

14:38:30 16-MAY-80

```

000.000      1 .PIP.  EQU   0      ASSEMBLE AS PIP
000.001      2 ONECOPY EQU   1      DONT ASSEMBLE AS ONECOPY

```

```

000.000      4      IF      .PIP.
              6      ELSE
              7      TITLE  'ONECOPY - ONE DRIVE COPY UTILITY'
              8      ENDIF

```

```

10
11 ***      PIP - PERIPHERAL INTERCHANGE PROGRAM.
12 *
13 *      J.G. LETWIN, 11/1977 FOR *HEATH* COMPANY
14 *
15 *      COPYRIGHT 1977 BY HEATH COMPANY
16 *
17 *      G. Chandler,   78/09  Maintenance Release
18 *                   79/04
19 *
20 *                   79/11  50.05.00
21 *

```

```

23 ***      USE:
24 *
25 *      DEST=SOURCE1 [ ,SOURCE2, ...,SOURCEN] [/SWITCH1.../SWITCHN]
26 *
27 *      SWITCHES:
28 *
29 *      /RENAMEJ      RENAME
30 *      /DELETEJ     DELETE
31 *      /LISTJ       LIST
32 *      /BRIEFJ      BRIEF LIST
33 *      /SYSTEMJ     ENCLUDE SYSTEM FILES
34 *      /VERSIONJ    PIP VERSION NUMBER
35 *      /MOUNTJ      MOUNT DEVICE
36 *      /DISMOUNTJ   DISMOUNT DEVICE
37 *      /RESETJ      RESET DEVICE
38 *
39 *      /SUPPRESSJ   SUPPRESS
40 *      /JGL         WHO?

```

## 42 \*\* SYSTEM EQUIVALENCES

```

43
000.000      44 CN.SDU EQU   0      SOURCE CHANNEL NUMBER
000.001      45 CN.DES EQU   1      DESTINATION CHANNEL NUMBER
000.002      46 CN.DIR EQU   2      DIRECTORY CHANNEL NUMBER

```

## 48 \*\* PROGRAM ERROR CODES

```

49
000.200      50 PEC.DF EQU  2000     DEVICE FORMAT ERROR
000.201      51 PEC.DNC EQU  2010     DEVICES NOT CONSISTANT

```

000.203	52	PEC.TFI	EQU	2030	TARGET FILE ILLEGAL
000.204	53	PEC.CS	EQU	2040	CONTRADICTIONARY SWITCHES
000.205	54	PEC.IUW	EQU	2050	ILLEGAL USE OF WILDCARD
000.206	55	PEC.IDF	EQU	2060	ILLEGAL DESTINATION FILE FORMAT
000.207	56	PEC.SFI	EQU	2070	SOURCE FILE ILLEGAL
000.001	57		IF	ONECOPY	
	58	PEC.FCI	EQU	2100	FILE CONCATINATION ILLEGAL
	59		ENDIF		
	60				
000.000	61		XTEXT	U8250	
	63X	**		8250	UART CONTROL AND BIT DEFINITIONS.
	64X				
000.350	65X	SC.ACE	EQU	3500	SYSTEM CONSOLE PORT IF 8250 ACE
000.154	66X	AC.DLY	EQU	110	220 MIL. SEC. DELAY FOR 8250
	67X				
000.000	68X	UR.RBR	EQU	0	RECEIVER BUFFER REGISTER (READ ONLY)
	69X				
000.000	70X	UR.THR	EQU	0	TRANSMITTER HOLDING REGISTER (WRITE ONLY)
	71X				
000.000	72X	UR.DLL	EQU	0	DIVISOR LATCH (LEAST SIGNIFICANT)
	73X				
000.001	74X	UR.DLM	EQU	1	DIVISOR LATCH (MOST SIGNIFICANT)
	75X				
000.001	76X	UR.IER	EQU	1	INTERRUPT ENABLE REGISTER
000.001	77X	UC.EDA	EQU	00000001B	ENABLE RECEIVED DATA AVAILABLE INTERRUPT
000.002	78X	UC.TRE	EQU	00000010B	ENABLE TRANSMIT HOLD REGISTER EMPTY INTERRUPT
000.004	79X	UC.RSI	EQU	00000100B	ENABLE RECEIVE STATUS INTERRUPT
000.010	80X	UC.MSI	EQU	00001000B	ENABLE MODEM STATUS INTERRUPT
	81X				
000.002	82X	UR.IIR	EQU	2	INTERRUPT IDENTIFICATION REGISTER
000.001	83X	UC.IIF	EQU	00000001B	INVERTED INTERRUPT PENDING (0 MEANS PENDING)
000.006	84X	UC.IID	EQU	00000110B	INTERRUPT ID
	85X				
000.003	86X	UR.LCR	EQU	3	LINE CONTROL REGISTER
000.000	87X	UC.5BW	EQU	00000000B	5 BIT WORDS
000.001	88X	UC.6BW	EQU	00000001B	6 BIT WORDS
000.002	89X	UC.7BW	EQU	00000010B	7 BIT WORDS
000.003	90X	UC.8BW	EQU	00000011B	8 BIT WORDS
000.004	91X	UC.2SB	EQU	00000100B	TWO STOP BITS SELECTED
000.010	92X	UC.PEN	EQU	00001000B	PARITY COMPUTATION ENABLED
000.020	93X	UC.EPS	EQU	00010000B	EVEN PARITY SELECT
000.040	94X	UC.SKP	EQU	00100000B	STICK PARITY
000.100	95X	UC.SB	EQU	01000000B	SET BREAK
000.200	96X	UC.DLA	EQU	10000000B	DIVISOR LATCH ACCESS
	97X				
000.004	98X	UR.MCR	EQU	4	MODEM CONTROL REGISTER
000.001	99X	UC.DTR	EQU	00000001B	DATA TERMINAL READY
000.002	100X	UC.RTS	EQU	00000010B	REQUEST TO SEND
000.004	101X	UC.OU1	EQU	00000100B	OUT 1
000.010	102X	UC.OU2	EQU	00001000B	OUT 2
000.020	103X	UC.LOO	EQU	00010000B	LOOP
	104X				

U8250

14:38:37 16-MAY-80

000.005	105X	UR.LSR	EQU	5	LINE STATUS REGISTER
000.001	106X	UC.DR	EQU	00000001B	DATA READY
000.002	107X	UC.DR	EQU	00000010B	OVERRUN
000.004	108X	UC.PE	EQU	00000100B	PARITY ERROR
000.010	109X	UC.FE	EQU	00001000B	FRAMING ERROR
000.020	110X	UC.BI	EQU	00010000B	BREAK INTERRUPT
000.040	111X	UC.THE	EQU	00100000B	TRANSMITTER HOLDING REGISTER EMPTY
000.100	112X	UC.TSE	EQU	01000000B	TRANSMITTER SHIFT REGISTER EMPTY
	113X				
000.006	114X	UR.MSR	EQU	6	MODEM STATUS REGISTER
000.001	115X	UC.DCS	EQU	00000001B	DELTA CLEAR TO SEND
000.002	116X	UC.DDR	EQU	00000010B	DELTA DATA SET READY
000.004	117X	UC.TER	EQU	00000100B	TRAILING EDGE OF RING
000.010	118X	UC.DRL	EQU	00001000B	DELTA RECEIVE LINE SIGNAL DETECT
000.020	119X	UC.CTS	EQU	00010000B	CLEAR TO SEND
000.040	120X	UC.DSR	EQU	00100000B	DATA SET READY
000.100	121X	UC.RI	EQU	01000000B	RING INDICATOR
000.200	122X	UC.RLS	EQU	10000000B	RECEIVED LINE SIGNAL DETECT
000.000	123		XTEXT	U8251	

```

126X **      8251 USART BIT DEFINITIONS.
127X *
128X
129X **      PORT ADDRESSES
130X
000.000     131X UDR   EQU   0          DATA REGISTER IS EVEN
000.001     132X USR   EQU   1          STATUS REGISTER IS NEXT
133X
000.372     134X SC.UART EQU  3720        CONSOLE USART ADDRESS (IFF 8251)
135X
136X
137X **      MODE INSTRUCTION CONTROL BITS.
138X
000.100     139X UMI.1R  EQU  01000000B      1 STOP BIT
000.200     140X UMI.HB  EQU  10000000B      1 1/2 STOP BITS
000.300     141X UMI.2B  EQU  11000000B      2 STOP BITS
000.040     142X UMI.PE  EQU  00100000B      EVEN PARITY
000.020     143X UMI.PA  EQU  00010000B      USE PARITY
000.000     144X UMI.L5  EQU  00000000B      5 BIT CHARACTERS
000.004     145X UMI.L6  EQU  00000100B      6 BIT CHARACTERS
000.010     146X UMI.L7  EQU  00001000B      7 BIT CHARACTERS
000.014     147X UMI.L8  EQU  00001100B      8 BIT CHARACTERS
000.001     148X UMI.1X  EQU  00000001B      CLOCK X 1
000.002     149X UMI.16X EQU  00000010B      CLOCK X 16
000.003     150X UMI.64X EQU  00000011B      CLOCK X 64
151X
152X **      COMMAND INSTRUCTION BITS.
153X
000.100     154X UCI.IR  EQU  01000000B      INTERNAL RESET
000.040     155X UCI.RO  EQU  00100000B      READER-DN CONTROL FLAG
000.020     156X UCI.ER  EQU  00010000B      ERROR RESET
000.004     157X UCI.RE  EQU  00000100B      RECEIVE ENABLE
000.002     158X UCI.IE  EQU  00000010B      ENABLE INTERRUPTS FLAG
000.001     159X UCI.TE  EQU  00000001B      TRANSMIT ENABLE
160X
161X **      STATUS READ COMMAND BITS.
162X
000.040     163X USR.FE  EQU  00100000B      FRAMING ERROR
000.020     164X USR.DE  EQU  00010000B      OVERRUN ERROR
000.010     165X USR.PE  EQU  00001000B      PARITY ERROR
000.004     166X USR.TXE  EQU  00000100B      TRANSMITTER EMPTY
000.002     167X USR.RXR  EQU  00000010B      RECEIVER READY
000.001     168X USR.TXR  EQU  00000001B      TRANSMITTER READY
000.000     169X XTEXT  DIRDEF
170X
171X **      DIRECTORY ENTRY FORMAT.
172X
000.000     173X          ORG   0
174X
175X
000.377     176X DF.EMP  EQU  3770        FLAGS ENTRY EMPTY
000.376     177X DF.CLR  EQU  3760        FLAGS ENTRY EMPTY, REST OF DIR ALSO CLEAR
178X
000.000     179X DIR.NAM DS   8          NAME

```

000.010	180X	DIR.EXT	DS	3	EXTENSION
000.013	181X	DIR.PRO	DS	1	PROJECT
000.014	182X	DIR.VER	DS	1	VERSION
000.015	183X	DIRIDL	EQU	*	FILE IDENTIFICATION LENGTH
	184X				
000.015	185X	DIR.CLU	DS	1	CLUSTER FACTOR
000.016	186X	DIR.FLB	DS	1	FLAGS
000.017	187X	DIR	DS	1	RESERVED
000.020	188X	DIR.FGN	DS	1	FIRST GROUP NUMBER
000.021	189X	DIR.LGN	DS	1	LAST GROUP NUMBER
000.022	190X	DIR.LSI	DS	1	LAST SECTOR INDEX (IN LAST GROUP)
000.023	191X	DIR.CRD	DS	2	CREATION DATE
000.025	192X	DIR.ALD	DS	2	LAST ALTERATION DATE
	193X				
000.027	194X	DIRELEN	EQU	*	DIRECTORY ENTRY LENGTH
000.027	195		XTEXT	DIFDEF	

197X \*\* DIRECTORY FILE FLAGS.

	199X				
000.200	199X	DIF.SYS	EQU	10000000B	SYSTEM FILE
000.100	200X	DIF.LOC	EQU	01000000B	LOCKED FOR CHANGE
000.040	201X	DIF.WP	EQU	00100000B	WRITE PROTECTED
000.020	202X	DIF.CNT	EQU	00010000B	CONTIGUOUS FILE
	203X				
000.027	204		XTEXT	OVLDEF	

206X \*\* OVERLAY TABLE ENTRIES.

	207X				
000.000	208X		ORG	0	
	209X				
000.000	210X	OVL.COD	DS	2	FIRST SECTOR OF OVERLAY CODE
000.002	211X	OVL.SIZ	DS	2	OVERLAY SIZE
000.004	212X	OVL.ENT	DS	2	OVERLAY ENTRY POINT
000.006	213X	OVL.FLB	DS	1	OVERLAY FLAG BYTE
000.007	214X		DS	1	DUMMY BYTE TO ROUND TABLE SIZE UP TO 8
000.010	215X	OVL.ENS	EQU	*	OVERLAY ENTRY SIZE
	216X				
	217X	*			OVERLAY INDICES
	218X				
000.000	219X		ORG	0	
	220X				
000.000	221X	OVL0	DS	1	
000.001	222X	OVL1	DS	1	
000.002	223		XTEXT	DEVDEF	

DEV

225X \*\* DEVICE TABLE ENTRIES.

000.000	226X					
	227X	ORG		0		
	228X					
000.000	229X	DEV.NAM	DS	2	DEVICE NAME	
000.000	230X	DEV.EL	EQU	00000000B	END OF DEVICE LIST FLAG	
000.001	231X	DEV.NU	EQU	00000001B	DEVICE ENTRY NOT IN USE	
	232X					
000.002	233X	DEV.RES	DS	1	DRIVER RESIDENCE CODE	
000.001	234X	DR.IM	EQU	00000001B	DRIVER IN MEMORY	
000.002	235X	DR.PR	EQU	00000010B	DRIVER PERMINANTLY RESIDENT	
	236X					
000.003	237X	DEV.JMP	DS	1	JMP TO PROCESSOR	
000.004	238X	DEV.DDA	DS	2	DRIVER ADDRESS	
000.006	239X	DEV.FLG	DS	1	FLAG BYTE	
000.001	240X	DT.DD	EQU	00000001B	DIRECTORY DEVICE	
000.002	241X	DT.CR	EQU	00000010B	CAPABLE OF READ OPERATION	
000.004	242X	DT.CW	EQU	00000100B	CAPABLE OF WRITE OPERATION	
	243X					
000.007	244X	DEV.SFG	DS	1	SECTORS PER GROUP THIS DEVICE	
000.010	245X	DEV.MUM	DS	1	MOUNTED UNIT MASK	
000.011	246X	DEV.MNU	DS	1	MAXIMUM NUMBER OF UNITS	
000.012	247X	DEV.UNT	DS	2	ADDRESS OF UNIT SPECIFIC DATA TABLE	
	248X					
000.014	249X	DEV.DVL	DS	2	DRIVER BYTE LENGTH	
000.016	250X	DEV.DVG	DS	1	DRIVER ROUTINE GROUP ADDRESS	
	251X					
000.017	252X	DEVELEN	EQU	*	DEVICE TABLE ENTRY LENGTH	

254X \*\* UNIT SPECIFIC DEVICE DATA TABLE ENTRIES

000.000	255X					
	256X	ORG		0		
	257X					
000.000	258X	UNT.FLG	DS	1	UNIT SPECIFIC *DEV.FLG*	
000.001	259X	UNT.GRT	DS	2	ADDRESS OF GROUP RESERVATION TABLE (IF DT.DD)	
000.003	260X	UNT.GTS	DS	2	GRT SECTOR NUMBER	
000.005	261X	UNT.DIS	DS	2	DIRECTORY FIRST SECTOR NUMBER	
	262X					
000.007	263X	UNT.SIZ	EQU	*	SIZE OF UNIT SPECIFIC DATA TABLE PER UNIT	
000.007	264	XTEXT		IOCODEF		

266X \*\* I/O CHANNEL DEFINITIONS.

