



H17 ROM

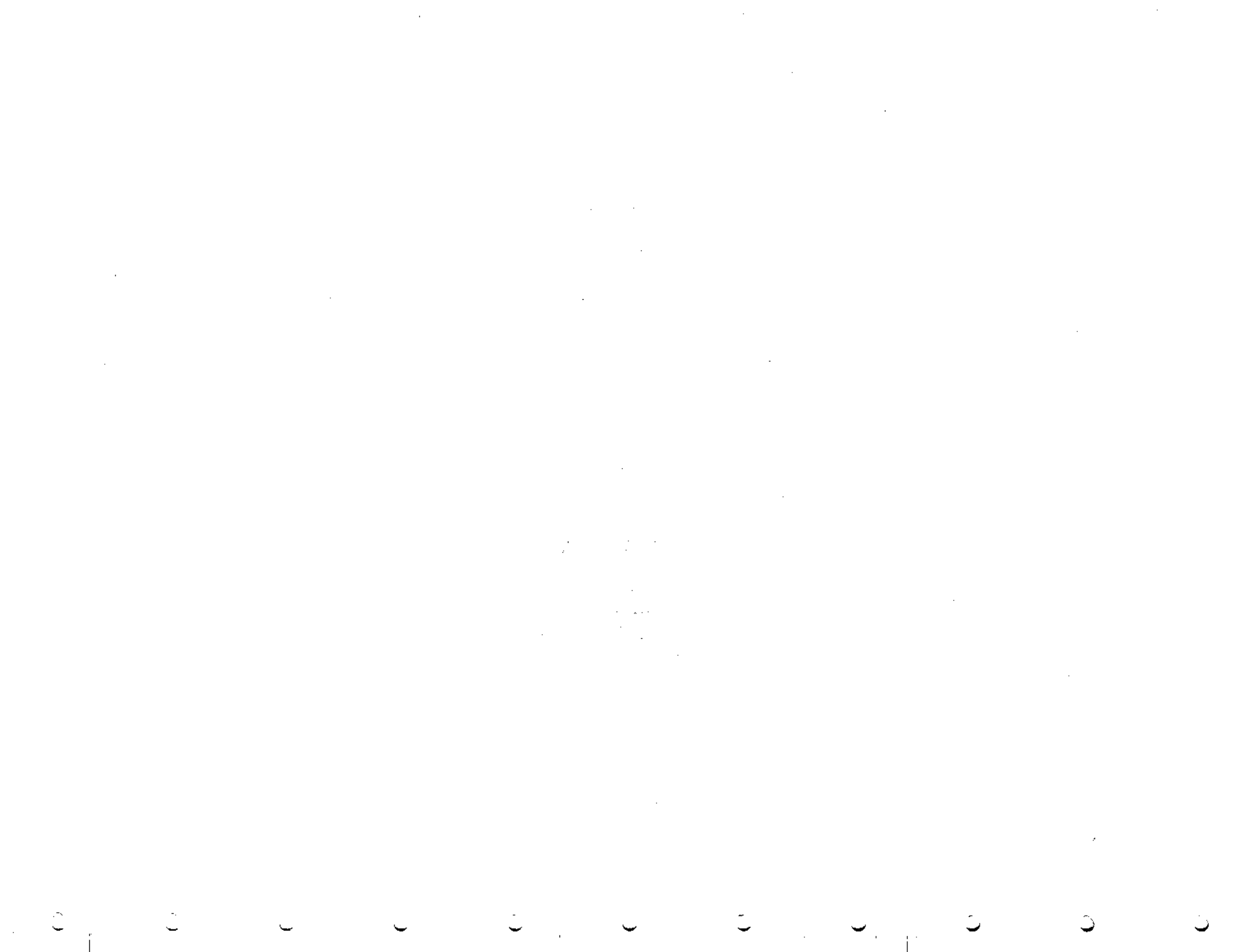
595-2464

The documentation of Software contained herein is owned by the Heath Company of Benton Harbor, Michigan.

Reproduction of this Software, except with the written consent of Heath Company, is prohibited.

Heath Company disclaims all warranties with regard to this material, including all implied warranties of merchantability and fitness for a special purpose.

If you find this material not suitable for your purposes, it may be returned to Heath Company for full refund within 30 days, providing that the special binding of this product has not been broken.



4 *** SYDD - SYSTEM DEVICE DRIVER.
 5 *
 6 * J.G. LETWIN, OCT 77
 7 *
 8 * COPYRIGHT HEATH CO.
 9
 10
 11 *** SYDD - SYSTEM DEVICE DRIVER.
 12 *
 13 * SYDD IS THE DEVICE DRIVER FOR THE SYSTEM DEVICE, AN H17 MINI/-FLOPPY.

15 **** ASSEMBLY CONSTANTS

16
 000.376 17 MI.CPI EQU 3760 CPI INSTRUCTION
 000.012 18 ERPTCNT EQU 10 SOFT ERROR RETRY COUNT
 19
 000.000 20 XTEXT MTR
 000.000 106 XTEXT U8251
 000.000 152 XTEXT ASCII
 180 LON C
 000.000 181 XTEXT HOSDEF

183X ** HOSDEF - DEFINE HOS PARAMETER.

184X *
 185X
 000.377 186X SYSCALL EQU 3770 SYSCALL INSTRUCTION
 187X
 000.000 188X ORG 0
 189X

190X * RESIDENT FUNCTIONS

191X
 000.000 192X .EXIT DS 1 EXIT (MUST BE FIRST)
 000.001 193X .SCIN DS 1 SCIN
 000.002 194X .SCOUT DS 1 SCOUT
 000.003 195X .PRINT DS 1 PRINT
 000.004 196X .READ DS 1 READ
 000.005 197X .WRITE DS 1 WRITE
 000.006 198X .CONSL DS 1 SET/CLEAR CONSOLE OPTIONS
 000.007 199X .CLRCD DS 1 CLEAR CONSOLE BUFFER
 000.010 200X .SYSRES DS 1 PRECEDING FUNCTIONS ARE RESIDENT
 201X

202X * HOSOVLSYS FUNCTIONS

203X
 000.040 204X ORG 40A
 205X
 000.040 206X .LINK DS 1 LINK (MUST BE FIRST)
 000.041 207X .CTLG DS 1 CTL-C
 000.042 208X .OPENR DS 1 OPENR
 000.043 209X .OPENW DS 1 OPENW
 000.044 210X .OPENU DS 1 OPENU
 000.045 211X .OPENC DS 1 OPENC
 000.046 212X .CLOSE DS 1 CLOSE

HOSDEF

000.047	213X	.POSIT	DS	1	POSITION
000.050	214X	.DELET	DS	1	DELETE
000.051	215X	.RENAM	DS	1	RENAME
000.052	216X	.SETTP	DS	1	SETTOP
000.053	217X	.DECODE	DS	1	NAME DECODE
000.054	218X	.NAME	DS	1	GET FILE NAME FROM CHANNEL
000.055	219X	.CLEAR	DS	1	CLEAR CHAN
000.056	220X	.CLEARA	DS	1	CLEAR ALL CHANS
000.057	221X	.ERROR	DS	1	LOOKUP ERROR
000.060	222X	.CHFLG	DS	1	CHANGE FLAGS
000.061	223X	.DISMT	DS	1	FLAG SYSTEM DISK DISMOUNTED
000.062	224		XTEXT	DIRDEF	

226X ** DIRECTORY ENTRY FORMAT.

	227X				
000.000	228X		ORG	0	
	229X				
	230X				
000.377	231X	DF.EMP	EQU	3770	FLAGS ENTRY EMPTY
000.376	232X	DF.CLR	EQU	3760	FLAGS ENTRY EMPTY, REST OF DIR ALSO CLEAR
	233X				
000.000	234X	DIR.NAM	DS	8	NAME
000.010	235X	DIR.EXT	DS	3	EXTENSION
000.013	236X	DIR.PRO	DS	1	PROJECT
000.014	237X	DIR.VER	DS	1	VERSION
000.015	238X	DIRIDL	EQU	*	FILE IDENTIFICATION LENGTH
	239X				
000.015	240X	DIR.CLU	DS	1	CLUSTER FACTOR
000.016	241X	DIR.FLG	DS	1	FLAGS
000.017	242X		DS	1	RESERVED
000.020	243X	DIR.FGN	DS	1	FIRST GROUP NUMBER
000.021	244X	DIR.LGN	DS	1	LAST GROUP NUMBER
000.022	245X	DIR.LSI	DS	1	LAST SECTOR INDEX (IN LAST GROUP)
000.023	246X	DIR.CRD	DS	2	CREATION DATE
000.025	247X	DIR.ALD	DS	2	LAST ALTERATION DATE
	248X				
000.027	249X	DIRELEN	EQU	*	DIRECTORY ENTRY LENGTH
000.027	250		XTEXT	DEVDEF	

252X ** DEVICE TABLE ENTRIES.

	253X				
000.000	254X		ORG	0	
	255X				
000.000	256X	DEV.NAM	DS	2	DEVICE NAME
000.002	257X	DEV.RES	DS	1	DRIVER RESIDENSE CODE
000.001	258X	DR.IM	EQU	00000001B	DRIVER IN MEMORY
000.002	259X	DR.PR	EQU	00000010B	DRIVER PERMINANTLY RESIDENT
	260X				
000.003	261X	DEV.JMP	DS	1	JMP TO PROCESSOR
000.004	262X	DEV.DDA	DS	2	DRIVER ADDRESS
000.006	263X	DEV.FLG	DS	1	FLAG BYTE
000.001	264X	DT.DD	EQU	00000001B	DIRECTORY DEVICE

DISK_DRIVER_CODE	DEV	HEATH HBASH V1.4 01/20/78	09:59:19 02-APR-80	PAGE 3
000.002	265X DT.CR EQU	00000010B	CAPABLE OF READ OPERATION	
000.004	266X DT.CW EQU	00000100B	CAPABLE OF WRITE OPERATION	
	267X			
000.007	268X DEV.GRT DS	2	ADDRESS OF GROUP RESERVATION TABLE (IF DIRECTORY)	
000.011	269X DEV.SPG DS	1	SECTORS PER GROUP THIS DEVICE	
000.012	270X DEV.MUM DS	0	MOUNTED UNIT MASK	
000.012	271X DEV.MNU DS	1	MAXIMUM NUMBER OF UNITS	
000.013	272X DEV.DVL DS	2	DRIVER BYTE LENGTH	
000.015	273X DEV.DVG DS	1	DRIVER ROUTINE GROUP ADDRESS	
000.016	274X DEV.DIR DS	2	DIRECTORY FIRST SECTOR ADDRESS	
000.020	275X DEV.GTS DS	2	GRT SECTOR NUMBER	
	276X			
000.022	277X DEVELEN EQU	*	DEVICE TABLE ENTRY LENGTH	
000.022	278	XTEXT	H17DEF	

280X ** H17 CONTROL INFORMATION.

	281X			
000.177	282X DF.DC EQU	07FH	DISK CONTROL PORT	
	283X			
000.001	284X DF.HD EQU	00000001B	HOLE DETECT	
000.002	285X DF.TO EQU	00000010B	TRACK 0 DETECT	
000.004	286X DF.WP EQU	00000100B	WRITE PROTECT	
000.010	287X DF.SD EQU	00001000B	SYNC DETECT	
	288X			
000.001	289X DF.WG EQU	00000001B	WRITE GATE ENABLE	
000.002	290X DF.DS0 EQU	00000010B	DRIVE SELECT 0	
000.004	291X DF.DS1 EQU	00000100B	DRIVE SELECT 1	
000.010	292X DF.DS2 EQU	00001000B	DRIVE SELECT 2	
000.020	293X DF.MO EQU	00010000B	MOTOR ON (BOTH DRIVES)	
000.040	294X DF.DI EQU	00100000B	DIRECTION (0=OUT)	
000.100	295X DF.ST EQU	01000000B	STEP COMMAND (ACTIVE HIGH)	
000.200	296X DF.WR EQU	10000000B	WRITE ENABLE RAM	
	297X			
	298X			
	299X			

300X ** DISK UART PORTS AND CONTROL FLAGS.

	301X			
000.174	302X UF.DP EQU	07CH	DATA PORT	
000.175	303X UF.FC EQU	07DH	FILL CHARACTER	
000.175	304X UF.ST EQU	07EH	STATUS FLAGS	
000.176	305X UF.SC EQU	07FH	SYN CHARACTER (OUTPUT)	
000.176	306X UF.SR EQU	07EH	SYNC RESET (INPUT)	
	307X			
000.001	308X UF.RDA EQU	00000001B	RECEIVE DATA AVAILABLE	
000.002	309X UF.RDR EQU	00000010B	RECEIVER OVERRUN	
000.004	310X UF.RPE EQU	00000100B	RECEIVER PARITY ERROR	
000.100	311X UF.FCT EQU	01000000B	FILL CHAR TRANSMITTED	
000.200	312X UF.TBM EQU	10000000B	TRANSMITTER BUFFER EMPTY	
	313X			
	314X			
	315X			

316X ** CHARACTER DEFINITIONS.

	317X			
000.375	318X C.DSYN EQU	0FDH	PREFIX SYNC CHARACTER	

000.022 319 XTEXT ECDEF

321X ** ERROR CODE DEFINITIONS.

000.000	322X				
	323X	ORG	0		
000.000	324X	DS	1		NO ERROR #0
000.001	325X EC.EOF	DS	1		END OF FILE
000.002	326X EC.EOM	DS	1		END OF MEDIA
000.003	327X EC.ILC	DS	1		ILLEGAL SYSCALL CODE
000.004	328X EC.CNA	DS	1		CHANNEL NOT AVAILABLE
000.005	329X EC.DNS	DS	1		DEVICE NOT SUITABLE
000.006	330X EC.IDN	DS	1		ILLEGAL DEVICE NAME
000.007	331X EC.IFN	DS	1		ILLEGAL FILE NAME
000.010	332X EC.NRD	DS	1		NO ROOM FOR DEVICE DRIVER
000.011	333X EC.FNO	DS	1		CHANNEL NOT OPEN
000.012	334X EC.ILR	DS	1		ILLEGAL REQUEST
000.013	335X EC.FUC	DS	1		FILE USAGE CONFLICT
000.014	336X EC.FNF	DS	1		FILE NAME NOT FOUND
000.015	337X EC.UND	DS	1		UNKNOWN DEVICE
000.016	338X EC.ICN	DS	1		ILLEGAL CHANNEL NUMBER
000.017	339X EC.DIF	DS	1		DIRECTORY FULL
000.020	340X EC.IFC	DS	1		ILLEGAL FILE CONTENTS
000.021	341X EC.NEM	DS	1		NOT ENOUGH MEMORY
000.022	342X EC.RF	DS	1		READ FAILURE
000.023	343X EC.WF	DS	1		WRITE FAILURE
000.024	344X EC.WPV	DS	1		WRITE PROTECTION VIOLATION
000.025	345X EC.WP	DS	1		DISK WRITE PROTECTED
000.026	346X EC.FAP	DS	1		FILE ALREADY PRESENT
000.027	347X EC.DDA	DS	1		DEVICE DRIVER ABORT
000.030	348X EC.FL	DS	1		FILE LOCKED
000.031	349X EC.FAD	DS	1		FILE ALREADY OPEN
000.032	350X EC.IS	DS	1		ILLEGAL SWITCH
000.033	351X EC.UUN	DS	1		UNKNOWN UNIT NUMBER
000.034	352X EC.FNR	DS	1		FILE NAME REQUIRED
000.035	353X EC.DIW	DS	1		DEVICE IS NOT WRITABLE (OR WRITE LOCKED)
000.036	354X EC.UNA	DS	1		UNIT NOT AVAILABLE
000.037	355X EC.ILV	DS	1		ILLEGAL VALUE
000.040	356X EC.ILO	DS	1		ILLEGAL OPTION
000.041	357	XTEXT	DDDEF		

359X ** DEVICE DRIVER COMMUNICATION FLAGS.

	360X *				
	361X				
	362X	ORG	0		
	363X				
000.000	364X DC.REA	DS	1		READ
000.001	365X DC.WRI	DS	1		WRITE
000.002	366X DC.RER	DS	1		READ REGARDLESS
000.003	367X DC.OPR	DS	1		OPEN FOR READ
000.004	368X DC.OPW	DS	1		OPEN FOR WRITE
000.005	369X DC.OPU	DS	1		OPEN FOR UPDATE
000.006	370X DC.CLO	DS	1		CLOSE

000.007 371X DC.ABT DS 1 ABORT
 000.010 372X DC.MDU DS 1 MOUNT DEVICE
 000.011 373 XTEXT PICDEF

375X ** PIC FORMAT EQUIVALENCES.

000.000 376X
 377X ORG 0
 378X
 000.000 379X PIC.ID DS 1 377Q = BINARY FILE FLAG
 000.001 380X DS 1 FILE TYPE (FT.PIC)
 000.002 381X PIC.LEN DS 2 LENGTH OF ENTIRE RECORD
 000.004 382X PIC.PTR DS 2 INDEX OF START OF PIC TABLE
 383X
 000.006 384X PIC.COD DS 0 CODE STARTS HERE
 000.006 385 XTEXT HOSEQU

387X ** HDOS SYSTEM EQUIVALENCES.

388X *
 389X
 024.000 390X S.GRT EQU 24000A SYSTEM AREA FOR GRT0
 025.000 391X S.GRT1 EQU 25000A SYSTEM AREA FOR GRT 1
 026.000 392X SECSCR EQU 26000A SYSTEM S12 BYTE SCRATCH AREA
 030.000 393X ROMBOOT EQU 30000A ROM BOOT ENTRY
 394X
 040.100 395X ORG 40100A FREE SPACE FROM PAM-8
 396X
 040.100 397X DS 8 JUMP TO SYSTEM EXIT
 040.110 398X D.CON DS 16 DISK CONSTANTS
 040.130 399X SYDD EQU * SYSTEM DISK ENTRY POINT
 040.130 400X D.VEC DS 24*3 SYSTEM ROM ENTRY VECTORS
 040.240 401X D.RAM DS 31 SYSTEM ROM WORK AREA
 040.277 402X S.VAL DS 38 SYSTEM VALUES
 040.345 403X S.INT DS 113 SYSTEM INTERNAL WORK AREAS
 041.126 404X DS 16
 041.146 405X S.SOVR DS 2 STACK OVERFLOW WARNING
 041.150 406X DS 42200A-* SYSTEM STACK
 001.032 407X STACKL EQU *-S.SOVR STACK SIZE
 408X
 042.200 409X STACK EQU * LWA+1 SYSTEM STACK
 042.200 410X USERFWA EQU * USER FWA
 042.200 411 XTEXT EDCON

	413X **	D.CON DETAILED EQUIVALENCES.			
	414X *				
	415X *	HOSEQU MUST BE MODIFIED WHEN THIS TABLE IS MODIFIED.			
	416X				
040.110	417X	ORG	D.CON		
	418X				
040.110	419X	D.XITA	DS	2	SEE SYSTEM ROM FOR DESCRIPTION
040.112	420X	D.WRITA	DS	1	
040.113	421X	D.WRITB	DS	1	
040.114	422X	D.WRITC	DS	1	
040.115	423X	D.MAIA	DS	1	
040.116	424X	D.LPSA	DS	1	
040.117	425X	D.SDFA	DS	1	
040.120	426X	D.SDFB	DS	1	
040.121	427X	D.STSA	DS	1	
040.122	428X	D.STSB	DS	1	
040.123	429X	D.WHDA	DS	1	
040.124	430X	D.WNHA	DS	1	
040.125	431X	D.WSCA	DS	1	
	432X				
040.126	433X	D.ERTS	DS	2	TRACK AND SECTOR OF LAST DISK ERRORS
040.130	434	XTEXT	EDVEC		

