORD
050.316	303 255 050	1752	JMP	LOA.1	
		1753			
		1754	*	ALL DONE. PRINT NEXT FREE ADDRESS	
		1755			
050.321	041 215 057	1756	LOA.2 LXI	H, MEMFB	
050.324	315 273 055	1757	CALL	\$FCLO	CLOSE INPUT FILE
050.327	041 000 106	1758	LXI	H, 'F' *256	
050.332	042 022 045	1759	SHLD	CHD.8A	FORMAT DOUBLE OCTAL VALUE
050.335	041 000 000	1760	LXI	H, 0	
050.340	071	1761	DAD	SP	(HL) = ADDRESS OF VALUE
050.341	315 136 031	1762	CALL	\$TYPTX	
050.344	114 127 101	1763	DB	'LWA+1 =', ' '+2000	
050.354	315 045 052	1764	CALL	FVD	FORMAT VALUE FOR DISPLAY
050.357	341	1765	POP	H	
		1766			
		1767	*	RE-INITIALIZE THE DEFAULT CONSOLE DEFINITION BYTES	/79.12.6C/
		1768			
050.360	257	1769	XRA	A	/79.12.6C/
050.361	062 114 057	1770	STA	CSLMD	/79.12.6C/
050.364	062 115 057	1771	STA	CONFL	/79.12.6C/
		1772			
050.367	311	1773	RET		
		1774			
050.370	123 131 060	1775	LOAB DB	'SYOPIC'	DEFAULTS FOR PIC LOAD

```

1779 **      ANV - ACCEPT NEW VALUE.
1780 *
1781 *      ANV IS CALLED TO ACCEPT A NEW SINGLE OR DOUBLE BYTE VALUE.
1782 *      THE OLD VALUE IS TYPED OUT, FOLLOWED BY A '/', AND THEN
1783 *      A NEW VALUE MAY BE ENTERED.
1784 *
1785 *      IF MODE IS OCTAL OR DECIMAL, A BLANK TERMINATES THE
1786 *      CURRENT VALUE, A 'CR' TERMINATES THE CURRENT VALUE AND
1787 *      THE OPERATION. A NULL VALUE CAUSES THAT BYTE TO REMAIN
1788 *      UNCHANGED.
1789 *
1790 *      IN ASCII MODE, AN 'ESC' TERMINATES ENTRY.
1791 *
1792 *      ENTRY (HL) = START ADDRESS
1793 *      (DE) = LIMIT ADDRESS
1794
1795
050.376      1796 ANV EQU *
050.376 076 303 1797 MVI A,MI JMP
051.000 062 217 044 1798 STA FICA SET FLAG TO READ FROM ITY
1799
1800 *      TYPE OUT 'OLD VALUE'
1801
051.003 325      1802 ANV1 PUSH D SAVE (DE)
051.004 345      1803 PUSH H
051.005 315 045 052 1804 CALL FVD FORMAT VALUE FOR DISPLAY
051.010 341      1805 POP H
051.011 315 065 054 1806 CALL $TYFCH
051.014 057      1807 DB '/'
051.015 072 022 045 1808 LDA CMD,8A (A) = DISPLAY OPTION
051.020 026 012      1809 MVI D,10
051.022 376 104      1810 CPI 'D'
051.024 312 036 051 1811 JE ANV2 IF DECIMAL
051.027 026 010      1812 MVI D,8 ASSUME OCTAL (NOT SPECIFIED)
051.031 376 101      1813 CPI 'A'
051.033 312 104 051 1814 JE ANV6 IS ASCII
1815
1816 *      ACCUMULATE A DIGIT VALUE
1817
051.036 076 120      1818 ANV2 MVI A,80 (A) = DIGIT COUNT
051.040 315 353 043 1819 CALL ACN ACCUMULATE NUMBER
051.043 072 023 045 1820 LDA CMD,8A+1 (A) / 0 IF FOLLWORD
051.046 312 077 051 1821 JZ ANV5 IS NULL ENTRY
1822
1823 *      STORE ENTRY
1824
051.051 163      1825 MOV M,E STORE
051.052 043      1826 INX H
051.053 247      1827 ANA A
051.054 312 061 051 1828 ANV3 JZ ANV4 IF SINGLE BYTE
051.057 162      1829 MOV M,D
051.060 043      1830 INX H
1831
1832 *      ACCEPTED VALUE, IF HE TYPED '/', CONTINUE
1833 *      IF IS A CARRIAGE RETURN, STOP.
1834

```

```

051.061 321      1835 ANV4 POP      D
051.062 072 214 057 1836 LDA      $LSTIN
051.065 376 040      1837 CPI
051.067 300      1838 RNE
051.070 315 312 051 1839 ANV4*5 CALL    CUB      STOP IF NOT ' '
051.073 330      1840 RC        CHECK TO SEE IF DONE      /80.02.GC/
051.074 303 003 051 1841 JMP      ANV1    IF DONE
                                MORE DATA
                                1842
                                1843 *      NULL ENTRY
                                1844
051.077 043      1845 ANV5 INX      H
051.100 106      1846 MOV      B,M
051.101 303 054 051 1847 JMP      ANV3    ADJUST MEMORY POINTER
                                1848
                                1849
                                1850 **     IS ASCII VALUE
                                1851
051.104 315 150 053 1852 ANV6 CALL    $INCHA
051.107 376 004      1853 CPI      CTLD
051.111 312 001 046 1854 JE        EXIT      CTL-D
051.114 315 071 054 1855 CALL    $TYPC.    ECHO
051.117 321      1856 POP      D
051.120 376 033      1857 CPI      ESC
051.122 310      1858 RE
051.123 167      1859 MOV      M,A    EXIT IF BREAK
051.124 043      1860 INX      H
051.125 303 070 051 1861 JMP      ANV4*5
                                1862

                                1863 **     CEA - COMPUTE EFFECTIVE ADDRESS.
                                1864 *
                                1865 *     ENTRY (HL) = ADDRESS BLOCK
                                1866 *     EXIT  (HL) = EFFECTIVE ADDRESS
                                1867
                                1868
051.130 176      1869 CEA  MOV      A,M    (A) = FLAGS
051.131 017      1870 RRC
051.132 332 147 051 1871 JC      CEA1    IS BOTTOM VALUE
051.135 017      1872 RRC
051.136 332 153 051 1873 JC      CEA2    IS TOP VALUE
                                1874
                                1875 *     HAVE SPECIFIED ADDRESS.
                                1876
051.141 043      1877 INX      H
051.142 176      1878 MOV      A,M
051.143 043      1879 INX      H
051.144 146      1880 MOV      H,M
051.145 157      1881 MOV      L,A
051.146 311      1882 RET
                                1883
                                1884 *     HAVE BOTTOM (LAST+1) VALUE
                                1885
051.147 052 112 057 1886 CEA1 LHLD    BOTVAL
051.152 311      1887 RET

```

```

1888
1889 *      HAVE TOP (FIRST) VALUE
1890
051.153 052 110 057 1891 CEA2  LHL0  TOPVAL
051.156 311         1892      RET

1894 **     CLL - CHECK LINE LENGTHS.
1895 *
1896 *      CLL IS CALLED TO CHECK IF THE CURRENT LINE IS TOO LONG TO
1897 *      CONTINUE
1898 *
1899 *      USES      A,F
1900
1901
051.157         1902 CLL  EQU      *
051.157 072 330 040 1903      LDA  S.CUSOR
051.162 306 010     1904      ADI      8          SEE IF WILL RUN OVER
051.164 305         1905      PUSH     B
051.165 107         1906      MOV      B,A          (B) = CURRENT COLUMN NUMBER
051.166 072 331 040 1907      LDA  S.CONWI
051.171 270         1908      CMP      B
051.172 301         1909      POP      B
051.173 320         1910      RNC          NOT AT END
051.174 315 142 053 1911      CALL    $CRLF      NEW LINE
051.177 072 307 044 1912      LDA  PATCNT      DONT PRINT ADDRESS FOR CB.DAR
051.202 376 004     1913      CPI      TB.DARI
051.204 310         1914      RE          SKIP IT
051.205 174         1915      MOV      A,H
051.206 315 037 054 1916      CALL    $TOD      TYPE OCTAL DIGIT
051.211 175         1917      MOV      A,L
051.212 315 037 054 1918      CALL    $TOD      TYPE OCTAL DIGITS
051.215 315 136 031 1919      CALL    $TYPTX
051.220 040 240     1920      DB      ' ',' '+2000
051.222 311         1921      RET

1923 **     CLR - CHECK LOAD RANGE.
1924 *
1925 *      CLR IS CALLED BEFORE A MEMORY LOAD IS PERFORMED. IT REQUESTS
1926 *      SUFFICIENT MEMORY FROM HDOS, AND MAKES SURE THAT THE PROGRAM WILL
1927 *      NOT LOAD OVER DEBUG.
1928 *
1929 *      ENTRY  (BC) = TOTAL LENGTH OF LOAD
1930 *      (DE) = LOAD FWA
1931 *      EXIT   TO CALLER IF OK
1932 *      (HL) = $MEMFB
1933 *      TO APPROPRIATE ERROR HANDLER (AND THUS TO TBGX) IF ERROR
1934 *      USES   A,F,H,L
1935
1936
051.223 305         1937 CLR  PUSH     B

```

051.224	325	1938	PUSH	D	SAVE REGISTERS
051.225	353	1939	XCHG		
051.226	011	1940	DAD	B	(HL) = NEW LWA
051.227	377 052	1941	DB		SYS CALL, .SETIP
051.231	041 215 057	1942	LXI	H, MEMFB	POINT TO FILE IF ERROR
051.234	332 000 056	1943	JC	%FERROR	MEMORY OVERFLOW
051.237	321	1944	POP	D	
051.240	301	1945	POP	B	RESTORE REGISTERS
051.241	041 020 317	1946	LXI	H, -RMEML	
051.244	031	1947	DAD	D	SEE IF OVERLAYING DEBUG
051.245	041 215 057	1948	LXI	H, MEMFB	
051.250	330	1949	RC		NOT OVERLAYING DEBUG
051.251	315 136 031	1950	CALL	%TYP TX	
051.254	007 101 164	1951	DB	BELL, Attempt to Load Over DEBUG, ENL	
051.307	303 211 045	1952	JMP	RESTART	RESET FILES, ENTER COMMAND MODE

1954 \*\* CUB - CHECK UPPER BOUND /80.02.GC/

1955 \*

1956 \*

CUB check bounds to see if enough have been processed.

1957 \*

1958 \*

1959 \*

ENTRY: HL = NEXT BYTE

1960 \*

DE = LAST BYTE

1961 \*

1962 \*

EXIT: PSW = 'C' SET IF DONE

1963 \*

1964 \*

USES: PSW

1965 \*

1966

051.312	173	1967	CUB	MOV	A, E
051.313	225	1968		SUB	L
051.314	172	1969		MOV	A, D
051.315	234	1970		SBB	H
051.316	330	1971		RC	DONE

1972

051.317	174	1973		MOV	A, H
---------	-----	------	--	-----	------

051.320	265	1974		ORA	L
---------	-----	------	--	-----	---

051.321	300	1975		RNZ	NEXT ONE IS NOT ZERO
---------	-----	------	--	-----	----------------------

1976

051.322	067	1977		STC	FLAG IT DONE FOR NO WRAP THROUGH THE TOP
---------	-----	------	--	-----	------------------------------------------

051.323	311	1978		RET	
---------	-----	------	--	-----	--

1980 \*\* DRA - DETERMINE REGISTER ADDRESS.