000.000	267X					
	268X	ORG		0		
	269X					
000.000	270X	IOC.LNK	DS	2	ADDRESS OF NEXT CHANNEL; =0 IF LAST	
000.002	271X	IOC.DDA	DS	2	THREAD JUMP TO DEVICE DRIVER (VIA DEV TABLE)	
	272X					
000.004	273X	IOC.FLG	DS	1	FILE TYPE FLAGS	
000.001	274X	FT.DD	EQU	00000001B	=1 IF DIRECTORY DEVICE	
000.002	275X	FT.OR	EQU	00000010B	=1 IF OPEN FOR READ	

000.004	276X	FT.DW	ERU	00000100B	=1 IF OPEN FOR WRITE
000.010	277X	FT.DU	ERU	00001000B	=1 IF OPEN FOR UPDATE
000.003	278X	IOC.SQL	ERU	*-IOC.DDA	LENGTH OF INFO FOR SEQUENTIAL FILE (FROM IOC)
	279X				
000.005	280X	IOC.GRT	DS	2	ADDRESS OF GROUP RESERVATION TABLE
000.007	281X	IOC.SPG	DS	1	SECTORS PER GROUP, THIS DEVICE
000.010	282X	IOC.CGN	DS	1	CURRENT GROUP NUMBER
000.011	283X	IOC.CSI	DS	1	CURRENT SECTOR INDEX (IN CURRENT GROUP)
000.012	284X	IOC.LGN	DS	1	LAST GROUP NUMBER
000.013	285X	IOC.LSI	DS	1	LAST SECTOR INDEX (IN LAST GROUP)
000.010	286X	IOC.DRL	ERU	*-IOC.FLG	LENGTH OF INFO NORMALLY COPIED BACK TO
	287X	*			THE CHANNEL TABLE
000.014	288X	IOC.DTA	DS	2	DEVICE TABLE ADDRESS FOR THIS DEVICE
000.016	289X	IOC.DES	DS	2	SECTOR NUMBER OF DIRECTORY ENTRY
000.020	290X	IOC.DEV	DS	2	DEVICE CODE
000.022	291X	IOC.UNI	DS	1	UNIT NUMBER (0-9)
000.021	292X	IOC.DIL	ERU	*-IOC.DDA	LENGTH OF INFO FOR DIRECTORY FILE (FROM IOC)
	293X				
000.023	294X	IOC.DIR	DS	DIRELEN	DIRECTORY ENTRY
	295X				
000.052	296X	IOCELEN	ERU	*	IOC ENTRY LENGTH
	297X				
000.001	298X	IOCCTD	ERU	1	INDEX OF USER CHANNEL #0 IN CHANTAB (FIRST = 0)
000.052	299	XTEXT		DISDEF	

301X \*\* DIRECTORY BLOCK FORMAT.

	302X				
000.000	303X		ORG	0	
	304X				
000.000	305X	DIS.ENT	ERU	*	FIRST ENTRY ADDRESS
000.000	306X		DS	22*DIRELEN	22 DIRECTORY ENTRIES PER BLOCK
001.372	307X		DS	1	0 BYTE = END OF ENTRIES IN THIS BLOCK
	308X				
001.373	309X		ORG	512-5	AT END OF BLOCK
001.373	310X	DIS.ENL	DS	1	LENGTH OF EACH ENTRY (=DIRELEN)
001.374	311X	DIS.SEC	DS	2	BLOCK # OF THIS BLOCK
001.374	312X	DIS.LNK	DS	2	BLOCK # OF NEXT BLOCK, =0 IF THIS IS LAST
002.000	313	XTEXT		FBDEF	

315X \*\* FILE BLOCK DEFINITIONS.

	316X				
000.000	317X		ORG	0	
000.000	318X	FB.CHA	DS	1	CHANNEL NUMBER
000.001	319X	FB.FLG	DS	1	FLAGS
000.002	320X	FB.FWA	DS	2	BUFFER FWA
000.004	321X	FB.PTR	DS	2	BUFFER POINTER
000.006	322X	FB.LIM	DS	2	LIMIT OF DATA IN BUFFER (READ OPERATIONS)
000.010	323X	FB.LWA	DS	2	LWA OF BUFFER
000.012	324X	FB.NAM	DS	4+8+4+1	NAME OF FILE
000.021	325X	FB.NAML	ERU	*-FB.NAM	
000.033	326X	FBENL	ERU	*	ENTRY LENGTH

000.033 327 XTEXT ECDEF

329X \*\* ERROR CODE DEFINITIONS.

000.000	330X				
000.000	331X	DRG	0		
000.000	332X	DS	1	NO ERROR #0	
000.001	333X	EC.EOF	DS	1	END OF FILE
000.002	334X	EC.EOM	DS	1	END OF MEDIA
000.003	335X	EC.ILC	DS	1	ILLEGAL SYSCALL CODE
000.004	336X	EC.CNA	DS	1	CHANNEL NOT AVAILABLE
000.005	337X	EC.BNS	DS	1	DEVICE NOT SUITABLE
000.006	338X	EC.IDN	DS	1	ILLEGAL DEVICE NAME
000.007	339X	EC.IFN	DS	1	ILLEGAL FILE NAME
000.010	340X	EC.NRD	DS	1	NO ROOM FOR DEVICE DRIVER
000.011	341X	EC.FNO	DS	1	CHANNEL NOT OPEN
000.012	342X	EC.ILR	DS	1	ILLEGAL REQUEST
000.013	343X	EC.FUC	DS	1	FILE USAGE CONFLICT
000.014	344X	EC.FNF	DS	1	FILE NAME NOT FOUND
000.015	345X	EC.UND	DS	1	UNKNOWN DEVICE
000.016	346X	EC.ICN	DS	1	ILLEGAL CHANNEL NUMBER
000.017	347X	EC.DIF	DS	1	DIRECTORY FULL
000.020	348X	EC.IFC	DS	1	ILLEGAL FILE CONTENTS
000.021	349X	EC.NEM	DS	1	NOT ENOUGH MEMORY
000.022	350X	EC.RF	DS	1	READ FAILURE
000.023	351X	EC.WF	DS	1	WRITE FAILURE
000.024	352X	EC.WPV	DS	1	WRITE PROTECTION VIOLATION
000.025	353X	EC.WP	DS	1	DISK WRITE PROTECTED
000.026	354X	EC.FAP	DS	1	FILE ALREADY PRESENT
000.027	355X	EC.DDA	DS	1	DEVICE DRIVER ABORT
000.030	356X	EC.FL	DS	1	FILE LOCKED
000.031	357X	EC.FAO	DS	1	FILE ALREADY OPEN
000.032	358X	EC.IS	DS	1	ILLEGAL SWITCH
000.033	359X	EC.UUN	DS	1	UNKNOWN UNIT NUMBER
000.034	360X	EC.FNR	DS	1	FILE NAME REQUIRED
000.035	361X	EC.DIW	DS	1	DEVICE IS NOT WRITABLE (OR WRITE LOCKED)
000.036	362X	EC.UNA	DS	1	UNIT NOT AVAILABLE
000.037	363X	EC.ILV	DS	1	ILLEGAL VALUE
000.040	364X	EC.ILO	DS	1	ILLEGAL OPTION
000.041	365X	EC.VPM	DS	1	VOLUME PRESENTLY MOUNTED ON DEVICE
000.042	366X	EC.NVM	DS	1	NO VOLUME PRESENTLY MOUNTED
000.043	367X	EC.FOD	DS	1	FILE OPEN ON DEVICE
000.044	368X	EC.NPM	DS	1	NO PROVISIONS MADE FOR REMOUNTING MORE DISKS
000.045	369X	EC.DNI	DS	1	DISK NOT INITIALIZED
000.046	370X	EC.DNR	DS	1	DISK IS NOT READABLE
000.047	371X	EC.DSC	DS	1	DISK STRUCTURE IS CORRUPT
000.050	372X	EC.NCV	DS	1	NOT CORRECT VERSION OF HDOS
000.051	373X	EC.NOS	DS	1	NO OPERATING SYSTEM MOUNTED
000.052	374X	EC.IOI	DS	1	ILLEGAL OVERLAY INDEX
000.053	375X	EC.OTL	DS	1	OVERLAY TOO LARGE
000.054	376	XTEXT	HOSEQU		



## 378X \*\* HDOS SYSTEM EQUIVALENCES.

```
379X *
380X *
024.000 381X S.GRT0 EQU 24000A SYSTEM AREA FOR GRT0
025.000 382X S.GRT1 EQU 25000A SYSTEM AREA FOR GRT1
026.000 383X S.GRT2 EQU 26000A SYSTEM AREA FOR GRT2
384X *
030.000 385X ROMBOOT EQU 30000A ROM BOOT ENTRY
386X *
040.100 387X * ORG 40100A FREE SPACE FROM PAM-8
388X *
040.100 389X * DS 8 JUMP TO SYSTEM EXIT
040.110 390X D.CONV DS 16 DISK CONSTANTS
040.130 391X SYDD EQU * SYSTEM DISK ENTRY POINT
040.130 392X D.VEC DS 24*3 SYSTEM ROM ENTRY VECTORS
040.240 393X D.RAM DS 31 SYSTEM ROM WORK AREA
040.277 394X S.VAL DS 36 SYSTEM VALUES
040.343 395X S.INT DS 115 SYSTEM INTERNAL WORK AREAS
041.126 396X * DS 16
041.146 397X S.SOVR DS 2 STACK OVERFLOW WARNING
041.150 398X * DS 42200A-* SYSTEM STACK
001.032 399X STACKL EQU *-S.SOVR STACK SIZE
400X *
042.200 401X STACK EQU * LWA+1 SYSTEM STACK
042.200 402X USERFWA EQU * USER FWA
042.200 403 XTEXT HOSDEF
```

## 405X \*\* HOSDEF - DEFINE HOS PARAMETER.

```
406X *
407X *
408X *
000.026 409X VERS EQU 1*16+6 VERSION 1.6
410X *
000.377 411X SYSCALL EQU 3770 SYSCALL INSTRUCTION
412X *
413X *
000.000 414X * ORG 0
```

## 416X \* RESIDENT FUNCTIONS

```
417X *
000.000 418X .EXIT DS 1 EXIT (MUST BE FIRST)
000.001 419X .SCIN DS 1 SCIN
000.002 420X .SCOUT DS 1 SCOUT
000.003 421X .PRINT DS 1 PRINT
000.004 422X .READ DS 1 READ
000.005 423X .WRITE DS 1 WRITE
000.006 424X .CONSL DS 1 SET/CLEAR CONSOLE OPTIONS
000.007 425X .CLRCD DS 1 CLEAR CONSOLE BUFFER
000.010 426X .LOAD0 DS 1 LOAD AN OVERLAY
000.011 427X .VERS DS 1 RETURN HDOS VERSION NUMBER
000.012 428X .SYSRES DS 1 PRECEDING FUNCTIONS ARE RESIDENT
429X *
430X *
```

## 431X \* \*HDOSVOL0.SYS\* FUNCTIONS

	432X				
000.040	433X	ORG	40A		
	434X				
000.040	435X	.LINK	DS	1	LINK (MUST BE FIRST)
000.041	436X	.CTL	DS	1	CTL-C
000.042	437X	.OPENR	DS	1	OPENR
000.043	438X	.OPENW	DS	1	OPENW
000.044	439X	.OPENU	DS	1	OPENU
000.045	440X	.OPENC	DS	1	OPENC
000.046	441X	.CLOSE	DS	1	CLOSE
000.047	442X	.POSIT	DS	1	POSITION
000.050	443X	.DELET	DS	1	DELETE
000.051	444X	.RENAM	DS	1	RENAME
000.052	445X	.SETTP	DS	1	SETTOP
000.053	446X	.DECODE	DS	1	NAME DECODE
000.054	447X	.NAME	DS	1	GET FILE NAME FROM CHANNEL
000.055	448X	.CLEAR	DS	1	CLEAR CHAN
000.056	449X	.CLEARA	DS	1	CLEAR ALL CHANS
000.057	450X	.ERROR	DS	1	LOOKUP ERROR
000.060	451X	.CHFLG	DS	1	CHANGE FLAGS
000.061	452X	.DISMT	DS	1	FLAG SYSTEM DISK DISMOUNTED
000.062	453X	.LOADD	DS	1	LOAD DEVICE DRIVER
	454X				
	455X				
	456X	*	*HDOSVLI.SYS*		FUNCTIONS
	457X				
000.200	458X	ORG	200Q		
	459X				
000.200	460X	.MOUNT	DS	1	MOUNT (MUST BE FIRST)
000.201	461X	.DMOUN	DS	1	DISMOUNT
000.202	462X	.DMNMS	DS	1	MOUNT/NO MESSAGE
000.203	463X	.DMNMS	DS	1	DISMOUNT/NO MESSAGE
000.204	464X	.RESET	DS	1	RESET = DISMOUNT/MOUNT OF UNIT
000.205	465	XTEXT	ASCII		
	467X	**			ASCII CHARACTER EQUIVALENCES.
	468X				
000.015	469X	CR	EQU	13	CARRIAGE RETURN
000.012	470X	LF	EQU	10	LINE FEED
000.200	471X	NULL	EQU	200Q	PAD CHARACTER
000.000	472X	NUL2	EQU	0	
000.007	473X	BELL	EQU	7	BELL CHARACTER
000.177	474X	RUBOUT	EQU	177Q	
000.010	475X	BKSP	EQU	10Q	CTL-H
000.026	476X	C.SYN	EQU	26Q	SYNC
000.002	477X	C.STX	EQU	2	STX
000.047	478X	QUOTE	EQU	47Q	
000.011	479X	TAB	EQU	11Q	
000.033	480X	ESC	EQU	33Q	
000.012	481X	NL	EQU	12Q	NEW LINE (HDOS SYSTEMS)
000.212	482X	ENL	EQU	NL+200Q	NL + END-OF-LINE-FLAG
000.014	483X	FF	EQU	14Q	FORM FEED
000.001	484X	CTLA	EQU	01Q	CTL-A
000.002	485X	CTLB	EQU	02Q	CTL-B
000.003	486X	CTLC	EQU	03Q	CTL-C