	436X **	JMP VECTORS FOR ROM CODE			
	437X *				
	438X *	SEE DISK ROM FOR ADDRESSES			
	439X *				
	440X *	HOSEQU MUST BE ALTERED WHEN THIS TABLE IS ALTERED.			
	441X				
040.130	442X	ORG	D.VEC		
	443X				
040.130	444X	D.SYDD	DS	3	JMP R.SYDD (MUST BE FIRST)
040.133	445X	D.MOUNT	DS	3	JMP R.MOUNT
040.136	446X	D.XOK	DS	3	JMP R.XOK
040.141	447X	D.ABORT	DS	3	JMP R.ABORT
040.144	448X	D.XIT	DS	3	JMP R.XIT
040.147	449X	D.READ	DS	3	JMP R.READ
040.152	450X	D.READR	DS	3	JMP R.READR
040.155	451X	D.WRITE	DS	3	JMP R.WRITE
040.160	452X	D.CDE	DS	3	JMP R.CDE
040.163	453X	D.DTS	DS	3	JMP R.DTS
040.166	454X	D.SDT	DS	3	JMP R.SDT
040.171	455X	D.MAI	DS	3	JMP R.MAI
040.174	456X	D.MAD	DS	3	JMP R.MAD
040.177	457X	D.LPS	DS	3	JMP R.LPS
040.202	458X	D.RDR	DS	3	JMP R.RDR
040.205	459X	D.SDP	DS	3	JMP R.SDP
040.210	460X	D.STS	DS	3	JMP R.STS
040.213	461X	D.STZ	DS	3	JMP R.STZ
040.216	462X	D.UDLY	DS	3	JMP R.UDLY
040.221	463X	D.WSC	DS	3	JMP R.WSC
040.224	464X	D.WSP	DS	3	JMP R.WSP
040.227	465X	D.WNB	DS	3	JMP R.WNB

040.232	466X	D.ERRT	DS	3	JMP	R.ERRT
040.235	467X	D.DLY	DS	3	JMP	R.DLY
040.240	468	XTEXT	EDRAM			
	470X	**			EDRAM - DISK RAM WORKAREA DEFINITION.	
	471X	*				
	472X	*			ZEROED UPON BOOTING UP.	
	473X	*				
	474X	*			HOSEQU MUST BE CHANGED WHEN THIS DECK IS CHANGED.	
	475X					
	476X					
040.240	477X	ORG	D.RAM			
	478X					
040.240	479X	D.TT	DS	1		TARGET TRACK (CURRENT OPERATION)
040.241	480X	D.TS	DS	1		TARGET SECTOR (CURRENT OPERATION)
	481X					
040.242	482X	D.DUCTL	DS	1		DEVICE CONTROL BYTE
	483X					
040.243	484X	D.DLYMO	DS	1		MOTOR ON DELAY COUNT
040.244	485X	D.DLYHS	DS	1		HEAD SETTLE DELAY COUNTER
	486X					
040.245	487X	D.TRKFT	DS	2		ADDRESS IN D.DRVTB FOR TRACK NUMBER
040.247	488X	D.VOLFT	DS	2		ADDRESS IN D.DRVTB FOR VOLUME NUMBER
	489X					
040.251	490X	D.DRVTB	DS	2*4		TRACK NUMBER AND VOLUME NUMBER FOR 4 DRIVES
	491X					
040.261	492X	D.HECNT	DS	1		HARD ERROR COUNT
040.262	493X	D.SECNT	DS	2		SOFT ERROR COUNT
040.264	494X	D.OECNT	DS	1		OPERATION ERROR COUNT
	495X					
	496X	*			GLOBAL DISK ERROR COUNTERS	
	497X					
040.265	498X	D.ERR	DS	0		BEGINNING OF ERROR BLOCK
040.265	499X	D.E.MDS	DS	1		MISSING DATA SYNC
040.266	500X	D.E.HSY	DS	1		MISSING HEADER SYNC
040.267	501X	D.E.CHK	DS	1		DATA CHECKSUM
040.270	502X	D.E.HCK	DS	1		HEADER CHECKSUM
040.271	503X	D.E.VOL	DS	1		WRONG VOLUME NUMBER
040.272	504X	D.E.TRK	DS	1		BAR TRACK SEEN
040.273	505X	D.ERRL	DS	0		LIMIT OF ERROR COUNTERS
	506X					
	507X	*			I/O OPERATION COUNTS	
	508X					
040.273	509X	D.OPR	DS	2		
040.275	510X	D.OPW	DS	2		
	511X					
000.037	512X	D.RAML	EQU	*-D.RAM		
040.277	513	XTEXT	ESVAL			

```
.....
515X **      S.VAL - SYSTEM VALUE DEFINITIONS.
516X *
517X *      THESE VALUES ARE SET AND MAINTAINED BY THE SYSTEM.
518X *
519X *      THE DECK HOSEQU MUST BE MODIFIED WHEN THIS IS MODIFIED.
520X
521X
040.277      522X      ORG      S.VAL
523X
040.277      524X S.DATE  DS      9      SYSTEM DATE (IN ASCII)
040.310      525X S.DATC  DS      2      CODED DATE
040.312      526X S.TIME  DS      4      TIME FROM MIDNIGHT (IN TICS)
040.316      527X S.HINEM DS      2      HARDWARE HIGH MEMORY ADDRESS+1
528X
040.320      529X S.SYSM  DS      2      FWA RESIDENT SYSTEM
530X
040.322      531X S.USRM  DS      2      LWA USER MEMORY
532X
040.324      533X S.OMAX  DS      2      MAX OVERLAY SIZE FOR SYSTEM
534X
535X
536X **      THE FOLLOWING FIVE CELLS SHOULD BE MODIFIED/READ ONLY VIA THE .CONSL SYSCALL
537X
000.200      538X CSL.ECH  EQU      10000000B  SUPPRESS ECHO
000.002      539X CSL.WRP  EQU      00000010B  WRAP LINES AT WIDTH
000.001      540X CSL.CHR  EQU      00000001B  OPERATE IN CHARACTER MODE
541X
000.000      542X I.CSLMD  EQU      0      S.CSLMD IS FIRST BYTE
040.326      543X S.CSLMD  DS      1      CONSOLE MODE
544X
000.200      545X CTP.BKS  EQU      10000000B  TERMINAL PROCESSES BACKSPACES
000.040      546X CTP.NLI  EQU      00100000B  MAP LOWER CASE TO UPPER ON INPUT
000.020      547X CTP.NLO  EQU      00010000B  MAP LOWER CASE TO UPPER ON OUTPUT
000.010      548X CTP.2SB  EQU      00001000B  TERMINAL NEEDS TWO STOP BITS
000.002      549X CTP.BKH  EQU      00000010B  MAP BKSP (UPON INPUT) TO RUBOUT
000.001      550X CTP.TAB  EQU      00000001B  TERMINAL SUPPORTS TAB CHARACTERS
551X
000.001      552X I.CONTY  EQU      1      S.CONTY IS 2ND BYTE
000.000      553X      ERRNZ  *-S.CSLMD-I.CONTY
040.327      554X S.CONTY  DS      1      CONSOLE TYPE FLAGS
000.002      555X I.CUSOR  EQU      2      S.CUSOR IS 3RD BYTE
000.000      556X      ERRNZ  *-S.CSLMD-I.CUSOR
040.330      557X S.CUSOR  DS      1      CURRENT CURSOR POSITION
000.003      558X I.CONWI  EQU      3      S.CONWI IS 4TH BYTE
000.000      559X      ERRNZ  *-S.CSLMD-I.CONWI
040.331      560X S.CONWI  DS      1      CONSOLE WIDTH
561X
000.001      562X CO.FLG  EQU      00000001B  CTL-D FLAG
000.200      563X CS.FLG  EQU      10000000B  CTL-S FLAG
564X
000.004      565X I.CONFL  EQU      4      S.CONFL IS 5TH BYTE
000.000      566X      ERRNZ  *-S.CSLMD-I.CONFL
040.332      567X S.CONFL  DS      1      CONSOLE FLAGS
568X
040.333      569X S.CAADR  DS      2      ADDRESS FOR ABORT PROCESSING (>256 IF VALID)
040.335      570X S.CCTAR  DS      6      ADDR FOR CTL-A, CTL-B, CTL-C PROCESSING
.....
```

040.343

571 XTEXT ESINT

573X ** S.INT - SYSTEM INTERNAL WORKAREA DEFINITIONS.

574X *

575X * THESE CELLS ARE REFERENCED BY OVERLAYS AND MAIN CODE, AND
576X * MUST THEREFORE RESIDE IN FIXED LOW MEMORY.

577X

578X

040.345

579X ORG S.INT

580X

581X ** CONSOLE STATUS FLAGS

582X

040.345

583X S.CDB DS 1 CONSOLE DESCRIPTOR BYTE

000.000

584X CDB,H85 EQU 00000000B

000.001

585X CDB,H84 EQU 00000001B =0 IF H8-5, =1 IF H8-4

040.346

586X S.BAUD DS 2 [0-14] H8-4 BAUD RATE, =0 IF H8-5

587X *

588X

589X ** TABLE ADDRESS WORDS

590X

040.350

591X S.DLINK DS 2 ADDRESS OF DATA IN HDOS CODE

040.352

592X S.CFWA DS 2 FWA CHANNEL TABLE

040.354

593X S.IFWA DS 2 FWA DEVICE TABLE

040.356

594X S.RFWA DS 2 FWA RESIDENT HDOS CODE

595X

596X ** DEVICE DRIVER DELAYED LOAD FLAGS

597X

040.360

598X S.DDLDA DS 2 DRIVER LOAD ADDRESS (HIGH BYTE=0 IF NO LOAD PENDING)

040.362

599X S.DDLEN DS 2 CODE LENGTH IN BYTES

040.364

600X S.DDGRF DS 1 GROUP NUMBER FOR DRIVER

040.365

601X DS 1 HOLD PLACE

602X

602X *S.DDSEC DS 2 SECTOR NUMBER FOR DRIVER (* OBSOLETE ! *)

040.366

603X S.DDUTA DS 2 DEVICE'S ADDRESS IN DEVLST +DEV.RES

040.370

604X S.DDOPC DS 1 OPEN OP CODE PENDING

605X

606X ** OVERLAY MANAGEMENT FLAGS

607X

000.001

608X OVL.IN EQU 00000001B IN MEMORY

000.002

609X OVL.RES EQU 00000010B PERMINANTLY RESIDENT

000.200

610X OVL.UCS EQU 10000000B USER CODE SWAPPED FOR OVERLAY

611X

040.371

612X S.OVLFL DS 1 OVERLAY FLAG

040.372

613X S.UCSF DS 2 FWA SWAPPED USER CODE

040.374

614X S.UCSL DS 2 LENGTH SWAPPED USER CODE

040.376

615X S.OVLS DS 2 SIZE OF OVERLAY CODE

041.000

616X S.OVLE DS 2 ENTRY POINT OF OVERLAY CODE

617X

041.002

618X S.SSN DS 2 SWAP AREA SECTOR NUMBER

041.004

619X S.OSN DS 2 OVERLAY SECTOR NUMBER

620X

621X * SYSCALL PROCESSING WORK AREAS

622X

041.006

623X S.CACC DS 1 (ACC) UPON SYSCALL

DISK DRIVER CODE

ESINT

09:59:37 02-APR-80

041.007	624X	S.CODE	DS	1	SYSCALL INDEX IN PROGRESS
	625X				
	626X	*			JUMPS TO ROUTINES IN RESIDENT HDOS CODE
	627X				
041.010	628X	S.JUMPS	DS	0	START OF DUMP VECTORS
041.010	629X	S.SDD	DS	3	JUMP TO STAND-IN DEVICE DRIVER
041.013	630X	S.FASER	DS	3	JUMP TO FATSERK (FATAL SYSTEM ERROR)
041.016	631X	S.DIREA	DS	3	JUMP TO DIREAD (DISK FILE READ)
041.021	632X	S.FCI	DS	3	JUMP TO FCI (FETCH CHANNEL INFO)
041.024	633X	S.SCI	DS	3	JUMP TO SCI (STORE CHANNEL INFO)
041.027	634X	S.MOUNT	DS	1	<>0 IF THE SYSTEM DISK IS MOUNTED
041.030	635X	S.DCS	DS	1	DEFAULT CLUSTER SIZE-1
	636X				
041.031	637X		DS	1	UNUSED
	638X				
	639X	*			STACK VALUE SAVED FOR OVERLAY SYSCALLS
	640X				
041.032	641X	S.OVSTK	DS	2	VALUE OF SP UPON SYSCALLS USING OVERLAY
	642X				
	643X	*			VOLUME DEPENDANT VALUES FOR SY1:
	644X				
041.034	645X	S.SIBIS	DS	2	DIRECTORY SECTOR
041.036	646X	S.SIGRT	DS	2	GRT SECTOR
	647X				
	648X				
	649X	**			ACTIVE I/O AREA.
	650X	*			
	651X	*			THE AIO.XXX AREA CONTAINS INFORMATION ABOUT THE I/O OPERATION
	652X	*			CURRENTLY BEING PERFORMED. THE INFORMATION IS OBTAINED FROM
	653X	*			THE CHANNEL TABLE, AND WILL BE RESTORED THERE WHEN DONE.
	654X	*			
	655X	*			NORMALLY, THE AIO.XXX INFORMATION WOULD BE OBTAINED DIRECTLY
	656X	*			FROM VARIOUS SYSTEM TABLES VIA POINTER REGISTERS. SINCE THE
	657X	*			8080 HAS NO GOOD INDEXED ADDRESSING, THE DATA IS MANUALLY
	658X	*			COPIED INTO THE AIO.XXX CELLS BEFORE PROCESSING, AND
	659X	*			BACKDATED AFTER PROCESSING.
	660X				
041.040	661X	AIO.VEC	DS	3	JUMP INSTRUCTION
041.041	662X	AIO.IDA	EQU	*-2	DEVICE DRIVER ADDRESS
041.043	663X	AIO.FLG	DS	1	FLAG BYTE
041.044	664X	AIO.GRT	DS	2	ADDRESS OF GROUP RESERV TABLE
041.046	665X	AIO.SPB	DS	1	SECTORS PER GROUP
041.047	666X	AIO.CGN	DS	1	CURRENT GROUP NUMBER
041.050	667X	AIO.CSI	DS	1	CURRENT SECTOR INDEX
041.051	668X	AIO.LGN	DS	1	LAST GROUP NUMBER
041.052	669X	AIO.LSI	DS	1	LAST SECTOR INDEX
041.053	670X	AIO.ITA	DS	2	DEVICE TABLE ADDRESS
041.055	671X	AIO.DES	DS	2	DIRECTORY SECTOR
041.057	672X	AIO.DEV	DS	2	DEVICE CODE
041.061	673X	AIO.UNI	DS	1	UNIT NUMBER (0-9)
	674X				
041.062	675X	AIO.DIR	DS	DIRELEN	DIRECTORY ENTRY
	676X				
041.111	677X	AIO.CNT	DS	1	SECTOR COUNT
041.112	678X	AIO.EDM	DS	1	END OF MEDIA FLAG
041.113	679X	AIO.EOF	DS	1	END OF FILE FLAG

DISK DRIVER CODE

ESINT

09:59:38 02-APR-80

041.114

680X AIO.TFP DS

2

TEMP FILE POINTERS

041.116

681X AIO.CHA DS

2

ADDRESS OF CHANNEL BLOCK (IOC.DDA)

030.060

734

XTEXT COMP

```

736X ** $COMP - COMPARE TWO CHARACTER STRINGS.
737X *
738X * $COMP COMPARES TWO BYTE STRINGS.
739X *
740X * ENTRY (C) = COMPARE COUNT
741X * (DE) = FWA OF STRING #1
742X * (HL) = FWA OF STRING #2
743X * EXIT 'Z' CLEAR, IS MIS-MATCH
744X * (C) = LENGTH REMAINING
745X * (DE) = ADDRESS OF MISMATCH IN STRING#1
746X * (HL) = ADDRESS OF MISMATCH IN STRING #2
747X * 'C' SET, HAVE MATCH
748X * (C) = 0
749X * (DE) = (DE) + (OC)
750X * (HL) = (HL) + (OC)
751X * USES A,F,C,D,E,H,L
752X
753X
030.060 032 754X $COMP LDAX D
030.061 276 755X CMP M COMPARE
030.062 300 756X RNE NO MATCH
030.063 023 757X INX D
030.064 043 758X INX H
030.065 015 759X DCR C
030.066 302 060 030 760X JNZ $COMP TRY SOME MORE
030.071 311 761X RET HAVE MATCH
030.072 762 XTEXT DADA

```

```

764X ** $DADA - PERFORM (H,L) = (H,L) + (0,A)
765X *
766X * ENTRY (H,L) = BEFORE VALUE
767X * (A) = BEFORE VALUE
768X * EXIT (H,L) = (H,L) + (0,A)
769X * 'C' SET IF OVERFLOW
770X * USES F,H,L
771X
772X
030.072 325 773X $DADA PUSH D
030.073 137 774X MOV E,A
030.074 026 000 775X MVI D,0
030.076 031 776X DAD D
030.077 321 777X POP D
030.100 311 778X RET EXIT
030.101 779 XTEXT DADA2

```

```
781X ** $DADA, - ADD (0,A) TO (H,L)
782X *
783X * ENTRY NONE
784X * EXIT (HL) = (HL) + (0A)
785X * USES A,F,H,L
786X
787X
030.101 205 788X $DADA, ADD L
030.102 157 789X MOV L,A
030.103 320 790X RNC
030.104 044 791X INR H
030.105 311 792X RET
030.106 793 XTEXT DU66

795X ** $DU66 - UNSIGNED 16 / 16 DIVIDE.
796X *
797X * (HL) = (BC)/(DE)
798X *
799X * ENTRY (BC), (DE) PRESET
800X * EXIT (HL) = RESULT
801X * (DE) = REMAINDER
802X * USES ALL
803X
804X
030.106 172 805X $DU66 MOV A,D TWOS COMPLEMENT (DE)
030.107 057 806X CMA
030.110 127 807X MOV D,A
030.111 173 808X MOV A,E
030.112 057 809X CMA
030.113 137 810X MOV E,A
030.114 023 811X INX D
030.115 172 812X MOV A,D
030.116 263 813X ORA E
030.117 312 205 030 814X JZ DU665 IF DIVIDE BY 0
030.122 257 815X XRA A
816X
817X * SHIFT (DE) LEFT UNTIL:
818X *
819X * 1) DE > BL
820X * 2) OVERFLOW
821X
030.123 142 822X DU661 MOV H,D
030.124 153 823X MOV L,E
030.125 011 824X DAD B
030.126 322 143 030 825X JNC DU662 IS TOO LARGE
030.131 074 826X INR A COUNT SHIFT
030.132 142 827X MOV H,D
030.133 153 828X MOV L,E
030.134 051 829X DAD H
030.135 353 830X XCHG (DE) = (DE)*2
030.136 332 123 030 831X JC DU661 IF NOT OVERFLOW
832X
833X * (DE) OVERFLOWED, PUT IT BACK.
```



```

834X
030.141 353 835X XCHG
030.142 075 836X DCR A REMOVE EXTRA COUNT
837X
838X * READY TO START SUBTRACTING. (A) = LOOP COUNT
839X
030.143 140 840X DU662 MOV H,B (H,L) = WORKING VALU
030.144 151 841X MOV L,C
030.145 001 000 000 842X LXI B,0 (BC) = RESULT
030.150 365 843X DU663 PUSH PSW SAVE (A)
030.151 031 844X DAD D
030.152 332 163 030 845X JC DU664 IF SUBTRACT OK
030.155 175 846X MOV A,L ADD BACK IN
030.156 223 847X SUR E
030.157 157 848X MOV L,A
030.160 174 849X MOV A,H
030.161 232 850X SBB D
030.162 147 851X MOV H,A
030.163 171 852X DU664 MOV A,C
030.164 027 853X RAL
030.165 117 854X MOV C,A
030.166 170 855X MOV A,B
030.167 027 856X RAL
030.170 107 857X MOV B,A
858X
859X * RIGHT SHFT (DE)
860X
030.171 067 861X STC
030.172 172 862X MOV A,D
030.173 037 863X RAR
030.174 127 864X MOV D,A
030.175 173 865X MOV A,E
030.176 037 866X RAR
030.177 137 867X MOV E,A
030.200 361 868X POP PSW
030.201 075 869X DCR A
030.202 362 150 030 870X JP DU663 IF NOT DONE
030.205 353 871X DU645 XCHG (D,E) = REMAINDER
030.206 140 872X MOV H,B (HL) = RESULT
030.207 151 873X MOV L,C
030.210 311 874X RET
030.211 875 XTEXT HLIHL

```

```

877X ** $HLIHL - LOAD HL INDIRECT THROUGH HL.
878X *
879X * (HL) = ((HL))
880X *
881X * ENTRY NONE
882X * EXIT NONE
883X * USES A,H,L
884X
030.211 176 885X $HLIHL MOV A,M
030.212 043 886X INX H