1981 \*

1982 \*

ENTRY (DE) = ADDRESS OF \*DARA\* ENTRY

1983 \*

EXIT (HL) = ADDRESS OF VALUE IN MEMORY

1984 \*

'M' SET IF DOUBLE BYTE VALUE

1985 \*

USES A, F, D, E, H, L

1986

1987

051.324	021	136	057	1988	DRA.	LXI	D,DARAP	
051.327	023			1989	DRA	INX	D	
051.330	032			1990		LDAX	D	(A) = CODE
051.331	346	177		1991		ANI	1770	
051.333	052	226	045	1992		LHLD	REGPTR	
051.336	315	072	030	1993		CALL	\$DADA	
051.341	032			1994		LDAX	D	(A) = CODE
051.342	247			1995		ANA	A	SET CODE
051.343	311			1996		RET		

1998 \*\* DRI - DETERMINE REGISTER INDEX.  
1999 \*  
2000 \* ENTRY CMD.8A+1 = REGISTER CODE  
2001 \* EXIT (BC) = ADDRESS OF ENTRY IN \*PARA\*  
2002 \* USES A,B,C,D,F  
2003  
2004

051.344	072	024	045	2005	DRI	LDA	CMD.8A+2	
051.347	041	116	057	2006		LXI	H,DARA	
051.352	315	304	053	2007		CALL	\$TBLS	TABLE LOOKUP AND RETURN
051.355	053			2008		DCX	H	
051.356	353			2009		XCHG		
051.357	311			2010		RET		

2012 \*\* DRV - DISPLAY REGISTER VALUE.  
2013 \*  
2014 \* DRV DISPLAYS A REGISTER AS  
2015 \*  
2016 \* R=XXX IF 8 BIT, OR  
2017 \* R=XXXXX IF 16 BIT  
2018 \*  
2019 \* THE DISPLAY FORMAT OPTIONS MUST BE SET IN CMD.8A  
2020 \*  
2021 \* ENTRY (BC) = POINTER TO DARA ENTRY  
2022  
2023

051.360				2024	DRV	EQU	*	
051.360	032			2025		LDAX	D	
051.361	315	071	054	2026		CALL	\$TYPC.	TYPE REGISTER NAME
051.364	315	065	054	2027	DRV.	CALL	\$TYPCH	
051.367	075			2028		DB	'='	
051.370	315	327	051	2029		CALL	DRA	DETERMINE ADDRESS
051.373	346	200		2030		ANI	2000	
051.375	062	023	045	2031		STA	CMD.8A+1	SET NON-ZERO IF DOUBLE

```

2033 **      DVB - DISPLAY VALUE WITH BLANK.
2034 *
2035 *      DVB CALLS 'FVD', AND THEN FOLLOWS WITH A BLANK.
2036 *
2037 *
052.000 315 045 052 2038 DVB CALL FVD
052.003 076 040 2039 MVI A, ' '
052.005 303 071 054 2040 JMP $TYPC.          TYPE BLANK

```

```

2042 **      FBT - FIND BREAKPOINT IN TABLE.
2043 *
2044 *      ENTRY (BC) = ADDRESS
2045 *      EXIT (DE) = BKPT TABLE ADDRESS
2046 *      'Z' SET IF FOUND
2047 *      USES A,F
2048 *
2049 *
052.010 021 145 057 2050 FBT LXI D,BKPTAB
052.013 345 2051 PUSH H
052.014 046 010 2052 MVI H,BKPTBL
2053 *
052.016 032 2054 FBT1 LDAX D
052.017 251 2055 XRA C
052.020 302 032 052 2056 JNZ FBT2          IF NO MATCH
052.023 023 2057 INX D
052.024 032 2058 LDAX D
052.025 033 2059 DCX D
052.026 250 2060 XRA B
052.027 312 043 052 2061 JZ FBT3          BOTH MATCH: FOUND IT
2062 *
2063 *      CHECK NEXT ENTRY
2064 *
052.032 023 2065 FBT2 INX D
052.033 023 2066 INX D
052.034 023 2067 INX D
052.035 023 2068 INX D
052.036 045 2069 DCR H
052.037 302 016 052 2070 JNZ FBT1          IF MORE TO GO
052.042 262 2071 ORA D          CLEAR 'Z', NOT FOUND
052.043 341 2072 FBT3 POP H
052.044 311 2073 RET          EXIT

```

```

2075 **      FVD - FORMAT VALUE FOR DISPLAY.
2076 *
2077 *      FVD FORMATS THE SPECIFIED BYTE (OR DOUBLE-BYTE) AS SPECIFIED,
2078 *      AND ADDS IT TO THE LINE BEING BUILT.
2079 *
2080 *      IF NO FORMAT IS SPECIFIED, *OCTAL BYTE* IS USED.
2081 *
2082 *      IF A LINE IS LARGE ENOUGH ALREADY, IT IS TYPED AND

```



		2083	*	A NEW LINE IS STARTED.		
		2084	*			
		2085	*	ENTRY	(HL) = ADDRESS OF VALUE	
		2086	*	EXIT	(HL) ADVANCED	
		2087				
		2088				
052.045		2089	FVD	EQU	*	
052.045	315 157 051	2090		CALL	CLL	CHECK LINE LENGTH
		2091				
		2092	*	OUTPUT LEADING BLANK		
		2093				
052.050	325	2094		PUSH	D	SAVE (DE)
052.051	345	2095		PUSH	H	SAVE (HL)
052.052	072 022 045	2096		LDA	CMD.8A	
052.055	041 126 052	2097		LXI	H,FVDA	
052.060	247	2098		ANA	A	
052.061	312 073 052	2099		JZ	FVD0.1	/78.10.GC/
052.064	315 304 053	2100		CALL	\$TBLS	FIND IN TABLE
052.067	126	2101		MOV	D,M	(D) = PROCESSOR INDEX
052.070	303 074 052	2102		JMP	FVD0.2	/78.10.GC/
		2103				
052.073	127	2104	FVD0.1	MOV	D,A	/78.10.GC/
		2105				
052.074	041 124 052	2106	FVD0.2	LXI	H,FVD1	/78.10.GC/
052.077	343	2107		XTLH		SET RETURN ADDRESS, RESTORE (HL)
052.100	072 023 045	2108		LDA	CMD.8A+1	(A) = SINGLE/DOBLE FLAG
052.103	247	2109		ANA	A	'Z' SET IF SINGLE BYTE
052.104	365	2110		PUSH	PSW	
052.105	172	2111		MOV	A,D	(A) = FORMAT INDEX
052.106	126	2112		MOV	D,M	(D) = 1ST VALUE
052.107	043	2113		INX	H	
052.110	312 116 052	2114		JZ	FVD0	IF ONLY ONE BYTE
052.113	132	2115		MOV	E,D	(E) = 2ND VALUE
052.114	126	2116		MOV	D,M	
052.115	043	2117		INX	H	
052.116	315 076 031	2118	FVD0	CALL	\$TBRA	BRANCH TO PROCESSOR
		2119				
052.121	012	2120		DB	FVD.Q-*	OCTAL
052.122	023	2121		DB	FVD.D-*	DECIMAL
052.123	040	2122		DB	FVD.A-*	ASCII
		2123				
		2124				
052.124	321	2125	FVD1	POP	D	RESTORE (DE)
052.125	311	2126		RET		
		2127				
		2128				
052.126	104 001	2129	FVDA	DB	'D',1	DECIMAL
052.130	101 002	2130		DB	'A',2	ASCII
052.132	000	2131		DB	0	OCTAL

```

2133 **      FVD.Q - TYPE OCTAL VALUE.
2134
052.133 172 2135 FVD.Q MOV A,D
052.134 315 037 054 2136 CALL $TOD TYPE OCTAL DIGITS
052.137 361 2137 POP PSW
052.140 310 2138 RZ IF ONLY 1 BYTE
052.141 173 2139 MOV A,E
052.142 303 037 054 2140 JMP $TOD TYPE OCTAL DIGITS

```

```

2142 **      FVD.D - TYPE DECIMAL VALUE.
2143
052.145 361 2144 FVD.D POP PSW
052.146 076 005 2145 MVI A,5 ASSUME 5 DIGITS
052.150 302 160 052 2146 JNZ FVD.D1
052.153 132 2147 MOV E,D
052.154 026 000 2148 MVI D,0
052.156 076 003 2149 MVI A,3 3 DIGITS
052.160 303 337 053 2150 FVD.D1 JMP $TOD TYPE DECIMAL DIGITS

```

```

2152 **      FVD.A - TYPE ASCII VALUE.
2153
052.163 172 2154 FVD.A MOV A,D
052.164 315 020 054 2155 CALL $TPA TYPE PRINTING ASCII
052.167 361 2156 POP PSW
052.170 310 2157 RZ EXIT IF SINGLE
052.171 173 2158 MOV A,E
052.172 303 020 054 2159 JMP $TPA TYPE PRINTING ASCII

```

```

2161 **      RAS - RESOLVE ADDRESS SPECIFICATION.
2162 *
2163 *      ENTRY (HL) = CMD.AA
2164 *      (A) = ODD IF ADDRESS PAIR SPECIFIED
2165 *      EXIT (DE) = LWA
2166 *      (HL) = FWA
2167
052.175 2168
052.175 365 2169 RAS EQU *
052.176 315 130 051 2170 PUSH PSW SAVE (A)
052.201 353 2171 CALL CEA COMPUTE EFFECTIVE ADDRESS
052.202 041 034 045 2172 XCHG (DE) = FWA
052.205 361 2173 LXI H,CMD.AA+4
052.206 037 2174 POP PSW
052.207 322 230 052 2175 RAR
2176 JNC RAS1 IF DOUBLE ADDRESS SPECIFICATION
2177
2178 *      ADDR-ADDR
2179

```

052.212	315	130	051	2180	CALL	CEA	COMPUTE EFFECTIVE ADDRESS
052.215	353			2181	XCHG		(HL) = FWA, (DE) = LWA
052.216	173			2182	MOV	A,E	
052.217	225			2183	SUB	L	COMPARE TWO ADDRESSES
052.220	172			2184	MOV	A,D	
052.221	234			2185	SBB	H	
052.222	332	276	047	2186	JC	DMP1	FIRST > LAST
052.225	303	240	052	2187	JMP	RAS2	
				2188			
				2189	*	ADDRC/CNTJ	
				2190			
052.230	053			2191	RAS1	DCX	H
052.231	176			2192	MOV	A,M	(A) = (CMD,AA+3)
052.232	075			2193	DCR	A	
052.233	157			2194	MOV	L,A	
052.234	046	000		2195	MVI	H,0	(HL) = LENGTH SPECIFIED (0 IF NONE)
052.236	031			2196	DAD	D	(HL) = LWA
052.237	353			2197	XCHG		
				2198			
052.240	042	110	057	2199	RAS2	SHLD	TOPVAL
052.243	353			2200	XCHG		
052.244	043			2201	INX	H	
052.245	042	112	057	2202	SHLD	BOTVAL	
052.250	053			2203	DCX	H	
052.251	353			2204	XCHG		
052.252	311			2205	RET		
				2207	**	RBM - REMOVE BREAKPOINT FROM MEMORY.	
				2208	*		
				2209	*	RBM REMOVES SET BREAKPOINTS FROM MEMORY, BY RESTOREING THE	
				2210	*	ORIGINAL VALUES.	
				2211			
				2212			
052.253	001	145	057	2213	RBM	LXI	B,BKPTAB
052.256	026	011		2214	MVI	D,BKPTBL+1	
052.260	072	213	057	2215	LDA	BKFFLG	
052.263	247			2216	ANA	A	
052.264	310			2217	RZ		NO BREAKPOINTS SET
052.265	363			2218	DI		NO CTL-B WHILE SETTING BREAKPOINTS
				2219			
052.266	012			2220	RBM1	LDAX	B
052.267	157			2221	MOV	L,A	
052.270	003			2222	INX	B	
052.271	012			2223	LDAX	B	
052.272	147			2224	MOV	H,A	(HL) = ADDRESS OF BKPT
052.273	003			2225	INX	B	
052.274	003			2226	INX	B	
				2227			
				2228	*	RESTORE ORIGINAL VALUE	
				2229			
052.275	012			2230	LDAX	B	(A) = VALUE
052.276	167			2231	MOV	M,A	SET IN MEMORY
052.277	003			2232	INX	B	

052.300	025	2233	DCR	D	
052.301	302 266 052	2234	JNZ	RBM1	IF MORE IN TABLE
052.304	373	2235	EI		RESTORE INTERRUPTS
052.305	311	2236	RET		

2238	**	RFD - RESTORE FRONT PANEL DISPLAY.
2239	*	
2240	*	RFD IS CALLED TO RESTORE THE .CTLFLG OPTIONS STORED IN
2241	*	RFDA.
2242	*	
2243	*	ENTRY *RFDA* = CTLFLG VALUE
2244	*	EXIT .CTLFLG RESTORED
2245	*	USES A
2246		
2247		
052.306	076 000	2248 RFD MVI A,0
052.307		2249 RFDA EQU *-1
052.310	062 011 040	2250 STA .CTLFLG
052.313	323 360	2251 OUT OP.CTL

2253	**	RUC - RESTORE USER CONSOLE INVIRONMENT.
2254	*	
2255	*	RUC RESTORES THE USER CONSOLE FLAGS.
2256	*	
2257	*	ENTRY NONE
2258	*	EXIT NONE
2259	*	USES A,F
2260		
2261		

052.315	072 114 057	2262 RUC LDA CSLMD	
052.320	062 326 040	2263 STA S.CSLMD	STORE USER CONSOLE MODE
052.323	072 115 057	2264 LDA CONFL	
052.326	062 332 040	2265 STA S.CONFL	STORE CONSOLE FLAGS
052.331	311	2266 RET	
052.332	311	2267 RET	

2269	**	SBL - SET BREAKPOINT LIST.
2270	*	
2271	*	SBL IS CALLED TO SET A LIST OF BREAKPOINTS INTO THE TABLE.
2272	*	
2273	*	ENTRY (CMD.DA) = BREAKPOINTS
2274	*	EXIT SET IN TABLE
2275		
2276		