000.004	487X	CTLD	EQU	040	CTL-D
000.017	488X	CTLO	EQU	170	CTL-O
000.020	489X	CTLP	EQU	200	CTL-P
000.021	490X	CTLQ	EQU	210	CTL-Q
000.023	491X	CTLS	EQU	230	CTL-S
000.032	492X	CTLZ	EQU	320	CTL-Z
000.205	493		XTEXT	EDRAM	
	495X	**		EDRAM - DISK RAM WORKAREA DEFINITION.	
	496X	*			
	497X	*		ZERDED UPON BOOTING UP.	
	498X	*			
	499X	*		HOSERU MUST BE CHANGED WHEN THIS DECK IS CHANGED.	
	500X				
	501X				
040.240	502X		ORG	D.RAM	
	503X				
040.240	504X	D.TT	DS	1	TARGET TRACK (CURRENT OPERATION)
040.241	505X	D.TS	DS	1	TARGET SECTOR (CURRENT OPERATION)
	506X				
040.242	507X	D.DVCTL	DS	1	DEVICE CONTROL BYTE
	508X				
040.243	509X	D.DLYMD	DS	1	MOTOR ON DELAY COUNT
040.244	510X	D.DLYHS	DS	1	HEAD SETTLE DELAY COUNTER
	511X				
040.245	512X	D.TRKPT	DS	2	ADDRESS IN D.DRVTB FOR TRACK NUMBER
040.247	513X	D.VOLPT	DS	2	ADDRESS IN D.DRVTB FOR VOLUME NUMBER
	514X				
040.251	515X	D.DRVTB	DS	2*4	TRACK NUMBER AND VOLUME NUMBER FOR A DRIVES
	516X				
040.261	517X	D.HECNT	DS	1	HARD ERROR COUNT
040.262	518X	D.SECNT	DS	2	SOFT ERROR COUNT
040.264	519X	D.OECNT	DS	1	OPERATION ERROR COUNT
	520X				
	521X	*		GLOBAL DISK ERROR COUNTERS	
	522X				
040.265	523X	D.ERR	DS	0	BEGINNING OF ERROR BLOCK
040.265	524X	D.E.MDS	DS	1	MISSING DATA SYNC
040.266	525X	D.E.HSY	DS	1	MISSING HEADER SYNC
040.267	526X	D.E.CHK	DS	1	DATA CHECKSUM
040.270	527X	D.E.HCK	DS	1	HEADER CHECKSUM
040.271	528X	D.E.VOL	DS	1	WRONG VOLUME NUMBER
040.272	529X	D.E.TRK	DS	1	BAD TRACK SEEK
040.273	530X	D.ERRL	DS	0	LIMIT OF ERROR COUNTERS
	531X				
	532X	*		I/O OPERATION COUNTS	
	533X				
040.273	534X	D.OPR	DS	2	
040.275	535X	D.OPW	DS	2	
	536X				
000.037	537X	D.RAML	EQU	*-D.RAM	
040.277	538		XTEXT	ESINT	

```

540X **      S.INT - SYSTEM INTERNAL WORKAREA DEFINITIONS.
541X *
542X *      THESE CELLS ARE REFERENCED BY OVERLAYS AND MAIN CODE, AND
543X *      MUST THEREFORE RESIDE IN FIXED LOW MEMORY.
544X
545X
040.343     546X      ORG      S.INT
547X
548X **      CONSOLE STATUS FLAGS
549X
040.343     550X S.CDB   DS      1      CONSOLE DESCRIPTOR BYTE
000.000     551X CDB.H85 EQU     00000000B
000.001     552X CDB.H84 EQU     00000001B      =0 IF H8-5, =1 IF H8-4
040.344     553X S.BAUD  DS      2      [0-14] H8-4 BAUD RATE, =0 IF H8-5
554X *      [15] =1 IF BAUD RATE => 2 STOP BITS
555X
556X **      TABLE ADDRESS WORDS
557X
040.346     558X S.DLINK DS      2      ADDRESS OF DATA IN HDOS CODE
040.350     559X S.OPWA  DS      2      FWA OVERLAY TABLE
040.352     560X S.CFWA  DS      2      FWA CHANNEL TABLE
040.354     561X S.DFWA  DS      2      FWA DEVICE TABLE
040.356     562X S.RFWA  DS      2      FWA RESIDENT HDOS CODE
563X
564X **      DEVICE DRIVER DELAYED LOAD FLAGS
565X
040.360     566X S.DDLDA DS      2      DRIVER LOAD ADDRESS (HIGH BYTE=0 IF NO LOAD PENDING)
040.362     567X S.DDLEN DS      2      CODE LENGTH IN BYTES
040.364     568X S.DDGRP DS      1      GROUP NUMBER FOR DRIVER
040.365     569X      DS      1      HOLD PLACE
570X *S.DDSEC DS      2      2      SECTOR NUMBER FOR DRIVER ( * OBSOLETE ! * )
040.366     571X S.DDDTA DS      2      DEVICE'S ADDRESS IN DEVLST +DEV.RES
040.370     572X S.DDOPC DS      1      OPEN OPCODE PENDING
573X
574X **      OVERLAY MANAGEMENT FLAGS
575X
000.001     576X OVL.IN  EQU     00000001B      IN MEMORY
000.002     577X OVL.RES EQU     00000010B      PERMANENTLY RESIDENT
000.014     578X OVL.NUM EQU     00001100B      OVERLAY NUMBER MASK
000.200     579X OVL.UCS EQU     10000000B      USER CODE SWAPPED FOR OVERLAY
580X
040.371     581X S.OVLFL DS      1      OVERLAY FLAG
040.372     582X S.UCSF  DS      2      FWA SWAPPED USER CODE
040.374     583X S.UCSL  DS      2      LENGTH SWAPPED USER CODE
040.376     584X S.OVLS  DS      2      SIZE OF OVERLAY CODE
041.000     585X S.OVLE  DS      2      ENTRY POINT OF OVERLAY CODE
586X
041.002     587X S.SSN  DS      2      SWAP AREA SECTOR NUMBER
041.004     588X S.OSN  DS      2      OVERLAY SECTOR NUMBER
589X
590X *      SYSCALL PROCESSING WORK AREAS
591X
041.006     592X S.CACC  DS      1      (ACC) UPON SYSCALL
041.007     593X S.CGDE  DS      1      SYSCALL INDEX IN PROGRESS
594X
595X *      JUMPS TO ROUTINES IN RESIDENT HDOS CODE

```

	596X				
041.010	597X	S.JUMPS DS	0		START OF DUMP VECTORS
041.010	598X	S.SDD DS	3		JUMP TO STAND-IN DEVICE DRIVER
041.013	599X	S.FASER DS	3		JUMP TO FATERR (FATAL SYSTEM ERROR)
041.016	600X	S.DIREA DS	3		JUMP TO DIREAD (DISK FILE READ)
041.021	601X	S.FCI DS	3		JUMP TO FCI (FETCH CHANNEL INFO)
041.024	602X	S.SCI DS	3		JUMP TO SCI (STORE CHANNEL INFO)
041.027	603X	S.GUP DS	3		JUMP TO GUP (GET UNIT POINTER)
	604X				
041.032	605X	S.MOUNT DS	1		0 IF THE SYSTEM DISK IS MOUNTED
041.033	606X	S.DCS DS	1		DEFAULT CLUSTER SIZE-1
	607X				
041.034	608X	S.BOOTF DS	1		BOOT FLAGS
000.001	609X	BOOT.P EQU	00000001B		EXECUTE PROLOGUE UPON BOOTUP
	610X				
	611X	*			STACK VALUE SAVED FOR OVERLAY SYSCALLS
	612X				
041.035	613X	S.OVSTK DS	2		VALUE OF SP UPON SYSCALLS USING OVERLAY
	614X				
041.037	615X	DS	1		RESERVED
	617X	**			ACTIVE I/O AREA.
	618X	*			
	619X	*			THE AIO.XXX AREA CONTAINS INFORMATION ABOUT THE I/O OPERATION
	620X	*			CURRENTLY BEING PERFORMED. THE INFORMATION IS OBTAINED FROM
	621X	*			THE CHANNEL TABLE, AND WILL BE RESTORED THERE WHEN DONE.
	622X	*			
	623X	*			NORMALLY, THE AIO.XXX INFORMATION WOULD BE OBTAINED DIRECTLY
	624X	*			FROM VARIOUS SYSTEM TABLES VIA POINTER REGISTERS. SINCE THE
	625X	*			9080 HAS NO GOOD INDEXED ADDRESSING, THE DATA IS MANUALLY
	626X	*			COPIED INTO THE AIO.XXX CELLS BEFORE PROCESSING, AND
	627X	*			BACKDATED AFTER PROCESSING.
	628X				
041.040	629X	AIO.VEC DS	3		JUMP INSTRUCTION
041.041	630X	AIO.IDA EQU	*-2		DEVICE DRIVER ADDRESS
041.043	631X	AIO.FLG DS	1		FLAG BYTE
041.044	632X	AIO.BRT DS	2		ADDRESS OF GROUP RESERV TABLE
041.046	633X	AIO.SPG DS	1		SECTORS PER GROUP
041.047	634X	AIO.CGN DS	1		CURRENT GROUP NUMBER
041.050	635X	AIO.CSI DS	1		CURRENT SECTOR INDEX
041.051	636X	AIO.LGN DS	1		LAST GROUP NUMBER
041.052	637X	AIO.LSI DS	1		LAST SECTOR INDEX
041.053	638X	AIO.DTA DS	2		DEVICE TABLE ADDRESS
041.055	639X	AIO.DES DS	2		DIRECTORY SECTOR
041.057	640X	AIO.DEV DS	2		DEVICE CODE
041.061	641X	AIO.UNI DS	1		UNIT NUMBER (0-9)
	642X				
041.062	643X	AIO.DIR DS	DIRELEN		DIRECTORY ENTRY
	644X				
041.111	645X	AIO.CNT DS	1		SECTOR COUNT
041.112	646X	AIO.EOM DS	1		END OF MEDIA FLAG
041.113	647X	AIO.EOF DS	1		END OF FILE FLAG
041.114	648X	AIO.TFF DS	2		TEMP FILE POINTERS

041.116 649X AID.CHA DS 2 ADDRESS OF CHANNEL BLOCK (IOC.DDA)

041.120 651X S.SCR DS 2 SYSTEM SCRATCH AREA ADDRESS  
041.122 652 XTEXT ESVAL

654X \*\* S.VAL - SYSTEM VALUE DEFINITIONS.  
655X \*  
656X \* THESE VALUES ARE SET AND MAINTAINED BY THE SYSTEM.  
657X \*  
658X \* THE DECK HOSEQU MUST BE MODIFIED WHEN THIS IS MODIFIED.  
659X \*

040.277 660X  
661X ORG S.VAL  
662X  
040.277 663X S.DATE DS 9 SYSTEM DATE (IN ASCII)  
040.310 664X S.DATC DS 2 CODED DATE  
040.312 665X S.TIME DS 4 TIME FROM MIDNIGHT (IN TICS)  
040.316 666X S.HIMEM DS 2 HARDWARE HIGH MEMORY ADDRESS+1  
667X  
040.320 668X S.BYSM DS 2 FWA RESIDENT SYSTEM  
669X  
040.322 670X S.USRM DS 2 LWA USER MEMORY  
671X  
040.324 672X S.OMAX DS 2 MAX OVERLAY SIZE FOR SYSTEM  
673X

674X  
675X \*\* THE FOLLOWING FIVE CELLS SHOULD BE MODIFIED/READ ONLY VIA THE .CONSL SYSCALL  
676X

000.200 677X CSL.ECH EQU 10000000B SUPPRESS ECHO  
000.002 678X CSL.WRP EQU 00000010B WRAP LINES AT WIDTH  
000.001 679X CSL.CHR EQU 00000001B OPERATE IN CHARACTER MODE  
680X  
000.000 681X I.CSLMD EQU 0 S.CSLMD IS FIRST BYTE  
040.326 682X S.CSLMD DS 1 CONSOLE MODE  
683X  
000.200 684X CTF.BKS EQU 10000000B TERMINAL PROCESSES BACKSPACES  
000.040 685X CTF.MLI EQU 00100000B MAP LOWER CASE TO UPPER ON INPUT  
000.020 686X CTF.MLO EQU 00010000B MAP LOWER CASE TO UPPER ON OUTPUT  
000.010 687X CTF.2SB EQU 00001000B TERMINAL NEEDS TWO STOP BITS  
000.002 688X CTF.BKM EQU 00000010B MAP BKSP (UPON INPUT) TO RUBOUT  
000.001 689X CTF.TAB EQU 00000001B TERMINAL SUPPORTS TAB CHARACTERS  
690X  
000.001 691X I.CONTY EQU 1 S.CONTY IS 2ND BYTE  
000.000 692X ERRNZ \*-S.CSLMD-I.CONTY  
040.327 693X S.CONTY DS 1 CONSOLE TYPE FLAGS  
000.002 694X I.CUSOR EQU 2 S.CUSOR IS 3RD BYTE  
000.000 695X ERRNZ \*-S.CSLMD-I.CUSOR  
040.330 696X S.CUSOR DS 1 CURRENT CURSOR POSITION  
000.003 697X I.CONWI EQU 3 S.CONWI IS 4TH BYTE  
000.000 698X ERRNZ \*-S.CSLMD-I.CONWI

040.331	699X	S.CONWI	DS	1	CONSOLE WIDTH
	700X				
000.001	701X	CS.FLG	ERU	00000001B	CTL-D FLAG
000.200	702X	CS.FLG	ERU	10000000B	CTL-S FLAG
	703X				
000.004	704X	I.CONFL	EQU	4	S.CONFL IS 5TH BYTE
000.000	705X		ERRNZ	*-S.CSLMD-I.CONFL	
040.332	706X	S.CONFL	DS	1	CONSOLE FLAGS
	707X				
040.333	708X	S.CAADR	DS	2	ADDRESS FOR ABORT PROCESSING (>256 IF VALID)
040.335	709X	S.CCTAB	DS	6	ADDR FOR CTL-A, CTL-B, CTL-C PROCESSING
040.343	710		XTEXT	DDDEF	

712X \*\* DEVICE DRIVER COMMUNICATION FLAGS.

	713X	*			
	714X				
000.000	715X	ORG		0	
	716X				
000.000	717X	DC.REA	DS	1	READ
000.001	718X	DC.WRI	DS	1	WRITE
000.002	719X	DC.RER	DS	1	READ REGARDLESS
000.003	720X	DC.OPR	DS	1	OPEN FOR READ
000.004	721X	DC.OPW	DS	1	OPEN FOR WRITE
000.005	722X	DC.OPU	DS	1	OPEN FOR UPDATE
000.006	723X	DC.CLO	DS	1	CLOSE
000.007	724X	DC.ABT	DS	1	ABORT
000.010	725X	DC.MOU	DS	1	MOUNT DEVICE
000.011	726X	DC.LOD	DS	1	LOAD DEVICE DRIVER
000.012	727X	DC.MAX	DS	1	MAXIMUM ENTRY INDEX
000.013	728		XTEXT	MTR	

731X \*\* MTR - PAM/8 EQUIVALENCES.

732X \*

733X \* THIS DECK CONTAINS SYMBOLIC DEFINITIONS USED TO

734X \* MAKE USE OF THE PAM/8 CODE AND CONTROL BYTES.

736X \*\* IO PORTS

737X

000.360	738X	IP.FAD	EQU	360Q	PAD INPUT PORT
000.360	739X	OP.CTL	EQU	360Q	CONTROL OUTPUT PORT
000.360	740X	OP.DIG	EQU	360Q	DIGIT SELECT OUTPUT PORT
000.361	741X	OP.SEG	EQU	361Q	SEGMENT SELECT OUTPUT PORT

743X \*\* FRONT PANEL CONTROL BITS.

744X

000.020	745X	CB.SSI	EQU	00010000B	SINGLE STEP INTERRUPT
000.040	746X	CB.MTL	EQU	00100000B	MONITOR LIGHT
000.100	747X	CB.CLI	EQU	01000000B	CLOCK INTERRUPT ENABLE
000.200	748X	CB.SFK	EQU	10000000B	SPEAKER ENABLE

750X \*\* MONITOR MODE FLAGS.

751X

000.000	752X	DM.MR	EQU	0	MEMORY READ
000.001	753X	DM.MW	EQU	1	MEMORY WRITE
000.002	754X	DM.RR	EQU	2	REGISTER READ
000.003	755X	DM.RW	EQU	3	REGISTER WRITE

757X \*\* USER OPTION BITS.

758X \*

759X \* THESE BITS ARE SET IN CELL .MFLAG.