```

030.213	146	887X	MOV	H,H
030.214	157	888X	MOV	L,A
030.215	311	889X	RET	
030.216		890	XTEXT	CDEHL

892X	**	\$CDEHL = COMPARE (DE) TO (HL)		
893X	*			
894X	*	\$CDEHL COMPARES (DE) TO (HL) FOR EQUALITY.		
895X	*			
896X	*	ENTRY	NONE	
897X	*	EXIT	'Z' SET IF (DE) = (HL)	
898X	*	USES	A,F	
899X				
900X				

030.216	173	901X	\$CDEHL	MOV	A,E	
030.217	255	902X		XRA	L	
030.220	300	903X		RNZ		IF DIFFERENT
030.221	172	904X		MOV	A,B	
030.222	254	905X		XRA	H	
030.223	311	906X		RET		
030.224		907	XTEXT	CHL		COMPLEMENT (HL)

909X	**	\$CHL = COMPLEMENT (HL).		
910X	*			
911X	*	(HL) = -(HL)		TWO'S COMPLEMENT
912X	*			
913X	*	ENTRY	NONE	
914X	*	EXIT	NONE	
915X	*	USES	A,F,H,L	
916X				
917X				

030.224	174	918X	\$CHL	MOV	A,H	
030.225	057	919X		CMA		
030.226	147	920X		MOV	H,A	
030.227	175	921X		MOV	A,L	
030.230	057	922X		CMA		
030.231	157	923X		MOV	L,A	
030.232	043	924X		INX	H	
030.233	311	925X		RET		
030.234		926	XTEXT	INHL		INDEXED LOAD

```

928X ** $INDL - INDEXED LOAD.
929X *
930X * $INDL LOADS DE WITH THE TWO BYTES AT (HL)+DISPLACEMENT
931X *
932X * THIS ACTS AS AN INDEXED FULL WORD LOAD.
933X *
934X * (DE) = ( (HL) + DSPLACEMENT )
935X *
936X * ENTRY ((RET)) = DISPLACEMENT (FULL WORD)
937X * (HL) = TABLE ADDRESS
938X * EXIT TO (RET+2)
939X * USES A,F,D,E
940X
941X
030.234 343 942X $INDL XTHL (HL) = RET, ((SP)) = TBL ADDRESS
030.235 136 943X MOV E,M
030.236 043 944X INX H
030.237 126 945X MOV D,M (DE) = DISPLACEMENT
946X
030.240 043 947X INX H
030.241 343 948X XTHL ((SP)) = RET, (HL) = TBL ADDRESS
030.242 353 949X XCHG (DE) = TBL ADDRESS, (HL) = DISPLACEMENT
030.243 031 950X DAD D (HL) = TARGET ADDRESS
030.244 176 951X MOV A,M
030.245 043 952X INX H
030.246 146 953X MOV H,M
030.247 157 954X MOV L,A (HL) = ((HL))
030.250 353 955X XCHG (DE) = VALUE, (HL) = TABLE ADDRESS
030.251 311 956X RET

```

```

958 ** $MOVE - MOVE DATA
959 *
960 * $MOVE MOVES A BLOCK OF BYTES TO A NEW MEMORY ADDRESS.
961 * IF THE MOVE IS TO A LOWER ADDRESS, THE BYTES ARE MOVED FROM
962 * FIRST TO LAST.
963 *
964 * IF THE MOVE IS TO A HIGHER ADDRESS, THE BYTES ARE MOVED FROM
965 * LAST TO FIRST.
966 *
967 * THIS IS DONE SO THAT AN OVERLAPED MOVE WILL NOT 'RIPPLE'.
968 *
969 * ENTRY (BC) = COUNT
970 * (DE) = FROM
971 * (HL) = TO
972 * EXIT MOVED
973 * (DE) = ADDRESS OF NEXT FROM BYTE
974 * (HL) = ADDRESS OF NEXT *TO* BYTE
975 * 'C' CLEAR
976 * USES ALL
977
978
030.252 979 $MOVE EQU *
030.252 170 980 MOV A,B

```

*MOVE

```

030.253 261          981      ORA      C
030.254 310          982      RZ
030.255 175          983      MOV     A,L      NONE TO MOVE
030.256 223          984      SUB     E        COMPARE *FROM* TO *TO*
030.257 174          985      MOV     A,H
030.260 232          986      SBB    D
030.261 332 311 030  987      JC      MOV2     IS MOVE DOWN (TO LOWER ADDRESSES)
                                988
                                989 *      IS MOVE UP (TO HIGHER ADDRESSES)
                                990
030.264 013          991      DCX    B
030.265 011          992      DAD    B        (HL) = *TO* LWA
030.266 345          993      PUSH   H        SAVE *TO* LIMIT
030.267 353          994      XCHG
030.270 011          995      DAD    B        (HL) = *FROM* LWA
030.271 345          996      PUSH   H        SAVE *FROM* LIMIT
                                997
030.272 176          998      MOV1   MOV     A,M      MOVE BYTE
030.273 022          999      STAX  D
030.274 033          1000     DCX    D        INCREMENT *TO* ADDRESS
030.275 053          1001     DCX    H        INCREMENT *FROM* ADDRESS
030.276 013          1002     DCX    B        DECREMENT COUNT
030.277 170          1003     MOV     A,B
030.300 247          1004     ANA    A
030.301 362 272 030  1005     JP     MOV1     MORE TO GO
030.304 321          1006     POP    B        (DE) = *FROM* LIMIT
030.305 341          1007     POP    H        (HL) = *TO* LIMIT
030.306 023          1008     INX    D
030.307 043          1009     INX    H
030.310 311          1010     RET
                                1011
                                1012 *      IS MOVE DOWN (TO LOWER ADDRESSES)
                                1013
030.311 032          1014     MOV2   LDAX   D        MOVE BYTE
030.312 167          1015     MOV     M,A
030.313 043          1016     INX    H        INCREMENT *FROM*
030.314 023          1017     INX    D        INCREMENT *TO*
030.315 013          1018     DCX    B        DECREMENT COUNT
030.316 170          1019     MOV     A,B
030.317 261          1020     ORA    C
030.320 302 311 030  1021     JNZ   MOV2     IF NOT DONE
030.323 311          1022     RET        DONE
030.324          1023     XTEXT  MU10

```

```

1025X **      *MU10 - MULTIPLY UNSIGNED 16 BIT QUANTITY BY 10.
1026X *
1027X *      (HL) = (DE)*10
1028X *
1029X *      ENTRY (DE) = MULTIPLIER
1030X *      EXIT  'C' CLEAR IF OK
1031X *      (HL) = PRODUCT
1032X *      'C' SET IF ERROR
1033X *      USES  B,E,H,L,F

```

```

1034X
1035X
030.324 353 1036X $MU10 XCHG (HL) = MULTIPLIER
030.325 051 1037X DAD H (HL) = X*2
030.326 330 1038X RC
030.327 124 1039X MOV D,H
030.330 135 1040X MOV E,L
030.331 051 1041X DAD H (HL) = X*4
030.332 330 1042X RC
030.333 051 1043X DAD H (HL) = X*8
030.334 330 1044X RC
030.335 031 1045X DAD D (HL) = X*10
030.336 311 1046X RET
030.337 1047 XTEXT MU66

1049X ** $MU66 - UNSIGNED 16X16 MULTIPLY.
1050X *
1051X * ENTRY (BC) = MULTIPLICAND
1052X * (DE) = MULTIPLIER
1053X * EXIT (HL) = RESULT
1054X * 'Z' SET IF NOT OVERFLOW
1055X * USES ALL
1056X
1057X
030.337 257 1058X $MU66 XRA A
030.340 365 1059X PUSH PSW SAVE OVERFLOW STATUS
030.341 041 000 000 1060X LXI H,0 (HL) = RESULT ACCUMULATOR
1061X
030.344 170 1062X MU661 MOV A,B
030.345 037 1063X RAR
030.346 107 1064X MOV B,A
030.347 171 1065X MOV A,C
030.350 037 1066X RAR
030.351 117 1067X MOV C,A
030.352 322 364 030 1068X JNC MU662 IF BIT CLEAR
030.355 031 1069X DAD D
030.356 322 364 030 1070X JNC MU662 IF NOT OVERFLOW
030.361 361 1071X POP PSW
030.362 074 1072X INR A
030.363 365 1073X PUSH PSW
030.364 170 1074X MU662 MOV A,B
030.365 261 1075X ORA C SEE IF MULTIPLIER 0
030.366 312 005 031 1076X JZ MU663 IS ZERO? AM DONE
030.371 353 1077X XCHG
030.372 051 1078X DAD H (D,E) = (DE)*2
030.373 353 1079X XCHG
030.374 322 344 030 1080X JNC MU661 IF NOT OVERFLOW
030.377 361 1081X POP PSW
031.000 074 1082X INR A
031.001 365 1083X PUSH PSW FLAG OVERFLOW
031.002 303 344 030 1084X JMP MU661 PROCESS NEXT BIT
1085X
031.005 361 1086X MU663 POP PSW (A,E) = OVERFLOW STATUS

```

```

031.006 311      1087X      RET
031.007          1088      XTEXT  MUB6

1090X **      $MUB6 - MULTIPLY 8X16 UNSIGNED.
1091X *
1092X *      $MUB6 MULTIPLIES A 16 BIT VALUE BY A 8
1093X *      BIT VALUE.
1094X *
1095X *      ENTRY  (A) = MULTIPLIER
1096X *      (DE) = MULTIPLICAND
1097X *      EXIT   (HL) = RESULT
1098X *      'Z' SET IF NOT OVERFLOW
1099X *      USES  A,F,H,L
1100X
1101X
031.007 041 000 000 1102X $MUB6 LXI  H,0      (HL) = RESULT ACCUMULATOR
031.012 305          1103X      PUSH  B
031.013 104          1104X      MOV   B,H      (B) = OVERFLOW FLAG
031.014 267          1105X MUB60  ORA   A      CLEAR CARRY
1106X
031.015 037          1107X MUB61  RAR
031.016 322 026 031 1108X      JNC   MUB62   IF NOT TO ADD
031.021 031          1109X      DAD   D
031.022 322 026 031 1110X      JNC   MUB62   NOT OVERFLOW
031.025 004          1111X      INR   B
031.026 267          1112X MUB62  ORA   A
031.027 312 044 031 1113X      JZ    MUB63   IF DONE
031.032 353          1114X      XCHG
031.033 051          1115X      DAD   H
031.034 353          1116X      XCHG
031.035 322 015 031 1117X      JNC   MUB61   LOOP IF NOT OVERFLOW
031.040 004          1118X      INR   B
031.041 303 014 031 1119X      JMP   MUB60
1120X
031.044 260          1121X MUB63  ORA   B      SET *Z* FLAG IF NOT OVERFLOW
031.045 301          1122X      POP  B      RESTORE (BC)
031.046 311          1123X      RET
031.047          1124      XTEXT  SAVALL

```

```

1126X **      *RSTALL - RESTORE ALL REGISTERS.
1127X *
1128X *      *RSTALL RESTORES ALL THE REGISTERS OFF THE STACK, AND
1129X *      RETURNS TO THE PREVIOUS CALLER.
1130X *
1131X *      ENTRY  (SP) = PSW
1132X *      (SP+2) = BC
1133X *      (SP+4) = DE
1134X *      (SP+6) = HL
1135X *      (SP+8) = RET
1136X *      EXIT  TO *RET*, REGISTERS RESTORED
1137X *      USES  ALL

```

```

1138X
1139X
031.047 361 1140X $RSTALL POP PSM
031.050 301 1141X POP B
031.051 321 1142X POP D
031.052 341 1143X POP H
031.053 311 1144X RET

1146X ** $SAVALL - SAVE ALL REGISTERS ON STACK.
1147X *
1148X * $SAVALL SAVES ALL THE REGISTERS ON THE STACK.
1149X *
1150X * ENTRY NONE
1151X * EXIT (SP) = PSM
1152X * (SP+2) = BC
1153X * (SP+4) = DE
1154X * (SP+6) = HL
1155X * USES H,L
1156X
1157X
031.054 343 1158X $SAVALL XTHL PUSH H; (HL) = RETURN ADDRESS
031.055 325 1159X PUSH D
031.056 305 1160X PUSH B
031.057 365 1161X PUSH PSM
031.060 351 1162X PCHL RETURN TO CALLER
031.061 1163X XTEXT TJMP

```

```

1165X ** $TJMP - TABLE JUMP.
1166X *
1167X * USAGE
1168X *
1169X * CALL $TJMP (A) = INDEX
1170X * DW ADDR1 INDEX = 0
1171X * .
1172X * .
1173X * .
1174X * DW ADDRn INDEX = N-1
1175X *
1176X * ENTRY (A) = INDEX
1177X * EXIT TO PROCESSOR
1178X * (A) = INDEX*2
1179X * USES A,F
1180X
1181X
031.061 007 1182X $TJMP RLC (A) = INDEX*2
1183X
031.062 1184X $TJMP. EQU *
031.062 343 1185X XTHL (HL) = TABLE ADDRESS
031.063 365 1186X PUSH PSM SAVE INDEX*2
031.064 315 101 030 1187X CALL $BADA.
031.067 176 1188X MOV A,M

```

```

031.070 043      1189X      INX      H
031.071 146      1190X      MOV      M,M
031.072 157      1191X      MOV      C,A
031.073 361      1192X      POP      PSW      (A) = INDEX*2
031.074 343      1193X      XTHL           ADDRESS ON STACK
031.075 311      1194X      RET              JUMP TP PROCESSOR
031.076          1195      XTEXT      TBRA
    
```

```

1197X **      $TBRA - BRANCH RELATIVE THOUGH TABLE.
1198X *
1199X *      $TBRA USES THE SUPPLIED INDEX TO SELECT A BYTE FROM THE
1200X *      JUMP TABLE. THE CONTENTS OF THIS BYTE ARE ADDED TO THE
1201X *      ADDRESS OF THE BYTE, YEILDING THE PROCESSOR ADDRESS.
1202X *
1203X *      CALL      $TBRA
1204X *      DB      LAB1-*      INDEX = 0 FOR LAB1
1205X *      DB      LAB2-*      INDEX = 1 FOR LAB2
1206X *      DB      LABN-*      INDEX = N-1 FOR LABN
    
```

```

1207X *
1208X *      ENTRY   (A) = INDEX
1209X *      (RET) = TABLE FWA
1210X *      EXIT    TO COMPUTED ADDRESS
1211X *      USES    F,H,L
1212X *
1213X *
031.076          1214X $TBRA EQU      *
031.076 343      1215X XTHL           (HL) = TABLE ADDRESS
031.077 325      1216X PUSH      D
031.100 137      1217X MOV      E,A
031.101 026 000 1218X MVI      D,0
031.103 031      1219X DAD      D      (HL) = ADDRESS OF ELEMENT
031.104 136      1220X MOV      E,M
031.105 031      1221X DAD      D      (HL) = PROCESSOR ADDRESS
031.106 321      1222X POP      B
031.107 343      1223X XTHL
031.110 311      1224X RET
    
```

```

1226 **      $TBLS - TABLE SEARCH
1227 *
1228 *      TABLE FORMAT
1229 *
1230 *      DB      KEY1,VAL1,
1231 *      .
1232 *      .
1233 *      DB      KEYN,VALN
1234 *      DB      0
1235 *
1236 *      ENTRY   (A) = PATIERN
1237 *      (H>L) = TABLE FWA
1238 *      EXIT    (A) = PATTERN IF FOUND
    
```



```

1239 *           'Z' SET IF FOUND
1240 *           USES   A,F,H,L
1241
1242
031.111 305     1243 $TBLS  PUSH  B
031.112 107     1244      MOV  B+A
031.113 176     1245 $TBL1  MOV  A,M      (A) = CHARACTER
031.114 270     1246      CMP  B
031.115 312 133 031 1247      JZ   $TBL2      IF MATC
031.120 247     1248      ANA  A
031.121 043     1249      INX  H
031.122 043     1250      INX  H      SKIP PAST
031.123 302 113 031 1251      JNZ  $TBL1      IF NOT END OF TABLE
031.126 053     1252      DCX  H
031.127 053     1253      DCX  H
031.130 264     1254      ORA  H      CLEAR 'Z'
031.131 076 000 1255      MVI  A,O      SET '(A)' = 'O' FOR OLD USERS
1256
1257 *           DONE
1258
031.133 301     1259 $TBL2  POP  B
031.134 043     1260      INX  H
031.135 311     1261      RET

```

```

1263 **          $TYPTX - TYPE TEXT.
1264 *
1265 *          $TYPTX IS CALLED TO TYPE A BLOCK OF TEXT ON THE SYSTEM CONSOLE.
1266 *
1267 *          IMBEDDED ZERO BYTES INDICATE A CARRIAGE RETURN LINE FEED,
1268 *          A BYTE WITH THE '0000' BIT SET IS THE LAST BYTE IN THE MESSAGE.
1269 *
1270 *          ENTRY (RET) = TEXT
1271 *          EXIT  TO (RET+LENGTH)
1272 *          USES  A,F
1273
1274
031.136 343     1275 $TYPTX  XTHL      (HL) = TEXT ADDRESS
031.137 315 144 031 1276      CALL $TYPTX.  TYPE IT
031.142 343     1277      XTHL
031.143 311     1278      RET
1279
031.144 176     1280 $TYPTX. MOV  A,M
031.145 346 177 1281      ANI  1770
031.147 377 002 1282      DB   SYSCALL,.SCOUT
031.151 276     1283      CMP  H
031.152 043     1284      INX  H
031.153 312 144 031 1285      JE   $TYPTX.  MORE TO GO
031.156 311     1286      RET
031.157         1287      XTEXT  UDD

```

```

1289X ** $UDD - UNPACK DECIMAL DIGITS.
1290X *
1291X * UDD CONVERTS A 16 BIT VALUE INTO A SPECIFIED NUMBER OF
1292X * DECIMAL DIGITS. THE RESULT IS ZERO FILLED.
1293X *
1294X * ENTRY (B,C) = ADDRESS VALUE
1295X * (A) = DIGIT COUNT
1296X * (H,L) = MEMORY ADDRESS
1297X * EXIT (HL) = (HL) + (A)
1298X * USES ALL
1299X
1300X
031.157 1301X $UDD EQU *
031.157 315 072 030 1302X CALL $DADA
031.162 345 1303X PUSH H SAVE FINAL (H,L) VALUE
1304X
031.163 365 1305X UDD1 PUSH PSW
031.164 345 1306X PUSH H
031.165 021 012 000 1307X LXI D,10
031.170 315 106 030 1308X CALL $DU66 (H,L) = VALUE/10
031.173 345 1309X PUSH H
031.174 301 1310X POP R (R,C) = REMAINDER
031.175 341 1311X POP H
031.176 076 060 1312X MVI A,'0'
031.200 203 1313X ADD E ADD REMAINDER
031.201 053 1314X DCX H
031.202 167 1315X MOV M,A STORE DIGIT
031.203 361 1316X POP PSW
031.204 075 1317X DCR A
031.205 302 163 031 1318X JNZ UDD1 IF MORE TO GO
031.210 341 1319X POP H RESTORE H
031.211 311 1320X RET RETURN
031.212 1321X XTEXT ZERO

```

```

1323X ** $ZERO - ZERO MEMORY
1324X *
1325X * $ZERO ZERDS A BLOCK OF MEMORY.
1326X *
1327X * ENTRY (HL) = ADDRESS
1328X * (B) = COUNT
1329X * EXIT (A) = 0
1330X * USES A,B,F,H,L
1331X
1332X
031.212 257 1333X $ZERO XRA A
031.213 167 1334X ZR01 MOV M,A
031.214 043 1335X INX H
031.215 005 1336X DCR R
031.216 302 213 031 1337X JNZ ZR01 IF MORE
031.221 311 1338X RET