052.333		2277 SBL EQU *	CALLED AS SUBROUTINE
052.333	041 040 045	2278 LXI H,CMD.DA	
		2279	

```

2280 *      EXAMINE NEXT BREAKPOINT
2281
052.336 176 2282 SBL1  MOV    A,M          (A) = OPTION
052.337 017 2283      RRC
052.340 330 2284      RC          IF END OF LIST
052.341 043 2285      INX    H
2286
2287 *      FIND BREAKPOINT ALREADY IN LIST, OR EMPTY SPOT
2288
052.342 116 2289      MOV    C,M
052.343 043 2290      INX    H
052.344 106 2291      MOV    B,M          (BC) = ADDRESS
052.345 315 010 052 2292      CALL  FBT          FIND BREAKPOINT IN TABLE
052.350 312 366 052 2293      JE     SBL2          IF FOUND
052.353 305 2294      PUSH  B
052.354 001 000 000 2295      LXI    B,0
052.357 315 010 052 2296      CALL  FBT          FIND EMPTY SPOT
052.362 302 003 053 2297      JNE   SBL3          NO SPACE
052.365 301 2298      POP    B
2299
2300 *      HAVE SPOT. STORE VALUE
2301
052.366 353 2302 SBL2  XCHG
052.367 161 2303      MOV    M,C          SET VALUE IN TAL
052.370 043 2304      INX    H
052.371 160 2305      MOV    M,B
052.372 023 2306      INX    D          (DE) = ADDRESS OF REPEAT COUNT
052.373 032 2307      LDAX  D
052.374 043 2308      INX    H
052.375 167 2309      MOV    M,A          SET REPEAT COUNT
052.376 353 2310      XCHG
052.377 043 2311      INX    H
053.000 303 336 052 2312      JMP    SBL1          PROCESS NEXT BREAKPOINT
2313
2314 *      OUT OF SPACE
2315
053.003 315 136 031 2316 SBL3  CALL  $TPTX
053.006 007 116 117 2317      DB    BELL,'NO ROOM',ENL
053.017 303 225 045 2318      JMP    TBGX
2319

2320 **     SBM - SET BREAKPOINT IN MEMORY.
2321 *
2322 *     SBM SETS THE BREAKPOINT INSTRUCTIONS IN MEMORY PREPARATORY
2323 *     TO EXECUTION.
2324
053.022 001 145 057 2326 SBM   LXI    B,BKPTAB
053.025 026 011 2327      MVI    D,BKPTBL+1
053.027 072 213 057 2328      LDA    BKPFLG
053.032 247 2329      ANA    A
053.033 300 2330      RNZ
053.034 363 2331      DI          ALREADY IN MEMORY
2332      NO INTERRUPTS WHILE SETTING

```

```

053.035 012      2333 SBM1 LDAX B
053.036 157      2334 MOV L,A
053.037 003      2335 INX B
053.040 012      2336 LDAX B
053.041 147      2337 MOV H,A
053.042 003      2338 INX B
053.043 003      2339 INX B
                2340
                2341 * SET IT
                2342
053.044 176      2343 MOV A,M (A) = INSTRUCTION TO BE SAVED
053.045 002      2344 STAX B
053.046 066 327  2345 MVI M,MI.BKP SET BREAKPOINT
053.050 003      2346 SBM2 INX B
053.051 025      2347 DCR B
053.052 302 035 053 2348 JNZ SBM1 IF MORE TO CHECK
053.055 373      2349 EI RESTORE INTERRUPTS
053.056 311      2350 RET EXIT

```

```

                2352 ** SDC - SET DEBUGGER CONSOLE INVIRONMENT.
                2353 *
                2354 * SDC SAVES THE USER'S CONSOLE CONTROL FLAGS, AND INSTUTITES
                2355 * HBUG'S
                2356 *
                2357 * ENTRY NONE
                2358 * EXIT NONE
                2359 * USES A,F,H,L
                2360
                2361
053.057 041 326 040 2362 SDC LXI H,S.CSLMD
053.062 176      2363 MOV A,M
053.063 062 114 057 2364 STA CSLMD CLEAR CONSOLE MODE
053.066 066 201  2365 MVI M,CSL.ECH+CSL.CHR SET NO ECHO, CHAR MODE
053.070 056 332  2366 MVI L,#S.CONFL
000.040          2367 SET S.CSLMD/256
000.000          2368 ERRNZ S.CONFL/256-. MUST BE IN SAME PAGE
053.072 176      2369 MOV A,M
053.073 062 115 057 2370 STA CONFL SAVE USER CONSOLE FLAGS
053.076 066 000  2371 MVI M,0 CLEAR FLAGS
053.100 311      2372 RET

```

```

                2374 ** SSA - SET STARTING ADDRESS.
                2375 *
                2376 * SSA SETS AN ENTERED VALUE INTO THE USER PROGRAM PC REGISTER.
                2377 *
                2378 * ENTRY (HL) = ADDRESS OF VALUE BLOCK
                2379 * EXIT ADDRESS SET.
                2380
                2381
053.101 176      2382 SSA MOV A,M (A) = DEFAULT OPTION

```

SUBROUTINES.

SSA

15:30:11 16-MAY-80

053.102	017	2383	RRC		
053.103	330	2384	RC		IF DEFAULT
053.104	315 130 051	2385	CALL	CEA	COMPUTE EFFECTIVE ADDRESS
053.107	104	2386	MOV	B,H	
053.110	115	2387	MOV	C,L	
053.111	315 324 051	2388	CALL	DRA.	DETERMINE ADDRESS
053.114	161	2389	MOV	M,C	
053.115	043	2390	INX	H	
053.116	160	2391	MOV	M,B	
053.117	311	2392	RET		EXIT

053.120

2395

XTEXT MOVE

2397X \*\* \$MOVE - MOVE DATA  
2398X \*  
2399X \* \$MOVE MOVES A BLOCK OF BYTES TO A NEW MEMORY ADDRESS.  
2400X \* IF THE MOVE IS TO A LOWER ADDRESS, THE BYTES ARE MOVED FROM  
2401X \* FIRST TO LAST.  
2402X \*  
2403X \* IF THE MOVE IS TO A HIGHER ADDRESS, THE BYTES ARE MOVED FROM  
2404X \* LAST TO FIRST.  
2405X \*  
2406X \* THIS IS DONE SO THAT AN OVERLAPED MOVE WILL NOT 'RIPPLE'.  
2407X \*  
2408X \* ENTRY (BC) = COUNT  
2409X \* (DE) = FROM  
2410X \* (HL) = TO  
2411X \* EXIT MOVED  
2412X \* (DE) = ADDRESS OF NEXT FROM BYTE  
2413X \* (HL) = ADDRESS OF NEXT \*TO\* BYTE  
2414X \* 'C' CLEAR  
2415X \* USES ALL  
2416X  
2417X

030.252

053.120

2418X \$MOVE EQU 30252A IN H17 ROM  
2419 XTEXT SAVALL

2421X \*\* \$RSTALL - RESTORE ALL REGISTERS.  
2422X \*  
2423X \* \$RSTALL RESTORES ALL THE REGISTERS OFF THE STACK, AND  
2424X \* RETURNS TO THE PREVIOUS CALLER.  
2425X \*  
2426X \* ENTRY (SP) = PSW  
2427X \* (SP+2) = BC  
2428X \* (SP+4) = DE  
2429X \* (SP+6) = HL  
2430X \* (SP+8) = RET  
2431X \* EXIT TO \*RET\*, REGISTERS RESTORED  
2432X \* USES ALL  
2433X  
2434X  
031.047 2435X \$RSTALL EQU 31047A IN H17 ROM



```

2437X ** $SAVALL - SAVE ALL REGISTERS ON STACK.
2438X *
2439X * $SAVALL SAVES ALL THE REGISTERS ON THE STACK.
2440X *
2441X * ENTRY NONE
2442X * EXIT (SP) = PSW
2443X * (SP+2) = BC
2444X * (SP+4) = DE
2445X * (SP+6) = HL
2446X * USES H,L
2447X
2448X
031.054 2449X $SAVALL EQU 31054A IN H17 ROM
053.120 2450 XTEXT MCU

```

```

2452X ** MCU - MAP LOWER CASE TO UPPER CASE.
2453X *
2454X * MCU MAPS A LOWER CASE ALPHABETIC TO UPPER
2455X * CASE.
2456X *
2457X * ENTRY (A) = CHARACTER
2458X * EXIT (A) = CHARACTER RESULT
2459X * USES A,F
2460X
2461X
053.120 376 141 2462X $MCU CFI 'a'
053.122 330 2463X RC NOT LOWER CASE
053.123 376 173 2464X CFI 'z'+1
053.125 320 2465X RNC NOT LOWER CASE
053.126 326 040 2466X SUI 'a'-'A'
053.130 311 2467X RET
053.131 2468 XTEXT INDL

```

```

2470X ** $INDL - INDEXED LOAD.
2471X *
2472X * $INDL LOADS DE WITH THE TWO BYTES AT (HL)+DISPLACEMENT
2473X *
2474X * THIS ACTS AS AN INDEXED FULL WORD LOAD.
2475X *
2476X * (DE) = ( (HL) + DSPLACEMENT )
2477X *
2478X * ENTRY ((RET)) = DISPLACEMENT (FULL WORD)
2479X * (HL) = TABLE ADDRESS
2480X * EXIT TO (RET+2)
2481X * USES A,F,D,E
2482X
2483X
030.234 2484X $INDL EQU 30234A IN H17 ROM
053.131 2485 XTEXT HLIHL

```

\$HLIHL

```

2487X **      $HLIHL - LOAD HL INDIRECT THROUGH HL.
2488X *
2489X *      (HL) = ((HL))
2490X *
2491X *      ENTRY  NONE
2492X *      EXIT   NONE
2493X *      USES   A,H,L
2494X
030.211      2495X $HLIHL EQU 30211A      IN H17 ROM
053.131      2496      XTEXT TYPTX

```

```

2498X **      $TYPTX - TYPE TEXT.
2499X *
2500X *      $TYPTX IS CALLED TO TYPE A BLOCK OF TEXT ON THE SYSTEM CONSOLE.
2501X *
2502X *      IMBEDDED ZERO BYTES INDICATE A CARRIAGE RETURN LINE FEED,
2503X *      A BYTE WITH THE 200Q BIT SET IS THE LAST BYTE IN THE MESSAGE.
2504X *
2505X *      ENTRY  (RET) = TEXT
2506X *      EXIT   TO (RET+LENGTH)
2507X *      USES   A,F
2508X
031.136      2509X
2510X $TYPTX EQU 31136A      IN H17 ROM
2511X
031.144      2512X $TYPTX EQU 31144A      IN H17 ROM
053.131      2513      XTEXT RCHAR

```

```

2515X **      $RCHAR - READ SINGLE CHARACTER FROM CONSOLE.
2516X *
2517X *      ENTRY  NONE
2518X *      EXIT   (A) = CHARACTER
2519X *      USES   A,F
2520X
053.131 377 001      2521X
053.133 332 131 053 2522X $RCHAR DB SYSCALL,.SCIN
2523X JC $RCHAR      NOT READY
053.136 311      2524X RET
2525X
053.137 377 002      2526X $WCHAR DB SYSCALL,.SCOUT
053.141 311      2527X RET
053.142      2528      XTEXT CRLF

```

```

2530X **      $CRLF - TYPE CARRIAGE RETURN/ LINE FEED
2531X *
2532X *      $CRLF IS USED TO GENERATE PADDED CRLF'S.
2533X *
2534X *      ENTRY  NONE
2535X *      EXIT   (A) = 0
2536X *      USES   A,F
2537X
2538X
053.142 076 012 2539X $CRLF MVI    A,NL
053.144 377 002 2540X      DB     SYSCALL,.SCOUT
053.146 257      2541X      XRA     A
053.147 311      2542X      RET
053.150      2543X      XTEXT   DADA

```

```

2545X **      $DADA - PERFORM (H,L) = (H,L) + (0,A)
2546X *
2547X *      ENTRY  (H,L) = BEFORE VALUE
2548X *      (A) = BEFORE VALUE
2549X *      EXIT   (H,L) = (H,L) + (0,A)
2550X *      'C' SET IF OVERFLOW
2551X *      USES   F,H,L
2552X
2553X
030.072      2554X $DADA EQU     30072A      IN H17 ROM
053.150      2555X      XTEXT   DADA2

```

```

2557X **      $DADA. - ADD (0,A) TO (H,L)
2558X *
2559X *      ENTRY  NONE
2560X *      EXIT   (HL) = (HL) + (0A)
2561X *      USES   A,F,H,L
2562X
2563X
030.101      2564X $DADA. EQU     30101A      IN H17 ROM
053.150      2565X      XTEXT   INCHA

```

```

2567X **      $INCHA - READ ONE CHARACTER.
2568X *
2569X *      $INCHA READS ONE CHARACTER FROM THE TERMINAL.
2570X *
2571X *      CHAR = CTL-U: ERASE LINE
2572X *      = BKSP: BACKSPACE CHARACTER
2573X *      = RUBOUT: BACKSPACE CHARACTER
2574X
2575X *****8
2576X **

```

P 000.001 2577X ERRNZ 1 THIS ROUTINE IS OBSOLETE

2578X  
2579X \*\*\*\*\*  
2580X

2581X  
053.150 315 131 053 2582X \$INCHA CALL \$RCHAR READ A CHARACTER  
053.153 376 010 2583X CPI BKSP  
053.155 312 216 053 2584X JE INCO IS BKSP  
053.160 376 177 2585X CPI RUBOUT  
053.162 312 216 053 2586X JE INCO IS RUBOUT  
053.165 365 2587X PUSH PSW SAVE CODE  
053.166 072 303 053 2588X LDA \$INCHAA (A) = RUBOUT FLAG  
053.171 247 2589X ANA A  
053.172 304 137 053 2590X CNZ \$WCHAR ECHO RUBOUT CHAR. IF ANY  
053.175 257 2591X XRA A  
053.176 062 303 053 2592X STA \$INCHAA CLEAR FLAG  
053.201 361 2593X POP PSW  
053.202 376 025 2594X CPI 'U'-'@'  
053.204 300 2595X RNE NOT CTL-U, RETURN