760X

000.200	761X	UD.HLT	EQU	10000000B	DISABLE HALT PROCESSING
000.100	762X	UD.NFR	EQU	CB.CLI	NO REFRESH OF FRONT PANEL
000.002	763X	UD.DDU	EQU	00000010B	DISABLE DISPLAY UPDATE
000.001	764X	UD.CLK	EQU	00000001B	ALLOW PRIVATE INTERRUPT PROCESSING

766X \*\* MONITOR IDENTIFICATION FLAGS

767X \*

768X \* THESE BYTES IDENTIFY THE ROM MONITOR.

769X \* THEY ARE THE VARIOUS VALUES OF LOCATION .IDENT

770X

000.021	771X	M.PAMB	EQU	021Q	'LXI' INSTRUCTION AT 000.000 IN PAM-8
000.303	772X	M.FOX	EQU	303Q	'JMP' INSTRUCTION AT 000.000 IN FOX ROM



774X \*\* ROUTINE ENTRY POINTS.

775X \*

776X

000.000	777X	.IDENT	EQU	0000A	IDENTIFICATION LOCATION
000.053	778X	.DLY	EQU	0053A	DELAY
001.267	779X	.LOAD	EQU	1267A	TAPE LOAD
001.374	780X	.DUMP	EQU	1374A	TAPE DUMP
002.136	781X	.ALARM	EQU	2136A	ALARM ROUTINE
002.140	782X	.HORN	EQU	2140A	HORN
002.172	783X	.CTC	EQU	2172A	CHECK TAPE CHECKSUM
002.205	784X	.TPERR	EQU	2205A	TAPE ERROR ROUTINE
002.264	785X	.FCHL	EQU	2264A	FCHL INSTRUCTION
002.265	786X	.SRS	EQU	2265A	SCAN RECORD START
002.325	787X	.RNP	EQU	2325A	READ NEXT PAIR
002.331	788X	.RNB	EQU	2331A	READ NEXT BYTE
002.347	789X	.CRC	EQU	2347A	CRC-16 CALCULATOR
003.017	790X	.WNP	EQU	3017A	WRITE NEXT PAIR
003.024	791X	.WNB	EQU	3024A	WRITE NEXT BYTE
003.122	792X	.DOD	EQU	3122A	DECODE FOR OCTAL DISPLAY
003.260	793X	.RCK	EQU	3260A	READ CONSOLE KEYS
003.356	794X	.DODA	EQU	3356A	SEGMENT CODE TABLE

796X \*\* RAM CELLS USED BY HBMT.

797X \*

798X

040.000	799X	.START	EQU	40000A	START DUMP ADDRESS
040.002	800X	.IDWRK	EQU	40002A	IN OR OUT INSTRUCTION
040.005	801X	.REGI	EQU	40005A	DISPLAYED REGISTER INDEX
040.006	802X	.DSPROT	EQU	40006A	PERIOD FLAG BYTE
040.007	803X	.DSPMOD	EQU	40007A	DISPLAY MODE
040.010	804X	.MFLAG	EQU	40010A	USER OPTION BYTE
040.011	805X	.CTLFLG	EQU	40011A	PANEL CONTROL BYTE
040.013	806X	.ALEDS	EQU	40013A	ABUSS LEADS
040.021	807X	.DLEDS	EQU	40021A	DBUSS LEADS
040.024	808X	.ABUSS	EQU	40024A	ABUSS REGISTER
040.027	809X	.CRCSUM	EQU	40027A	CRCSUM WORD
040.031	810X	.TPERRX	EQU	40031A	TAPE ERROR EXIT VECTOR
040.033	811X	.TICNT	EQU	40033A	CLOCK TICK COUNTER
040.035	812X	.REGPTR	EQU	40035A	REGISTER POINTER
040.037	813X	.UIVEC	EQU	40037A	USER INTERRUPT VECTORS
000.013	814	.XTEXT	DDDEF		

816X \*\* DIRECTORY DEVICE FORMAT DEFINITION.

817X \*

818X

819X

000.002	820X	HDS.SPG	EQU	2	2 SECTORS PER GROUP REQUIRED FOR NOW
	821X				
000.000	822X	.DRG	DS	0	
000.000	823X	.DDF.BOD	DS	9	2K BOOT PROGRAM
000.011	824X	.DDF.BOL	EQU	*	LENGTH OF BOOT
000.011	825X	.DDF.LAB	DS	1	LABEL SECTOR

000.012	826X	DDF.RGT	DS	2	
000.014	827X	DDF.USR	DS	0	RESERVED GROUP TABLE
000.014	828		XTEXT	LABDEF	BEGINNING OF OPEN SPACE

830X \*\* DISK LABEL SECTOR FORMATS.

	831X				
000.000	832X		ORG	0	
000.000	833X	LAB.SER	DS	1	SERIAL NUMBER OF VOLUME
000.001	834X	LAB.IND	DS	2	INITIALIZATION DATE
000.003	835X	LAB.DIS	DS	2	SECTOR NUMBER OF 1ST DIRECTORY SECTOR
000.005	836X	LAB.GRT	DS	2	INDEX OF GRT SECTOR
000.007	837X	LAB.SPG	DS	1	SECTORS PER GROUP
	838X				
000.000	839X	LAB.DAT	EQU	0	DATA VOLUME ONLY
000.001	840X	LAB.SYS	EQU	1	SYSTEM VOLUME
000.002	841X	LAB.NOD	EQU	2	=> LAB.NOD MEANS VOLUME HAS NO DIRECTORY
	842X				
000.010	843X	LAB.VLT	DS	1	VOLUME TYPE
000.011	844X	LAB.VER	DS	1	VERSION OF INIT17 THAT INITED DISK
000.012	845X		DS	7	UNUSED
000.021	846X	LAB.LAB	DS	60	LABEL
000.074	847X	LAB.LBL	EQU	*-LAB.LAB	LABEL LENGTH
000.115	848		XTEXT	FILDEF	

850X \*\* FILDEF - FILE TYPE DEFINITIONS.

	851X	*			
	852X	*	DB	3770,FT.XXX	
	853X				
	854X				
000.000	855X	FT.ABS	EQU	0	ABSOLUTE BINARY
000.001	856X	FT.PIC	EQU	1	POSITION INDEPENDANT CODE
000.002	857X	FT.REL	EQU	2	RELOCATABLE CODE
000.003	858X	FT.BAC	EQU	3	COMPILED BASIC CODE
000.115	859		XTEXT	ABSDEF	

861X \*\* ABS FORMAT EQUIVALENCES.

	862X				
000.000	863X		ORG	0	
	864X				
000.000	865X	ABS.ID	DS	1	3770 = BINARY FILE FLAG
000.001	866X		DS	1	FILE TYPE (FT.ABS)
000.002	867X	ABS.LDA	DS	2	LOAD ADDRESS
000.004	868X	ABS.LEN	DS	2	LENGTH OF ENTIRE RECORD
000.006	869X	ABS.ENT	DS	2	ENTRY POINT
	870X				
000.010	871X	ABS.COD	DS	0	CODE STARTS HERE

```

042.170          874      ORG      USERFWA-ABS.COD
042.170 377 000      875      DB      377Q,FT.ABS
042.172 200 042      876      DW      USERFWA      LOAD ADDRESS
042.174 240 021      877      DW      HEML-USERFWA      SIZE
042.176 332 063      878      DW      ENTRY      ENTRY
                        879
042.200          880 PIP      EQU      *
                        881
                        882 *      COMMAND INTERPRETATION COMES HERE
                        883
042.200          884 RESTART EQU      *
                        885
042.200 072 244 063 886      LDA      MODE
042.203 247          887      ANA      A
042.204 302 347 042 888      JNZ      EXIT      ENTERED WITH COMMAND, WILL NOW EXIT
042.207 061 200 042 889 START LXI      SP,STACK      CLEAN STACK
042.212 315 220 042 890      CALL     PIP1      EXECUTE COMMAND
                        891
                        892 *      COMMANDS EXIT HERE IF NO ERRORS FOUND
                        893
042.215 303 200 042 894      JMP      RESTART
                        895
                        896 *      GET READY TO PROCESS COMMAND
                        897
042.220 315 271 056 898 PIP1  CALL     SDD      SET DEFAULT DEFAULT
                        899
                        900 *      CLEAR CHANNELS AND FILE BUFFER
                        901
042.223 377 056      902      DB      SYSCALL,CLEARA CLEAR CHANNELS
042.225 257          903      XRA      A
042.226 062 274 063 904      STA      DESTFB+FB.FLG  FLAG FILE NOT OPEN
                        905
                        906 *      CLEAR DYNAMIC BUFFERS
                        907
042.231 041 000 000 908      LXI      H,0
042.234 042 271 063 909      SHLD   BUFSIZ      EMPTY BUFFER
042.237 042 326 063 910      SHLD   NAMTLEN     CLEAR NAMTAB
042.242 042 330 063 911      SHLD   NAMTMAX     CLEAR NAMTAB AREA
042.245 041 154 065 912      LXI      H,BUFF
042.250 042 267 063 913      SHLD   BUFPTR      SET BUFFER AGAINST END OF NAMTAB
                        914
                        915 *      INPUT COMMAND LINE
                        916
042.253 315 021 057 917      CALL     $CCO      CLEAR CONTROL-0
042.256 072 244 063 918      LDA      MODE
042.261 247          919      ANA      A
042.262 314 274 043 920      CZ      ACL      ACCEPT COMMAND LINE (UNLESS WAS PASSED ONE BY CALLER)
042.265 332 347 042 921      JC      EXIT      EOF
042.270 041 034 065 922      LXI      H,LINE     (HL) = COMMAND ADDRESS
042.273 021 364 042 923      LXI      D,PIPA     (DE) = SWITCH LIST
000.000          924      ERRNZ  I.COP
042.276 257          925      XRA      A      (A) = #I.COP
042.277 062 243 063 926      STA      COMAND     ASSUME COPY COMMAND
042.302 062 246 063 927      STA      SUPRES     CLEAR /SUP FLAG
042.305 074          928      INR      A      FLAG 'NO' /S FLAG
042.306 062 247 063 929      STA      SYSTEM     CLEAR /S FLAG

```

042.311	315	311	060	930	CALL	\$DRS	DETECT AND REMOVE SWITCHES
042.314	332	265	051	931	JC	ERROR	ERROR
042.317	072	243	043	932	LDA	COMMAND	
042.322	315	061	031	933	CALL	\$TJMP	PROCESS COMMAND

```

935 **      COMMAND LIST
936
042.325      937 PIPB  DS      0      COMMAND PROCESSOR TABLE
000.000      938 I.COP  EQU     *-PIPB/2  COMMAND INDEX
042.325 317 043 939      DW     COPY
000.001      940 I.LIS  EQU     *-PIPB/2  COMMAND INDEX
042.327 324 045 941      DW     LIST
000.002      942 I.BRE  EQU     *-PIPB/2  COMMAND INDEX
042.331 332 045 943      DW     BRIEF      /BR
000.003      944 I.VER  EQU     *-PIPB/2  COMMAND INDEX
042.333 373 050 945      DW     VERSN      /V
000.004      946 I.MOU  EQU     *-PIPB/2  /MOU,/M
042.335 371 044 947      DW     MOUNT
000.000      948      IF      .PIP.
000.005      949 I.DEL  EQU     *-PIPB/2
042.337 100 045 950      DW     DELETE      /DEL
000.006      951 I.REN  EQU     *-PIPB/2
042.341 157 045 952      DW     RENAME      /RE
000.007      953 I.DIS  EQU     *-PIPB/2
042.343 377 044 954      DW     DISMOU     /DIS
000.010      955 I.RES  EQU     *-PIPB/2
042.345 005 045 956      DW     RESET      /RES
957      ENDIF
958
959 *      CTL-D HIT
960
042.347 257      961 EXIT  XRA     A
042.350 377 000 962      DB     SYSCALL,.EXIT  EXIT

964 **      CCHIT - CTL-C HIT
965 *
966 *      ENTRY FROM SYSTEM
967
968
042.352 315 136 031 969 CCHIT CALL  $TYPTX
042.355 136 303      970      DB     C1,C12000
042.357 377 007      971      DB     SYSCALL,.CLRCD  CLEAR CONSOLE TYPEAHEAD
042.361 303 200 042 972      JMP    RESTART    GET NEW COMMAND

```

```

975 *** SWITCH PROCESSING TABLES AND ROUTINES.
976 *
977 * COMMAND SWITCHES ARE PROCESSED VIA THE ROUTINE $DRS, 'DECODE AND
978 * REMOVE SWITCHES'. $DRS IS SUPPLIED WITH A SWITCH DESCRIPTION
979 * TABLE, WHICH CONTAINS THE ADDRESSES OF ROUTINES
980 * WHICH ARE ENVOKED WHEN THE SWITCHES ARE ENCOUNTERED.
981
982
983 ** SWITCH TABLE
984
042.364 985 PIFA DS 0 FWA SWITCH TABLE
000.000 986 IF ,PIP,
042.364 104 105 114 987 DB ,DEL, /DELETE
042.367 305 324 305 988 DB ,E'+200Q,'T'+200Q,'E'+200Q,200Q
042.373 124 043 989 DW SW.DEL PROCESSING ROUTINES
990
042.375 122 991 DB ,R, /RENAME
042.376 305 316 301 992 DB ,E'+200Q,'N'+200Q,'A'+200Q,'M'+200Q,'E'+200Q,200Q
043.004 131 043 993 DW SW.REN PROCESS RENAME
994
043.006 104 111 123 995 DB ,DIS, /DISMOUNT
043.011 315 317 325 996 DB ,M'+200Q,'O'+200Q,'U'+200Q,'N'+200Q,'T'+200Q,200Q
043.017 136 043 997 DW SW.DIS
998
043.021 122 105 123 999 DB ,RES, /RESET
043.024 305 324 200 1000 DB ,E'+200Q,'T'+200Q,200Q
043.027 143 043 1001 DW SW.RES
1002
1003
043.031 114 1004 DB ,L, /LIST
043.032 311 323 324 1005 DB ,I'+200Q,'S'+200Q,'T'+200Q,200Q
043.036 241 043 1006 DW SW.LIS PROCESS LIST
1007
043.040 102 1008 DB ,B, /BRIEF
043.041 322 311 305 1009 DB ,R'+200Q,'I'+200Q,'E'+200Q,'F'+200Q,200Q
043.046 216 043 1010 DW SW.BRE PROCESS BRIEF
1011
043.050 126 1012 DB ,V, /VERSION
043.051 305 322 323 1013 DB ,E'+200Q,'R'+200Q,'S'+200Q,'I'+200Q,'O'+200Q,'N'+200Q,200Q
043.060 262 043 1014 DW SW.VER PROCESS VERSION
1015
043.062 115 117 125 1016 DB ,MOU, /MOUNT
043.065 316 324 200 1017 DB ,N'+200Q,'T'+200Q,200Q
043.070 267 043 1018 DW SW.MOU
1019
043.072 123 1020 DB ,S, /SYSTEM
043.073 331 323 324 1021 DB ,Y'+200Q,'S'+200Q,'T'+200Q,'E'+200Q,'M'+200Q,200Q
043.101 166 043 1022 DW SW.SYS PROCESS SYSTEM
1023
043.103 123 125 1024 DB ,SU, /SUPRESS
043.105 320 322 305 1025 DB ,P'+200Q,'R'+200Q,'E'+200Q,'S'+200Q,'S'+200Q,200Q
043.113 173 043 1026 DW SW.SUP
1027
043.115 112 107 114 1028 DB ,JGL, /JGL INTERNAL SWITCH
043.120 200 1029 DB 200Q
043.121 201 043 1030 DW SW.JGL
    
```