```

```
1340 **      $WDR - WRITE DISABLE RAM.
1341 *
1342 *      $WDR IS CALLED TO DISABLE THE WRITABILITY OF THE
1343 *      H17 CONTROLLER RAM AREA.
1344 *
1345 *      ENTRY  NONE
1346 *      EXIT   NONE
1347 *      USES  NONE
1348
1349
1350 $WDR      PUSH  PSW
1351          DI
1352          LDA   D.DVCTL
1353          ANI  377Q-DF.WR
1354 $WDR1     STA  D.DVCTL
1355          OUT  DP.BC
1356          EI
1357          POP  PSW
1358          RET

1360 **      $WER - WRITE ENABLE RAM.
1361 *
1362 *      $WER IS CALLED TO ENABLE WRITING TO THE H17 CONTROLLERS RAM AREA.
1363 *
1364 *      ENTRY  NONE
1365 *      EXIT   NONE
1366 *      USES  NONE
1367
1368
1369 $WER      PUSH  PSW
1370          DI
1371          LDA  D.DVCTL
1372          ORI  DF.WR
1373          JMP  $WDR1

1375 **      D.DISK - DEVICE DRIVER READ CODE.
1376 *
1377 *      ENTRY  (BC) = COUNT (IN SECTORS)
1378 *            (DE) = ADDRESS
1379 *            (HL) = SECTOR
1380 *      EXIT  'C' CLEAR IF OK, EXIT TO CALLER
1381 *            'C' SET IF ERROR
1382 *            TO S.FASER (FATAL SYSTEM ERROR) IF UNIT 0
1383 *            TO CALLER IF OTHER UNIT
1384 *            (A) = ERROR CODE
1385
1386
1387 $WRITE    MVI  A,DC,WRI
1388          DB   MI,CPI          SKIP NEXT
1389 $DREAD    XRA  A              SET READ CODE
```

D.DISK

				1390				
				1391	ERRNZ	DC.REA		
031.257	315	130	040	1392	CALL	SYDD	CALL DEVICE DRIVER	
031.262	320			1393	RNC		IF OK	
031.263	365			1394	PUSH	PSW	SAVE CODE	
031.264	072	061	041	1395	LDA	AID.UNI		
031.267	247			1396	ANA	A		
031.270	314	013	041	1397	CZ	S.FASER	IS SYO:	
031.273	361			1398	POP	PSW		
031.274	311			1399	RET		RETURN WITH BAD NEWS	

				1401	**	SREAD - READ FROM SYSTEM DISK.		
				1402	*			
				1403	*			
				1404	*	ENTRY (BC) = COUNT (IN SECTORS)		
				1405	*	(DE) = ADDRESS		
				1406	*	(HL) = SECTOR		
				1407	*	EXIT TO CALLER IF OK		
				1408	*	TO S.FASER (FATAL SYSTEM ERROR) IF ERROR		
				1409	*			

				1410				
031.275	072	061	041	1411	SREAD	LDA AID.UNI		
031.300	365			1412	PUSH	PSW	SAVE CURRENT UNIT	
031.301	257			1413	XRA	A		
000.000				1414	ERRNZ	DC.REA		
031.302	062	061	041	1415	STA	AID.UNI		
031.305	315	130	040	1416	SREAD1	CALL SYDD		
031.310	334	013	041	1417	CC	S.FASER	READ ERROR	
031.313	361			1418	POP	PSW		
031.314	062	061	041	1419	STA	AID.UNI		
031.317	311			1420	RET			

				1421				
				1422	**	CONSTANT ZEROS		
				1423	*			

031.320	000	000	000	1424	ZEROS	DR	0,0,0,0,0,0,0,0	
---------	-----	-----	-----	------	-------	----	-----------------	--

				1426	**	SWRITE - WRITE TO SYSTEM DISK.		
				1427	*			
				1428	*			
				1429	*	ENTRY (BC) = COUNT (IN SECTORS)		
				1430	*	(DE) = ADDRESS		
				1431	*	(HL) = SECTOR		
				1432	*	EXIT TO CALLER IF OK		
				1433	*	TO S.FASER (FATAL SYSTEM ERROR) IF ERROR		
				1434	*			

				1435				
031.330	072	061	041	1436	SWRITE	LDA AID.UNI		
031.333	365			1437	PUSH	PSW	SAVE OLD UNIT #	
031.334	257			1438	XRA	A		
031.335	062	061	041	1439	STA	AID.UNI	SET SYSTEM UNIT	

000.000			1440	ERRNZ	DC.WRI-1	
031.340	074		1441	INR	A	(A) = DC.WRI
031.341	303 305 031		1442	JMP	SREADI	

1446 ** ERR.FNO - ERROR: FILE NOT OPEN
 1447
 031.344 076 011 1448 ERR.FNO MVI A,EC.FNO FILE NOT OPEN
 031.346 067 1449 STC
 031.347 311 1450 RET ERROR CODE

1452 ** ERR.ILR - ERROR: ILLEGAL REQUEST
 1453
 031.350 076 012 1454 ERR.ILR MVI A,EC.ILR ILLEGAL REQUEST
 031.352 067 1455 STC
 031.353 311 1456 RET

1458 ** CFF - CHAIN FREE BLOCK TO FILE.
 1459 *
 1460 * CFF UNCHAINS A FREE BLOCK FROM THE FREE LIST, AND CHAINS
 1461 * IT ONTO THE END OF THE ACTIVE FILE.
 1462 *
 1463 * ENTRY (HL) = ADDRESS IN GROUP TABLE OF THE GROUP IN QUESTION.
 1464 * (E) = INDEX OF PREVIOUS GROUP IN LIST
 1465 * AIO,XXX SETUP
 1466 * EXIT AIO,LGN = (L) (AT ENTRY)
 1467 * AIO,LSI = 0
 1468 * USES A,F,D,H,L
 1469
 1470
 031.354 176 1471 CFF MOV A,H (A) = NEXT FREE
 031.355 066 000 1472 MVI M,0 NEW BLOCK IS END OF CHAIN FOR FILE
 031.357 125 1473 MOV D,0 (D) = NEW INDEX
 031.360 153 1474 MOV L,E (HL) = ADDRESS OF PREVIOUS BLOCK
 031.361 167 1475 MOV M,A UNCHAIN FROM FREE CHAIN
 031.362 072 051 041 1476 LDA AIO,LGN (A) = LAST GROUP #
 031.365 157 1477 MOV L,A (HL) = ADDRESS OF FILE LAST GROUP
 031.366 162 1478 MOV M,D LINK TO NEW LAST BLOCK
 031.367 041 051 041 1479 LXI M,AIO,LGN
 031.372 162 1480 MOV M,D SET NEW LGN
 000.000 1481 ERRNZ AIO,LSI-AIO,LGN-1
 031.373 043 1482 INX H
 031.374 066 000 1483 MVI M,0 CLEAR LSI
 031.376 311 1484 RET

1486 ** DCA - DETERMINE CONTIGUOUS AREA.
 1487 *
 1488 * DCA IS CALLED TO FIND HOW MANY OF THE SECTORS WHICH ARE TO BE
 1489 * READ ARE CONTIGUOUS.
 1490 *
 1491 * ENTRY (B) = SECTORS DESIRED
 1492 * AIO,XXX SETUP
 1493 * EXIT (B) = SECTORS-AIO,CN1

```
1494 * AIO.CNT = SECTORS WHICH ARE CONTIGUOUS
1495 * AIO.EOF = EC.EOF*2+1 IF EOF
1496 * AIO.TFF = SETUP WITH GROUP AND INDEX OF START OF AREA
1497 * USES ALL
1498
1499
031.377 315 205 032 1500 DCA1 CALL FFL FOLLOW FORWARD LINK
1501
032.002 052 047 041 1502 DCA LHL D AIO.CGN (H) = CURRENT GP #; (L) = CUR SECT INDEX
000.000 ERRNZ AIO.CSI-AIO.CGN-1
032.005 042 114 041 1504 SHLD AIO.TFF TEMP FILE POINTER
032.010 315 233 033 1505 CALL TFE TEST FOR EOF
032.013 062 113 041 1506 STA AIO.EOF SET FLAG
032.016 062 111 041 1507 STA AIO.CNT SET CNT=0 IF EOF
032.021 310 1508 RE IS EOF
032.022 072 050 041 1509 LDA AIO.CSI (A) = CURRENT SECTOR INDEX
032.025 147 1510 MOV H,A
032.026 072 046 041 1511 LDA AIO.SPG
032.031 274 1512 CMP H SEE IF GROUP EXHAUSTED
032.032 312 377 031 1513 JE DCA1 WAS POINTING AT END OF GROUP
1514
1515 * SEE IF MORE NEEDED.
1516
032.035 170 1517 DCA2 MOV A,B
032.036 247 1518 ANA A
032.037 310 1519 RZ NO MORE SECTORS TO CHECK
1520
1521 * SEE HOW MANY SECTORS ARE LEFT IN THIS GROUP
1522
032.040 052 051 041 1523 LHL D AIO.LGN (L) = AIO.LGN; (H) = AIO.LSI
000.000 ERRNZ AIO.LSI-AIO.LGN-1
032.043 072 047 041 1525 LDA AIO.CGN
032.046 275 1526 CMP L SEE IF WE ARE POINTED AT LAST GROUP
032.047 312 055 032 1527 JE DCA3 WE ARE
032.052 052 045 041 1528 LHL D AIO.SPG-1 (H) = AIO.SPG
032.055 365 1529 DCA3 FUSH PSW SAVE STATUS
032.056 072 050 041 1530 LDA AIO.CSI (A) = CURRENT SECTOR INDEX
032.061 224 1531 SUB H (A) = -SECTORS LEFT IN GROUP
032.062 057 1532 CMA
032.063 074 1533 INR A (A) = +SECTORS LEFT IN GROUP
032.064 270 1534 CMP B
032.065 332 071 032 1535 JC DCA4 NEED STILL MORE
032.070 170 1536 MOV A,B DONT TAKE MORE THAN WE NEED
032.071 117 1537 DCA4 MOV C,A (C) = AMOUNT TO TAKE
032.072 041 050 041 1538 LXI H,AIO.CSI
032.075 206 1539 ADD M UPDATE CSI TO INDICATE NUMBER TO BE READ
032.076 167 1540 MOV H,A
032.077 171 1541 MOV A,C (A) = NUMBER TO BE READ
032.100 041 111 041 1542 LXI H,AIO.CNT
032.103 206 1543 ADD M ADD TO COUNT
032.104 167 1544 MOV H,A
032.105 170 1545 MOV A,B (A) = AMOUNT NEEDED
032.106 221 1546 SUB C
032.107 107 1547 MOV B,A
032.110 361 1548 POP PSW
032.111 310 1549 RE WAS ON LAST TRACK; AM DONE
```

```

032.112 170      1550      MOV      A,B
032.113 247      1551      ANA      A
032.114 310      1552      RZ              NO MORE NEEDED; AM DONE
1553
1554 *          USED UP THIS BLOCK, LINK TO THE NEXT.
1555 *          IF NOT CONTIGUOUS, STOP HERE
1556
032.115 072 047 041 1557      LDA      AIO,CGN
032.120 074      1558      INR      A
032.121 365      1559      PUSH     PSW      SAVE NEXT CONTIGUOUS BLOCK #
032.122 315 205 032 1560      CALL     FFL      FOLLOW FILE LINK
032.125 361      1561      POP      PSW
032.126 275      1562      CMP      L
032.127 312 035 032 1563      JE       DCA2
032.132 311      1564      RET              GOT IT, WAS CONTIGUOUS
                          STOP HERE

1566 **        FFB - FIND FREE BLOCK.
1567 *
1568 *          FFB IS CALLED TO LOCATE A FREE BLOCK IN THE GRT'S
1569 *          FREE CHAIN.
1570 *
1571 *          FFB WILL ATTEMPT TO GET A 'PREFERRED BLOCK', IF POSSIBLE,
1572 *          IF THE PREFERRED BLOCK IS NOT AVAILABLE, FFB WILL (OPTIONALLY) DO
1573 *          THE BEST HE CAN; START A VIRGIN CLUSTER, IF POSSIBLE, THEN
1574 *          JUST SETTLE FOR ANYTHING.
1575 *
1576 *          ENTRY (D) = PREFERRED BLOCK NUMBER (0 IF NONE)
1577 *          (C) = PREFERRED FLAG (=0, WILL TAKE SOMETHING ELSE,
1578 *          <math>\diamond</math>0, MUST HAVE PREFERRED BLOCK (OR NOTHING))
1579 *          EXIT 'C' SET, EOM ON DEVICE
1580 *          'C' CLEAR, NOT EOM
1581 *          'Z' CLEAR, COULDN'T GET PREFERRED BLOCK (ONLY IF (C)<math>\diamond</math>0 ON ENTRY)
1582 *          'Z' SET, GOT A BLOCK (PREFERRED OR NOT)
1583 *          (HL) = ADDRESS OF BLOCK IN GRT TABLE
1584 *          (E) = INDEX OF FREE BLOCK BEFORE THE FOUND ONE
1585 *          USES  A,F,E,H,L
1586
032.133 052 044 041 1588      FFB      LHLD     AIO,GRT
032.136 176      1589      MOV      A,M      (A) = FIRST FREE BLOCK
032.137 247      1590      ANA      A
032.140 067      1591      STC              ASSUME EOM
032.141 310      1592      RZ              END OF MEDIA
1593
1594 *          NOT END OF MEDIA. TRY TO FIND THE CONTIGUOUS BLOCK IN THE FREE LIST.
1595
032.142 135      1596      MOV      E,i      (E) = INDEX OF PREVIOUS BYTE
032.143 054      1597      INR      L
032.144 147      1598      MOV      M,A      FLAG CHANGE IN GRT
032.145 157      1599      FFB4     MOV      L,A      (HL) = ADDRESS OF NEXT BYTE IN FREE CHAIN
032.146 272      1600      CMP      D
032.147 310      1601      RE              GOT THE ONE WE NEED
032.150 322 161 032 1602      JNC     FFB5     GONE TOO FAR
  
```



```
032.153 135 1603 MOV E,L SAVE THIS BLOCK INDEX
032.154 176 1604 MOV A,M
032.155 247 1605 ANA A
032.156 302 145 032 1606 JNZ FFR4 TRY AGAIN
1607
1608 * COULDN'T FIND CONTIGUOUS BLOCK. THIS MEANS A BREAK IN
1609 * CONTINUITY. IF WE HAVE ANYTHING, RETURN WITH IT.
1610 * IF WE HAVE NOTHING YET, TRY TO FIND A VIRGIN CLUSTER.
1611
032.161 171 1612 FFR5 MOV A,C
032.162 247 1613 ANA A
032.163 300 1614 RNZ MUST NOT CONTINUE
032.164 157 1615 MOV L,A (HL) = (AIO,BRT)
032.165 135 1616 FFR6 MOV E,L (E) = INDEX OF PREVIOUS NODE
032.166 156 1617 MOV L,M LINK FORWARD
032.167 072 077 041 1618 LDA AIO,DIR+DIR,CLU
032.172 245 1619 ANA L SEE IF START OF CLUSTER
032.173 310 1620 KZ GOT VIRGIN CLUSTER
032.174 176 1621 MOV A,M
032.175 247 1622 ANA A
032.176 302 165 032 1623 JNZ FFR6 TRY AGAIN
1624
1625 * CANT FIND VIRGIN CLUSTER. WILL TAKE WHATEVER WE CAN GET
1626
032.201 157 1627 MOV L,A
032.202 135 1628 MOV E,L (E) = INDEX OF PREVIOUS NODE
032.203 156 1629 MOV L,M (HL) = ADDRESS OF FIRST FREE BLOCK BYTE
032.204 311 1630 RET RETURN WITH 'Z': GOT ONE

1632 ** FFL - FOLLOW FORWARD LINK.
1633 *
1634 * FFL LINKS AIO,CGN TO THE NEXT GROUP
1635 *
1636 * ENTRY NONE
1637 * EXIT AIO,CGN = LINK(AIO,CGN)
1638 * AIO,CSI = 0
1639 * (L) = AIO,CGN
1640 * USES A,F,H,L
1641
1642
032.205 052 044 041 1643 FFL LHLD AIO,BRT
032.210 072 047 041 1644 LDA AIO,CGN
032.213 157 1645 MOV L,A (HL) = ADDRESS
032.214 156 1646 MOV L,M (L) = LINK
032.215 046 000 1647 MVI H,0
032.217 042 047 041 1648 SHLD AIO,CGN SET CGN, CLEAR CSI
000.000 1649 ERRNZ AIO,CSI-AIO,CGN-1
032.222 311 1650 RET
```

```
1652 ** LDD - LOAD DEVICE DRIVER.
1653 *
1654 * LDD IS CALLED TO PERFORM THE SUSPENDED LOAD OF A DEVICE DRIVER.
1655 *
1656 * IF SOME OVL CODE WISHES TO LOAD A DEVICE DRIVER, IT MUST
1657 * SUSPEND THE REQUEST, SINCE THE DEVICE DRIVER WILL OVERLAY THE
1658 * OVL CODE. AFTER THE OVL CODE EXITS, THE RESIDENT CODE WILL CALL
1659 * LDD TO PERFORM THE ACTUAL LOAD, OVER THE OVL.
1660 *
1661 * ENTRY DD.IOC = POINTER TO IOC.DDA
1662 * DD.LDA = LOAD ADDRESS
1663 * DD.LEN = LOAD LENGTH
1664 * DD.SEC = SECTOR INDEX ON SYSTEM DEVICE
1665 * DD.DTA = DEV.RES ADDRESS
1666 * DD.OPE = OPEN CODE (DC.OPR, DC.OPW, DC.OPU)
1667 * EXIT OVL CODE DESTROYED
1668 * USES NONE
1669
040.364 1670 S.DDSEC EQU S.DDGRP REFERENCE TO MAKE ASSEMBLE OK
1671
032.223 315 054 031 1672 LDD CALL $SAVALL SAVE REGS
1673
1674 * CLEAR OVL RESIDENT FLAG
1675
032.226 041 371 040 1676 LXI H,S.OVLFL
032.231 176 1677 MOV A,M
032.232 346 376 1678 ANI 377B-OVL.IN
032.234 167 1679 MOV M,A CLEAR IN FLAG
1680
1681 * LOAD OVERLAY
1682
032.235 052 362 040 1683 LHLD S.DDLEN (HL) = LENGTH
032.240 104 1684 MOV R,H
032.241 115 1685 MOV C,L (BC) = LENGTH
032.242 052 360 040 1686 LHLD S.DDLDA (HL) = LOAD ADDRESS
032.245 345 1687 PUSH H SAVE FOR LATER
032.246 353 1688 XCHG
032.247 041 377 026 1689 LXI H,SECSR+255 FORCE NEW DISK READ RIGHT AWAY
1690
1691 * LOAD BINARY
1692
032.252 315 362 032 1693 LDD2 CALL LDD8 FIND NEXT BYTE
032.255 176 1694 MOV A,M (A) = NEXT BYTE
032.256 022 1695 STAX D COPY
032.257 023 1696 INX D
032.260 013 1697 DCX B
032.261 170 1698 MOV A,B
032.262 261 1699 ORA C
032.263 302 252 032 1700 JNZ LDD2 MORE TO GO
1701
1702 * CODE ALL LOADED. RELOCATE IT
1703
032.266 301 1704 POP B (BC) = REL. FACTOR
032.267 005 1705 DCR B
032.270 005 1706 DCR B
HP 376.000 1707 ERRM7 DD.ENT-2000A ASSUME DRIVER ENTRY = 2000A
```

```

032.271 315 362 032 1708 LDD3 CALL LDD8
032.274 136 1709 MOV E,M
032.275 315 362 032 1710 CALL LDD8
032.300 126 1711 MOV D,M (DE) = REL ADDRESS OF WORD TO RELOCATE
032.301 172 1712 MOV A,D
032.302 263 1713 ORA E
032.303 312 323 032 1714 JZ LDD4 ALL DONE
032.306 353 1715 XCHG (HL) = REL ADDRESS OF WORD TO RELOCATE
032.307 011 1716 DAD B (HL) = ABS ADDRESS OF WORD TO RELOCATE
032.310 176 1717 MOV A,M
032.311 201 1718 ADD C
032.312 167 1719 MOV M,A
032.313 043 1720 INX H
032.314 176 1721 MOV A,M
032.315 210 1722 ADC B
032.316 167 1723 MOV M,A
032.317 353 1724 XCHG RESTORE (HL)
032.320 303 271 032 1725 JMP LDD3
1726