2596X  
2597X \* IS CTL-U  
2598X

053.205 041 312 044 2599X LXI H,LINE  
053.210 315 142 053 2600X CALL \$CRLF  
053.213 303 245 053 2601X JMP INCI CLEAR LINE AND SET LINPTR

2602X  
2603X \* IS BKSP  
2604X

053.216 052 020 045 2605X INCO LHLD LINPTR  
053.221 076 312 2606X MVI A,\$LINE  
053.223 275 2607X CMP L  
053.224 312 150 053 2608X JE \$INCHA IF ALREADY AT FRONT  
053.227 053 2609X DCX H  
053.230 072 327 040 2610X LDA S,CONTY SEE IF BACKSPACING  
053.233 247 2611X ANA A  
053.234 362 255 053 2612X JP INC3 IS NON-CRT  
053.237 315 136 031 2613X CALL \$TYPTX  
053.242 010 040 210 2614X DB BKSP,' ',BKSP+2000 BACKSPACE FOR CRT  
053.245 042 020 045 2615X INC1 SHLD LINPTR  
053.250 066 000 2616X MVI M,0  
053.252 303 150 053 2617X JMP \$INCHA CLEAR ENTRY AGAIN

2618X  
2619X \* BACKSPACE FOR NON-CRT  
2620X

053.255 072 303 053 2621X INC3 LDA \$INCHAA (A) = FLAG  
053.260 247 2622X ANA A  
053.261 302 274 053 2623X JNZ INC4 AM STILL BACKSPACING  
053.264 076 057 2624X MVI A, '/'  
053.266 062 303 053 2625X STA \$INCHAA SET FLAG  
053.271 315 137 053 2626X CALL \$WCHAR TYPE  
053.274 176 2627X INC4 MOV A,M  
053.275 315 137 053 2628X CALL \$WCHAR SHOW CHARACTER BEING REMOVED  
053.300 303 245 053 2629X JMP INCI CLEAR IT  
2630X

053.303 000 2631X \$INCHAA DB 0 RUBOUT FLAG  
053.304 2632 XTEXT MUB6

```

2634X **      $MUB6 - MULTIPLY 8X16 UNSIGNED.
2635X *
2636X *      $MUB6 MULTIPLIES A 16 BIT VALUE BY A 8
2637X *      BIT VALUE.
2638X *
2639X *      ENTRY  (A) = MULTIPLIER
2640X *      (DE) = MULTIPLICAND
2641X *      EXIT  (HL) = RESULT
2642X *      'Z' SET IF NOT OVERFLOW
2643X *      USES  A,F,H,L
2644X
2645X
031.007      2646X $MUB6 EQU 31007A      IN H17 ROM
053.304      2647      XTEXT  TBL5

2649X **      $TBL5 - TABLE SEARCH
2650X *
2651X *      TABLE FORMAT
2652X *
2653X *      DB      KEY1,VAL1,
2654X *      .
2655X *      .
2656X *      DB      KEYN,VALN
2657X *      DB      0
2658X *
2659X *      ENTRY  (A) = PATTERN
2660X *      (H,L) = TABLE FWA
2661X *      EXIT  (A) = PATTERN IF FOUND
2662X *      'Z' SET IF FOUND
2663X *      'Z' CLEAR IF NOT FOUND OR PATTERN=0      /78.10.GC/
2664X *      USES  A,F,H,L
2665X
2666X
053.304 305      2667X $TBL5 PUSH B
053.305 376.000      2668X CPI 0
053.307 312 331 053 2669X JZ TBL2      /78.10.GC/
053.312 107      2670X MOV B,A      /78.10.GC/
053.313 176      2671X TBL1 MOV A,M      (A) = CHARACTER
053.314 043      2672X INX H
053.315 270      2673X CMP B
053.316 312 333 053 2674X JZ TBL3      IF MATCH
053.321 247      2675X ANA A
053.322 043      2676X INX H      SKIP PAST
053.323 302 313 053 2677X JNZ TBL1      IF NOT END OF TABLE
053.324 053      2678X DCX H
053.327 053      2679X DCX H
053.330 257      2680X XRA A      SET TO ZERO FOR OLD USERS      /78.10.GC/
053.331 376.001      2681X TBL2 CPI 1      CLEAR ZERO      /78.10.GC/
2682X
2683X *      DONE
2684X
053.333 301      2685X TBL3 POP B
053.334 311      2686X RET

```

053.335

2687

XTEXT TJMP

2689X \*\* \$TJMP - TABLE JUMP.  
2690X \*  
2691X \* USAGE  
2692X \*  
2693X \* CALL \$TJMP (A) = INDEX  
2694X \* DW ADDR1  
2695X \*  
2696X \*  
2697X \*  
2698X \* DW ADDR2  
2699X \*  
2700X \* ENTRY (A) = INDEX  
2701X \* EXIT TO PROCESSOR  
2702X \* (A) = INDEX\*2  
2703X \* USES NONE.  
2704X  
2705X

031.061

2706X \$TJMP EQU 31061A IN H17 ROM, (A) = INDEX\*2

031.062

2708X \$TJMP EQU 31062A IN H17 ROM

053.335

2709 XTEXT TDD

2711X \*\* \$TDD - TYPE DECIMAL DIGITS.  
2712X \*  
2713X \* \$TDD TYPES A 16 BIT VALUE AS 1 TO 5 DECIMAL DIGITS.  
2714X \*  
2715X \* ENTRY (D,E) = VALUE  
2716X \* (A) = DIGIT COUNT  
2717X \* EXIT VALUE TYPED.  
2718X \* USES A,B,C,F  
2719X  
2720X

053.335 076 005

2721X \$TDD, MVI A,5

053.337 345

2722X \$TDD, PUSH H

053.340 365

2723X TDD1 PUSH PSW

053.341 041 004 054

2724X LXI H,TDDA-2

053.344 007

2725X RLC (A) = DIGIT NUMBER\*2

053.345 315 101 030

2726X CALL \$DADA.

053.350 176

2727X MOV A,M

053.351 043

2728X INX H

053.352 146

2729X MOV H,M

053.353 157

2730X MOV L,A (HL) = MULTIPLE OF 10

053.354 353

2731X XCHG (DE) = DIVISOR, (HL) = VALUE

053.355 076 377

2732X MVI A,377Q

053.357 031

2733X TDD2 DAD D

053.360 074

2734X INR A

053.361 332 357 053

2735X JC TDD2 IF MORE TO GO

053.364 306 060

2736X ADI '0'

\$TDD

15:31:03 16-MAY-80

```

053.366 315 071 054 2737X CALL $TYFC. TYPE DIGIT
053.371 175 2738X MOV A,L
053.372 223 2739X SUB E
053.373 137 2740X MOV E,A REMOVE EXTRA SUBTRACTION
053.374 174 2741X MOV A,H
053.375 232 2742X SBB D
053.376 127 2743X MOV D,A
053.377 361 2744X POP PSW
054.000 075 2745X DCR A
054.001 302 340 053 2746X JNZ TDD1 IF MORE DIGITS
054.004 341 2747X POP H
054.005 311 2748X RET EXIT
2749X
054.006 2750X TDDA EQU *
054.006 377 377 2751X DW -1
054.010 366 377 2752X DW -10
054.012 234 377 2753X DW -100
054.014 030 374 2754X DW -1000
054.016 360 330 2755X DW -10000
054.020 2756 XTEXT TEA

```

```

2758X ** $TPA - TYPE PRINTING ASCII.
2759X *
2760X * $TPA TYPES AN ASCII CHARACTER. ALL NON-PRINTING CHARACTERS
2761X * ARE TYPED AS BLANKS.
2762X *
2763X * ENTRY (A) = CHARACTER
2764X * EXIT TYPED
2765X * USES A,F
2766X
2767X
054.020 376 040 2768X $TPA CPI 400
054.022 372 032 054 2769X JM TPA1 IF BAD
054.025 376 177 2770X CPI 1770
054.027 332 071 054 2771X JC $TYFC. OK, TYPE AND RETURN
054.032 076 040 2772X TPA1 MVI A, ' '
054.034 303 071 054 2773X JMP $TYFC. TYPE AND RETURN
054.037 2774 XTEXT TERA

```

```

2776X ** $TBRA - BRANCH RELATIVE THROUGH TABLE.
2777X *
2778X * $TBRA USES THE SUPPLIED INDEX TO SELECT A BYTE FROM THE
2779X * JUMP TABLE. THE CONTENTS OF THIS BYTE ARE ADDED TO THE
2780X * ADDRESS OF THE BYTE, YEILDING THE PROCESSOR ADDRESS.
2781X *
2782X * CALL $TBRA
2783X * DB LAB1-* INDEX = 0 FOR LAB1
2784X * DB LAB2-* INDEX = 1 FOR LAB2
2785X * DB LABN-* INDEX = N-1 FOR LABN
2786X *

```

	2787X *	ENTRY	(A) = INDEX	
	2788X *		(RET) = TABLE FWA	
	2789X *	EXIT	TO COMPUTED ADDRESS	
	2790X *	USES	F,H,L	
	2791X			
	2792X			
031.076	2793X \$TBRA	EDU	31076A	IN HI7 ROM
054.037	2794	XTEXT	TOD	

	2796X **	\$TOD - TYPE OCTAL DIGITS.		
	2797X *			
	2798X *	\$TOD TYPES AN OCTAL BYTE AS 3 OCTAL DIGITS, ZERO FILL.		
	2799X *			
	2800X *	ENTRY	(A) = VALUE	
	2801X *	EXIT	VALUE TYPES	
	2802X *	USES	A,F	
	2803X			
	2804X			
054.037 305	2805X \$TOD	PUSH	B	
054.040 006 003	2806X	MVI	B,3	
054.042 247	2807X	ANA	A	CLEAR CARRY
	2808X			
054.043 027	2809X TOD1	RAL		
054.044 027	2810X	RAL		
054.045 027	2811X	RAL		
054.046 365	2812X	PUSH	PSW	
054.047 346 007	2813X	ANI	7	
054.051 306 060	2814X	ADI	'0'	
054.053 315 071 054	2815X	CALL	\$TYPCH	TYPE CHARACTER
054.056 361	2816X	POP	PSW	
054.057 005	2817X	DCR	B	
054.060 302 043 054	2818X	JNZ	TOD1	IF MORE TO GO
054.063 301	2819X	POP	B	
054.064 311	2820X	RET		EXIT
054.065	2821	XTEXT	TYPCH	

	2823X **	\$TYPCH - TYPE SINGLE CHARACTER.		
	2824X *			
	2825X *	ENTRY	(RET) = CHARACTER	
	2826X *	EXIT	TO (RET)+1	
	2827X *		(A) = CHARACTER TYPED	
	2828X			
	2829X			
054.065 343	2830X \$TYPCH	XTHL		(HL) = RETURN ADDRESS
054.066 176	2831X	MOV	A,H	(A) = CHARACTER
054.067 043	2832X	INX	H	
054.070 343	2833X	XTHL		RESTORE ADVANCED EXIT ADDRESS
	2834X			
	2835X **	\$TYPCH - TYPE SINGLE CHARACTER.		
	2836X *			



		2837X *	ENTRY	(A) = CHARACTER	
		2838X *	EXIT	TO (RET)	
		2839X *			
054.071	377 002	2840X \$TYPCH.	DB	SYSCALL, SCOUT	
054.073	311	2841X *	RET		
054.074		2842 *	XTEXT	ZERO	

		2844X **	\$ZERO	- ZERO MEMORY	
		2845X *			
		2846X *	\$ZERO	ZEROS A BLOCK OF MEMORY.	
		2847X *			
		2848X *	ENTRY	(HL) = ADDRESS	
		2849X *		(B) = COUNT	
		2850X *	EXIT	(A) = 0	
		2851X *	USES	A,B,F,H,L	
		2852X *			
		2853X *			
031.212		2854X \$ZERO	EQU	31212A	IN H17 ROM
054.074		2855 *	XTEXT	FOPE	