043.123 000 1031  
1032 DB 0 END OF TABLE

```

000.000          1034      IF      ,FIP,
................................................................
................................................................
          1036 **      SW.DEL - /DELETE SWITCH DETECTED.
          1037
043.124 076 005      1038 SW.DEL MVI      A,I,DEL
043.126 303 150 043 1039      JMP      SWIT1          IS MAJOR FUNCTION
................................................................
................................................................
          1041 **      SW.REN - /RENAME SWITCH DETECTED.
          1042
043.131 076 006      1043 SW.REN MVI      A,I,REN
043.133 303 150 043 1044      JMP      SWIT1          IS MAJOR FUNCTION
................................................................
................................................................
          1046 **      SW.DIS - /DISMOUNT SWITCH DETECTED
          1047
043.136 076 007      1048 SW.DIS MVI      A,I,DIS
043.140 303 150 043 1049      JMP      SWIT1          IS MAJOR FUNCTION
................................................................
................................................................
          1051 **      SW.RES - /RESET SWITCH DETECTED.
          1052
043.143 076 010      1053 SW.RES MVI      A,I,RES
043.145 303 150 043 1054      JMP      SWIT1          IS MAJOR FUNCTION
          1055      ENDIF
................................................................
................................................................
          1057 *      SWIT1 - PROCESS MAJOR FUNCTION SWITCH.
          1058 *
          1059 *      SWIT1 IS ENTERED TO PROCESS SWITCHES WHICH DETERMINE THE FUNCTION
          1060 *      FIP IS TO PERFORM. I.E. 'VERB' SWITCHES, SUCH
          1061 *      AS /DELETE (AS OPOSED TO 'MODIFIER' SWITCHES, LIKE /SYSTEM)
          1062
043.150 001 243 063 1063 SWIT1 LXI      B,COMMAND
043.153 365          1064      PUSH     PSW          SAVE COMMAND
043.154 012          1065      LDAX    B              (A) = PREVIOUS COMMAND
043.155 247          1066      ANA     A              (A) = PREVIOUS COMMAND
043.156 076 204      1067      MVI     A,PEC,CS      CONTRADICTION SWITCHES
043.159 302 265 051 1068      JNZ     ERROR        IF SO
043.163 361          1069      POP     PSW          (A) = NEW CODE
043.164 002          1070      STAX    B              STORE IT
043.165 311          1071      RET
    
```



```

1073 ** SW.SYS - /SYSTEM SWITCH DETECTED.
1074
043.166 257 1075 SW.SYS XRA A SET /S FLAG
043.167 062 247 063 1076 STA SYSTEM
043.172 311 1077 RET
  
```

```

1079 ** SW.SUP - /SUPPRESS SWITCH.
1080
1081
043.173 076 001 1082 SW.SUP MVI A,I
043.175 062 246 063 1083 STA SUPRES
043.200 311 1084 RET
  
```

```

1086 ** SW.JGL - /JGL SYSTEM SWITCH.
1087
1088
043.201 076 001 1089 SW.JGL MVI A,I
043.203 062 245 063 1090 STA JGL
043.206 076 103 1091 MVI A,'C'
043.210 062 365 050 1092 STA PFIB1 SET 'C' CHARACTER FOR FLAGS DISPLAY
043.213 303 166 043 1093 JMP SW.SYS
  
```

```

1095 ** SW.BRE - /BRIEF SWITCH DETECTED.
1096
043.216 072 243 063 1097 SW.BRE LDA COMMAND ALLOW TO SUPERCEDE /LIST
043.221 247 1098 ANA A
043.222 312 233 043 1099 JZ SW.BRE1 NO OTHER COMMAND
000.000 1100 ERRNZ I.LIS-1
043.225 075 1101 ICR A
043.226 076 204 1102 MVI A,PEC,CS ASSUME CONTRADICTIONARY SWITCHES
043.230 302 265 051 1103 JNZ ERROR
043.233 076 002 1104 SW.BRE1 MVI A,I,BRE IS /BREIF
043.235 062 243 063 1105 STA COMMAND
043.240 311 1106 RET
  
```

```

1108 ** SW.LIS - /LIST SWITCH DETECTED.
1109
043.241 072 243 063 1110 SW.LIS LDA COMMAND
043.244 247 1111 ANA A
043.245 312 254 043 1112 JZ SW.LIS1 NO FUNCTION
000.000 1113 ERRNZ I.BRE-2
000.000 1114 ERRNZ I.LIS-1
043.250 326 003 1115 SUI 3
043.252 077 1116 CMC
043.253 320 1117 RNC
043.254 076 001 1118 SW.LIS1 MVI A,I,LIS ALREADY HAVE ONE SPECIFIED, I.BRE OVERRULES /LIST
  
```

043.256 062 243 063 1119 STA COMAND  
043.261 311 1120 RET

1122 \*\* SW.VER - /VERSION SWITCH DETECTED  
1123

043.262 076 003 1124 SW.VER MVI A,I.VER  
043.264 303 150 043 1125 JMP SWIT1

1127 \*\* SW.MOU - /MOUNT SWITCH DETECTED  
1128

043.267 076 004 1129 SW.MOU MVI A,I.MOU  
043.271 303 150 043 1130 JMP SWIT1

```
1134 *** ACL = ACCEPT COMMAND LINE.  
1135 *  
1136 * ACL PROMPTS FOR AND READS A COMMAND LINE FROM  
1137 * THE CONSOLE.  
1138 *  
1139 * ENTRY NONE  
1140 * EXIT 'C' CLEAR; 'GOT' LINE  
1141 * 'LINE' = COMMAND LINE  
1142 * 'C' SET IF 'EOF'  
1143 * USES ALL  
1144  
043.274 315 036 057 1146 ACL CALL $GNL GUARANTEE NEW LINE  
043.277 315 136 031 1147 CALL $TYPTX  
000.000 1148 IF ,PIP,  
043.302 072 120 272 1149 DB 'P',',',+2000  
1150 ELSE ONECOPY  
1151 DB 'OC',',',+2000  
1152 ENDIF  
043.305 257 1153 XRA A  
043.306 062 326 040 1154 STA S,CSLMD CLEAR SPECIAL MODES  
043.311 041 034 065 1155 LXI H,LINE  
043.314 303 103 057 1156 JMP $RTL. READ UPPER CASE LINE AND EXIT
```

000.000

1159 IF ,PIP, PIP USES 'COPY'  
 1160 \*\*\* COPY - PROCESS COPY COMMAND.

1161 \*  
 1162 \* SYNTAX!  
 1163 \*

1164 \* DEST=SOURCE1,...,SOURCEN  
 1165 \*

1166 \* D'DEST' IS THE DESTINATION FILE DESIGNATOR. IF NULL  
 1167 \* (IN WHICH CASE THE '=' MAY BE OMITTED) IT DEFAULTS TO  
 1168 \* KB:PIFDEST.JGL  
 1169 \*

1170 \* THE 'SOURCE' FIELDS ARE THE SOURCE FILE DESIGNATORS. WILDCARDS  
 1171 \* MAY BE USED FOR FILE NAME AND EXTENSION.  
 1172 \* IF NO WILDCARDS ARE USED IN THE DESTINATION, MULTIPLE SOURCE FILES  
 1173 \* ARE CONCATINATED TOGETHER.  
 1174 \*

1175 \* IF WILDCARDS ARE PRESENT IN THE DESTINATION FILE DESCRIPTION,  
 1176 \* THE SOURCE FILES ARE COPIED TO INDIVIDUAL OUTPUT FILES. THE  
 1177 \* NAMES OF THE OUTPUT FILES ARE CREATED BY FILLING  
 1178 \* THE 'WILD' SPOTS IN THE DESTINATION NAME WITH THE CORRESPONDING  
 1179 \* CHARACTERS IN THE SOURCE NAME.  
 1180 \*

1181

043.317 1182 COPY EQU \*  
 043.317 257 1183 XRA A

043.320 042 347 044 1184 STA COPYC CLEAR FILE COUNT  
 043.323 315 264 053 1185 CALL DDF DECODE DESTINATION FILE

043.326 332 265 051 1186 JC ERROR ERROR  
 043.331 062 346 044 1187 STA COPYA SAVE DESTINATION TYPE

043.334 315 271 056 1188 CALL SDD RESET DEFAULT DEFAULTS  
 043.337 257 1189 XRA A ALLOW \*\*

043.340 315 092 053 1190 CALL BSL BUILD SOURCE FILE LIST  
 043.343 332 265 051 1191 JC ERROR

043.346 315 244 060 1192 CALL \$MOVEL  
 043.351 021 000 1193 DW COPYDL

043.353 305 063 1194 DW DESTFB+FB.NAM  
 043.355 350 044 1195 DW COPYD SAVE WILDCARD DESTINATION

1196 \*  
 1197 \* HAVE DESTINATION AND SOURCE FILE NAMES. DO THE COPYING.  
 1198 \*

1199 \* IF NO DESTINATION WILD CARDS, THUS COPIING TO A SINGLE OUTPUT  
 1200 \* FILE, OPEN THAT FILE NOW.  
 1201 \*

043.357 072 346 044 1202 LDA COPYA  
 043.362 247 1203 ANA A

043.363 312 003 044 1204 JZ COPY1 IS WILDCARDED  
 043.366 041 305 063 1205 LXI H,DESTFB+FB.NAM

043.371 076 001 1206 MVI A,CN,DES (A) = DESTINATION CHANNEL  
 043.373 377 043 1207 DB SYSCALL, OPENW OPEN IT

043.375 041 273 063 1208 LXI H,DESTFB  
 044.000 332 161 063 1209 JC \$FERROR IF ERROR

1210  
 1211 \* OPEN NEXT SOURCE FILE  
 1212 \*

044.003 052 326 063 1213 COPY1 LHLD NAMLLEN  
 044.006 174 1214 MOV A,H

```

044.007 265      1215      ORA      L
044.010 312 215 044 1216      JZ       COPY5      NO MORE INPUT FILES
044.013 041 347 044 1217      LXI      H,COPYC
044.016 064      1218      INR      M          COUNT FILE
044.017 041 154 065 1219      LXI      H,NAHTAB   (HL) = NAME ADDRESS
044.022 076 000      1220      MVI      A,CN,SOU   SOURCE CHANNEL
044.024 377 042      1221      DB       SYSCALL,OPENR OPEN FOR READ
044.026 332 046 051 1222      JC       NAMERR     IF ERROR
                    1223
                    1224 *      OPEN DESTINATION FILE IFF WILDCARDS
                    1225
044.031 072 346 044 1226      LDA      COPYA
044.034 247      1227      ANA      A
044.035 302 070 044 1228      JNZ      COPY2      NOT WILDCARDS
044.040 001 350 044 1229      LXI      B,COPYD   (BC) = WILDCARD PATTERN ADDRESS
044.043 021 154 065 1230      LXI      D,NAHTAB   (DE) = SOURCE NAME
044.046 041 305 063 1231      LXI      H,DESTFB+FB,NAM (HL) = RESULT AREA
044.051 345      1232      PUSH    H          SAVE POINTER TO RESULT AREA
044.052 315 147 056 1233      CALL    MWN        MERGE WILDCARD NAME
044.055 341      1234      POP     H          (HL) = #DESTFB+FB,NAM
044.056 076 001      1235      MVI      A,CN,DES
044.060 377 043      1236      DB       SYSCALL,OPENW
044.062 041 273 063 1237      LXI      H,DESTFB
044.065 332 161 063 1238      JC       $FERROR   CANT GET FILE OPEN
                    1239
                    1240 *      INPUT AND OUTPUT FILES OPEN, COPY
                    1241
044.070 315 354 054 1242 COPY2  CALL    EBM          EXPAND BUFFER TO MAX SIZE
044.073 052 271 063 1243 COPY3  LHL    BUFSIZ
044.076 104      1244      MOV     B,H
044.077 115      1245      MOV     C,L        (BC) = LENGTH OF BUFFER
044.100 052 267 063 1246      LHL    BUFPTR
044.103 353      1247      XCHG   (DE) = BUFFER FWA
044.104 076 000      1248      MVI      A,CN,SOU
044.106 325      1249      PUSH    D
044.107 377 004      1250      DB       SYSCALL,READ
044.111 321      1251      POP     D          (DE) = BUFFER FWA
044.112 365      1252      PUSH    PSW
044.113 322 127 044 1253      JNC     COPY4      GOT IT ALL
044.116 376 001      1254      CPI     EC,EOF
044.120 312 127 044 1255      JE      COPY4      IS EOF
044.123 361      1256      POP     PSW        RESTORE ERROR CODE
044.124 303 046 051 1257      JMP     NAMERR
                    1258
044.127 072 272 063 1259 COPY4  LDA      BUFSIZ+1   (A) = # OF SECTORS IN BUFFER
044.132 220      1260      SUB     B
044.133 107      1261      MOV     B,A        (B) = SECTORS READ
044.134 016 000      1262      MVI      C,0
044.136 076 001      1263      MVI      A,CN,DES
044.140 377 005      1264      DB       SYSCALL,WRITE WRITE IT OUT
044.142 041 273 063 1265      LXI      H,DESTFB
044.145 332 161 063 1266      JC       $FERROR   ERROR ON WRITE
044.150 361      1267      POP     PSW        (PSW) = STATUS FROM READ
044.151 322 073 044 1268      JNC     COPY3      NOT EOF
044.154 315 250 056 1269      CALL    SBE        SHRINK BUFFER TO MINIMUM SIZE
044.157 076 000      1270      MVI      A,CN,SOU

```

```

044.161 377 046 1271 DB SYSCALL,CLOSE CLOSE SOURCE
044.163 332 046 051 1272 JC NAMERR ERROR ON CLOSE
044.166 315 223 056 1273 CALL REN REMOVE ENTRY FROM NAMTAB
1274
1275 * IF DOING INDIVIDUAL FILE COPIES, CLOSE OUTPUT FILE.
1276
044.171 072 346 044 1277 LDA COPYA
044.174 247 1278 ANA A
044.175 302 003 044 1279 JNZ COPY1 CONCATINATING
044.200 076 001 1280 MVI A,CN.DES
044.202 377 046 1281 DB SYSCALL,CLOSE CLOSE DESTINATION
044.204 041 273 063 1282 LXI H,DESTFB
044.207 332 161 063 1283 JC $FERROR ERROR ON CLOSE
044.212 303 003 044 1284 JMP COPY1 GET NEXT FILE
1285
1286 ** ALL COPIES COMPLETE. CLOSE FILES AND CLEAN UP
1287
044.215 072 347 044 1288 COPY5 LDA COPYC
044.220 247 1289 ANA A
044.221 302 255 044 1290 JNZ COPY6
1291
1292 * NO FILES COPIED
1293
044.224 315 136 031 1294 CALL $TYPTX
044.227 007 116 157 1295 DB BELL,'No Files Copied',ENL
044.250 076 001 1296 MVI A,CN.DES
044.252 377 055 1297 DB SYSCALL,CLEAR CLEAR CHANNEL
044.254 311 1298 RET
1299
044.255 006 000 1300 COPY6 MVI B,0 (BC) = COUNT OF FILES COPIED
044.257 117 1301 MOV C,A
044.260 072 346 044 1302 LDA COPYA
044.263 247 1303 ANA A
044.264 312 303 044 1304 JZ COPY7 WILDCARDED
044.267 305 1305 PUSH B SAVE COUNT
044.270 076 001 1306 MVI A,CN.DES
044.272 377 046 1307 DB SYSCALL,CLOSE CLOSE DESTINATION
044.274 301 1308 POP B (BC) = FILES COPIED COUNT
044.275 041 273 063 1309 LXI H,DESTFB
044.300 332 161 063 1310 JC $FERROR ERROR ON CLOSE
1311
1312 * TYPE FILE COUNT
1313
044.303 072 246 063 1314 COPY7 LDA SUPRES
044.306 247 1315 ANA A
044.307 300 1316 RNZ SUPPRESS TRAIL MESSAGE
044.310 076 003 1317 MVI A,3
044.312 041 324 044 1318 LXI H,COPYE
044.315 315 171 060 1319 CALL $UDDN UNPACK COUNT INTO MESSAGE
044.320 315 136 031 1320 CALL $TYPTX
044.323 012 1321 DB NL
044.324 130 130 130 1322 COPYE DB 'XXX'
044.327 040 106 151 1323 DB ' Files Copied',ENL
044.345 311 1324 RET
1325
044.346 000 1326 COPYA DB 0 DESTINATION FILE WILDCARD FLAG (=0 IF WC)