```

1727 * SETUP ENTRY ADDRESSES IN TABLES

```

1728
032.323 052 360 040 1729 LDD4 LHLD S,DDLDA (DE) = ENTRY ADDRESS
032.326 353 1730 XCHG (HL) = ADDRESS OF DEVLST ENTRY
032.327 052 366 040 1731 LHLD S,DDDTA
032.332 176 1732 MOV A,M
032.333 366 001 1733 ORI DR,IM SET IN MEMORY
032.335 167 1734 MOV M,A
032.336 043 1735 INX H
032.337 043 1736 INX H
000.000 1737 ERRNZ DEV.DDA-DEV.RES-2
032.340 163 1738 MOV M,E
032.341 043 1739 INX H
032.342 162 1740 MOV M,D SET ADDRESS IN TABLE
032.343 353 1741 XCHG (HL) = ENTRY POINT ADDRESS
032.344 257 1742 XRA A
032.345 062 361 040 1743 STA S,DDLDA+1 CLEAR LOAD FLAG
032.350 072 370 040 1744 LDA S,DDOPC (A) = OPEN CODE
032.353 315 361 032 1745 CALL FCHL CALL CODE
032.356 303 047 031 1746 JMP $RSTALL RESTORE REGISTERS
1747

```

```

032.361 351 1748 FCHL FCHL
1749

```

```

1751 ** LDD8 - READ A BYTE FROM THE FILE.
1752 *
1753 * ENTRY (HL) = SECSR POINTER OF CURRENT BYTE
1754 * S,DDDESC = SECTOR NUMBER OF NEXT SECTOR
1755 * EXIT (HL) = ADDRESS OF NEXT BYTE
1756 * USES L
1757
1758
032.362 054 1759 LDD8 INR L POINT TO NEXT BYTE
032.363 300 1760 RNZ GOT IT
1761
1762 * MUST READ ANOTHER

```

			1763					
032.364	305		1764	PUSH	B			
032.365	325		1765	PUSH	D			
032.366	345		1766	PUSH	H			
032.367	353		1767	XCHG		(DE) = ADDRESS		
032.370	001 000 001		1768	LXI	B,256			
032.373	052 364 040		1769	LHLD	S.DDSEC	(HL) = SECTOR NUMBER TO READ		
032.376	043		1770	INX	H			
032.377	042 364 040		1771	SHLD	S.DDSEC	(HL) = NEXT SECTOR NUMBER TO READ		
033.002	053		1772	DCX	H	RESTORE (HL)		
033.003	315 275 031		1773	CALL	SREAD	READ IT		
033.006	341		1774	POP	H			
033.007	321		1775	POP	D			
033.010	301		1776	POP	B			
033.011	311		1777	RET				

1779 ** LDD - LOAD OVL CODE.
1780 *
1781 * LDD IS CALLED WHEN THE OVL OVERLAY MUST BE LOADED.
1782 *
1783 * IF USER HIGH MEM IS TOO HIGH, PART OF THE USER CODE WILL
1784 * HAVE TO BE SAVED ON THE SWAP AREA, BEFORE THE OVL CODE CAN BE
1785 * LOADED.
1786 *
1787 * ENTRY NONE
1788 * EXIT NONE
1789 * USES A,F,H,L

			1790					
			1791					
033.012	325		1792	LDD	PUSH D			
033.013	305		1793	PUSH	B			
			1794					
			1795	*	SEE IF WILL HAVE TO PAGE USER CODE			
			1796					
033.014	052 376 040		1797	LHLD	S.OVLS	(HL) = SIZE OF HDOSOVL		
033.017	315 224 030		1798	CALL	%CHL	COMPLEMENT (HL)		
033.022	353		1799	XCHG		(DE) = -SIZE		
033.023	052 320 040		1800	LHLD	S.SYSM	(HL) = CURRENT FWA		
033.026	031		1801	DAD	D	(HL) = NEW FWA WITH OVL		
033.027	042 372 040		1802	SHLD	S.UCSF	SET USER SWAP (IN CASE IT IS SWAPPED)		
033.032	353		1803	XCHG				
033.033	052 322 040		1804	LHLD	S.USRM			
033.036	175		1805	MOV	A,L			
033.037	223		1806	SUB	E			
033.040	157		1807	MOV	L,A			
033.041	174		1808	MOV	A,H			
033.042	232		1809	SBB	D			
033.043	147		1810	MOV	H,A	(HL) = AMOUNT TO SWAP		
033.044	332 073 033		1811	JC	LDD1	NO NEED TO SWAP		
			1812					
			1813	*	MUST DUMP (HL) BYTES OF USER CODE STARTING AT (DE)			
			1814					
033.047	325		1815	PUSH	D	SAVE ADDRESS		

```

033.050 042 374 040 1816 SHLD S,UCSL SET LENGTH OF DUMP
033.053 104 1817 MOV B,H
033.054 115 1818 MOV C,L (BC) = COUNT
033.055 052 002 041 1819 LHLD S,SSN (HL) = SECTOR FOR SWAP (SET BY ROOT)
033.060 315 330 031 1820 CALL SWRITE
033.063 041 371 040 1821 LXI H,S,OVLFL
033.066 076 200 1822 MVI A,OVL,UCS
033.070 266 1823 ORA M SET USER CODE SWAPPED
033.071 167 1824 MOV M,A
033.072 321 1825 POP D (DE) = ADDRESS TO LOAD
1826
1827 * AM READY TO LOAD OVL OVERLAY,
1828 *
1829 * (DE) = ADDRESS
1830
033.073 052 376 040 1831 LD01 LHLD S,OVL5
033.076 104 1832 MOV B,H
033.077 115 1833 MOV C,L (BC) = SIZE OF OVERLAY
033.100 052 004 041 1834 LHLD S,OSN
033.103 315 275 031 1835 CALL SREAD READ IT
033.106 041 371 040 1836 LXI H,S,OVLFL
033.111 176 1837 MOV A,M
033.112 366 001 1838 ORI OVL,IN SET IT IN
033.114 167 1839 MOV M,A
1840
1841 * RELOCATE OVL
1842
033.115 052 372 040 1843 LHLD S,UCSF (HL) = FWA OVERLAY LOAD
033.120 021 006 000 1844 LXI D,PIC,COD
033.123 104 1845 MOV B,H
033.124 115 1846 MOV C,L (BC) = OVL FWA
033.125 031 1847 DAD D (HL) = ADDRESS OF ENTRY POINT
033.126 042 000 041 1848 SHLD S,OVLE SET ENTRY POINT
000.000 1849 ERNZ PIC,PTR-PIC,COD+2
033.131 053 1850 DCX H
033.132 176 1851 MOV A,M
033.133 053 1852 DCX H
033.134 156 1853 MOV L,M
033.135 147 1854 MOV H,A (HL) = REL ADDRESS OF TABLE
033.136 011 1855 DAD R (HL) = ABS ADDRESS OF TABLE
033.137 315 175 033 1856 CALL REL, RELOCATE OVL
1857
033.142 301 1858 POP B
033.143 321 1859 POP D
033.144 311 1860 RET
  
```

```

1862 ** FDI = PREPARE FOR DEVICE I/O
1863 *
1864 * FDI PREPARES FOR PHYSICAL I/O BY
1865 *
1866 * 1) COMPUTING THE PHYSICAL ADDRESS
1867 * 2) PREPARE THE COUNT
1868 *
  
```

PDI

```

1869 * ENTRY AIO,XXX SETUP
1870 * EXIT (BC) = COUNT
1871 * (HL) = SECTOR
1872 * (A) = 0
1873 * USES A,F,B,C,H,L
1874
1875
033.145 052.114 041 1876 PDI LHLD AIO,TFP (L) = AIO,CGN, (H) = AIO,CSI
000.000 1877 ERRNZ AIO,CSI-AIO,CGN-1
033.150 072.046 041 1878 LDA AIO,SPG (A) = SECTORS PER GROUP
033.153 117 1879 MOV C,A
033.154 175 1880 MOV A,L (A) = GROUP NUMBER
033.155 154 1881 MOV L,H
033.156 046.000 1882 MVI H,0 (HL) = (0,CSI)
033.160 104 1883 MOV B,H (BC) = (0,SPG)
1884
1885 * COMPUTE SECTOR NUMBER BY ADDING SPG 'BLOCK NUMBER' TIMES.
1886
033.161 011 1887 PDI1 DAD B ADD
033.162 075 1888 DCR A
033.163 302 161 033 1889 JNZ PDI1 MORE TO GO
033.166 072 111 041 1890 LDA AIO,CNT (A) = COUNT
033.171 110 1891 MOV C,B (C) = 0
033.172 107 1892 MOV B,A (B) = SECTOR COUNT
033.173 257 1893 XRA A CLEAR A
033.174 311 1894 RET

1896 ** REL - RELOCATE CODE.
1897 *
1898 * REL PROCESSES A RELOCATION LIST.
1899 *
1900 * ENTRY (BC) = DISPLACEMENT FROM ASSEMBLED ADDRESS
1901 * (DE) = RELOCATION FACTOR (FROM CURRENT ADDRESS)
1902 * (HL) = FWA RELOCATION LIST
1903 * EXIT NONE
1904 * USES ALL
1905
1906
033.175 120 1907 REL MOV D,B ENTRY FOR CODE DISPLACE = REL FACTOR
033.176 131 1908 MOV E,C
1909
033.177 325 1910 REL PUSH D SAVE RELOCATION FACTOR
033.200 136 1911 MOV E,M
033.201 043 1912 INX H
033.202 126 1913 MOV D,M
033.203 043 1914 INX H (DE) = REL ADDRESS OF WORD TO RELOCATE
033.204 172 1915 MOV A,D
033.205 263 1916 ORA E
033.206 302 213 033 1917 JNZ REL1 MORE TO TO
033.211 321 1918 POP D
033.212 311 1919 RET EXIT
1920
1921 * (DE) = INDEX OF WORD TO RELOCATE
  
```

				1922	*		(HL) = RELOCATION TABLE ADDRESS	
				1923	*		(BC) = CODE DISPLACEMENT FACTOR	
				1924	*		((SP)) = CODE RELOCATION FACTOR	
				1925				
				1926	RELY	XCHG		
033.213	353			1927		DAD	B	(HL) = ABS ADDR OF WORD TO REL
033.214	011			1928		XCHG		(DE) = ABS CODE ADDRESS; (HL) = REL TABLE ADDR
033.215	353			1929		XTHL		(HL) = CODE REL FACTOR
033.216	343			1930		LDAX	D	
033.217	032			1931		ADD	L	RELOCATE WORD OF CODE
033.220	205			1932		STAX	D	
033.221	022			1933		INX	D	
033.222	023			1934		LDAX	D	
033.223	032			1935		ADC	H	
033.224	214			1936		STAX	D	RELOCATE
033.225	022			1937		XCHG		(DE) = RELOCATION FACTOR
033.226	353			1938		POP	H	(HL) = RELOCATION TABLE ENTRY ADDRESS
033.227	341			1939		JMP	REL	DO IT AGAIN
				1941	**		TFE - TEST FOR EOF.	
				1942	*			
				1943	*		TFE CHECKS FOR AND END OF FILE, INDICATED BY	
				1944	*		AIO.CGN = AIO.LGN	
				1945	*		AIO.CSI = AIO.LSI	
				1946	*			
				1947	*	ENTRY	NONE	
				1948	*	EXIT	'Z' CLEAR IF NOT EOF	
				1949	*		(A) = 0	
				1950	*		'Z' SET IF EOF	
				1951	*		'C' SET	
				1952	*		(A) = EC.EOF	
				1953	*	USES	A;F;H;L	
				1954				
				1955				
033.233	052	051	041	1956	TFE	LHLD	AIO.LGN	
000.000				1957		ERRNZ	AIO.LSI-AIO.LGN-1	
033.236	072	047	041	1958		LDA	AIO.CGN	
033.241	275			1959		CMP	L	
033.242	076	000		1960		MVI	A,0	
033.244	300			1961		RNE		NOT EOF
033.245	072	050	041	1962		LDA	AIO.CSI	
033.250	274			1963		CMP	H	
033.251	076	000		1964		MVI	A,0	
033.253	300			1965		RNE		NOT EOF
033.254	076	003		1966		MVI	A,EC.EOF*2+1	SET EOF CODE
033.256	311			1967		RET		

				1969	**	RUC - RESTORE USER CODE.		
				1970	*			
				1971	*	RUC RESTORES THE USER PROGRAM CODE WHICH WAS SWAPPED		
				1972	*	FOR THE OVL CODE.		
				1973	*			
				1974	*	SINCE RUC RESIDES IN THE OVL AREA, IT MAY NOT RETURN AFTER THE		
				1975	*	DISK I/O CALL.		
				1976	*			
				1977	*	ENTRY	NONE	
				1978	*	EXIT	NONE	
				1979	*	USES	NONE	
				1980				
				1981				
033.257	315	054	031	1982	RUC	CALL	\$SAVALL	SAVE REGISTERS
033.262	041	047	031	1983		LXI	H,\$RSTALL	
033.265	345			1984		PUSH	H	RETURN VIA \$RSTALL
033.266	041	371	040	1985		LXI	H,\$OVLFL	
033.271	176			1986		MOV	A,M	
033.272	247			1987		ANA	A	
000.000				1988		ERRNZ	OVL,UCS-2000	
033.273	360			1989		RF		NOT SWAPPED
033.274	346	176		1990		ANI	3770-OVL,UCS-OVL,IN	RESTORE USER CODE, REMOVE OVL
033.276	167			1991		MOV	M,A	
				1992				
				1993	*	RESTORE USER CODE		
				1994				
033.277	052	374	040	1995		LHLD	S,UCSL	
033.302	104			1996		MOV	B,H	
033.303	115			1997		MOV	C,L	(BC) = COUNT
033.304	052	372	040	1998		LHLD	S,UCSF	
033.307	353			1999		XCHG		(DE) = ADDRESS
033.310	052	002	041	2000		LHLD	S,SSN	(HL) = SECTOR FOR SWAP
033.313	303	275	031	2001		JMP	SREAD	READ AND EXIT


```

2004 *** SYDD - SYSTEM DISK DEVICE DRIVER.
2005 *
2006 * SYDD IS THE HDOS SYSTEM H17 DEVICE DRIVER.
2007 *
2008 * ENTRY (A) = DC,XXX FUNCTION CODE
2009 * OTHER REGISTERS SET AS NEEDED BY FUNCTION
2010 * EXIT 'C' CLEAR, OK
2011 * REGISTERS SET BY FUNCTION
2012 * 'C' SET, ERRDR
2013 * (A) = ERROR CODE
2014 * USES ALL
2015
2016
033.314 2017 R.SYDD EQU *
000.000 2018 ERRNZ DC,REA
033.316 247 2019 ANA A
033.317 312 147 040 2020 JZ D,READ
000.000 2021 ERRNZ DC,WRI-1
033.322 075 2022 DCR A
033.323 312 155 040 2023 JZ D,WRITE
000.000 2024 ERRNZ DC,RER-2
033.326 075 2025 DCR A
033.327 312 152 040 2026 JZ D,READR READ REGARDLESS
033.332 376 005 2027 CPI DC,ABT-2
033.334 332 136 040 2028 JC D,XOK IS NOT ABORT OK MOUNT, IGNORE
000.000 2029 ERRNZ DC,MOU-DC,ABT-1
033.337 312 141 040 2030 JE D,ABORT IS ABORT
033.342 303 133 040 2031 JMP D,MOUNT
  
```

```

2033 *** MOUNT - MOUNT NEW DEVICE.
2034 *
2035 * MOUNT PROCESSES DEVICE DEPENDANT MOUNTING OF A NEW MEDIA.
2036 *
2037 * THE VOLUME SERIAL IS READ INTO THE VOLUME TABLE, AND
2038 * THE HEADS ARE HOMED.
2039 *
2040 * ENTRY (L) = VOLUME NUMBER (IF ANY)
2041
2042
033.345 2043 R.MOUNT EQU *
033.345 105 2044 MOV B,L (B) = VOLUME SERIAL
033.346 041 000 000 2045 LXI H,0 SET SECTOR INDEX
033.351 315 205 040 2046 CALL D,SDP SET DEVICE PARAMETERS
033.354 315 213 040 2047 CALL D,STZ SEEK TRACK ZERO
033.357 052 247 040 2048 LHLD D,VOLFT
033.362 160 2049 MOV M,B SET VOLUME NUMBER
033.363 303 136 040 2050 JMP D,XOK EXIT WITH STUFF OK
  
```

```

2053 *** ABORT - ABORT ANY ACTIVE I/O.
2054 *
2055 * ABORT CAUSES ANY ON/-GOING I/O TO BE ABORTED.
2056
2057
033,366 2058 R.ABORT EQU *
033,366 315 205 040 2059 CALL D.SDF SET DEVICE PARAMETERS
033,371 315 213 040 2060 CALL D.STZ SEEK TRACK ZERO
033,374 2061 SET R.XOK IMPLICIT REFERENCE TO R:XOK
2062 * JMP D.XOK EXIT AS IF OK
  
```

```

2064 ** XOK - EXIT WITH ALL OK FLAG.
2065
033,374 257 2066 R.XOK XRA A
033,375 365 2067 R.XIT PUSH PSW SAVE STATUS
033,376 072 244 040 2068 XITI LDA D.DLYHS
034,001 247 2069 ANA A
034,002 302 376 033 2070 JNZ XITI WAIT FOR HARDWARE DELAYS
034,005 363 2071 DI LOCK OUT CLOCK
034,006 072 242 040 2072 LDA D.DVCTL
034,011 346 220 2073 ANI DF.MO+DF.WR REMOVE DEVICE SELECT
034,013 323 177 2074 OUT DF.DC DESELECT MOTOR
034,015 062 242 040 2075 STA D.DVCTL UPDATE BYTE
034,020 052 110 040 2076 LHL D.XITA
034,023 042 243 040 2077 SHLD D.DLYMO SET 120/2 SECONDS OF MOTOR ON
000,000 2078 ERNZ D.DLYHS-D.DLYMO-1 SET 7*2 MILLISECONDS OF HEAD UNSETTLE
034,026 361 2079 POP PSW
034,027 373 2080 EIXIT EI RESTORE INTERRUPTS
034,030 311 2081 RET
  
```

```

2083 ** CLOCK - PROCESS CLOCK INTERRUPTS.
2084 *
2085
034,031 072 033 040 2087 CLOCK LDA .TICCNT
034,034 017 2088 RRC
034,035 330 2089 RC NOT EVEN
034,036 247 2090 ANA A
034,037 041 243 040 2091 LXI H,D.DLYMO
034,042 302 070 034 2092 JNZ CLOCK1 NOT HALF SECOND
034,045 075 2093 DCR A (A) = -1
034,046 206 2094 ADD H SUBTRACT ONE
034,047 322 070 034 2095 JNC CLOCK1 WAS 0
034,052 167 2096 MOV M,A UPDATE
034,053 302 070 034 2097 JNZ CLOCK1 NOT TIME FOR MOTOR OFF
034,056 072 242 040 2098 LDA D.DVCTL
034,061 346 200 2099 ANI DF.WR REMOVE ALL BUT RAM/WRITE
034,063 062 242 040 2100 STA D.DVCTL
034,066 323 177 2101 OUT DF.DC OFF MOTOR
034,070 043 2102 CLOCK1 INX H (HL) = #DLYHS
  