		2857X **	\$FOPEX	- OPEN FILE BLOCK FOR I/O	
		2858X *			
		2859X *	\$FOPEX	IS CALLED BEFORE ANY I/O IS DONE VIA A	
		2860X *	FILE BLOCK. \$FOPEX	SETS UP THE FILE BLOCK, AND OPENS	
		2861X *	THE FILE VIA *HDOS*.		
		2862X *			
		2863X *	ENTRY	(DE) = ADDRESS OF DEFAULT BLOCK	
		2864X *		(HL) = ADDRESS OF FILE BLOCK	
		2865X *	EXIT	TO \$FERROR IF ERROR	
		2866X *		TO CALLER IF OK	
		2867X *	USES	A,F,B,C,D,E	
		2868X *			
		2869X *			
054.074	315 121 054	2870X \$FOPER	CALL	\$FOPER.	
054.077	320	2871X *	RNC		
054.100	303 000 056	2872X *	JMP	\$FERROR	IN ERROR
		2873X *			
054.103	315 124 054	2874X \$FOPEW	CALL	\$FOPEW.	
054.106	320	2875X *	RNC		
054.107	303 000 056	2876X *	JMP	\$FERROR	IN ERROR
		2877X *			
054.112	315 127 054	2878X \$FOPEU	CALL	\$FOPEU.	
054.115	320	2879X *	RNC		
054.116	303 000 056	2880X *	JMP	\$FERROR	IN ERROR
		2881X *			
		2882X *			
054.121	076 002	2883X \$FOPER.	MVI	A,FT,OR	FILE TYPE OF OPEN FOR READ
054.123	001	2884X *	DB	001Q	LXI,B TO SKIP NEXT MVI
054.124	076 004	2885X \$FOPEW.	MVI	A,FT,OW	OPEN FOR WRITE
054.126	001	2886X *	DB	001Q	LXI,B TO SKIP NEXT MIV

```

054.127 076 006      2887X $FOPEU. MVI      A,FT.0R+FT.0W
                      2888X
                      2889X *      (A) = FILE FLAGS
                      2890X
054.131 345          2891X      PUSH      H      SAVE FILE BLOCK ADDRESS
054.132 365          2892X      PUSH      PSW     SAVE NEW FLAGS
000.000              2893X      ERRNZ     FB.CHA
054.133 106          2894X      MOV       B,M      (B) = CHANNEL NUMBER
054.134 305          2895X      PUSH      B      SAVE HANNEL NUMBER
000.000              2896X      ERRNZ     FB.FLG-FB.CHA-1
054.135 043          2897X      INX        H
054.136 117          2898X      MOV       C,A      (C) = NEW FILE FLAGS
054.137 176          2899X      MOV       A,M      (A) = CURRENT TYPE
054.140 247          2900X      ANA        A
054.141 171          2901X      MOV       A,C      (A) = NEW FLAGS TO BE SET
054.142 312 154 054 2902X      JZ        $FOPE1    NOT ALREADY OPEN
                      2903X
                      2904X *      ALREADY OPEN. SQUACK
                      2905X
054.145 301          2906X      POP        B      RESTORE (BC)
054.146 361          2907X      POP        PSW     DISCARD NEW FLAGS
054.147 341          2908X      POP        H      (HL) = FB ADDRESS
054.150 076 031      2909X      MVI       A,EC.FAO  FILE ALREADY OPEN
054.152 067          2910X      STC
054.153 311          2911X      RET
                      2912X
000.000              2913X      ERRNZ     FB.FWA-FB.FLG-1
054.154 043          2914X $FOPE1      INX        H      (HL) = $FB.FWA
054.155 116          2915X      MOV       C,M
054.156 043          2916X      INX        H
054.157 106          2917X      MOV       B,M      (BC) = FB.FWA
054.160 043          2918X      INX        H
000.000              2919X      ERRNZ     FB.PTR-FB.FWA-2
054.161 161          2920X      MOV       M,C      SET FB.PTR = FB.FWA
054.162 043          2921X      INX        H
054.163 160          2922X      MOV       M,B
054.164 043          2923X      INX        H
000.000              2924X      ERRNZ     FB.LIM-FB.PTR-2
054.165 161          2925X      MOV       M,C      SET FB.LIM = FB.FWA
054.166 043          2926X      INX        H
054.167 160          2927X      MOV       M,B
054.170 043          2928X      INX        H
000.000              2929X      ERRNZ     FB.NAM-FB.LIM-4
054.171 043          2930X      INX        H
054.172 043          2931X      INX        H      (HL) = $FB.NAM
                      2932X
                      2933X *      FILE BLOCK POINTERS SETUP. OPEN FILE
                      2934X
054.173 345          2935X      PUSH      H      SAVE NEW ADDRESS FOR NAME
054.174 041 225 054 2936X      LXI       H,$FOPEB
054.177 247          2937X      ANA        A
054.200 312 207 054 2938X      JZ        $FOPE2
000.000              2939X      ERRNZ     .EXIT
054.203 315 304 053 2940X      CALL     $TBLS    FIND CODE
054.206 176          2941X      MOV       A,M
054.207 062 215 054 2942X $FOPE2      STA     $FOPEA    SET SYSCALL CODE

```

/78.10.6C/

054.212	341	2943X	POP	H	(HL) = #FB.NAM
054.213	361	2944X	POP	PSW	(A) = CHANNEL NUMBER
054.214	377 000	2945X	DB		SYSCALL,.EXIT
054.215		2946X	EQU	*-1	SYSCALL CODE
054.216	321	2947X	POP	D	(D) = NEW FLAG
054.217	341	2948X	POP	H	(HL) = FILE BLOCK ADDRESS
054.220	330	2949X	RC		EXIT IF ERROR
054.221	043	2950X	INX	H	
000.000		2951X	ERRNZ	FB.FLG-1	
054.222	162	2952X	MOV	M,D	SET NEW FLAGS
054.223	053	2953X	DCX	H	RESTORE (HL)
054.224	311	2954X	RET		
		2955X			
054.225	002 042	2956X	DB	FT.0R,.OPENR	TABLE OF SYSCALL CODES
054.227	004 043	2957X	DB	FT.0W,.OPENW	
054.231	006 044	2958X	DB	FT.0R+FT.0W,.OPENU	
054.233	000	2959X	DB	0	SHOULD NOT OCCUR
054.234		2960	XTEXT	FREAB	

		2962X	**		\$FREAB - READ BYTES FROM FILE BUFFER.
		2963X	*		
		2964X	*		\$FREAB IS CALLED TO READ A NUMBER OF BYTES FROM A FILE BUFFER.
		2965X	*		
		2966X	*	ENTRY	(BC) = BYTE COUNT
		2967X	*		(DE) = FWA FOR BYTES
		2968X	*		(HL) = ADDRESS OF FILE BUFFER
		2969X	*	EXIT	TO \$FERROR* IF ERROR
		2970X	*		TO CALLER IF OK
		2971X	*		(BC) = UNREAD BYTE COUNT (ONLY IF EOF)
		2972X	*		(DE) = ADDRESS OF FIRST UNUSED BYTE
		2973X	*		'C' SET IF EOF DURING READ
		2974X	*	USES	A,F,B,C,D,E
		2975X			
		2976X			
054.234	315 247 054	2977X	\$FREAB	CALL	\$FREAB.
054.237	320	2978X	RNC		RETURN IF OK
054.240	376 001	2979X	CPI	EC.EOF	
054.242	302 000 056	2980X	JNE	\$FERROR	ERROR IS NOT EOF
054.245	067	2981X	STC		
054.246	311	2982X	RET		ERROR IS SIMPLY EOF
		2983X			
		2984X			
054.247		2985X	\$FREAB	EQU	*
054.247	257	2986X	XRA	A	
054.250	062 236 056	2987X	STA	EOFFLG	CLEAR EOF FLAG
054.253	345	2988X	PUSH	H	
054.254	315 062 056	2989X	CALL	CBT	COPY BUFFER POINTERS TO TEMP CELLS
		2990X			
		2991X	*		COPY DATA FROM BUFFER TO TARGET
		2992X			
054.257	325	2993X	\$REAB2	PUSH	D
054.260	072 225 056	2994X	LDA	T.FLG	SAVE TARGET ADDRESS
054.263	346 002	2995X	ANI	FT.0R	

```

054.265 076 011 2996X MVI A,EC.FND ASSUME FILE NOT OPEN FOR READ
054.267 067 2997X STC
054.270 312 000 055 2998X JZ $REAB8 NOT OPEN FOR READ
054.273 170 2999X MOV A,B
054.274 261 3000X ORA C
054.275 312 000 055 3001X JZ $REAB8 ALL DONE
3002X
3003X * COMPUTE MIN( DATA IN BUFFER, DATA REQUESTED)
3004X
054.300 052 230 056 3005X $REAB3 LHLD T.PTR
054.303 353 3006X XCHG (DE) = (FB.PTR) = ADDRESS OF DATA
054.304 052 232 056 3007X LHLD T.LIM (HL) = LIMIT ADDRESS
054.307 175 3008X MOV A,L
054.310 223 3009X SUB E
054.311 157 3010X MOV L,A
054.312 174 3011X MOV A,H
054.313 232 3012X SBB D
054.314 147 3013X MOV H,A (HL) = NUMBER OF BYTES IN BUFFER
054.315 171 3014X MOV A,C
054.316 225 3015X SUB L COMPARE REQUESTED TO AVAILABLE
054.317 170 3016X MOV A,B
054.320 234 3017X SBB H
054.321 322 326 054 3018X JNC $REAB4 MORE REQUESTED THEN AVAILABLE
054.324 140 3019X MOV H,B
054.325 151 3020X MOV L,C LIMIT TRANSFER TO REQUEST COUNT
054.326 174 3021X $REAB4 MOV A,H
054.327 265 3022X ORA L
054.330 302 344 054 3023X JNZ $REAB6 SOME IN BUFFER
3024X
3025X * BUFFER IS EMPTY, RE-FILL IT
3026X
054.333 315 142 056 3027X CALL $FFB FILL FILE BUFFER
054.336 332 000 055 3028X JC $REAB8 ERROR CONDITION
054.341 303 300 054 3029X JMP $REAB3 COUNT NEW DATA
3030X
3031X * GOT THE DATA, MOVE IT FROM BUFFER TO TARGET
3032X *
3033X * (BC) = REQUESTED COUNT
3034X * (DE) = FROM
3035X * (HL) = COUNT
3036X * ((SP)) = TO
3037X
054.344 171 3038X $REAB6 MOV A,C
054.345 225 3039X SUB L
054.346 117 3040X MOV C,A
054.347 170 3041X MOV A,B
054.350 234 3042X SBB H
054.351 107 3043X MOV B,A REMOVE BYTES ABOUT TO BE MOVED FROM REQUEST COUNT
054.352 305 3044X PUSH B
054.353 343 3045X XTHL (HL) = REMAINING REQUEST COUNT
054.354 301 3046X POP B (BC) = COUNT FOR THIS COPY
054.355 343 3047X XTHL (HL) = TARGET ADDR, ((SP)) = REMAINING REQ. COUNT
054.356 032 3048X $REAB7 LDAX D
054.357 167 3049X MOV M,A
054.360 023 3050X INX D
054.361 043 3051X INX H

```

```

054.362 013      3052X      DCX      B
054.363 170      3053X      MOV      A,B
054.364 261      3054X      ORA      C
054.365 302 356 054 3055X      JNZ      $REAB7      MORE TO GO
054.370 353      3056X      XCHG
054.371 042 230 056 3057X      SHLD     T,PTR      UPDATE POINTER
054.374 301      3058X      POP      B      (BC) = REMAINING COUNT
054.375 303 257 054 3059X      JMP      $REAB2      SEE IF MORE IN BUFFER
                3060X
                3061X *      READ COMPLETE.
                3062X *
                3063X *      (PSW) = COMPLETION FLAGS
                3064X
055.000 321      3065X $REAB8 POP      D      RESTORE TARGET ADDRESS
055.001 341      3066X      POP      H
055.002 303 110 056 3067X      JMP      CTB      COPY TEMP POINTERS BACK TO BLOCK, EXIT
055.005      3068X      XTEXT    FWRIB

```

```

                3070X **      $FWRIB - WRITE BYTES FROM FILE BUFFER.
                3071X *
                3072X *      $FWRIB IS CALLED TO WRITE A NUMBER OF BYTES FROM A FILE BUFFER.
                3073X *
                3074X *      ENTRY (BC) = BYTE COUNT
                3075X *      (DE) = FWA FOR BYTES
                3076X *      (HL) = ADDRESS OF FILE BUFFER
                3077X *      EXIT TO *FERROR* IF ERROR
                3078X *      TO CALLER IF OK
                3079X *      (DE) = ADDRESS OF FIRST UNWRITTEN BYTE
                3080X *      USES A,F,B,C,D,E
                3081X
                3082X
055.005 315 014 055 3083X $FWRIB CALL    $FWRIB.
055.010 320      3084X      RNC      RETURN IF OK
055.011 303 000 056 3085X      JMP      $FERROR      ERROR
                3086X
                3087X
055.014      3088X $FWRIB, EQU      *
055.014 345      3089X      PUSH     H
055.015 315 062 056 3090X      CALL    CBT      COPY BUFFER POINTERS TO TEMP CELLS
                3091X
                3092X *      COPY DATA FROM USER AREA TO BUFFER
                3093X
055.020 325      3094X $WRIB2 PUSH     D      SAVE AREA ADDRESS
055.021 072 225 056 3095X      LDA      T,FLG
055.024 346 004      3096X      ANI      FT,0W      SEE IF OPEN FOR WRITE
055.026 312 162 055 3097X      JZ      $WRIB8      FILE NOT OPEN FOR WRITE
055.031 170      3098X      MOV      A,B
055.032 261      3099X      ORA      C
055.033 312 162 055 3100X      JZ      $WRIB8      ALL DONE
                3101X
                3102X *      COMPUTE MIN( ROOM IN BUFFER, WRITE COUNT REQUESTED)
                3103X
055.036 052 230 056 3104X $WRIB3 LHL D      T,PTR