```

044.347 000  
044.350  
000.021

1327 COPYC DB 0  
1328 COPYD DS FB,NAML  
1329 COPYDL EQU \*-COPYD

FILES COPIED COUNT  
HOLD AREA FOR WILDCARD DESTINATION

```
1332 *** MOUNT - MOUNT A NEW DISK
1333 *
1334 * MOUNT MOUNTS A NEW DISK ON THE SPECIFIED UNIT OF THE SELECTED
1335 * DEVICE.
1336 *
1337 * DEV:/MOUNTJ
1338 *
1339
044.371 1340 MOUNT EQU *
044.371 076 200 1341 MVI A,MOUNT
044.373 315 013 045 1342 CALL MDR. MOUNT/DISMOUNT/RESET
044.376 311 1343 RET
```



```
1347 *** DISMOUNT - DISMOUNT CURRENT DISK
1348 *
1349 * DISMOUNT DISMOUNTS THE CURRENT DISK ON THE SPECIFIED UNIT OF THE
1350 * SELECTED DEVICE.
1351 *
1352 * DEV:/DISMOUNT]
1353 *
1354
044.377 1355 DISMOUNT EQU *
044.377 076 201 1356 MVI A, DDMOUN
045.001 315 013 045 1357 CALL MDR. MOUNT/DISMOUNT/RESET
045.004 311 1358 RET
```

RESET - RESET CURRENT DISK

RESET

14:40:15 16-MAY-80

```

1362 *** RESET - RESET THE CURRENT DISK
1363 *
1364 * RESET RESETS THE SPECIFIED UNIT OF THE SELECTED DEVICE BY ISSUING
1365 * THE HDOS RESET CALL, WHICH IN TURN ISSUES A DISMOUNT AND MOUNT
1366 * ASKING THE USER TO OPEN THE DRIVE IN BETWEEN THE TWO.
1367 *
1368 * DEV://RESET]
1369 *
1370 *
045.005 1371 RESET EQU *
045.005 076 204 1372 MVI A,RESET
045.007 315 013 045 1373 CALL MDR. MOUNT/DISMOUNT/RESET
045.012 311 1374 RET

1376 ** MDR. - MOUNT/DISMOUNT/RESET
1377 *
1378 * MDR. PERFORMS THE SIMILAR FUNCTIONS OF MOUNT, DISMOUNT, AND RESET.
1379 *
1380 *
1381 * ENTRY (A) = SYSCALL CODE FOR OPERATION TO BE PERFORMED
1382 *
1383 * EXIT IF NO ERROR
1384 * TO CALLER
1385 * ELSE
1386 * TO ERROR
1387 *
1388 * USES ALL
1389 *
1390 *
045.013 062 044 045 1391 MDR. STA HDRA STORE SYSCALL VALUE
045.016 315 231 053 1392 CALL CTS CHECK FOR TARGET FILE SPECIFICATION
045.021 067 1393 STC
045.022 302 265 051 1394 JNZ ERROR THERE WAS A TARGET FILE
045.025 041 034 065 1395 LXI H,LINE
045.030 315 110 061 1396 CALL $DTB DELETE TRAILING BLANKS
045.033 376 001 1397 CFI I (A) = LINE LENGTH INCLUDING <00> BYTE
045.035 076 200 1398 MVI A,PEC,DF DEVICE FORMAT ERROR
045.037 312 265 051 1399 JZ ERROR NULL DEVICE IS ILLEGAL, ONLY BYTE IS NULL
045.042 345 1400 MDR1 PUSH H SAVE SPEC. ADDRESS FOR RETRY
045.043 377 000 1401 DB SYSCALL,0
045.044 1402 HDRA EQU *-1 SYSCALL VALUE
045.045 341 1403 POP H
045.046 320 1404 RNC NO ERROR
045.047 345 1405 PUSH H SAVE SPEC. ADDRESS
045.050 376 044 1406 CFI EC,NPM NO PROVISIONS MADE FOR REMOUNT
045.052 067 1407 STC
045.053 302 265 051 1408 JNZ ERROR ALL ERRORS BUT 'EC,NPM' CONSIDERED FATAL
045.056 076 000 1409 MVI A,DVLO
045.060 377 010 1410 DB SYSCALL,.LOAD0 LOAD *HDOSDVLO.SYS*
045.062 332 265 051 1411 JC ERROR
045.065 076 001 1412 MVI A,DVLI
045.067 377 010 1413 DB SYSCALL,.LOAD0 LOAD *HDOSDVLI.SYS*
045.071 332 265 051 1414 JC ERROR SYSCALL ERROR

```

```

045.074 341      1415      POP      H          RESTORE SPEC. ADDRESS
045.075 303 042 045 1416      JMP      MDR1       TRY AGAIN
1417      ELSE
1418      STL      'MOUNT - MOUNT A DIFFERENT DISK'
1419      EJECT
1420 MOUNT SPACE 4,10
1421 *** MOUNT - MOUNT A DIFFERENT DISK.
1422 *
1423 * MOUNT CAUSES A NEW DISK TO BE MOUNTED.
1424 *
1425 * INSERT THE DISK IN SY0, THEN TYPE
1426 *
1427 * /MOUNT
1428
1429
1430
1431 MOUNT EQU *
1432 LXI D,MOUNTA
1433 MVI B,377Q OFF PERIODS
1434 CALL MAD MOUNT ALTERNATE DISK
1435 RET
1436
1437 MOUNTA DB 244Q,306Q,307Q
1438 DB NL,'Insert New Disk',':'+200Q
1439 STL 'ONECOPY - COPY FILES BETWEEN VOLUMES.'
1440 EJECT
1441 ONECOPY SPACE 4,10
1442 *** ONECOPY - COPY FILES BETWEEN TWO VOLUMES, WITH ONLY ONE
1443 * DRIVE.
1444 *
1445 * (AND FOR MY NEXT TRICK...)
1446 *
1447 * ONECOPY COPIES FILES BETWEEN TWO VOLUMES BY ALTERNATING BETWEEN
1448 * TWO PHASES, THE READ PHASE AND THE WRITE PHASE.
1449 *
1450 * READ PHASE:
1451 *
1452 * DURING THE READ PHASE, THE SOURCE DISK IS MOUNTED. SOURCE FILES ARE
1453 * OPENED IN THE ORDER OF THEIR APPEARANCE. FOR EACH OPENED
1454 * FILE, A 'FILE DESCRIPTOR NODE' *FDN* IS ADDED TO THE ACTIVE
1455 * CHAIN. THEN, AS MUCH AS THE FILE AS POSSIBLE IS READ INTO MEMORY.
1456 *
1457 * THE PROCESS CONTINUES UNTIL
1458 * 1) THERE IS NO MORE FREE RAM
1459 * 2) OR, THERE ARE NO MORE FILE DESCRIPTOR NODES IN THE FREE CHAIN
1460 * 3) OR, THERE ARE NO MORE FILES IN NAMTAB (INPUT FILE LIST)
1461 *
1462 *
1463 * WRITE PHASE
1464 *
1465 * DURING THE WRITE PHASE, THE DESTINATION DISK IS MOUNTED. THE NODES
1466 * ARE TAKEN FROM THE ACTIVE CHAIN, AND PROCESSED. IF THE FILE HAD
1467 * BEEN PARTIALLY WRITTEN THE LAST PASS, IT IS RE-OPENED AND POSITIONED.
1468 * IF THERE IS NOT MORE DATA TO READ FOR A PROCESSED
1469 * NODE, IT IS REMOVED, AND THE CORRESPONDING ENTRY IN NAMTAB IS DELETED.
1470 *

```

MDR

```

1471 *      WRITE PHASE CONTINUES UNTIL
1472 *
1473 *      1) THERE ARE NO MORE FILE NODES IN THE ACTIVE LIST
1474 *      2) OR, THE FIRST (AND ONLY) ENTRY IN THE LIST HAS NO
1475 *      MORE DATA IN MEMORY, BUT HAS NOT BEEN COMPLETELY READ.
1476
1477
1478 OCOPY EQU *      CALLED OCOPY BY MAINLINE CODE
1479 OCOPY EQU *
1480 CALL IFL INITIALIZE FDN LISTS
1481 XRA A
1482 STA OCOPYC CLEAR FILE COUNT
1483 STA VOLFLAG FLAG SOURCE VOLUME MOUNTED
1484 LDA D,DRVTR+1
1485 STA VOLSER SET VOLUME SERIAL NUMBER
1486 CALL DDF DECODE DESTINATION FILE
1487 JC ERROR ERROR
1488 STA OCOPYA SAVE DESTINATION TYPE
1489 CALL SDD RESET DEFAULT DEFAULTS
1490 XRA A ALLOW *.*
1491 CALL BSL BUILD SOURCE FILE LIST
1492 JC ERROR
1493 CALL $MOVE
1494 DW OCOPYDL
1495 DW DESTFB+FB.NAM
1496 DW OCOPYD SAVE WILDCARD DESTINATION
1497 CALL EBM EXPAND BUFFER TO MAX
1498
1499 *      MAKE SURE HE'S NOT TRYING TO CONCATINATE
1500
1501 LDA OCOPYA
1502 ANA A
1503 JZ OCOPY1 HAVE WILDCARDS
1504 LHLD NAMLEN NO WILDCARDS, ONLY LET HIM SPECIFY ONE SOURCE
1505 LXI D,-FB.NAM
1506 DAD D
1507 MOV A,H
1508 ORA L
1509 MVI A,PEC.FCI FILE CONCATINATION IS ILLEGAL
1510 JNZ ERROR
1511
1512 *      START READ PHASE
1513
1514 OCOPY1 LDA BUFR+1 (A) = BUFFER FWA/256
1515 INR A ROUND UP TO NEXT PAGE
1516 STA OBUFR SET SECTOR BUFFER FWA/256
1517 LDA VOLFLAG
1518 ANA A
1519 JZ OCOPY2 SOURCE IS MOUNTED
1520 LXI D,OCOPYF
1521 MOV B,A (B) = 3770 = PERIODS MASK
1522 CALL MAD MOUNT ALTERNATE DISK
1523 OCOPY2 CALL RPH READ PHASE
1524 LDA FDNHEAD
1525 ANA A
1526 JZ OCOPY6 NO FILES ARE READ, ERGO NONE ARE LEFT

```

```

1527 LDA VOLFLAG
1528 ANA A
1529 JNZ OCOPY3
1530 MVI B,1770 (B) = PERIODS MASK
1531 LXI H,OCOPY6
1532 CALL MAD MOUNT ALTERNATE DISK
1533 OCOPY3 CALL WPH WRITE PHASE
1534 JMP OCOPY1
1535
1536 * ALL DONE, FINISH MESSAGE
1537
1538 OCOPY6 LDA OCOPYC (A) = FILE COUNT
1539 MVI B,0 (BC) = COUNT OF FILES COPIED
1540 MOV C,A
1541
1542 * TYPE FILE COUNT
1543
1544 MVI A,3
1545 LXI H,OCOPYE
1546 CALL $UDDN UNPACK COUNT INTO MESSAGE
1547 CALL $TYPTX
1548 OCOPYE DB 'XXX'
1549 DB ' Files Copied',ENL
1550 RET
1551
1552 OCOPYA DB 0 DESTINATION FILE WILDCARD FLAG (=0 IF WC)
1553 OCOPYC DB 0 FILES COPIED COUNT
1554 OCOPYD DB FB.NAML HOLD AREA FOR WILDCARD DESTINATION
1555 OCOPYDL EQU *-OCOPYD
1556 OCOPYF DB 2440,3060,3070
1557 DB NL,'Insert Source',':'+2000
1558 OCOPYG DB 1020,0140,440
1559 DB NL,'Insert Destination',':'+2000
1560 STL 'ONECOPY SUBROUTINES'
1561 EJECT
1562 RPH SPACE 4,10
1563 ** RPH - READ PHASE.
1564 *
1565 * RPH HANDLES THE READ PHASE OF THE COPY PROCESS.
1566 *
1567 * IT IS ENTERED WITH THE NAMTAB AND FDN TABLE SETUP, AND
1568 * WITH THE SOURCE DISK MOUNTED.
1569 *
1570 * READ PHASE:
1571 *
1572 * DURING THE READ PHASE, THE SOURCE DISK IS MOUNTED. SOURCE FILES ARE
1573 * OPENED IN THE ORDER OF THEIR APPEARANCE. FOR EACH OPENED
1574 * FILE, A 'FILE DESCRIPTOR NODE' *FDN* IS ADDED TO THE ACTIVE
1575 * CHAIN. THEN, AS MUCH AS THE FILE AS POSSIBLE IS READ INTO MEMORY.
1576 *
1577 * THE PROCESS CONTINUES UNTIL
1578 * 1) THERE IS NO MORE FREE RAM
1579 * 2) OR, THERE ARE NO MORE FILE DESCRIPTOR NODES IN THE 'FREE' CHAIN
1580 * 3) OR, THERE ARE NO MORE FILES IN NAMTAB (INPUT FILE LIST)
1581 *
1582 * ENTRY NONE
  
```

```

1583 *      EXIT      NONE
1584 *      USES      ALL
1585
1586
1587 RPH     EQU      *
1588
1589
1590 *      SEE IF ANY MEMORY TO HAVE
1591
1592      CALL      CBR          COMPUTE BUFFER ROOM
1593      RZ              NONE
1594
1595 *      SEE IF WE NEED TO READ SOME MORE INTO A PART-COPIED FILE
1596
1597      LXI      H,FDNHEAD
1598      MOV      L,M          (HL) = ADDRESS IF FIRST NODE
1599      MOV      A,L
1600      ANA      A
1601      JZ       RPH1        IS NO FIRST NODE; ERGO NO FILE
1602      INX      H
1603      ERRNZ   FDN,STA-1
1604      MOV      A,M          (A) = .STA
1605      ANI      ST.OPK
1606      LXI      D,NAMTAB
1607      JNZ     RPH2.S      FILE IS INCOMPLETELY READ
1608
1609 *      SEE IF ANY FREE FILE DESCRIPTOR NODES TO USE
1610
1611 RPH1     LDA      FDNFRE
1612      ANA      A
1613      RZ              NO MORE
1614
1615 *      SEE IF THERE IS A FILE IN NAMTAB WITHOUT AN ENTRY IN FDNLIST;
1616 *      SINCE THE FIRST ENTRY IN FDNLIST CORRESPONDS TO THE FIRST IN
1617 *      NAMTAB, ETC., WE'LL JUST RUN DOWN FDNLIST UNTIL THE END, AND
1618 *      THE NEXT NAMTAB FILE WILL BE THE ONE WE WANT..
1619
1620      LXI      B,FB.NAML   (BC) = ENTRY SIZE IN NAMTAB
1621      LXI      D,-FB.NAML  (DE) = POINTER INTO NAMTAB
1622      LXI      H,FDNHEAD
1623      MOV      A,L          START WITH FDNHEAD
1624 RPH2     MOV      L,A      FOLLOW LINK
1625      MOV      A,M          (A) = NEXT NODE
1626      XCHG
1627      DAD      B            ADVANCE POINTER INTO NAMTAB
1628      XCHG
1629      ANA      A
1630      JNZ     RPH2        LINK SOME MORE
1631      PUSH   H            (HL) = ADDRESS OF LAST NODE
1632      LHL    NAMTLEN
1633      CALL   $CDEHL      SEE IF HAVE ACCOUNTED FOR ALL NAMTAB ENTRIES
1634      POP    H
1635      RE              FILES ALL USED UP
1636
1637 *      HAVE ROOM FOR DATA, HAVE A NODE FOR THE FILE COUNTS, AND
1638 *      HAVE A FILE NAME. ALL SET FOR BUSINESS..