```

ABORT - ABORT ACTIVE I/O

CLOCK

10:00:02 02-APR-80

000.000		2103	ERRNZ	D.DLYHS-D.DLYHO-I	
034.071	176	2104	MOV	A,M	(A) = D.DLYHS
034.072	326 001	2105	SUI	1	
034.074	330	2106	RC		WAS 0
034.075	167	2107	MOV	N,A	
034.076	311	2108	RET		

```

2112 *** READ - READ FROM DISK.
2113 *
2114 * ENTRY (BC) = COUNT
2115 * (DE) = ADDRESS
2116 * (HL) = BLOCK #
2117 * INTERRUPTS ENABLED
2118 * EXIT (DE) = NEXT UNUSED ADDRESS
2119 * INTERRUPTS DISABLED
2120 * USES ALL
2121
2122
034.077 345 2123 R,READ PUSH H SAVE (HL)
034.100 315 205 040 2124 CALL D.SDP SETUP DEVICE PARAMETERS
034.103 052 273 040 2125 LHL D.OPR
034.106 043 2126 INX H
034.107 042 273 040 2127 SHLD D.OPR COUNT OPERATION
2128
2129 * READ TO READ SECTOR.
2130 *
2131 * (BC) = AMOUNT
2132 * (DE) = ADDRESS
2133 * ((SF)) = SECTOR NUMBER
2134
034.112 341 2135 READ1 POP H (HL) = SECTOR NUMBER
034.113 325 2136 PUSH D SAVE ADDR
034.114 171 2137 MOV A,C ADJUST (B) SO THAT (B) = # OF WHOLE OR PARTIAL
034.115 247 2138 ANA A SECTORS TO READ, (C) = BYTES OF LAST SECTOR TO
034.116 312 122 034 2139 JZ READ1.5 READ, (C)=0 IF TO READ ENTIRE LAST SECTOR
034.121 004 2140 INR B
2141
2142 * * * NOTE * *
2143 *
2144 * THIS CODE RUNS WITH INTERRUPTS DISABLED FROM HERE ON
2145
034.122 305 2146 READ1.5 PUSH B SAVE COUNT
034.123 315 163 040 2147 CALL D.DTS DECODE TRACK AND SECTOR
034.126 076 001 2148 READ2 MVI A,1 (A) = DELAY COUNT FOR START
2149
2150 * LOOK FOR RIGHT SECTOR.
2151 * (A) = DELAY COUNT BEFORE SEARCH
2152
034.130 315 216 040 2153 READ2.4 CALL D.UDLY DELAY SOME MICROSECONDS
034.133 315 177 040 2154 CALL D.LFS LOCATE PROPER SECTOR
034.136 332 300 034 2155 JC READ7 ERROR
034.141 301 2156 POP B (BC) = COUNT
034.142 341 2157 POP H (HL) = ADDRESS FOR DATA
2158
2159 * CHECK AMOUNT TO READ
2160
034.143 170 2161 READ3 MOV A,B
034.144 261 2162 ORA C
034.145 312 315 034 2163 JZ READ8 NO MORE TO READ
034.150 345 2164 PUSH H
034.151 305 2165 PUSH B SAVE COUNT AND ADDRESS IN CASE OF ERROR
034.152 005 2166 DCR B SEE IF ON LAST (MAYBE PARTIAL) SECTOR
034.153 312 160 034 2167 JZ READ3.5 ON LAST SECTOR, READ (C) COUNT
  
```

READ - READ FROM DISK

READ

10:00:03 02-APR-89

```

034.156 016 000 2168 MVI C,0 WILL READ ALL 256 BYTES
034.160 101 2169 READ3,5 MOV B,C (B) = # TO READ+1, (C) = # TO SKIP
034.161 315 221 040 2170 CALL D,WSC WAIT SYNC CHARACTER
034.164 332 261 034 2171 JC READ71 DIDNT GET ONE
2172
2173 * READ DATA
2174
034.167 315 202 040 2175 READ4 CALL D,RDB READ BYTE
034.172 167 2176 MOV M,A STORE
034.173 043 2177 INX H
034.174 005 2178 DCR B
034.175 302 167 034 2179 JNZ READ4 MORE TO GO
034.200 171 2180 MOV A,C
034.201 247 2181 ANA A
034.202 312 214 034 2182 JZ READ6 NONE TO DISCARD
2183
2184 * READ, CHECKSUM, AND DISCARD DATA
2185
034.205 315 202 040 2186 READ5 CALL D,RDB
034.210 014 2187 INR C
034.211 302 205 034 2188 JNZ READ5
034.214 102 2189 READ6 MOV B,D (B) = CHECKSUM
034.215 315 202 040 2190 CALL D,RDB
034.220 270 2191 CMP B
034.221 302 272 034 2192 JNE READ72 CHECKSUM ERROR
2193
2194 * GOT GOOD SECTOR
2195
034.224 301 2196 POP B (BC) = OLD COUNT
034.225 005 2197 DCR B COUNT SECTOR READ
034.226 312 315 034 2198 JZ READ8 JUST READ LAST ONE
2199
2200 * HAVE MORE TO READ
2201
034.231 343 2202 XTHL SAVE ADDRESS
034.232 305 2203 PUSH B SAVE COUNT
034.233 041 241 040 2204 LXI H,D,TS
034.236 064 2205 INR M COUNT SECTOR
034.237 076 012 2206 MVI A,10
034.241 226 2207 SUB M
034.242 076 000 2208 MVI A,0
000.000 2209 ERNZ 30*44*2/15-1000A (A) = TIME TO DELAY 30 CHARACTERS
034.244 302 130 034 2210 JNE READ2.4 NOT AT END OF TRACK
034.247 167 2211 MOV M,A SECTOR # = 0
000.000 2212 ERKNZ D,TS-D,TT-1
034.250 053 2213 DCX H
034.251 064 2214 INR M
034.252 373 2215 EI RESTORE INTERRUPTS UNTIL *STS* CALLED
034.253 315 166 040 2216 CALL D,SIT SEEK DESIRED TRACK
034.256 303 126 034 2217 JMP READ2
2218
2219 * CAN'T GET DATA, HEADER OR CHECKSUM PROBLEM
2220
034.261 041 265 040 2221 READ71 LXI H,D,E,MBS MISSING DATA SYNC ERROR
034.264 315 232 040 2222 CALL D,ERRT
034.267 303 300 034 2223 JMP READ7

```

```
2224
034.272 041 267 040 2225 READ72 LXI H,D.E.CHK CHECKSUM ERROR
034.275 315 232 040 2226 CALL D.ERRT
2227
034.300 315 160 040 2228 READ7 CALL D.CDE COUNT DISK ERROR
034.303 322 126 034 2229 JNC READ2 TRY AGAIN
034.306 301 2230 POP B
034.307 321 2231 POP D
034.310 076 022 2232 MVI A,EC.RF READ FAILURE
034.312 303 144 040 2233 JMP D.XIT TOO MANY ERRORS, TOO BAD...
2234
2235 * ENTIRE READ WAS OK
2236
034.315 341 2237 READ8 POP H CLEAN STACK
034.316 303 136 040 2238 JMP D.XOK EXIT OK
```

```
2241 *** READR - READ DISK REGARDLESS OF VOLUME PROTECTION.  
2242 *  
2243 * ENTRY (BC) = COUNT  
2244 * (DE) = ADDRESS  
2245 * (HL) = BLOCK #  
2246 * EXIT (DE) = NEXT UNUSED ADDRESS  
2247 * USES ALL  
2248  
2249  
034.321 345 2250 R.READR PUSH H SAVE (HL)  
034.322 315 205 040 2251 CALL D.SDF SETUP DEVICE PARAMETERS  
034.325 041 320 031 2252 LXI H,ZEROS  
034.330 042 247 040 2253 SHLD D.VOLPT  
034.333 303 112 034 2254 JMP READ1 PROCESS AS REGULAR READ
```

WRITE - PROCESS DISK WRITE.

WRITE

10:00:05 02-APR-80

```

2258 *** WRITE - PROCESS DISK WRITE.
2259 *
2260 * ENTRY (BC) = COUNT
2261 * (DE) = ADDRESS
2262 * (HL) = BLOCK #
2263 * EXIT (LINK) = LAST BLOCK #
2264 * USES ALL
2265
2266
034.336 2267 R.WRITE EQU *
034.336 345 2268 PUSH H SAVE BLOCK NUMBER
034.337 315 205 040 2269 CALL D,SDP SET DEVICE PARAMETERS
034.342 052 275 040 2270 LHLD D,OPW
034.345 043 2271 INX H
034.346 042 275 040 2272 SHLD D,OPW COUNT OPERATION
034.351 333 177 2273 IN DP,DC SEE IF DISK WRITE PROTECTED
034.353 346 004 2274 ANI DP,WP
034.355 067 2275 STC
034.356 076 025 2276 MVI A,EC,WP
034.360 302 132 035 2277 JNZ WRITE0 DISK IS WRITE PROTECTED
2278
2279 * READY TO WRITE SECTOR
2280 *
2281 * (BC) = COUNT
2282 * (DE) = ADDRESS
2283 * ((SP)) = SECTOR NUMBER
2284
034.363 041 377 000 2285 LXI H,3770
034.366 011 2286 DAD B
034.367 104 2287 MOV B,H (B) = # OF SECTORS TO WRITE
2288
034.370 341 2289 WRITE1 POP H (HL) = SECTOR NUMBER
034.371 325 2290 PUSH D SAVE ADDR
2291
2292 * * * NOTE * *
2293 *
2294 * THIS CODE RUNS WITH INTERRUPTS DISABLED FROM THIS POINT ON
2295
034.372 315 163 040 2296 CALL D,DTS DETERMINE TRACK AND SECTOR
034.375 076 001 2297 WRITE2 MVI A,1 (A) = SHORT DELAY COUNT
2298
2299 * FIND RIGHT SECTOR
2300 * (A) = DELAY COUNT
2301
034.377 315 216 040 2302 WRIT2.5 CALL D,UDLY DELAY SOME MICROSECONDS
035.002 305 2303 PUSH B SAVE COUNT
035.003 315 177 040 2304 CALL D,LFS LOCATE PROPER SECTOR
035.006 301 2305 POP B (BC) = COUNT
035.007 332 122 035 2306 JC WRITE7 CANT FIND IT
035.012 341 2307 POP H (HL) = ADDR
035.013 072 112 040 2308 LDA D,WRIT4 (A) = GUARDBAND DELAY
035.016 075 2309 WRITE4 DCR A
035.017 302 016 035 2310 JNZ WRITE4 FAUSE OVER GUARDBAND
035.022 072 113 040 2311 LDA D,WRIT4
035.025 117 2312 MOV C,A (C) = # OF 00 CHARACTERS
035.026 072 114 040 2313 LDA D,WRITC (A) = 128/8 = TWO CHARACTER TIMES BEFORE WRITING

```


WRITE

```

035.031 315 224 040 2314 CALL D,WSP WRITE SYNC PATTERN
                2315
                2316 * OUT WITH THE DATA
                2317
035.034 176 2318 WRITES MOV A,H
035.035 315 227 040 2319 CALL D,WNB
035.040 043 2320 INX H
035.041 015 2321 DCR C
035.042 302 034 035 2322 JNZ WRITES NOT DONE YET
035.045 172 2323 MOV A,D (A) = CHECKSUM
035.046 315 227 040 2324 CALL D,WNB WRITE CHECKSUM
                2325
                2326 * HAVE DONE WRITING. LEAVE WRITE-GATE OPEN FOR 3 CHARACTER TIMES
                2327 * TO FINISH TUNNEL ERASING.
                2328
035.051 315 227 040 2329 CALL D,WNB
035.054 315 227 040 2330 CALL D,WNB
035.057 315 227 040 2331 CALL D,WNB
035.062 072 242 040 2332 LDA D,IVCTL
035.065 323 177 2333 OUT BF,DC OFF DISK CONTROL
035.067 005 2334 DCR B
035.070 312 136 040 2335 JZ D,XOK ALL DONE
035.073 345 2336 PUSH H SAVE ADDRESS
035.074 041 241 040 2337 LXI H,D,TS
035.077 064 2338 INR M
035.100 076 012 2339 MVI A,10
035.102 226 2340 SUB H
035.103 076 000 2341 MVI A,0
000.000 2342 ERRNZ 30*64*2/15-1000A (A) = COUNT TO DELAY 30 CHARACTER TIMES
035.105 302 377 034 2343 JNZ WRIT2,5 NOT AT END OF TRACK
                2344
                2345 * MOVE TO NEXT TRACK
                2346
000.000 2347 ERRNZ D,TS-D,TT-1
035.110 167 2348 MOV M,A CLEAR CURRENT SECTOR INDEX
035.111 053 2349 DCX H
035.112 064 2350 INR M
035.113 373 2351 EI RESTORE INTERRUPTS UNTIL *STS* CALL
035.114 315 166 040 2352 CALL D,SDT SEEK DESIRED TRACK
035.117 303 375 034 2353 JMP WRITE2
                2354
                2355 * ERROR
                2356
035.122 315 160 040 2357 WRITE7 CALL D,CDE COUNT DISK ERROR
035.125 322 375 034 2358 JNC WRITE2 TRY AGAIN
035.130 076 023 2359 MVI A,EC,WF WRITE FAILURE
035.132 341 2360 WRITE8 POP H RESTORE STACK
035.133 303 144 040 2361 JMP D,XIT TOO MANY... TRY AGAIN

```

```
2365 ** CDE - COUNT DISK ERRORS.
2366 *
2367 * CDE IS CALLED WHEN A DISK SOFT ERROR OCCURS. IF THERE HAVE
2368 * OCCURED 10 SOFT ERRORS FOR THIS OPERATION, THEN A HARD ERROR
2369 * IS FLAGGED.
2370 *
2371 * ENTRY NONE
2372 * EXIT 'C' SET IF HARD ERROR
2373 * INTERRUPTS DISABLED
2374 * USES A,F,H,L
2375
2376
035.136 373 2377 R.CDE EI RESTORE INTERRUPTS
035.137 315 213 040 2378 CALL D,STZ SEEK TRACK ZERO
035.142 315 166 040 2379 CALL D,SDI SEEK DESIRED TRACK
035.145 247 2380 ANA A CLEAR CARRY
035.146 052 262 040 2381 LHLD D,SECNT
035.151 043 2382 INX H
035.152 042 262 040 2383 SHLD D,SECNT INCREMENT COUNT
035.155 041 264 040 2384 LXI H,D,DECNT (HL) = #OPERATION ERROR COUNT
035.160 065 2385 DCR M
035.161 360 2386 RP NOT TOO MANY
035.162 053 2387 DCX H
035.163 076 366 2388 MVI A,-ERPTCNT
035.165 206 2389 ADD M REMOVE SOFT COUNT
035.166 167 2390 MOV M,A
000.000 2391 ERRNZ D,SECNT-D,HECNT-1
035.167 064 2392 INR M COUNT HARD ERROR
035.170 067 2393 STC
035.171 311 2394 RET EXIT WITH 'C' SET

2396 ** DTS - DECODE TRACK AND SECTOR.
2397 *
2398 * DTS DECODES THE TRACK AND SECTOR NUMBER FROM
2399 * THE SUPPLIED SECTOR INDEX.
2400 *
2401 * ENTRY (HL) = SECTOR INDEX
2402 * INTERRUPTS ENABLED
2403 * EXIT D,TS = SECTOR NUMBER
2404 * D,TT = TRACK
2405 * INTERRUPTS DISABLED
2406 * USES A,F,H,L
2407
2408
035.173 305 2409 R.DTS PUSH R SAVE (BC)
035.173 001 366 377 2410 LXI R,-10
035.176 170 2411 MOV A,B (A) = 377R
035.177 074 2412 DTSI INR A
035.200 011 2413 DAD B
035.201 332 177 035 2414 JC DTSI
035.204 062 240 040 2415 STA D,TT SET TRACK NUMBER
035.207 175 2416 MOV A,L
035.210 306 012 2417 ADI 10
```

```
035.212 062 241 040 2418 STA D,TS SET SECTOR
035.215 301 2419 POP R RESTORE (BC)
035.216 303 225 035 2420 JMP R,SDT SEEK DESIRED TRACK
```

```
2422 ** SDT - SEEK DESIRED TRACK.
```

```
2423 *
```

```
2424 * SDT MOVES THE DISK ARM TO THE DESIRED (D,TT) TRACK.
```

```
2425 *
```

```
2426 * ENTRY NONE
```

```
2427 * EXIT NONE
```

```
2428 * USES A,F,H,L
```

```
2429
```

```
2430
```

```
2431 * MOVE ARM IN
```

```
2432
```

```
035.221 064 2433 SDT3 INR M
035.222 315 171 040 2434 CALL D,MAI
```

```
2435
```

```
2436
```

```
035.225 052 245 040 2437 R,SDT LHLD D,TRKPT
```

```
035.230 072 240 040 2438 LDA D,TT
```

```
035.233 276 2439 CMP M
```

```
035.234 312 210 040 2440 JE D,STS GOT THERE
```

```
035.237 362 221 035 2441 JP SDT3 MUST MOVE IN
```

```
2442
```

```
2443 * MOVE ARM OUT
```

```
2444
```

```
035.242 065 2445 SDT1 DCR M UPDATE TRACK NUMBER
```

```
035.243 315 174 040 2446 CALL D,MAO MOVE ARM OUT
```

```
035.246 303 225 035 2447 JMP R,SDT SEE IF THERE YET
```

```
2449 ** MAI - MOVE DISK ARM IN ONE TRACK.
```

```
2450 *
```

```
2451 * ENTRY NONE
```

```
2452 * EXIT NONE
```

```
2453 * USES A,F
```

```
2454
```

```
2455
```

```
2456 ** MAO - MOVE ARM OUT.
```

```
2457 *
```

```
2458 * USES A,F
```

```
2459
```

```
2460
```

```
035.251 076 040 2461 R,MAI MVI A,DF,DI SET DIRECTION
```

```
2462
```

```
035.253 376 2463 DB MI,CPI GORBLE XRA INSTRUCTION
```

```
035.254 257 2464 R,MAO XRA A SET DIRECTION
```

```
2465
```

```
035.255 345 2466 PUSH H
```

```
035.256 147 2467 MOV H,A
```

```

035.257 072 242 040 2468 LDA D,DVCTL
035.262 346 237 2469 ANI 377Q-DF,DI-DF,ST
035.264 264 2470 ORA H SET DIRECTION
035.265 323 177 2471 OUT DF,DC SET DIRECTION
035.267 341 2472 POP H RESTORE (HL)
035.270 366 100 2473 ORI DF,ST
035.272 323 177 2474 OUT DF,DC START STEP
035.274 356 100 2475 XRI DF,ST
035.276 323 177 2476 OUT DF,DC COMPLETE STEP
035.300 072 115 040 2477 LDA D,MAIA (A) = MS/2 FOR TRACK TIMING
035.303 2478 SET R,DLY SET REFERENCE TO ROM
2479 * JMP D,DLY DELAY 8 MS

```

```

2481 ** DLY - DELAY BY FRONT PANEL CLOCK.
2482 *
2483 * ENTRY (A) = MILLISECOND COUNT/2
2484 * EXIT NONE
2485 * USES A,F
2486
2487

```

```

035.303 345 2488 R,DLY PUSH H
035.304 041 033 040 2489 LXI H,TICCNT
035.307 206 2490 ADD M
035.310 276 2491 DLY1 CMP M
035.311 302 310 035 2492 JNE DLY1
035.314 341 2493 POP H
035.315 311 2494 RET

```

```

2496 ** LPS - LOCATE PROPER SECTOR.
2497 *
2498 * LPS READS OVER SECTOR HEADERS UNTIL THE PROPER SECTOR
2499 * IS FOUND.
2500 *
2501 * UPON ENTRY, THE ARM SHOULD BE POSITIONED OVER THE SECTOR.
2502 *
2503 * D,TT = DESIRED TRACK
2504 * D,TS = DESIRED SECTOR
2505 *
2506 * ENTRY NONE
2507 * EXIT INTERRUPTS DISABLED
2508 * 'C' SET IF ERROR
2509 * USES ALL BUT C
2510
2511

```

```

035.316 315 210 040 2512 LPS0 CALL D,STS SKIP THIS SECTOR
2513
035.321 072 116 040 2514 R,LPS LDA D,LPSA (A) = #OF TRYs FOR THIS SECTOR
035.324 107 2515 MOV R,A
035.325 072 244 040 2516 LDA D,DLYHS
035.330 247 2517 ANA A