```

```

055.041 353      3105X      XCHG      (DE) = (FB.PTR) = ADDRESS OF ROOM
055.042 052 234 056 3106X      LHLD      T,LWA      (HL) = LIMIT ADDRESS
055.045 175      3107X      MOV      A,L
055.046 223      3108X      SUB      E
055.047 157      3109X      MOV      L,A
055.050 174      3110X      MOV      A,H
055.051 232      3111X      SBB      D
055.052 147      3112X      MOV      H,A      (HL) = BYTES OF ROOM IN BUFFER
055.053 171      3113X      MOV      A,C      COMPARE REQUESTED COUNT TO BUFFER ROOM
055.054 225      3114X      SUB      L
055.055 170      3115X      MOV      A,B
055.056 234      3116X      SBB      H
055.057 322 064 055 3117X      JNC      $WRIB4      MORE REQUESTED THEN ROOM
055.062 140      3118X      MOV      H,B
055.063 151      3119X      MOV      L,C      USE REQUESTED COUNT
055.064 174      3120X $WRIB4 MOV      A,H
055.065 265      3121X      ORA      L
055.066 302 126 055 3122X      JNZ      $WRIB6      SOME ROOM IN BUFFER
3123X
3124X *      BUFFER IS FULL, EMPTY IT
3125X
055.071 305      3126X      PUSH     B      SAVE COUNT
055.072 052 226 056 3127X      LHLD      T,FWA
055.075 042 230 056 3128X      SHLD     T,PTR      CLEAR REMOVAL POINTER
055.100 353      3129X      XCHG
055.101 052 234 056 3130X      LHLD      T,LWA
055.104 175      3131X      MOV      A,L
055.105 223      3132X      SUB      E
055.106 117      3133X      MOV      C,A
055.107 174      3134X      MOV      A,H
055.110 232      3135X      SBB      D
055.111 107      3136X      MOV      B,A      (BC) = DATA IN BUFFER
055.112 072 224 056 3137X      LDA      T,CHA
055.115 377 005      3138X      DB      SYSCALL,WRITE WRITE BUFFER
055.117 301      3139X      POP      B      (BC) = DESIRED COUNT
055.120 322 036 055 3140X      JNC      $WRIB3      GOT THE DATA
3141X
3142X *      ERROR ON WRITE.
3143X
055.123 303 162 055 3144X      JMP      $WRIB8      HAVE ERROR
3145X
3146X *      GOT THE DATA, MOVE IT FROM BUFFER TO TARGET
3147X *
3148X *      (BC) = REQUEST COUNT
3149X *      (DE) = TO
3150X *      (HL) = COUNT
3151X *      ((SP)) = FROM
3152X
055.126 171      3153X $WRIB6 MOV      A,C
055.127 225      3154X      SUB      L
055.130 117      3155X      MOV      C,A
055.131 170      3156X      MOV      A,B
055.132 234      3157X      SBB      H
055.133 107      3158X      MOV      B,A      REMOVE BYTES ABOUT TO BE MOVED FROM REQUEST COUNT
055.134 305      3159X      PUSH     B
055.135 343      3160X      XTHL      (HL) = REMAINING REQUEST COUNT

```

```

055.136 301      3161X      POP      B      (BC) = COUNT FOR THIS COPY
055.137 343      3162X      XTHL             (HL) = TARGET ADDR, ((SP)) = REMAINING REQ. COUNT.
055.140 176      3163X $FWRIB7 MOV      A,M
055.141 022      3164X      STAX      D
055.142 023      3165X      INX       D
055.143 043      3166X      INX       H
055.144 013      3167X      DCX       B
055.145 170      3168X      MOV      A,B
055.146 261      3169X      ORA       C
055.147 302 140 055 3170X      JNZ      $FWRIB7      MORE TO GO
055.152 353      3171X      XCHG
055.153 042 230 056 3172X      SHLD    T,PTR      UPDATE POINTER
055.156 301      3173X      POP      B      (BC) = REMAINING COUNT
055.157 303 020 055 3174X      JMP      $FWRIB2      SEE IF MORE IN BUFFER
3175X
3176X *      WRITE COMPLETE.
3177X *
3178X *      (PSW) = COMPLETION FLAGS
3179X
055.162 321      3180X $FWRIB8 POP      D      RESTORE TARGET ADDRESS
055.163 341      3181X      POP      H
055.164 303 110 056 3182X      JMP      CTB      COPY TEMP POINTERS BACK TO BLOCK, EXIT

```

```

3184X **      $FWBRK - BREAKOUTPUT /80.02.GC/
3185X *
3186X *      $FWBRK empties the specified buffer by filling it with NULLs
3187X *      and then writing it. Note this is used to insure that block
3188X *      mode I/O is output if it is not really a serial device (es.
3189X *      writing to AT: from *EDIT*.
3190X *
3191X *
3192X *      ENTRY: HL = FILE BLOCK POINTER
3193X *
3194X *      EXIT: HL = FILE BLOCK POINTER
3195X *      TO $FERROR IF ERROR
3196X *
3197X *      USES: PSW,BC,DE
3198X *
3199X
055.167 315 176 055 3200X $FWBRK CALL    $FWBRK.
055.172 320      3201X      RNC              NO ERROR
3202X
055.173 303 000 056 3203X      JMP      $FERROR
3204X
055.176 345      3205X $FWBRK. PUSH     H
055.177 315 062 056 3206X      CALL    CBT      COPY BUFFER TO TEMPORARY
055.202 315 212 055 3207X      CALL    $FWBRK1
055.205 341      3208X      POP      H
055.206 315 110 056 3209X      CALL    CTB      COPY TEMPORARY TO BUFFER
055.211 311      3210X      RET
3211X
055.212 052 234 056 3212X $FWBRK1 LHLD    T,LWA
055.215 353      3213X      XCHG      DE = BUFFER LWA

```

```

055.216 052 230 056 3214X      LHLD    T,PTR      HL = BUFFER PTR
055.221 173          3215X      MOV     A,E
055.222 225          3216X      SUB     L
055.223 117          3217X      MOV     C,A
055.224 172          3218X      MOV     A,D
055.225 234          3219X      SBB     H
055.226 107          3220X      MOV     B,A      BC = DE - HL
055.227 261          3221X      ORA     C
055.230 310          3222X      RZ          THE BUFFER IS ALREADY FLUSHED
          3223X
          3224X *      FILL THE BUFFER WITH NULLS
          3225X
055.231 170          3226X FWBRK2 MOV     A,B
055.232 261          3227X      ORA     C
055.233 312 245 055 3228X      JZ      FWBRK3      NO MORE LEFT TO FILL
          3229X
055.236 066 000      3230X      MVI     M,0
055.240 043          3231X      INX     H
055.241 013          3232X      DCX     B
055.242 303 231 055 3233X      JMP     FWBRK2
          3234X
055.245 052 226 056 3235X FWBRK3 LHLD    T,FWA
055.250 042 230 056 3236X      SHLD   T,PTR
055.253 353          3237X      XCHG
055.254 052 234 056 3238X      LHLD    T,LWA      DE = BUFFER FWA
          HL = BUFFER LWA
055.257 175          3239X      MOV     A,L
055.260 223          3240X      SUB     E
055.261 117          3241X      MOV     C,A
055.262 174          3242X      MOV     A,H
055.263 232          3243X      SBB     D
055.264 107          3244X      MOV     B,A      BC = HL - DE ( BC = COUNT )
055.265 072 224 056 3245X      LDA     T,CHA
055.270 377 005      3246X      DB      SYSCALL,WRITE
055.272 311          3247X      RET
055.273          3248      XTEXT   FCLO

```

```

          3250X **      $FCLO - CLOSE FILE BLOCK.
          3251X *
          3252X *      $FCLO IS CALLED TO TERMINATE PROCESSING THROUGH A FILE
          3253X *      BLOCK.
          3254X *
          3255X *      ENTRY (HL) = FILE BLOCK ADDRESS
          3256X *      EXIT TO $FERROR IF ERROR
          3257X *      TO CALLER IF OK
          3258X *      USES A,F,B,C,D,E
          3259X
          3260X
055.273 315 302 055 3261X $FCLO CALL    $FCLO.
055.276 320          3262X      RNC          NO ERROR
055.277 303 000 056 3263X      JMP     $FERROR
          3264X
055.302 345          3265X $FCLO. PUSH   H      SAVE FILE BLOCK ADDRESS
000.000          3266X      ERNZ    FB,FLG-1

```



```

055.303 043      3267X      INX      H      (HL) = #FB.FLG
055.304 176      3268X      MOV      A,M
055.305 066 000    3269X      MVI      M,0      CLEAR FLAG
055.307 247      3270X      ANA      A
055.310 312 376 055 3271X      JZ      $FCLO4      FILE NOT OPEN
055.313 346 004    3272X      ANI      FT.OW
055.315 312 370 055 3273X      JZ      $FCLO3      NO WRITING, NO FLUSHING NEEDED
3274X
3275X *          WAS OPEN FOR WRITE. SEE IF NEED FLUSH THE LAST SECTOR
3276X
055.320 315 234 030 3277X      CALL     $INDL
055.323 003 000    3278X      DW      FB.PTR-FB.FLG
055.325 325      3279X      PUSH     D      SAVE (FB.PTR)
055.326 315 234 030 3280X      CALL     $INDL      (DE) = (FB.FWA)
055.331 001 000    3281X      DW      FB.FWA-FB.FLG
055.333 341      3282X      POP      H      (HL) = (FB.PTR)
055.334 175      3283X      MOV      A,L
055.335 223      3284X      SUB      E
055.336 117      3285X      MOV      C,A
055.337 174      3286X      MOV      A,H
055.340 232      3287X      SBB      D
055.341 107      3288X      MOV      B,A      (BC) = AMOUNT IN BLOCK
055.342 261      3289X      ORA      C
055.343 312 370 055 3290X      JZ      $FCLO3      NONE TO FLUSH
3291X
3292X *          NEED TO FLUSH BUFFER
3293X *
3294X *          (BC) = DATA AMOUNT
3295X *          (DE) = FWA
3296X *          (HL) = LWA+1
3297X
055.346 171      3298X      MOV      A,C
055.347 247      3299X      ANA      A
055.350 312 363 055 3300X      JZ      $FCLO2      DONT HAVE PARTIAL SECTOR
3301X
3302X *          ZERO FILL PARTIAL SECTOR
3303X
055.353 066 000    3304X $FCLO1 MVI      M,0
055.355 043      3305X      INX      H
055.356 014      3306X      INR      C
055.357 302 353 055 3307X      JNZ     $FCLO1
055.362 004      3308X      INR      B      COUNT ANOTHER FULL SECTOR
055.363 341      3309X $FCLO2 POP      H      (HL) = FB FWA
055.364 176      3310X      MOV      A,M      (A) = CHANNEL NUMBER
000.000      3311X      ERRNZ   FB.CHA
055.365 345      3312X      PUSH     H
055.366 377 005    3313X      DB      SYSCALL,.WRITE      FLUSH
3314X
3315X *          READY TO CLOSE FILE.
3316X *
3317X *          'C' SET IF ERROR
3318X *          (A) = ERROR CODE
3319X
055.370 341      3320X $FCLO3 POP      H      (HL) = FILE BLOCK ADDRESS
055.371 330      3321X      RC          ERROR
000.000      3322X      ERRNZ   FB.CHA

```

055.372	176	3323X	MOV	A,M	(A) = CHANNEL NUMBER
055.373	345	3324X	PUSH	H	
055.374	377 046	3325X	DB	SYSCALL,.CLOSE	CLOSE CHANNEL
055.376	341	3326X \$FCLO4	POP	H	(HL) = FILE BLOCK ADDRESS
055.377	311	3327X	RET		
056.000		3328	XTEXT	FERROR	

3330X \*\* \$FERROR - PROCESS FILE ERRORS.  
3331X \*  
3332X \* \$FERROR IS CALLED TO COMPLAIN ABOUT AN ERROR ENCOUNTERED  
3333X \* WHEN PROCESSING FILES.  
3334X \*  
3335X \* ENTRY (A) = ERROR CODE  
3336X \* (HL) = ADDRESS OF FILE NAME - FB.NAM  
3337X \* EXIT TO RESTART  
3338X \* USES ALL

056.000	365	3341X \$FERROR	PUSH	PSW	SAVE CODE
056.001	315 136 031	3342X	CALL	\$TYPTX	
056.004	012 007 105	3343X	DB	NL,BELL,'ERROR ON FILE', ' '+2000	
056.024	021 012 000	3344X	LXI	D,FB.NAM	
056.027	031	3345X	DAD	D	

3346X  
3347X \* PRINT FILE NAME  
3348X

056.030	176	3349X \$FERR1	MOV	A,M	
056.031	043	3350X	INX	H	ADVANCE MESSAGE
056.032	247	3351X	ANA	A	
056.033	312 044 056	3352X	JZ	\$FERR2	
056.036	315 137 053	3353X	CALL	\$WCHAR	
056.041	303 030 056	3354X	JMP	\$FERR1	

3355X  
3356X \* TYPE ERROR MESSAGE  
3357X

056.044	315 136 031	3358X \$FERR2	CALL	\$TYPTX	
056.047	040 055 240	3359X	DB	' - ',' '+2000	
056.052	046 012	3360X	MVI	H,NL	
056.054	361	3361X	POP	PSW	(A) = CODE
056.055	377 057	3362X	DB	SYSCALL,.ERROR	
056.057	303 211 045	3363X	JMP	RESTART	EXIT
056.062		3364	XTEXT	FUTIL	