```

```

1639 *
1640 * (DE) = INDEX INTO NAMTAB FOR FILE
1641 * (HL) = NODE ADDRESS OF LAST ENTRY IN LIST
1642 *
1643 * CHAIN THE FIRST FREE NODE ONTO THE END OF THE LIST
1644
1645 LDA FDNFRE
1646 MOV M,A CHAIN TO NEW END NODE
1647 MOV L,A
1648 MOV A,M (A) = NEXT NODE IN FREE CHAIN
1649 STA FDNFRE
1650 MVI B,FDNELEN
1651 PUSH H SAVE NODE ADDRESS
1652 CALL $ZERO ZERO ENTIRE NODE, INCLUDING CHAIN (AT END, NOW)
1653 LXI B,NAMTAB
1654 XCHG
1655 DAD B (HL) = ADDRESS OF NAMTAB ENTRY
1656 SHLD NAMTPTR POINTER TO CURRENT NAMTAB ENTRY
1657 XCHG
1658 POP H
1659 ERRNZ FDN.STA-1
1660 INX H (HL) = ADDR OF FDN.STA OF NODE
1661
1662 * READY TO OPEN FILE
1663 *
1664 * (DE) = NAMTAB ENTRY ADDRESS
1665 * (HL) = $FDN.STA OF ENTRY
1666
1667 RPH2.5 PUSH H SAVE ADDRESS
1668 XCHG
1669 XRA A
1670 ERRNZ CN.SOU (A) = SOURCE CHANNEL NUMBER
1671 DB SYSCALL, .OPENR OPEN
1672 JC NAMERR ERROR
1673 POP D
1674 LDAX D (A) = FDN.STA
1675 ANI ST.OPR
1676 PUSH D SAVE ADDRESS
1677 JNZ RPH3 ALREADY OPENED IN PREVIOUS PASSES
1678
1679 * FIRST TIME THIS FILE HAS BEEN OPENED. SEE IF CONTIGUOUS
1680
1681 PUSH H
1682 LXI H,OCOPYC
1683 INR M
1684 POP H
1685 LDAX D
1686 ORI ST.OPR SET OPEN FOR READ
1687 STAX D
1688 LHLD S.CFWA (HL) = CHANNEL 0 FWA
1689 ERRNZ IOCCTD-1 WE NEED TO CHAIN ONE TO GET TO USER #0
1690 CALL $HLIHL
1691 ERRNZ CN.SOU ASSUME WE WANT CHANNEL 0
1692 CALL $INDL
1693 DW IOC.DIR+DIR.FLG
1694 MOV A,E (A) = DIR.FLG

```

```

1695 ANI 0 DIF.CNT * * PATCH * *
1696 JZ RPH3 NOT CONTIG
1697
1698 * IS CONTIG, GET FILE SIZE
1699
1700 CALL $INDL
1701 DW IOC.GRT
1702 PUSH D SAVE GRT ADDRESS
1703 CALL $INDL
1704 DW IOC,DIR+DIR,FON.(E) = DIR,FON
1705 MOV A,E
1706 POP H (HL) = GRT TABLE ADDRESS
1707 CALL CFS. COMPUTE BLOCK SIZE
1708 POP H (HL) = ADDRESS OF FON,STA
1709 PUSH H
1710 MOV A,M (A) = FON,STA
1711 ORI ST,CNT FLAG CONTIG
1712 MOV M,A
1713 INX H
1714 ERNZ FON,SIZ-FON,STA-1
1715 MOV M,E SET BLOCK COUNT
1716
1717 * READY TO READ DATA. POSITION FILE (IN CASE SOME WAS READ IN
1718 * PREVIOUS PASSES), AND COMPUTE THE MAX POSSIBLE READ COUNT
1719 *
1720 * ((SP)) = ADDRESS OF FON,STA FOR NODE
1721
1722 RPH3 POP H (HL) = ADDRESS OF FON,STA
1723 PUSH H
1724 CALL $INDL
1725 DW FON,AMR-FON,STA (DE) = AMOUNT READ (IN SECTORS)
1726 MOV B,D
1727 MOV C,E (BC) = AMOUNT READ
1728 MVI A,CN,SOU
1729 DB SYSCALL,.POSIT POSIT
1730 JC IERR3 POSIT BLEW UP
1731 CALL CBR COMPUTE BUFFER ROOM
1732 XCHG (D) = POINTER/256, (E) = LIMIT/256
1733 POP H (HL) = #FON,STA
1734 LXI B,FON,ADR-FON,STA
1735 DAD B (HL) = #FON,ADR
1736 MOV M,D SET ADDRESS/256
1737 PUSH H SAVE #FON,ADR
1738 MVI E,0 (DE) = ADDRESS
1739 MOV B,A (B) = SECTORS OF RAM AVAILABLE
1740 MOV C,E (C) = 0
1741 PUSH B SAVE TRY COUNT
1742 MVI A,CN,SOU
1743 DB SYSCALL,.READ READ THE STUFF
1744
1745 * COMPUTE THE AMOUNT READ (IN CASE OF EOF)
1746
1747 POP B (DE) = TRY COUNT
1748 JNC RPH4 GOT ALL WE TRYED
1749 CPI EC,EOF
1750 JNE NAMERR NOT JUST EOF, GOT TROUBLES
  
```



```

1751      MOV      A,D
1752      SUB      B          REMOVE AMOUNT WE DIDNT GET
1753      MOV      D,A
1754      POP      H          (HL) = #FDN.ADR
1755      PUSH     H
1756      LXI      B,FDN.STA-FDN.ADR
1757      DAD      B
1758      MOV      A,M          (A) = FIN.STA
1759      ANI      3770-ST.DFR   EOF, NOT OPEN FOR READ ANYMORE
1760      MOV      M,A          POST READ COMPLETE FOR THIS GUY
1761
1762 *      STORE RESULTS OF READ IN NODE
1763 *
1764 *      (D) = SECTORS READ
1765 *      ((SP)) = #FDN.ADR
1766
1767 RPH4   POP      H          (HL) = #FDN.ADR
1768      INX      H
1769      ERRCZ    FDN.AIM-FDN.ADR-1 (HL) = ADDRESS IF AMOUNT IN MEMORY BYTE
1770      MOV      M,D          STORE SECTORS IN MEMORY COUNT
1771      LXI      B,FIN.AMR-FDN.AIM
1772      DAD      B          (HL) = #FIN.AMR (AMOUNT READ)
1773      MOV      A,M          (A) = AMOUNT READ BEFORE
1774      ADD      D          ADD NEW AMOUNT
1775      MOV      M,A
1776      INX      H
1777      MOV      A,M
1778      ACI      0          PROPAGATE FOR VERY LARGE FILES
1779      MOV      M,A
1780      LXI      H,ORUFPTR
1781      MOV      A,M
1782      ADD      D          ADVANCE FREE RAM POINTER BY AMOUNT READ
1783      MOV      M,A
1784      MVI      A,CN.SOU
1785      DB      SYSCALL,CLOSE   CLOSE FILE
1786      JMP      RPH          SEE IF MORE TO READ
1787 WPH   SPACE   4,10
1788 **   WPH - WRITE PHASE.
1789 *
1790 *      WPH HANDLES THE WRITE PHASE PROCESSING. IT IS ENTERED WITH
1791 *      THE FDN CHAIN SETUP, THE NAMTAB SETUP, AND
1792 *      THE DESTINATION DISK MOUNTED.
1793 *
1794 *
1795 *      WRITE PHASE
1796 *
1797 *      DURING THE WRITE PHASE, THE DESTINATION DISK IS MOUNTED. THE NODES
1798 *      ARE TAKEN FROM THE ACTIVE CHAIN, AND PROCESSED. IF THE FILE HAD
1799 *      BEEN PARTIALLY WRITTEN THE LAST PASS, IT IS RE-OPENED AND POSITIONED.
1800 *      IF THERE IS NOT MORE DATA TO READ FOR A PROCESSED
1801 *      NODE, IT IS REMOVED, AND THE CORRESPONDING ENTRY IN NAMTAB IS DELETED.
1802 *
1803 *      WRITE PHASE CONTINUES UNTIL
1804 *
1805 *      1) THERE ARE NO MORE FILE NODES IN THE ACTIVE LIST
1806 *      2) OR, THE FIRST (AND ONLY) ENTRY IN THE LIST HAS NO
  
```

```

1807 *          MORE DATA IN MEMORY, BUT HAS NOT BEEN COMPLETELY READ.
1808 *
1809 *          ENTRY  NONE
1810 *          EXIT   NONE
1811 *          USES   ALL
1812
1813
1814 WPH      EQU    *
1815
1816 *          SEE IF MORE TO WRITE
1817
1818          LXI    H,FDNHEAD
1819          MOV    L,M
1820          MOV    A,L          (A) = FIRST NODE INDEX
1821          ANA   A
1822          RZ              NO MORE
1823          CALL  $INCL
1824          DW    FDN.AIM      (E) = AMOUNT IN MEMORY FOR THIS GUY
1825          MOV    A,E
1826          ANA   A
1827          JNZ   WPHO        GOT DATA
1828
1829 *          NO DATA IN NODE. IF STILL READING, RETURN FOR MORE
1830
1831          INX    H
1832          MOV    A,M
1833          DCX   H
1834          ANI   ST.OPR
1835          RNZ              STILL READING, GET MORE
1836          XCHG          (DE) = ADDRESS
1837          JMP   WPH4        REMOVE NODE, AM DONE WITH FILE
1838
1839 *          HAVE DATA TO WRITE. SEE IF WE HAVE OPENED THIS FILE BEFORE.,
1840 *          OR IF THIS IS THE FIRST TIME
1841
1842 WPHO     PUSH   H          SAVE NODE POINTER
1843          INX    H
1844          ERRNZ  FDN.STA-1
1845          MOV    A,M          (A) = FDN.STA
1846          ANI   ST.OPW
1847          JNZ   WPH2        OPENED BEFORE
1848          ERRNZ  ST.OPW-1
1849          INR    H          SET '1' BIT
1850
1851 *          BUILD NAME INTO DESTFB
1852
1853          PUSH   H          SAVE NODE ADDRESS
1854          LXI    B,DCOPYD
1855          LXI    D,NAMTAB
1856          LXI    H,DESTFB+FB.NAM
1857          CALL  MNV          MERGE WILDCARD NAME
1858          POP    H
1859
1860 *          IS 1ST TIME FOR THIS FILE. IF CONTIGUOUS FLAG, OPEN THE FILE
1861 *          FOR CONTIGUOUS
1862

```

```

1863      MOV      A,H          (A) = FLAG BYTE
1864      ANI      ST.CNT
1865      JNZ      WPH1          IS CONTIG
1866      LXI      H,DESTFB+FB.NAM
1867      MOVI     A,CN.DES
1868      DB       SYSCALL,.OPENW  JUST OPEN FOR WRITE
1869      JC       DESTERR      ERROR
1870      JMP      WPH3          WRITE THE DATA
1871
1872      *          IS CONTIG FILE, OPEN IN CONTIG MODE
1873
1874      WPH1     INX      H
1875      ERRNZ    FDN.SIZ-FDN.STA-1
1876      MOV      C,H          (C) = COUNT (IN BLOCKS)
1877      MVI      B,0
1878      LXI      H,DESTFB+FB.NAM
1879      MOVI     A,CN.DES
1880      PUSH     B            SAVE COUNT
1881      DB       SYSCALL,.DELET  DELETE OLD ONE
1882      JNC     WPH1.5      DELETED
1883      CPI      EC.FNF
1884      JNE     ERROR      MUST BE WRITE PROTECTED, OR SOMETHING...
1885      WPH1.5  POP      B            (BC) = COUNT
1886      LXI      H,DESTFB+FB.NAM
1887      MOVI     A,CN.DES
1888      DB       SYSCALL,.OPENC  OPEN CONTIG
1889      JC       DESTERR
1890      JMP      WPH3
1891
1892      *          THIS FILE HAS ALREADY BEEN PARTIALLY WRITTEN, OPEN IN UPDATE MODE
1893      *          SO WE CAN EXTEND IT.
1894
1895      WPH2     LXI      H,DESTFB+FB.NAM
1896      MOVI     A,CN.DES
1897      DB       SYSCALL,.OPENU  OPEN FOR UPDATE
1898      JC       DESTERR      PROBLEMS
1899      POP      H
1900      PUSH     H            (HL) = #FDN.STA
1901      CALL     $INDL
1902      DW       FDN.AMW      (DE) = AMOUNT WRITTEN
1903      MOV      B,D
1904      MOV      C,E          (BC) = SECTORS WRITTEN
1905      MOVI     A,CN.DES
1906      DB       SYSCALL,.POSIT  POSITION FOR EXTEND
1907      JC       IERR1      COULDN'T GET THERE!
1908
1909      *          FILE OPEN AND POSITIONED, WRITE DATA
1910
1911      WPH3     POP      H
1912      PUSH     H            (HL) = #FDN.LNK
1913      CALL     $INDL
1914      DW       FDN.ADR      (E) = ADDR/256, (D) = CNT/256
1915      MOV      B,D
1916      MOV      D,E
1917      MOVI     E,0          (DE) = ADDRESS
1918      MOV      C,E          (BC) = COUNT
  
```

RESET - RESET CURRENT DISK

HDR

14:40:20 16-MAY-80

```

1919      MVI      A,CN,DES
1920      PUSH     B          SAVE WRITE COUNT
1921      DB       SYSCALL,,WRITE WRITE IT
1922      JC       DESTERR   PROBABLY OUT OF ROOM
1923      MVI      A,CN,DES
1924      DB       SYSCALL,,CLOSE CLOSE IT
1925      JC       DESTERR
1926      POP      B          (B) = SECTORS WRITTEN
1927      POP      H
1928      PUSH     H          (HL) = #FDN,LNK
1929      LXI      D,FDN,AMW-FDN,LNK
1930      DAD     D          (HL) = FDN,AMW
1931      MOV      A,M
1932      ADD     B
1933      MOV      M,A
1934      INX     H
1935      MOV      A,M
1936      ACI     0          INCREMENT AMOUNT WRITTEN
1937      MOV      M,A
1938
1939 *      CLEAR 'IN MEMORY' COUNT IN NODE. IF THE FILE HAS NO MORE TO
1940 *      READ, REMOVE IT FROM THE CHAIN AND NANTAB
1941
1942      POP      D          (DE) = FDN,LNK
1943      WPH4     LXI      H,FDN,AM
1944      DAD     D
1945      MVI      M,0       CLEAR AMOUNT IN MEMORY
1946      XCHG    (HL) = FDN,LNK
1947      INX     H
1948      ERNZ    FDN,STA-FDN,LNK-1
1949      MOV      A,M          (A) = FDN,STA
1950      ANI     ST,OPR
1951      RNZ
1952      STILL READING, AM DONE FOR THIS PHASE
1953 *      UNLINK NODE FROM LIST
1954
1955      DCX     H
1956      MOV      A,M
1957      STA     FDNHEAD   UNLINK FROM ACTIVE LIST
1958      LDA     FDNFRE
1959      MOV      M,A       PUT THIS GUY ON HEAD OF FREE LIST
1960      MOV      A,L
1961      STA     FDNFRE
1962      CALL    REN       REMOVE ENTRY FROM NANTAB
1963      JMP     WPH       TRY TO WRITE THE NEXT GUY
1964      CBR
1965 **     CBR - COMPUTE BUFFER ROOM.
1966 *
1967 *      CBR COMPUTES THE NUMBER OF SECTORS WORTH OF RAM
1968 *      STILL FREE.
1969 *
1970 *      ENTRY  NONE
1971 *      EXIT   (A) = SECTORS OF RAM FREE
1972 *           (%Z) SET IFF (A) = 0
1973 *           (H) = BUFPTR/256
1974 *           (L) = OBUFLIM/256