```

```

035.331 302 316 035 2518      JNZ     LPS0      WAIT FOR HEADS TO SETTLE
                2519
035.334 363                2520 LPS1      DI         DISABLE INTERRUPTS
035.335 315 221 040 2521      CALL    D.WSC    WAIT SYNC CHARACTER
035.340 332 025 036 2522      JC      LPS3      NONE
035.343 052 247 040 2523      LHL    D.VOLFT
035.346 315 202 040 2524      CALL    D.RDB
035.351 276                2525      CMP     M        SEE IF PROPER VOLUME
035.352 302 032 036 2526      JNE    LPS4      WRONG VOLUME
035.355 041 240 040 2527      LXI    H,D.TT
035.360 315 202 040 2528      CALL    D.RDB
035.363 276                2529      CMP     M        SEE IF PROPER TRACK
035.364 302 037 036 2530      JNE    LPS5      WRONG TRACK
000.000                2531      ERNZ   D.TS-D.TT-1
035.367 043                2532      INX   H
035.370 315 202 040 2533      CALL    D.RDB
035.373 276                2534      CMP     M
035.374 302 014 036 2535      JNE    LPS2      WRONG SECTOR
                2536
                2537 *      GOT RIGHT SECTOR. READ CHECKSUM
                2538
035.377 142                2539      MOV    H,D
036.000 315 202 040 2540      CALL    D.RDB
036.003 274                2541      CMP     H
036.004 310                2542      RE
                2543      ALL OK
036.005 056 270                2543      MVI    L,#D.E.HCK  HEADER CHECKSUM ERROR
036.007 046 040 2544 LPS1.5 MVI    H,D.ERR/256  (HL) = ERROR BYTE ADDRESS
000.040                2545      SET   D,ERR/256
000.000                2546      ERNZ   D.ERR/256-
036.011 315 232 040 2547      CALL    D.ERRT    MUST BE IN SAME BANK
                2548      COUNT ERROR
                2549 *      WRONG SECTOR OR BAD DATA. TRY SOME MORE
                2550
036.014 315 210 040 2551 LPS2      CALL    D.STS    SKIP THIS SECTOR
036.017 005                2552      DCR   B
036.020 302 334 035 2553      JNZ    LPS1      TRY AGAIN
036.023 067                2554      STC
                2555      ENOUGH TRYS
036.024 311                2556      RET
                2557
036.025 056 266                2557 LPS3      MVI    L,#D.E.HSY  HEADER SYNC ERROR
036.027 303 007 036 2558      JMP    LPS1.5
                2559
036.032 056 271                2560 LPS4      MVI    L,#D.E.VOL  BAD VOLUME NUMBER
036.034 303 007 036 2561      JMP    LPS1.5    COUNT ERROR
                2562
036.037 056 272                2563 LPS5      MVI    L,#D.E.TRK  BAD TRACK NUMBER
036.041 303 007 036 2564      JMP    LPS1.5

```

```
2566 ** RDB - READ BYTE FROM DISK.
2567 *
2568 * RDB READS THE NEXT BYTE FROM THE DISK.
2569 *
2570 * ENTRY (D) = CHECKSUM
2571 * EXIT (A) = BYTE
2572 * (D) UPDATED
2573 * USES A,F,I,E
2574
2575
036.044 333 175 2576 R,RDB IN UP,ST
000.000 2577 ERRNZ UF,RDA-1
036.046 037 2578 RAR
036.047 322 044 036 2579 JNC R,RDB NOT READY YET
036.052 333 174 2580 IN UP,DF (A) = DATA
036.054 137 2581 MOV E,A
036.055 252 2582 XRA B DIFFER
036.056 007 2583 RLC SHIFT LEFT
036.057 127 2584 MOV D,A REPLACE
036.060 173 2585 MOV A,E (A) = CHAR
036.061 311 2586 RET EXIT

2588 ** SDF - SET DEVICE PARAMETERS.
2589 *
2590 * SDF SETS UP ARGUMENTS FOR THE SPECIFIC UNIT.
2591 *
2592 * D.DVCTL = MOTOR ON, DEVICE SELECT
2593 * D.TRKPT = ADDRESS OF DEVICE TRACK NUMBER
2594 *
2595 * ENTRY AIO.UNI = UNIT NUMBER
2596 * EXIT (HL) = (D.TRKPT)
2597 * USES A,F,H,L
2598
2599
036.062 076 012 2600 R,SDF MVI A,ERPTCNT
036.064 062 264 040 2601 STA B,DECNT SET MAX ERROR COUNT FOR OPERATION
036.067 072 061 041 2602 LDA AIO.UNI
036.072 365 2603 PUSH FSW SAVE UNIT NUMBER
036.073 074 2604 INR A (A) = 1 IF DEV 0, 2 IF DEV 1
036.074 207 2605 ADD A
000.000 2606 ERRNZ DF,DS0-2 SELECT 0 OR 1
000.000 2607 ERRNZ DF,DS1-4
036.075 363 2608 DI INTERLOCK CLOCK INTERRUPTS
036.076 041 242 040 2609 LXI H,B.DVCTL
036.101 256 2610 XRA M
036.102 346 177 2611 ANI 377Q-DF,WR
036.104 256 2612 XRA M MERGE WITH DF.WR BIT FROM B.DVCTL
036.105 366 020 2613 ORL DF,MD MOTOR ON
036.107 167 2614 MOV M,A UPDATE
036.110 323 177 2615 OUT DF,DC SELECT DRIVE, LOAD HEAD
2616
2617 * SEE IF HEADS HAVE BEEN UNLOADED LONG ENOUGH TO REQUIRE LOAD DELAY
2618
```

SDP

```

036.112 041 244 040 2619 LXI H,D,DLYHS
036.113 176 2620 MOV A,M (A) = FLAG SET BY XIT
036.114 247 2621 ANA A
036.117 066 000 2622 MVI M,0 ASSUME NO RE-LOAD
036.121 302 130 036 2623 JNZ SDF1 NO RE-LOAD
036.124 072 117 040 2624 LDA D,SDPA (A) = HEAD SETTLE WAIT TIME/4
036.127 167 2625 MOV M,A SET FOR 'CLOCK' TIMER
036.130 053 2626 SDF1 DCX H
000.000 2627 ERRNZ D,DLYMD-D,DLYHS+1 (HL) = #D,DLYMD
036.131 176 2628 MOV A,M (A) = MOTOR ON DELAY
036.132 066 170 2629 MVI M,120 60 SECONDS BEFORE TURN OFF AGAIN
036.134 247 2630 ANA A 'Z' IF MOTOR TURNED OFF
036.135 043 2631 INX H (HL) = #D,DLYHS
036.136 302 145 036 2632 JNZ SDF2 MOTOR IS STILL ON
036.141 072 120 040 2633 LDA D,SDPB (A) = MOTOR WAIT TIME (MS/4)
036.144 167 2634 MOV M,A
036.145 373 2635 SDF2 EI RESTORE INTERRUPTS
036.146 361 2636 POP PSW (A) = UNIT NUMBER
036.147 207 2637 ADD A (A) = 2*UNIT NUMBER
036.150 041 251 040 2638 LXI H,D,DRVTB
036.153 205 2639 ADD L
036.154 157 2640 MOV L,A (HL) = ADDRESS OF TRACK ENTRY
036.155 042 245 040 2641 SHLD B,TRKPT
036.160 043 2642 INX H
036.161 042 247 040 2643 SHLD B,VOLPT SET VOLUME NUMBER
036.164 311 2644 RET

```

```

2646 ** STS - SKIP THIS SECTOR.
2647 *
2648 * STS IS CALLED TO SKIP THE CURRENT SECTOR, REGARDLESS OF WHERE
2649 * THE HEAD IS POSITIONED.
2650 *
2651 * STS WILL EXIT AT THE BEGINNING OF THE NEXT SECTOR.
2652 *
2653 * 1. IF THE HEAD IS NOT OVER A HOLE, WAIT 8 MS WHILE
2654 * HOLE CHECKING. IF NO HOLE IN THIS TIME, WHEN WE ARE IN
2655 * A REGULAR GAP, WAIT FOR THE NEXT HOLE AND EXIT.
2656 *
2657 * 2. IF THE HEAD IS OVER A HOLE, OR BECOMES SO DURING THE 8 MS
2658 * WAIT, THEN WAIT FOR THE HOLE TO PASS. WAIT 12 MILLISECONDS
2659 * IN CASE OF THE INDEX HOLE, THEN WAIT FOR THE NEXT HOLE AND EXIT.
2660 *
2661 * ENTRY NONE
2662 * EXIT INTERRUPTS DISABLED
2663 * USES A,F,H,L
2664
2665

```

```

036.165 373 2666 R.STS EI
036.166 305 2667 PUSH B SAVE (BC)
036.167 333 177 2668 IN DP,DC
000.000 2669 ERRNZ DF,HD-1
036.171 037 2670 RAR
036.172 332 222 036 2671 JC STS2 AM CURRENTLY OVER HOLE

```

```
2672
2673 * NO HOLE YET. WAIT 8 MS MIN (10 MAX) FOR HOLE TO
2674 * APPEAR
2675
036.175 041 033 040 2676 LXI H,TICCNT
036.200 106 2677 MOV B,M (B) = CURRENT TIME
036.201 333 177 2678 STS1 IN IP,DC
036.203 037 2679 RAR
000.000 2680 ERRNZ DF,HD-1
036.204 332 222 036 2681 JC STS2 GOT HOLE
036.207 072 121 040 2682 LDA D,STSA (A) = DELAY COUNT
036.212 200 2683 ADD B 10 MS MAX, 8 MS MIN
036.213 276 2684 CMF M
036.214 302 201 036 2685 JNE STS1 8 MS NOT UP YET
036.217 303 233 036 2686 JMP STS3 AM IN SECTOR GAP
2687
2688 * HAVE HOLE. SKIP IT AND WAIT 12 MS
2689
036.222 315 271 036 2690 STS2 CALL WNH WAIT FOR NO HOLE
036.225 072 122 040 2691 LDA D,STSB (A) = COUNT (10 MS MIN, 12 MS MAX)
036.230 315 235 040 2692 CALL D,DLY WAIT
036.233 301 2693 STS3 POP R RESTORE (BC)
036.234 363 2694 DI
2695 * JMP WHD WAIT HOLE DETECT

2697 ** WHD - WAIT HOLE DETECT.
2698 *
2699 * WHD WAITS UNTIL A HOLE IS LOCATED.
2700 *
2701 * ENTRY NONE
2702 * EXIT NONE
2703 * USES A,F
2704
036.235 333 177 2706 WHD IN DF,DC
000.000 2707 ERRNZ DF,HD-1
036.237 037 2708 RAR
036.240 322 235 036 2709 JNC WHD WAIT UNTIL FOUND
036.243 072 123 040 2710 LDA D,WHDA (A) = LOOP DELAY COUNT
036.246 303 216 040 2711 JMP D,UDLY

2713 ** STZ - SEEK TRACK ZERO.
2714 *
2715 * STZ SEEKS THE SELECTED UNIT ARM OUTWARDS UNTIL IT REACHES
2716 * TRACK ZERO.
2717 *
2718 * THE ARM POSITION BYTE IS THEN UPDATED TO 0.
2719 *
2720 * ENTRY INTERRUPTS ENABLED
2721 * EXIT INTERRUPTS ENABLED
```



```

2722 *      USES      A,F,H,L
2723
2724
036,251 315 174 040 2725 STZ0  CALL  D,MAO      MOVE ARM OUT
2726
036,254 333 177      2727 R,STZ  IN     DP,DC
036,256 346 002      2728      ANI    DF,TO
036,260 312 251 036 2729      JZ     STZ0      NOT TRACK 0 YET
036,263 052 248 040 2730      LHLD  D,TRKPT
036,266 066 000      2731      MVI   M,0      SET TRACK POINTER
036,270 311      2732      RET

2734 **      WNH - WAIT FOR NO HOLE.
2735 *
2736 *      WNH WAITS UNTIL THE CURRENT HOLE IS PAST.
2737 *
2738 *      ENTRY  NONE
2739 *      EXIT   NONE
2740 *      USES  A,F
2741
2742
036,271 333 177      2743 WNH   IN     DP,DC
000,000      2744      ERRNZ DF,HD-1
036,273 037      2745      RAR
036,274 332 271 036 2746      JC     WNH      STILL HOLE
036,277 072 124 040 2747      LDA   D,WNHA    (A) = DEBOUNCE COUNT
036,302      2748      SET   R,UDLY   REFERENCE TO R,UDLY
2749 *      JMP   D,UDLY   WAIT A LITTLE

2751 **      UDLY - MICROSECOND DELAY.
2752 *
2753 *      UDLY IS CALLED (WITH INTERRUPTS DISABLED)
2754 *      TO WAIT FOR A CERTAIN NUMBER OF MICROSECONDS.
2755 *
2756 *      EACH TIME THROUGH THE DELAY LOOP CAUSES A PAUSE OF 15/2,048
2757 *      MICROSECONDS.
2758 *
2759 *      ENTRY  (A) = LOOP COUNT (ZERO TAKEN AS 256)
2760 *      EXIT   (A) = 0
2761 *      USES  A,F
2762
2763
036,302 075      2764 R,UDLY DCR   A
036,303 302 302 036 2765      JNZ   R,UDLY
036,306 311      2766      RET

```

```
2768 ** WSC - WAIT SYNC CHARACTER.
2769 *
2770 * WSC WAITS FOR THE APPEARANCE OF A SYNC CHARACTER. THE DISK SHOULD BE
2771 * SELECTED, MOVING, AND THE HEAD SHOULD BE OVER THE PRE-SYNC
2772 * ZERO BAND.
2773 *
2774 * IF A SYNC IS NOT DETECTED IN 25 CHARACTER TIMES, AN ERROR IS RETURNED.
2775 *
2776 * ENTRY NONE
2777 * EXIT 'C' CLEAR IF OK, SYNC CHARACTER READ
2778 * (D) = 0 (CHECKSUM)
2779 * 'C' SET IF NO SYNC FOUND
2780 * USES A,F,D
2781
2782
036.307 076 375 2783 R.WSC MVI A,C.DSYN
036.311 323 176 2784 OUT UP.SC SET SYNC CHARACTER
036.313 333 176 2785 IN UP.SR RESET SYNC SEARCH
036.315 072 125 040 2786 LDA D.WSCA (A) = NUMBER OF LOOPS IN 25 CHARACTERS
036.320 127 2787 MOV D,A
036.321 333 177 2788 WSC1 IN DP.DC
036.323 346 010 2789 ANI DF.SD SEE IF SYNC
036.325 302 336 036 2790 JNZ WSC2 GOT SYNC
036.330 025 2791 DCR D
036.331 302 321 036 2792 JNZ WSC1 TRY SOME MORE
2793
2794 * COULDN'T FIND SYNC
2795
036.334 067 2796 STC CANT FIND IT
036.335 311 2797 RET
2798
2799 * FOUND IT
2800
036.336 333 174 2801 WSC2 IN UP.DP Gobble SYNC CHARACTER
036.340 026 000 2802 MVI D,0 CLEAR CHECKSUM
036.342 311 2803 RET

2805 ** WSP - WRITE SYNC PATTERN.
2806 *
2807 * WSP WRITES A SYNC PATTERN OF ZEROS, FOLLOWED BY A SYNC
2808 * CHARACTER.
2809 *
2810 * ENTRY (A) = INITIAL DELAY COUNTER
2811 * (C) = # OF ZERO BYTES TO WRITE
2812 * EXIT (D) = CHECKSUM
2813 * (C) = 0
2814 * USES A,F,C,D,E
2815
2816
036.343 075 2817 R.WSP DCR A
036.344 302 343 036 2818 JNZ R.WSP DELAY
2819
2820 * DELAY IS UP, ON WRITE GATE
```



```

2863 ** BOOT CODE.
2864 *
2865 * ENTERED TO BOOT DISK SYSTEM.
2866
2867
037.014 363 2868 BOOT DI WANT NO TROUBLES WITH INTERRUPTS !
037.015 061 200 042 2869 LXI SP,STACK CLEAR STACK
037.020 001 130 000 2870 LXI B,BOOTAL
037.023 021 132 037 2871 LXI D,BOOTA
037.026 041 110 040 2872 LXI H,D.CON
037.031 315 252 030 2873 CALL $MOVE MOVE IN CONSTANTS AND VECTORS
2874
2875 * ZERO WORK FIELD
2876
037.034 041 240 040 2877 LXI H,D.RAM
037.037 006 037 2878 MVI B,D.RAML
037.041 315 212 031 2879 CALL $ZERO ZERO MEMORY
037.044 062 061 041 2880 STA AIO.UNI
037.047 323 177 2881 OUT DP.DC OFF DISK
2882
2883 * SETUP ALL INTERRUPT VECTORS TO AN EI/RET SEQUENCE
2884
000.000 2885 ERRNZ UO.CLK-1
037.051 074 2886 INR A (A) = UO.CLK
037.052 062 010 040 2887 STA .MFLAG REQUEST CLOCK INTERRUPTS (*DI* STILL IN EFFECT NOW!)
2888
037.055 041 037 040 2889 LXI H,.UIVEC (HL) = .UIVEC ADDRESS, (A) = 1
037.060 066 303 2890 BOOT2 MVI M,3030
037.062 043 2891 INX H
037.063 066 027 2892 MVI M,#EIXIT
037.065 043 2893 INX H
037.066 066 034 2894 MVI M,EIXIT/256
037.070 043 2895 INX H
037.071 207 2896 ADD A SHIFT '1' IN (A) LEFT 1
037.072 362 060 037 2897 JP BOOT2 MORE TO GO
2898
2899 * SETUP CLOCK INTERRUPTS
2900
037.075 041 031 034 2901 BOOT3 LXI H,CLOCK
037.100 042 040 040 2902 SHLD .UIVEC+1
037.103 373 2903 EI RESTORE INTERRUPTS
2904
2905 * READ BOOT CODE
2906
037.104 315 366 033 2907 CALL R.ABORT RESET DISK 0
037.107 021 200 042 2908 LXI D,USERFWA
037.112 001 000 011 2909 LXI B,9*256
037.115 041 000 000 2910 LXI H,0
037.120 315 077 034 2911 CALL R.READ READ SYSTEM DISK BOOT CODE
037.123 322 200 042 2912 JNC USERFWA IS ALL OK
2913
2914 * WAIT FOR HIM TO HIT A CHARACTER
2915
037.126 166 2916 HLT
037.127 303 014 037 2917 JMP BOOT BOOT AGAIN
  
```

2923 ** DISK CONSTANT AND VECTOR INITIALIZATION TABLE.