3366X \*\* \$FUTIL - UTILITY ROUTINES FOR FILE BLOCK ROUTINES.  
3367X  
3368X \*\* CBT - COPY BLOCK POINTERS TO TEMP CELLS.  
3369X \*  
3370X \* ENTRY (HL) = FILE BLOK FWA  
3371X \* EXIT NONE  
3372X \* USES A,F,H,L

COMMON DECKS

\*FUTIL

15:31:58 16-MAY-80

```

3373X
056.062 325 3374X CBT PUSH D
056.063 305 3375X PUSH B SAVE REGISTERS
000.000 3376X ERRNZ TLEN-10 ASSUME 10 BYTES TO MOVE
056.064 021 224 056 3377X LXI D,T.CHA (DE) = TARGET FOR MOVE
056.067 006 005 3378X MVI B,10/2
056.071 176 3379X CBT1 MOV A,M COPY FILE BUFFER INTO WORK AREA
056.072 022 3380X STAX D
056.073 043 3381X INX H
056.074 023 3382X INX D
056.075 176 3383X MOV A,M
056.076 022 3384X STAX D
056.077 043 3385X INX H
056.100 023 3386X INX D
056.101 005 3387X DCR B
056.102 302 071 056 3388X JNZ CBT1 MORE TO GO
056.105 301 3389X POP B
056.106 321 3390X POP D (DE) = DATA TARGET ADDRESS
056.107 311 3391X RET
3392X
3393X
3394X ** CBT - COPY TEMP CELLS BACK TO FILE BLOCK.
3395X *
3396X * ENTRY (HL) = FILE BLOCK ADDRESS
3397X * EXIT NONE
3398X * USES NONE
3399X
056.110 365 3400X CBT PUSH PSW
056.111 325 3401X PUSH D
056.112 305 3402X PUSH B SAVE REGISTERS
056.113 345 3403X PUSH H
056.114 006 004 3404X MVI B,8/2
056.116 021 224 056 3405X LXI D,T.CHA
056.121 032 3406X CBT1 LDAX D
056.122 167 3407X MOV M,A
056.123 023 3408X INX D
056.124 043 3409X INX H
056.125 032 3410X LDAX D
056.126 167 3411X MOV M,A
056.127 023 3412X INX D
056.130 043 3413X INX H
056.131 005 3414X DCR B
056.132 302 121 056 3415X JNZ CBT1 RESTORE FILE BUFFER VALUES
056.135 341 3416X POP H
056.136 301 3417X POP B
056.137 321 3418X POP D
056.140 361 3419X POP PSW
056.141 311 3420X RET

```

```

3422X **      $FFB - FILE FILE BUFFER.
3423X *
3424X *      $FFB FILLS THE FILE BUFFER BY READING FROM THE FILE.
3425X *
3426X *      ENTRY  NONE
3427X *      EXIT   'C' SET IF READ INCOMPLETE
3428X *              (A) = ERROR CODE
3429X *              'C' CLEAR IF READ COMPLETE
3430X *              DATA IN BUFFER
3431X *      USES   A,F,D,E,H,L
3432X
3433X
056.142 072 236 056 3434X $FFB LDA EOFFLG
056.145 037          3435X RAR
056.146 330          3436X RC EOF
3437X
3438X *      CAN READ MORE. DO SO
3439X
056.147 305          3440X PUSH B SAVE COUNT
056.150 052 226 056 3441X LHL D T.FWA
056.153 042 230 056 3442X SHLD T.PTR CLEAR REMOVAL POINTER
056.156 353          3443X XCHG
056.157 052 234 056 3444X LHL D T.LWA
056.162 042 232 056 3445X SHLD T.LIM SET DATA LIMIT
056.165 175          3446X MOV A,L
056.166 223          3447X SUB E
056.167 117          3448X MOV C,A
056.170 174          3449X MOV A,H
056.171 232          3450X SBB D
056.172 107          3451X MOV B,A (BC) = ROOM IN BUFFER
056.173 072 224 056 3452X LDA T.CHA
056.176 377 004          3453X DB SYSCALL, READ READ BUFFER
056.200 120          3454X MOV D,B (D) = SECTORS UNREAD
056.201 301          3455X POP B (BC) = DESIRED COUNT
056.202 320          3456X RNC GOT THE DATA
3457X
3458X *      ERROR ON READ. SEE IF EOF
3459X
056.203 027          3460X RAL
056.204 062 236 056 3461X STA EOFFLG SET EOF, WE HOPE
056.207 376 003          3462X CPI EC.EOF*2+1
056.211 037          3463X RAR
056.212 300          3464X RNE IS NOT EOF, RETURN NOW!
056.213 072 233 056 3465X LDA T.LIM+1
056.216 222          3466X SUB D
056.217 062 233 056 3467X STA T.LIM+1 SET AMOUNT OF DATA WE DID GET
056.222 247          3468X ANA A
056.223 311          3469X RET EXIT WITH DATA
3470X
3471X
3472X **      TEMP CELLS TO HOLD FILE BLOCK POINTERS DURING I/O
3473X
000.000          3474X ERRNZ FB.CHA
056.224 000          3475X T.CHA DB 0 CHANNEL NUMBER
000.000          3476X ERRNZ *-T.CHA-FB.FLG
056.225 000          3477X T.FLG DB 0 FLAG BYTE

```

COMMON DECKS

\*FFB

15:31:59 16-MAY-80

000.000		3478X	ERRNZ	*-T.CHA-FB.FWA	
056.226	000 000	3479X T.FWA	DW	0	
000.000		3480X	ERRNZ	*-T.CHA-FB.PTR	
056.230	000 000	3481X T.PTR	DW	0	
000.000		3482X	ERRNZ	*-T.CHA-FB.LIM	
056.232	000 000	3483X T.LIM	DW	0	
000.000		3484X	ERRNZ	*-T.CHA-FB.LWA	
056.234	000 000	3485X T.LWA	DW	0	
000.012		3486X TLEN	EQU	*-T.CHA	LENGTH OF TEMP CELLS
		3487X			
056.236	000	3488X EOFFLG	DB	0	

```

3491 **      COMMAND TABLE.
3492 *
3493
056.237 000 3494 CMDTAB DB      0          DUMP FIRST ENTRY
3495
056.240 221 240 040 3496 *      0 - [OPT]ADDR
3497 DB      091H,0A0H,'',0
3498
056.244 221 241 055 3499 *      1 - [OPT]ADDR-ADDR
3500 DB      091H,0A1H,'-',0A5H,'',0
3501
056.252 221 240 075 3502 *      2 - [OPT]ADDR=VAL
3503 DB      091H,0A0H,'=',0
3504
056.256 221 241 055 3505 *      3 - [OPT]ADDR-ADDR=VAL
3506 DB      091H,0A1H,'-',0A5H,'=',0
3507
056.264 223 022 040 3508 *      4 - [OPT]CTL-R
3509 DB      093H,'R'-@',',',0
3510
056.270 223 222 224 3511 *      5 - [OPT]REGX
3512 DB      093H,092H,094H,'',0
3513
056.275 223 222 224 3514 *      6 - [OPT]REGX=
3515 DB      093H,092H,094H,'=',0
3516
056.302 105 130 105 3517 *      7 - EXEC A1-A2,...,AN
3518 DB      'EXEC ',0A1H,'-',0D0H,0
3519
056.313 123 124 105 3520 *      8 - STEP ADDR
3521 DB      'STEP ',0A0H,NL,0
3522
056.323 225 320 000 3523 *      9 - BKPT A1,...,AN
3524 DB      095H,0D0H,0
3525
056.326 225 104 123 3526 *     10 - BKPT DSPLY
3527 DB      095H,'DSPLY',',',0
3528
056.336 226 320 000 3529 *     11 - CLEAR A1,...,AN
3530 DB      096H,0D0H,00
3531
056.341 226 101 114 3532 **     12 - CLEAR ALL
3533 DB      096H,'ALL',NL,0
3534
056.347 104 125 115 3535 *     13 - DUMP
3536 DB      'DUMP ',0B0H,0B1H,'',',',0B1H,0A1H,'-',0A5H,NL,0
3537
056.366 114 117 101 3538 *     14 - LOAD
3539 DB      'LOAD ',0B0H,NL,0
3540
056.376 114 117 101 3541 *     15 - LOAD PIC
3542 DB      'LOAD PIC ',0B0H,0B1H,'',',',0B1H,0A3H,NL,0
3543
057.017 107 117 040 3544 *     16 - GO
3545 DB      'GO ',0A1H,NL,0
3546

```

057.025 000 3547 DB 0 END OF MAIN STRINGS.

3549 \*\* EXTENSION STRINGS.

3550

057.025 3551 CMDEXS EQU \*-1 START TABLE WITH 00

3552

057.026 202 106 202 3553 \* 1 - [OPT]  
3554 DB 082H,'F',082H,080H,'DA',080H,0C0H,0

3555

057.037 122 105 107 3556 \* 2 - REG  
3557 DB 'REG',0C0H,0

3558

057.044 200 104 101 3559 \* 3 - [OPT]  
3560 DB 080H,'DA',080H,0C0H,00

3561

057.052 205 101 102 3562 \* 4 - REGISTER ID  
3563 DB 085H,'ABCDEHLSFFM',085H,0C0H,0

3564

057.071 102 113 120 3565 \* 5 - BKPT  
3566 DB 'BKPT',0C0H,0

3567

057.100 103 114 105 3568 \* 6 - CLEAR  
3569 DB 'CLEAR',0C0H,0

```

3572 **      MEMORY TOP AND BOTTOM VALUES
3573
057.110 000 000 3574 TOPVAL DW      0
057.112 000 000 3575 BOTVAL DW      0
3576
057.114 000 3577 CSLMD  DB      0      SAVED VALUE OF USER S.CSLMD
057.115 000 3578 CONFL  DB      0      SAVED VALUE OF USER S.CONFL
3579
057.116 3580 DARA  EQU      *      REGISTER TABLE
057.116 101 003 3581      DB      'A',3
057.120 102 005 3582      DB      'B',5
057.122 103 004 3583      DB      'C',4
057.124 104 007 3584      DB      'D',7
057.126 105 006 3585      DB      'E',6
057.130 110 011 3586      DB      'H',9
057.132 114 010 3587      DB      'L',8
057.134 106 002 3588      DB      'F',2
057.136 120 212 3589 DARAF  DB      'P',10+80H
057.140 115 210 3590      DB      'M',08+80H
057.142 123 200 3591      DB      'S',00+80H
057.144 000 3592      DB      0
000.013 3593 DARAL  EQU      *-DARA-1/2

```

```

3595 **      BKPTAB - BREAKPOINT TABLE.
3596 *
3597 *      BKPTAB CONTAINS INFORMATION ABOUT BREAKPOINTS.
3598 *
3599 *      BYTE 0 - LOW ORDER ADDRESS
3600 *      1 - HIGH ORDER ADDRESS
3601 *      2 - BREAKPOINT REPEAT COUNT
3602 *      3 - INSTRUCTION AT BREAKPOINT
3603 *
3604 *      WHEN IN THE DEBUGGER PACKAGE, THE BREAKPOINT ARE NOT
3605 *      SET.
3606 *
3607
000.010 3608 BKPTBL EQU      8
3609
057.145 000 000 000 3610 BKPTAB DW      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
3611
3612 *      EXTRA ENTRY TO AUTOMATICALLY SET AND CLEAR BKPFLG
3613
057.205 213 057 001 3614      DW      BKPFLG,1,0
3615
057.213 000 3616 BKPFLG DB      0      NON-ZERO IF BREAKPOINTS ARE SET
3617
057.214 000 3618 $LSTIN DB      0      LAST READ BYTE

```



```

3620 **      LOAD/DUMP FILE BUFFER.
3621 *
3622
057.215 005 3623 MEMFB DB      CN.LD      CHANNEL NUMBER
057.216 000 3624          DB      0          FLAGS
057.217 360 057 360 3625          DW      MEMBUF, MEMBUF, MEMBUFE, MEMBUFE
3626
057.227 3627 CMD.BA DS      FB.NAML      SPACE FOR FILE NAME
3628
3629
057.250 3630 BFILHDR DS      ABS.COD      ROOM FOR BINARY AND PIC HEADERS FOR LOAD/DUMP
000.002 3631          ERRMI      ABS.COD-PIC.COD      MUST HAVE ROOM FOR EITHER
3632
057.260 3633 PATCH DS      64          PATCH AREA

```

```

3637 ** PRS - PRESET CODE
3638 *
3639 * THIS CODE IS ONLY USED AT ENTRY, IT IS THEN OVERLAID BY BUFFERS
3640 *
3641 *
057.360 3642 PRS EQU *
3643
3644 * CHECK THE VERSION OF HDOS
3645
057.360 3646 DB SYSCALL,.VERS
057.362 332 375 057 3647 JC PRSERR1 NO .VERS SYSTEM CALL
057.365 376 026 3648 CPI VERS
057.367 302 375 057 3649 JNZ PRSERR1
3650
3651 * GO TO THE REAL ENTRY
3652
057.372 303 101 045 3653 JMP HBUG
3654
057.375 076 050 3655 PRSERR1 MVI A,EC.NCV
3656
057.377 046 012 3657 PRSERR MVI H,NL
060.001 377 057 3658 DB SYSCALL,.ERROR
3659
060.003 303 046 046 3660 JMP EXIT1
3661
060.006 3662 MEML EQU * END OF LOAD IMAGE
3663
057.360 3664 ORG PRS OVERLAY PRS CODE
3665
057.360 3666 MEMBUF DS 256 BUFFER
060.360 3667 MEMBUFE EQU * END OF BUFFER
3668
3669
060.360 3670 RMEML EQU * RUNNING MEMORY LIMIT
3671
060.360 3672 END
ASSEMBLY COMPLETE
3672 STATEMENTS
1 ERRORS DETECTED
11038 BYTES FREE