037.132		2924	BOOTA	EQU	*	
000.000		2926				
037.132	170 002	2927		ERRNZ	*-BOOTA+D.CON-D.XITA	
000.000		2928		DW	2*256+120	HEAD UNSETTLE AND MOTOR ON TIMES
037.134	024	2929		ERRNZ	*-BOOTA+D.CON-D.WRITA	
000.000		2930		DB	20	GUARDBAND COUNT FOR WRITE
037.135	012	2931		ERRNZ	*-BOOTA+D.CON-D.WRITB	
000.000		2932		DB	10	NUMBER OF ZERO CHARACTERS AFTER HOLE EDGE
037.136	020	2933		ERRNZ	*-BOOTA+D.CON-D.WRITC	
000.000		2934		DB	128/8	TWO CHARACTER DELAY BEFORE WRITING
037.137	017	2935		ERRNZ	*-BOOTA+D.CON-D.MAIA	
000.000		2936		DB	15	TRACK-TO-TRACK STEP TIMES
037.140	024	2937		ERRNZ	*-BOOTA+D.CON-D.LPSA	
000.000		2938		DB	20	NUMBER OF TRYS FOR CORRECT SECTOR
037.141	021	2939		ERRNZ	*-BOOTA+D.CON-D.SDPA	
000.000		2940		DB	70/4	70 MILLISECONDS WAIT FOR HEAD SETTLE
037.142	372	2941		ERRNZ	*-BOOTA+D.CON-D.SDPB	
000.000		2942		DB	1000/4	1 SECOND WAIT FOR MOTOR ON
037.143	005	2943		ERRNZ	*-BOOTA+D.CON-D.STSA	
000.000		2944		DB	8/2+1	MS/2 TO WAIT FOR INDEX HOLE
037.144	007	2945		ERRNZ	*-BOOTA+D.CON-D.STSB	
000.000		2946		DB	12/2+1	MS/2 TO WAIT PAST INDEX HOLE
037.145	024	2947		ERRNZ	*-BOOTA+D.CON-D.WHDA	
000.000		2948		DB	20	UDLY COUNT FOR HOLE DEBOUNCE
037.146	024	2949		ERRNZ	*-BOOTA+D.CON-D.WNHA	
000.000		2950		DB	20	UDLY COUNT FOR HOLE DEBOUNCE
037.147	120	2951		ERRNZ	*-BOOTA+D.CON-D.WSCA	
		2952		DB	64*25/20	LOOP COUNT FOR 25 CHARACTERS

2954 ** ERRT - ERROR TEST LOOP

037.150	064	2955				
037.151	311	2956				
		2957	R.ERRT	INR	M	COUNT ERROR
		2958		RET		EXIT

2960 * JMP VECTORS

037.152	303 316 033	2961				
037.155	303 345 033	2962	JMP	R.SYDD		D.SYDD (MUST BE FIRST)
037.160	303 374 033	2963	JMP	R.MOUNT		D.MOUNT
037.163	303 366 033	2964	JMP	R.XOK		D.XOK
037.166	303 375 033	2965	JMP	R.ABORT		D.ABORT
037.171	303 077 034	2966	JMP	R.XIT		D.XIT
037.174	303 321 034	2967	JMP	R.READ		D.READ
037.177	303 336 034	2968	JMP	R.READR		D.READR
037.202	303 136 035	2969	JMP	R.WRITE		D.WRITE
037.205	303 172 035	2970	JMP	R.CDE		D.CDE
037.210	303 225 035	2971	JMP	R.DTS		D.DTS
037.213	303 251 035	2972	JMP	R.SDT		D.SDT
037.216	303 254 035	2973	JMP	R.MAI		D.MAI
037.221	303 321 035	2974	JMP	R.MAO		D.MAO
037.224	303 044 036	2975	JMP	R.LPS		D.LPS
		2976	JMP	R.RDB		D.RDB

ERRT

037.227	303	062	036	2977	JMP	R.SDP	D.SDP
037.232	303	165	036	2978	JMP	R.STS	D.STS
037.235	303	254	036	2979	JMP	R.STZ	D.STZ
037.240	303	302	036	2980	JMP	R.UDLY	D.UDLY
037.243	303	307	036	2981	JMP	R.WSC	D.WSC
037.246	303	343	036	2982	JMP	R.WSP	D.WSP
037.251	303	373	036	2983	JMP	R.WNB	D.WNB
037.254	303	150	037	2984	JMP	R.ERRT	D.ERRT
037.257	303	303	035	2985	JMP	R.DLY	D.DLY
000.130			2986	BOOTAL	EQU	*-BOOTA	

```

2989 *** DDIAG - INITIAL DRIVE DIAGNOSIS
2990
037,262 2991 DDIAG EQU *
037,262 076 031 2992 MVI A,DF,MO+DF,IS2+DF,NG
037,264 323 177 2993 OUT DP,DC ON DISK
037,266 016 372 2994 MVI C,250
037,270 171 2995 MOV A,C (A) = 250
037,271 315 303 035 2996 CALL R,DLY
037,274 171 2997 MOV A,C (A) = 250
037,275 315 303 035 2998 CALL R,DLY DELAY 1 SECOND
037,300 363 2999 DI DISABLE INTERRUPTS
037,301 315 306 037 3000 CALL DDIAGO NO CHECK, RETURN IF ERROR
3001
3002 * DISK DIAG ERROR
3003
037,304 373 3004 EI RESTORE INTERRUPTS
037,305 166 3005 HLT
3006
3007 * TEST DISK
3008
037,306 315 355 036 3009 DDIAGO CALL WSP1 WRITE SYNC PATTERN
037,311 001 134 014 3010 LXI R,3164
037,314 076 107 3011 DDIAG1 MVI A,'G'
037,316 315 373 036 3012 CALL R,WNR WRITE BYTE
037,321 013 3013 DCX B
037,322 170 3014 MOV A,B
037,323 261 3015 ORA C
037,324 302 314 037 3016 JNZ DDIAG1
037,327 076 030 3017 MVI A,300
037,331 323 177 3018 OUT DP,DC DEF WRITE SELECT
3019
3020 * NOW TRY READ
3021
037,333 076 333 3022 MVI A,219
037,335 062 125 040 3023 STA D,WSCA WAIT FOR 60 CHARS MAX
037,340 315 307 036 3024 CALL R,WSC WAIT FOR SYNC DETECT
037,343 330 3025 RC ERROR
037,344 001 132 014 3026 LXI R,3164-2 ALLOW USARY TO GORBLE TWO DURING WRITE
037,347 315 044 036 3027 DDIAG2 CALL R,RDB READ BYTE
037,352 376 107 3028 CPI 'G'
037,354 300 3029 RNE ERROR
037,355 013 3030 DCX B
037,356 170 3031 MOV A,B
037,357 261 3032 ORA C
037,360 302 347 037 3033 JNZ DDIAG2
037,363 373 3034 EI RESTORE INTERRUPTS
037,364 166 3035 HLT OK
3036
037,365 000 112 107 3037 DB 0,'JBL',0 ERROR ROUTING CODE
037,372 110 105 101 3038 DB 'HEATH'
037,377 000 3039 DB 0
3040
3041
3042
377,377 3043 ERRFL *-40001A OVERFLOW
3044
  
```

040.000 3045 END
ASSEMBLY COMPLETE
3045 STATEMENTS
1 ERRORS DETECTED
11072 BYTES FREE

SYDD - SYSTEM DEVICE / DEVICE DRIVER

CROSS REFERENCE TABLE

XREF V1.1

PAGE 63

\$CDEHL	030216	708	729	901L			
\$CHL	030224	918L	1798				
\$COMP	030060	754L	760				
\$DADA	030072	773L	1302				
\$DADA	030101	788L	1187				
\$DU66	030106	805L	1308				
\$HLIHL	030211	885L					
\$INDL	030234	942L					
\$MOVE	030252	979E	2673				
\$MU10	030324	1036L					
\$MU66	030337	1058L					
\$MUS6	031007	1102L					
\$RSTALL	031047	1140L	1744	1983			
\$SAVALL	031054	1158L	1672	1982			
\$TBL1	031113	1245L	1251				
\$TBL2	031133	1247	1259L				
\$TBLS	031111	1243L					
\$TBRA	031076	1214E					
\$TJMP	031061	1182L					
\$TJMP	031062	1184E					
\$TYPTX	031136	1275L					
\$TYPTX	031144	1276	1280L	1285			
\$UDD	031157	1301E					
\$WDR	031222	1350L					
\$WDR1	031231	1354L	1373				
\$WER	031241	1369L					
\$ZERO	031212	1333L	2829				
	036302	2061S	2478S	2545S	2546	2748S	
.ABUSS	040024	100E					
.ALARM	002136	73E					
.ALEDS	040013	98E					
.CHFLG	000060	222L					
.CLEAR	000055	219L					
.CLEARA	000056	220L					
.CLOSE	000046	212L					
.CLRCD	000007	199L					
.CONSL	000006	198L					
.CRC	002347	81E					
.CRCSUM	040027	101E					
.CYC	002172	75E					
.CTLG	000041	207L					
.CTLFLG	040011	97E					
.DECODE	000053	217L					
.DELET	000050	214L					
.DISMT	000061	223L					
.DLEDS	040021	99E					
.DLY	000053	70E					
.DOP	003122	84E					
.DODA	003356	86E					
.DSPHOB	040007	95E					
.DSPROT	040006	94E					
.DUMF	001374	72E					
.ERROR	000057	221L					
.EXIT	000000	192L					
.HORN	002140	74E					
.IDENT	000000	69E					
.IDWRK	040002	92E					
.LINK	000040	206L					

CROSS REFERENCE TABLE

D.MOUNT	040133	445L	2031						
D.OECNT	040264	494L	2384	2601					
D.OPR	040273	509L	2125	2127					
D.OPW	040275	510L	2270	2272					
D.RAM	040240	401L	477	512	2877				
D.RAML	000037	512E	2878						
D.RDR	040202	458L	2175	2186	2190	2524	2528	2533	2540
D.READ	040147	449L	2020						
D.READR	040152	450L	2026						
D.SDP	040205	459L	2046	2059	2124	2251	2289		
D.SDPA	040117	425L	2624	2939					
D.SDPB	040120	426L	2633	2941					
D.SDT	040166	454L	2216	2352	2379				
D.SECNT	040262	493L	2381	2383	2391				
D.STS	040210	460L	2440	2512	2551				
D.STSA	040121	427L	2682	2943					
D.STSR	040122	428L	2691	2945					
D.STZ	040213	461L	2047	2060	2378				
D.SYDD	040130	444L							
D.TRKPT	040245	487L	2437	2441	2730				
D.TS	040241	480L	2204	2212	2337	2347	2418	2531	
D.TT	040240	479L	2212	2347	2415	2438	2527	2531	
D.UDLY	040216	462L	2153	2302	2711				
D.VEC	040130	400L	442						
D.VOLFT	040247	488L	2048	2253	2523	2643			
D.WHDA	040123	429L	2710	2947					
D.WNB	040227	465L	2319	2324	2329	2330	2331	2830	2835
D.WNHA	040124	430L	2747	2949					
D.WRITA	040112	420L	2308	2929					
D.WRITB	040113	421L	2311	2931					
D.WRITC	040114	422L	2313	2933					
D.WRITE	040155	451L	2023						
D.WSC	040221	463L	2170	2521					
D.WSCA	040125	431L	2786	2951	3023				
D.WSP	040224	464L	2314						
D.XIT	040144	448L	2233	2361					
D.XITA	040110	419L	2076	2927					
D.XOK	040136	446L	2028	2050	2238	2335			
DC.ABT	000007	371L	2027	2029					
DC.CLD	000006	370L							
DC.MDU	000010	372L	2029						
DC.OPR	000003	367L							
DC.OPU	000005	369L							
DC.OPW	000004	368L							
DC.REA	000000	364L	1391	1414	2018				
DC.RER	000002	366L	2024						
DC.WRI	000001	365L	1387	1440	2021				
DCA	032002	1502L							
DCA1	031377	1500L	1513						
DCA2	032035	1517L	1563						
DCA3	032055	1527	1529L						
DCA4	032071	1535	1537L						
U DD.ENT	000000	1707							
DDIAG	037262	2991E							
DDIAG0	037306	3000	3009L						
DDIAG1	037314	3011L	3016						
DDIAG2	037347	3027L	3033						
DEV.DDA	000004	262L	1737						

SYDD - SYSTEM DEVICE / DEVICE DRIVER

XREF V1.1

CROSS REFERENCE TABLE

PAGE 68

DU665	030205	814	871L		
DWRITE	031253	1387L			
EC.CNA	000004	328L			
EC.DDA	000027	347L			
EC.DIF	000017	339L			
EC.DIW	000035	353L			
EC.DNS	000005	329L			
EC.EOF	000001	325L	1966		
EC.EQM	000002	326L			
EC.FAO	000031	349L			
EC.FAP	000026	346L			
EC.FL	000030	348L			
EC.FNF	000014	336L			
EC.FNO	000011	333L	1448		
EC.FNR	000034	352L			
EC.FUC	000013	335L			
EC.ICN	000016	338L			
EC.IDW	000006	330L			
EC.IFC	000020	340L			
EC.IFW	000007	331L			
EC.ILC	000003	327L			
EC.ILO	000040	356L			
EC.ILR	000012	334L	1454		
EC.ILV	000037	355L			
EC.IS	000032	350L			
EC.NEM	000021	341L			
EC.NRD	000010	332L			
EC.RF	000022	342L	2232		
EC.UNA	000036	354L			
EC.UND	000015	337L			
EC.UUN	000033	351L			
EC.WF	000023	343L	2359		
EC.WP	000025	345L	2276		
EC.WPV	000024	344L			
EIXIT	034027	2080L	2892	2894	
ENL	000212	169E			
ERFICNT	000012	18E	2388	2600	
ERR.FND	031344	1448L			
ERR.ILR	031350	1454L			
ESC	000033	167E			
FF	000014	170E			
FFB	032133	1588L			
FFB4	032145	1599L	1606		
FFB5	032161	1602	1612L		
FFB6	032165	1616L	1623		
FFL	032205	1500	1560	1643L	
I.CONFL	000004	565E	566		
I.CONTY	000001	552E	553		
I.CONWI	000003	558E	559		
I.CSLMB	000000	542E			
I.CUSOR	000002	555E	556		
IP.FAD	000360	30E			
LDB	032223	1672L			
LDB2	032252	1693L	1700		
LDB3	032271	1708L	1725		
LDB4	032323	1714	1729L		
LDB8	032362	1693	1708	1710	1759L
LDO	033012	1792L			

SYDD - SYSTEM DEVICE / DEVICE DRIVER
CROSS REFERENCE TABLE

XREF V1.1
PAGE 69

LDD1	033073	1811	1831L			
LF	000012	157E				
LPS0	035316	2512L	2518			
LPS1	035334	2520L	2553			
LPS1.5	036007	2544L	2558	2561	2564	
LPS2	036014	2535	2551L			
LPS3	036025	2522	2557L			
LPS4	036032	2526	2560L			
LPS5	036037	2530	2563L			
M.FDX	000303	64E				
M.PAMB	000021	63E				
MEM1	030017	706L	709			
MEM2	030032	715L	731			
MEM3	030036	718L	730			
MEM4	030046	721	728L			
MI.CPI	000376	17E	1388	2463		
MOV1	030272	998L	1005			
MOV2	030311	987	1014L	1021		
MU661	030344	1062L	1080	1084		
MU662	030364	1068	1070	1074L		
MU663	031005	1076	1086L			
MUB60	031014	1105L	1119			
MUB61	031015	1107L	1117			
MUB62	031026	1108	1110	1112L		
MUB63	031044	1113	1121L			
NL	000012	168E	169			
NUL2	000000	159E				
NULL	000200	158E				
OP.CTL	000360	31E				
OP.DIG	000360	32E				
OP.SEG	000361	33E				
OVL.IN	000001	608E	1678	1838	1990	
OVL.RES	000002	609E				
OVL.UCS	000200	610E	1822	1988	1990	
PCHL	032361	1745	1748L			
PDI	033145	1876L				
PDI1	033161	1887L	1889			
PIC.COD	000006	384L	1844	1849		
PIC.ID	000000	379L				
PIC.LEN	000002	381L				
PIC.PTR	000004	382L	1849			
QUOTE	000047	165E				
R.ABORT	033366	2058E	2907	2965		
R.CDE	035136	2377L	2970			
R.DLY	035303	2478	2488L	2985	2996	2998
R.DTS	035172	2409L	2971			
R.ERRT	037150	2957L	2984			
R.LPS	035321	2514L	2975			
R.MAI	035251	2461L	2973			
R.MAD	035254	2464L	2974			
R.MOUNT	033345	2043E	2963			
R.RDB	036044	2576L	2579	2976	3027	
R.READ	034077	2123L	2911	2967		
R.READR	034321	2250L	2968			
R.SDP	036062	2600L	2977			
R.SDT	035225	2420	2437L	2447	2972	
R.STS	036165	2666L	2978			
R.STZ	036254	2727L	2979			

SYDD - SYSTEM DEVICE / DEVICE DRIVER
 CROSS REFERENCE TABLE

XREF V1.1
 PAGE 70

R.SYDD	033316	2017E	2962				
R.UPLY	036302	2748	2764L	2765	2980		
R.WNB	036373	2848L	2983	3012			
R.WRITE	034336	2267E	2989				
R.WSC	036307	2783L	2981	3024			
R.WSP	036343	2817L	2818	2982			
R.XIT	033375	2067L	2966				
R.XOK	033374	2061	2066L	2964			
READ1	034112	2135L	2254				
READ1.5	034122	2139	2146L				
READ2	034126	2148L	2217	2229			
READ2.4	034130	2153L	2210				
READ3	034143	2161L					
READ3.5	034160	2167	2169L				
READ4	034167	2175L	2179				
READ5	034205	2186L	2188				
READ6	034214	2182	2189L				
READ7	034300	2155	2223	2228L			
READ71	034261	2171	2221L				
READ72	034272	2192	2225L				
READ8	034315	2163	2198	2237L			
REL	033177	1910L	1939				
REL.	033175	1856	1907L				
REL1	033213	1917	1926L				
ROMBOOT	030000	393E					
RUBOUT	000177	161E					
RUC	033257	1982L					
S.BAUD	040346	586L					
S.CAADR	040333	569L					
S.CACC	041006	623L					
S.CCTAR	040335	570L					
S.CDB	040345	583L					
S.CFWA	040352	592L					
S.CODE	041007	624L					
S.CONFL	040332	567L					
S.CONTY	040327	554L					
S.CONWI	040331	560L					
S.CSLMD	040326	543L	553	556	559	566	
S.CUSDF	040330	557L					
S.DATC	040310	525L					
S.DATE	040277	524L					
S.DCS	041030	635L					
S.DDATA	040366	603L	1731				
S.DDGRF	040364	600L	1670				
S.DDLDA	040360	598L	1686	1729	1743		
S.DDLEN	040362	599L	1683				
S.DDPC	040370	604L	1744				
S.DDSEC	040364	1670E	1769	1771			
S.DFWA	040354	593L					
S.DIREA	041016	631L					
S.DLINK	040350	591L					
S.FASER	041013	630L	1397	1417			
S.FCI	041021	632L					
S.GRT	024000	390E					
S.GRT1	025000	391E					
S.HIMEM	040316	527L					
S.INT	040345	403L	579				
S.JUMPS	041010	628L					

CROSS REFERENCE TABLE

S.MOUNT	041027	634L			
S.OMAX	040324	533L			
S.DSN	041004	619L	1834		
S.OVLE	041000	616L	1848		
S.OVLFL	040371	612L	1676	1821	1834 1985
S.OVLS	040376	615L	1797	1831	
S.OVSTK	041032	641L			
S.RFWA	040356	594L			
S.SIDIS	041034	645L			
S.SIGRT	041036	646L			
S.SCI	041024	633L			
S.SDD	041010	629L			
S.SDVR	041146	405L	407		
S.SSN	041002	618L	1819	2000	
S.SYSH	040320	529L	1800		
S.TIME	040312	526L			
S.UCSF	040372	613L	1802	1843	1998
S.UCSL	040374	614L	1816	1995	
S.USRH	040322	531L	1804		
S.VAL	040277	402L	522		
SC.UART	000372	117E			
SDP1	036130	2623	2626L		
SDP2	036145	2632	2635L		
SDT1	035242	2445L			
SDT3	035221	2433L	2441		
SECSCR	026000	392E	1689		
SREAD	031275	1411L	1773	1835	2001
SREAD1	031305	1416L	1442		
STACK	042200	409E	2869		
STACKL	001032	407E			
STS1	036201	2678L	2685		
STS2	036222	2671	2681	2690L	
STS3	036233	2686	2693L		
STZO	036251	2725L	2729		
SWRITE	031330	1436L	1820		
SYDD	040130	399E	1392	1416	
SYSCALL	000377	186E	1282		
TAB	000011	166E			
TFE	033233	1505	1956L		
UCI.ER	000020	139E			
UCI.IE	000002	141E			
UCI.IR	000100	137E			
UCI.RE	000004	140E			
UCI.RO	000040	138E			
UCI.TE	000001	142E			
UDD1	031163	1305L	1318		
UDB	000000	114E			
UF.FCT	000100	311E			
UF.RDA	000001	308E	2577		
UF.RDR	000002	309E			
UF.RPE	000004	310E			
UF.TBM	000200	312E	2851		
UMI.16X	000002	132E			
UMI.1B	000100	122E			
UMI.1X	000001	131E			
UMI.2B	000300	124E			
UMI.64X	000003	133E			
UMI.HB	000200	123E			

CROSS REFERENCE TABLE

UMI.L5	000000	127E			
UMI.L6	000004	128E			
UMI.L7	000010	129E			
UMI.L8	000014	130E			
UMI.FA	000020	126E			
UMI.FE	000040	125E			
UD.CLK	000001	56E	2885		
UD.DDU	000002	55E			
UD.HLT	000200	53E			
UD.NFR	000100	54E			
UP.DP	000174	302E	2580	2801	2854
UP.FC	000175	303E			
UP.SC	000176	305E	2784		
UP.SR	000176	306E	2785		
UP.ST	000175	304E	2576	2849	
USERFWA	042200	410E	2908	2912	
USR	000001	115E			
USR.FE	000040	146E			
USR.OE	000020	147E			
USR.PE	000010	148E			
USR.RXR	000002	150E			
USR.TXE	000004	149E			
USR.TXR	000001	151E			
WHD	036235	2704L	2709		
WNB1	036374	2849L	2852		
WNH	036271	2690	2743L	2746	
WRITE2.5	034377	2302L	2343		
WRITE1	034370	2289L			
WRITE2	034375	2297L	2353	2358	
WRITE4	035016	2309L	2310		
WRITE5	035034	2318L	2322		
WRITE7	035122	2306	2357L		
WRITE8	035132	2277	2360L		
WSC1	036321	2788L	2792		
WSC2	036336	2790	2801L		
WSF1	036355	2829L	2832	3009	
XIT1	033376	2068L	2070		
ZERDS	031320	1424L	2252		
ZR01	031213	1334L	1337		

22094 BYTES FREE