```

CROSS REFERENCE TABLE

\$CRLF	053142	1150	1276	1911	2539L	2600												
\$DADA	030072	759	896	1993	2554E													
\$DADA.	030101	2564E	2726															
\$FCLO	055273	1632	1757	3261L														
\$FCLO.	055302	3261	3265L															
\$FCLO1	055353	3304L	3307															
\$FCLO2	055363	3300	3309L															
\$FCLO3	055370	3273	3290	3320L														
\$FCLO4	055376	3271	3326L															
\$FERR1	056030	3349L	3354															
\$FERR2	056044	3352	3358L															
\$FERROR	056000	1943	2872	2876	2880	2980	3085	3203	3263	3341L								
\$FFB	056142	3027	3434L															
\$FOPE1	054154	2902	2914L															
\$FOPE2	054207	2938	2942L															
\$FOPEA	054215	2942	2946E															
\$FOPEB	054225	2936	2956L															
\$FOPER	054074	1648	1700	2870L														
\$FOPER.	054121	2870	2883L															
\$FOPEU	054112	2878L																
\$FOPEU.	054127	2878	2887L															
\$FOPEW	054103	1612	2874L															
\$FOPEW.	054124	2874	2885L															
\$FREAB	054234	1651	1679	1703	1722	1737	2977L											
\$FREAB.	054247	2977	2985E															
\$FWBRK	055167	3200L																
\$FWBRK.	055176	3200	3205L															
\$FWBRK1	055212	3207	3212L															
\$FWRIB	055005	1624	1631	3083L														
\$FWRIB.	055014	3083	3088E															
\$HLIHL	030211	1617	2495E															
\$INCHA	053150	524	1852	2582L	2608	2617												
\$INCHAA	053303	2588	2592	2621	2625	2631L												
\$INDL	030234	2484E	3277	3280														
\$LSTIN	057214	1039	1836	3618L														
\$MCU	053120	525	1221	2462L														
\$MOVE	030252	2418E																
\$MUB6	031007	891	2646E															
\$RCHAR	053131	1036	1218	2522L	2523	2582												
\$REAB2	054257	2993L	3059															
\$REAB3	054300	3005L	3029															
\$REAB4	054326	3018	3021L															
\$REAB6	054344	3023	3038L															
\$REAB7	054356	3048L	3055															
\$REAB8	055000	2998	3001	3028	3065L													
\$RSTALL	031047	2435E																
\$SAVALL	031054	1099	2449E															
\$TBLS	053304	2007	2100	2667L	2940													
\$TBRA	031076	633	2118	2793E														
\$TDD	053337	2150	2722L															
\$TDD.	053335	2721L																
\$TJMP	031061	1161	2706E															
\$TJMP.	031062	2708E																
\$TOD	054037	1916	1918	2136	2140	2805L												
\$TPA	054020	2155	2159	2768L														
\$TYPC.	054071	945	952	1043	1047	1855	2026	2040	2737	2771	2773	2815	2840L					
\$TYPCH	054065	573	1034	1387	1806	2027	2830L											
\$TYPTX	031136	1100	1151	1188	1216	1535	1541	1604	1684	1762	1919	1950	2316					

2510E	2613	3342	3358
\$.TYP TX.	031144	2512E	
\$.WCHAR	053137	2526L	2590 2626 2628 3353
\$.WRIB2	055020	3094L	3174
\$.WRIB3	055036	3104L	3140
\$.WRIB4	055064	3117	3120L
\$.WRIB6	055126	3122	3153L
\$.WRIB7	055140	3163L	3170
\$.WRIB8	055162	3097	3100 3144 3180L
\$.ZERO	031212	549	1455 2854E
\$.ABUSS	040024	2367S	2368
\$.ALARM	002136	202E	
\$.ALED5	040013	175E	
\$.BKP.	047052	200E	
\$.CHFLG	000060	1466	1497E
\$.CLEAR	000055	108L	
\$.CLEARA	000056	105L	1134 1139
\$.CLOSE	000046	106L	
\$.CLRCO	000007	98L	3325
\$.CONSL	000006	82L	
\$.CRC	002347	81L	
\$.CRCSUM	040027	183E	
\$.CTC	002172	203E	
\$.CTLG	000041	177E	
\$.CTLFLG	040011	93L	1105
\$.DECODE	000053	199E	1549 1552 2250
\$.DELET	000050	103L	
\$.DISMT	000061	100L	
\$.DLEDS	040021	109L	
\$.DLY	000053	201E	
\$.DMNMS	000203	172E	
\$.DMOUN	000201	120L	
\$.DOD	003122	118L	
\$.DODA	003356	186E	
\$.DSFMD	040007	188E	
\$.DSFMDT	040006	197E	
\$.DUMP	001374	194E	
\$.ERRDR	000057	174E	
\$.EXIT	000000	107L	3362 3658
\$.HORN	002140	75L	1225 2939 2945
\$.IDENT	000000	176E	
\$.IOWRK	040002	171E	
\$.LINK	000040	194E	
\$.LOAD	001267	92L	
\$.LOADD	000062	173E	
\$.LOADO	000010	110L	
\$.MFLAG	040010	83L	
\$.MONMS	000202	198E	
\$.MOUNT	000200	119L	
\$.NAME	000054	117L	
\$.OFENC	000045	104L	
\$.OFENR	000042	97L	
\$.OFENU	000044	94L	2956
\$.OFENW	000043	96L	2958
\$.PCHL	002264	95L	2957
\$.POSIT	000047	179E	
\$.PRINT	000003	99L	
		78L	

[illegible]

## XREF U1.1

PAGE 86

[illegible]

CROSS REFERENCE TABLE

CSR1	046142	1299	1301L			
CTB	056110	3067	3182	3209	3400L	
CTB1	056121	3406L	3415			
CTLA	000001	51E				
CTLB	000002	52E				
CTLC	000003	53E				
CTLD	000004	54E	524	1037	1219	1853
CTLO	000017	55E				
CTLP	000020	56E				
CTLQ	000021	57E				
CTLS	000023	58E				
CTLZ	000032	59E				
CTP.2SB	000010	270E				
CTP.BKM	000002	271E				
CTP.BKS	000200	267E				
CTP.MLI	000040	268E				
CTP.MLO	000020	269E				
CTP.TAB	000001	272E				
CUB	051312	1245	1839	1967L		
D.CON	040110	222L				
D.RAM	040240	225L				
D.VEC	040130	224L				
DARA	057116	1274	2006	3580E	3593	
DARAL	000013	1275	3593E			
DARAP	057136	1988	3589L			
DAS	044112	760	836	964L		
DAS1	044151	971	986L			
DAS2	044155	983	988L			
DBL1	046246	1377L	1403			
DBL2	046311	1384	1399L			
DF.CLR	000376	446E				
DF.EMP	000377	445E				
DIR.ALD	000025	461L				
DIR.CLU	000015	454L				
DIR.CRD	000023	460L				
DIR.EXT	000010	449L				
DIR.FGN	000020	457L				
DIR.FLG	000016	455L				
DIR.LGN	000021	458L				
DIR.LSI	000022	459L				
DIR.NAM	000000	448L				
DIR.PRO	000013	450L				
DIR.VER	000014	451L				
DIRELEN	000027	463E	494			
DIRIDL	000015	452E				
DM.MR	000000	146E				
DM.MW	000001	147E				
DM.RR	000002	148E				
DM.RW	000003	149E				
DMP0	047240	1579	1585L			
DMP1	047276	1598	1604L	2186		
DMP2	047317	1587	1600	1610L		
DMPA	050005	1610	1634L			
DRA	051327	1296	1989L	2029		
DRA	051324	1539	1616	1662	1988L	2388
DRI	051344	1287	1295	2005L		
DRV	051360	1277	2024E			
DRV	051364	1288	2027L			

## XREF V1.1

## PAGE 88

[illegible]



CROSS REFERENCE TABLE

FBNL	000033	355E						
FBT	052010	1428	1514	2050L	2292	2296		
FBT1	052016	2054L	2070					
FBT2	052032	2056	2065L					
FBT3	052043	2061	2072L					
FF	000014	50E						
FIC	044217	882	1023E					
FIC1	044231	1034L	1045	1048				
FIC2	044235	1025	1036L					
FIC2.5	044245	1039L						
FICA	044217	518	1024E	1798				
FNRA	044312	1070E						
FT.ABS	000000	363E	501	1619	1656			
FT.BAC	000003	366E						
FT.DD	000001	474E						
FT.DR	000002	475E	2883	2887	2956	2958	2995	
FT.QU	000010	477E						
FT.OW	000004	476E	2885	2887	2957	2958	3096	3272
FT.PIC	000001	364E	1708					
FT.REL	000002	365E						
FVD	052045	1386	1540	1764	1804	2038	2089E	
FVD.A	052163	2122	2154L					
FVD.D	052145	2121	2144L					
FVD.D1	052160	2146	2150L					
FVD.Q	052133	2120	2135L					
FVDO	052116	2114	2118L					
FVDO.1	052073	2099	2104L					
FVDO.2	052074	2102	2106L					
FVD1	052124	2106	2125L					
FVDA	052126	2097	2129L					
FWBRK2	055231	3226L	3233					
FWBRK3	055245	3228	3235L					
GO	047004	1470L						
G00	046366	1465L	1562					
G02	046374	1467L	1554					
G03	047206	1547	1558L					
G0A	047040	1476	1486E					
G0B	047044	1482	1489E					
G0C	047050	1484	1492E					
HBUG	045101	1094E	3653					
HBUG1	045171	1115L						
HBUGA	045270	1163L	1167					
I.CONFL	000004	287E	288					
I.CONTY	000001	274E	275					
I.CONWI	000003	280E	281					
I.CSLMD	000000	264E						
I.CUSOR	000002	277E	278					
INCO	053216	2584	2586	2605L				
INC1	053245	2601	2615L	2629				
INC3	053255	2612	2621L					
INC4	053274	2623	2627L					
INTRPT	045332	1104	1188L					
IOC.CGN	000010	482L						
IOC.CSI	000011	483L						
IOC.DDA	000002	471L	478	492				
IOC.DES	000016	489L						
IOC.DEV	000020	490L						
IOC.DIL	000021	492E						

```

XREF V1.1

```

PAGE 90

[illegible]

```

XREF V1.1

```

PAGE 91

[illegible]

TR.CSR	046126	1170	1294E						
TR.DAR	046077	1168	1274L						
TR.DAR1	046107	1277L	1280						
TR.DARI	000004	1167E	1913						
TR.DBL	046241	1174	1371E						
TR.DMP	047223	1177	1573E						
TR.DSR	046120	1169	1287L						
TR.DVP	046052	1164	1241E						
TR.DVS	046052	1163	1234E						
TR.EXE	046152	1171	1312E						
TR.GQ	046363	1180	1316	1463E					
TR.LOA	050013	1178	1645E						
TR.LOA	050155	1179	1697E						
TR.SBL	046236	1173	1359E						
TR.STP	046162	1172	1328E						
TBGX	045225	1143E	1158	1193	1223	1543	1606	1686	2318
TBL1	053313	2671L	2677						
TBL2	053331	2669	2681L						
TBL3	053333	2674	2685L						
TDD1	053340	2723L	2746						
TDD2	053357	2733L	2735						
TDDA	054006	2724	2750E						
TLEN	000012	3376	3486E						
TOD1	054043	2809L	2818						
TQFVAL	057110	1891	2199	3574L					
TPA1	054032	2769	2772L						
UCI.ER	000020	400E							
UCI.IE	000002	402E							
UCI.IR	000100	398E							
UCI.RE	000004	401E							
UCI.RD	000040	399E							
UCI.TE	000001	403E							
UMR	000000	375E							
UMI.16X	000002	393E							
UMI.1B	000100	383E							
UMI.1X	000001	392E							
UMI.2B	000300	385E							
UMI.64X	000003	394E							
UMI.HB	000200	384E							
UMI.L5	000000	388E							
UMI.L6	000004	389E							
UMI.L7	000010	390E							
UMI.L8	000014	391E							
UMI.PA	000020	387E							
UMI.PE	000040	386E							
UQ.CLK	000001	158E							
UQ.DDU	000002	157E							
UQ.HLT	000200	155E							
UQ.NFR	000100	156E							
USERFWA	042200	234E	500	502	503	1113			
USERMD	045340	1191E	1195	1340	1473	1499			
USR	000001	376E							
USR.FE	000040	407E							
USR.OE	000020	408E							
USR.PE	000010	409E							
USR.RXR	000002	411E							
USR.TXE	000004	410E							
USR.TXR	000001	412E							

DEBUG - HEATH TERMINAL DEBUGGER.

CROSS REFERENCE TABLE

XREF V1.1

PAGE 93

VERS 000026 66E 3648

19864 BYTES FREE