```

```

1975 *      USES      A,F
1976
1977
1978 CBR     LHL     OBUFLIM
1979         ERRNZ   OBUFPTR-OBUFLIM-I
1980         MOV     A,L
1981         SUB     H
1982         RET
1983 IFL     SPACE   4,10
1984 **      IFL - INITIALIZE FDN LIST.
1985 *
1986 *      IFL CHAINS ALL THE FDN NODES TO THE FREE LIST. THIS
1987 *      CLEANUP IS NECESSARY IN CASE A CTL-C OR SOMETHING
1988 *      LEFT THE LIST GARRAGED.
1989 *
1990 *      ENTRY     NONE
1991 *      EXIT      NONE
1992 *      USES     ALL
1993
1994
1995 IFL     LXI     H,FDN.1
1996         MOV     A,L          (A) = FIRST LINK
1997         STA     FDNFRE
1998         XRA     A
1999         STA     FDNHEAD     NONE IN LIST
2000         MVI     B,FDNCNT-1  (B) = NUMBER OF NODES-1
2001 IFL1   MVI     A,FDNELEN
2002         ADD     L          (A) = #ADDR OF NEXT NODE
2003         MOV     M,A        SET LINK
2004         MOV     L,A        FORWARD TO NEXT LINK
2005         DCR     B
2006         JNZ     IFL1      MORE TO GO
2007         MVI     M,0       LAST ONE CHAINS NOWHERE
2008         RET
2009 MAD     SPACE   4,10
2010 **      MAD - MOUNT ALTERNATE DISK.
2011 *
2012 *      MAD DISMOUNTS THE CURRENT DISK, HAS THE USER INSERT THE
2013 *      OTHER DISK, AND MOUNTS IT.
2014 *
2015 *      ENTRY     (B) = FRONT PANEL LED PATTERN
2016 *              (DE) = PROMPT PATTERNS FOR PANEL AND CONSOLE
2017 *      EXIT     (HL) = #VOLFLAG
2018 *      USES     ALL
2019
2020
2021 MAD     EQU     *
2022
2023 *      DISMOUNT CURRENT DISK
2024
2025         PUSH    D
2026         PUSH    B          SAVE ENTRY PARAMETERS IN CASE OF RETRY
2027         PUSH    D
2028         PUSH    B          SAVE ENTRY PARAMETERS OVER SYDD CALL
2029         LXI     H,MNDA     DEVICE SPECIFICATION
2030         DB     SYSCALL,DMNMS DISMOUNT WITHOUT MESSAGE

```

```

2031          JC      ERROR          IF ERROR
2032
2033 *        SETUP PROMPT ON FF LEDS AND CONSOLE FOR NEW DISK
2034
2035 MAD0      BI
2036          LXI     H,D:PLYMD
2037          MOV     A,M
2038          ANA     A
2039          JZ      MAD1          DISK ALREADY STOPPED
2040          MVI     M,1          STOP DISK VERY SOON
2041 MAD1      EI
2042          MVI     A,UO.DDU+UO.CLK+UO.HLT
2043          STA     ,MFLAG        HALT DISPLAY UPDATE
2044          LXI     H,,ALEDS
2045          MVI     A,9
2046          POP     B              (B) = PERIOD PATTERN
2047 MAD2      MOV     M,B          SET PATTERN
2048          INX     H
2049          DCR     A
2050          JNZ     MAD2          IF MORE TO BLANK
2051          LXI     H,,ALEDS+3
2052          LXI     B,3
2053          POP     D              (DE) = PROMPT LIST
2054          CALL    $MOVE          MOVE IN PROMPT PATTERN
2055          XCHG                    (HL) = PATTERN
2056          DB      SYSCALL,,PRINT CONSOLE PROMPT
2057          CALL    $TYPTX
2058          DB      BELL+200R      BEEP CONSOLE, TOO
2059          MVI     A,100
2060          CALL    ,HORN          BEEP A WARNING
2061
2062 *        WAIT FOR SIGNAL THAT NEW DISK IS IN
2063
2064 MAD3      DB      SYSCALL,,SCIN
2065          JNC     MAD4          GOT A CHARACTER
2066          IN      IP.PAD
2067          INR     A
2068          JZ      MAD3          NO REPLY THERE, EITHER
2069
2070 *        GOT REPLY. GOBBLE EXTRA CHARACTERS FROM CONSOLE
2071
2072 MAD4      DB      SYSCALL,,SCIN
2073          JNC     MAD4
2074
2075 *        READ NEW DISK'S LABEL
2076
2077          CALL    GETLAB
2078          JC      ERROR
2079
2080 *        SEE IF LABEL CHANGED FROM BEFORE
2081
2082          POP     B
2083          POP     D              RESTORE ENTRY PARAMETERS
2084          LXI     H,VOLSER
2085          LDA     LABEL+LAB,SER
2086          CMP     M
  
```

MDR.

```

2087 JNE MAD4.5 IS THE RIGHT DISK
2088 PUSH D SAVE PARAMS AS IN BEGINNING
2089 PUSH B
2090 PUSH D SAVE FOR RETRY
2091 PUSH B
2092 JMP MAD0 IT WAS NOT THE RIGHT DISK
2093
2094 MAD4.5 MOV M,A SET NEW SERIAL
2095 LXI H,VOLFLAG
2096 MOV A,M
2097 CMA
2098 MOV M,A COMPLEMENT VOLUME FLAG
2099
2100 * ERASE FRONT PANEL DISPLAY
2101
2102 LXI H,ALEDS
2103 MVI A,9
2104 MAD5 MOV M,R SET TO PATTERN
2105 INX H
2106 BCR A
2107 JNZ MAD5
2108 CALL MND MOUNT NEW DISK
2109 RET
2110 MND SPACE 4,10
2111 ** MND - MOUNT NEW DISK
2112 *
2113 * MOUNT NEW DISK ONTO DEVICE SPECIFIED IN MNDA
2114 *
2115 *
2116 * ENTRY NONE
2117 *
2118 * EXIT LABEL = LABEL SECTOR
2119 *
2120 * USES ALL
2121 *
2122
2123 MND LXI H,MNDA
2124 DB SYSCALL,,MDNMS MOUNT WITHOUT MESSAGE
2125 JC ERROR IF ERROR IN MOUNT
2126 CALL GETLAB GET LABEL
2127 RET
2128
2129 MNDA DB 'SY0:0'
2130 GETLAB SPACE 4,10
2131 ** GETLAB - GET LABEL
2132 *
2133 * GETLAB READS THE DISK LABEL
2134 *
2135 * ENTRY NONE
2136 *
2137 * EXIT LABEL IN LABEL
2138 * (PSM) = 'C' CLEAR IF NO ERROR
2139 * = 'C' SET IF ERROR
2140 * (A) = ERROR CODE
2141 *
2142 * USES ALL
  
```

```
2143 *  
2144  
2145 GETLAB LXI H,DDF.LAB  
2146 LXI D,LABEL  
2147 LXI B,256  
2148 CALL $WER WRITE ENABLE RAM  
2149 MVI A,DC.REK  
2150 CALL SYDD  
2151 RET  
2152 ENDIF
```



```

2155 *** DELETE - PROCESS DELETE COMMAND.
2156 *
2157 * SYNTAX:
2158 *
2159 * SOURCE1,...,SOURCE N/DELETE
2160 *
2161 * AT LEAST ONE SOURCE FILE MUST BE SPECIFIED.
2162 * IF *.* IS SPECIFIED, DELETE ASKS,
2163 * DELETE ALL 'Y'? ARE YOU SURE?
2164 *
2165 *
000.000 2166 IF .PIP.
045.100 2167 DELETE EQU *
045.100 041 034 065 2168 LXI H,LINE
2169 *
2170 * SEE IF A DESTINATION FILE SPECIFIED
2171 *
045.103 176 2172 DEL1 MOV A,M
045.104 043 2173 INX R
045.105 247 2174 ANA A
045.106 312 123 045 2175 JZ DEL2 END OF LINE
045.111 376 075 2176 CFI '='
045.113 302 103 045 2177 JNE DEL1
2178 *
2179 * HE SPECIFIED A DESTINATION FILE
2180 *
045.116 076 203 2181 MVI A,PEC.TFI TARGET FILE ILLEGAL
045.120 303 265 051 2182 JMP ERROR FORMAT ERROR
2183 *
2184 * NO TARGET FILE SPECIFIED
2185 *
045.123 076 001 2186 DEL2 MVI A,1 CHECK FOR *.*
045.125 315 002 053 2187 CALL BSL BUILD SOURCE FILE LIST
045.130 332 265 051 2188 JC ERROR NO GOOD
2189 *
2190 * DELETE FILES ONE BY ONE
2191 *
045.133 052 326 063 2192 DEL5 LHLD NAMTLEN
045.136 174 2193 MOV A,H
045.137 265 2194 ORA L
045.140 310 2195 RZ END OF LIST
045.141 041 154 065 2196 LXI H,NAMTAB
045.144 377 050 2197 DB SYSCALL,DELETE REMOVE IT
045.146 332 046 051 2198 JC NAMERR ERROR ON DELETE
045.151 315 223 056 2199 CALL REN REMOVE ENTRY FROM NAMTAB
045.154 303 133 045 2200 JMP DEL5 DELETE THE NEXT ONE

```

```

2203 *** RENAME - RENAME FILES.
2204 *
2205 * SYNTAX:
2206 *
2207 * DEST = SOURCE1,...,SOURCEN
2208 *
2209 * RENAME IS PROCESSED IN A MANNER SIMILAR TO COPY, EXCEPT THAT THE
2210 * FILE IS RENAMED, RATHER THAN COPIED.
2211 *
2212 *
045.157 2213 RENAME EQU *
045.157 315 264 053 2214 CALL DDF DECODE DESTINATION FILE
045.162 332 265 051 2215 JC ERROR
045.165 257 2216 XRA A ALLOW *.*
045.166 315 002 053 2217 CALL BSL BUILD SOURCEFILE LIST
045.171 332 265 051 2218 JC ERROR
2219 *
2220 * DO MULTIPLE RENAMES
2221 *
045.174 001 305 063 2222 REN1 LXI B,DESTFB+FB.NAM (BC) = WILDCARDED TARGET NAME
045.177 021 154 065 2223 LXI D,NAMTAB (DE) = NORMAL SOURCE NAME
045.202 041 303 045 2224 LXI H,RENA (HL) = BUFFER FOR RESULT NAME
045.205 305 2225 PUSH B SAVE #DESTFB+FB.NAM
045.206 325 2226 PUSH D SAVE #NAMTAB
045.207 315 147 056 2227 CALL MUM MERGE WILDCARD NAME
045.212 321 2228 POP D (DE) = #NAMTAB
045.213 341 2229 POP H (HL) = #DESTFB+FB.NAM
2230 *
2231 *
2232 * SEE IF SOURCE AND DEST FILE ON SAME DEVICE
2233 *
045.214 325 2234 PUSH D SAVE #NAMTAB (SOURCE NAME)
045.215 016 003 2235 MVI C,3
045.217 315 060 030 2236 CALL $COMP COMPARE DEVICES
045.222 076 201 2237 MVI A,$EC.DNC DEVICES NOT CONSISTANT
045.224 302 265 051 2238 JNE ERROR
2239 *
2240 * SEE IF TARGET ALREADY EXISTS
2241 *
045.227 041 303 045 2242 LXI H,RENA
045.232 076 000 2243 MVI A,CN.SOU
045.234 377 042 2244 DB SYSCALL,$OPENR
045.236 041 271 045 2245 LXI H,RENA+FB.NAM
045.241 332 251 045 2246 JC REN2 HAVE AN ERROR (AS WE SHOULD)
045.244 076 026 2247 MVI A,$EC.FAP FILE ALREADY PRESENT
045.246 303 161 063 2248 JMP $FERROR ALREADY THERE
2249 *
045.251 376 014 2250 REN2 CPI EC.FNF MUST BE NOT FOUND
045.253 302 161 063 2251 JNE $FERROR OTHER ERROR
045.256 341 2252 POP H (HL) = SOURCE NAME
045.257 001 303 045 2253 LXI B,RENA (BC) = NEW (TARGET) NAME
045.262 377 051 2254 DB SYSCALL,$RENAM RENAME IT
045.264 332 046 051 2255 JC NAMERR ERROR ON RENAME
2256 *
2257 * REMOVE NAME FROM NAMTAB
2258 *

```

RENAME - PROCESS RENAME COMMAND

14:40:26 16-MAY-80

```
045.267 315 223 056 2259 CALL REN REMOVE ENTRY FROM NAMTAB
045.272 052 326 063 2260 LHLD NAMTLEN
045.275 174 2261 MOV A,H
045.276 265 2262 ORA L
045.277 302 174 045 2263 JNZ REN1
045.302 311 2264 RET
2265
045.303 2266 RENA DS FB.NAML FILE NAME WORK AREA
2267 ENDIF
```

```

2270 *** LIST - INDEX DIRECTORY.
2271 *
2272 * DEST=SOURCE/LIST
2273 * /BRIEF
2274 *
2275 * THESE SWITCHES CAUSE THE DIRECTORY CONTENTS OF THE SPECIFIED FILE(S)
2276 * TO BE LISTED
2277 *
2278 * IN /LI FORM, THE OUTPUT IS:
2279 *
2280 * NAME EXT SIZE DATE FLAGS
2281 * XXX .XXX NNN DD-MMM-YY CWS
2282 * . . . . .
2283 * . . . . .
2284 * . . . . .
2285 * NNN FILES USING MMM SECTORS, XXX FREE
2286 *
2287 * IN /RR FORM, ONLY THE NAME AND EXTENSION ARE LISTED,
2288 * 4 ACROSS THE PAGE.
2289 *
2290 * SPECIAL CONSIDERATIONS:
2291 *
2292 * A NULL NAME OR EXTENSION IS TAKEN AS '*' (WILDCARD)
2293 *
2294 * IMPLIMENTATION:
2295 *
2296 * A FILE LIST OF SOURCE FILES IS BUILT, THE DEVICE DIRECTORY FILE
2297 * IS THEN READ, AND EACH FILE IN IT IS CHECKED FOR A MATCH
2298 * AGAINST ANY SOURCE SPECIFICATIONS, ELIGIBLE FILES ARE LISTED.
2299 *
2300
045.324 041.000.000 2301 LIST LXI H:0
045.327 303 335 045 2302 JMP LIST1
2303
045.332 041 001 000 2304 BRIEF LXI H:1
2305 * JMP LIST1
2306
045.335 042.070.047 2307 LIST1 SHLD LSTA (LSTA) = 0 IF LIST; 1 IF /BRIEF
000.000 2308 ERRNZ LSTB-LSTA-1 LSTB = FILE COUNT
045.340 041.000.000 2309 LXI H:0
045.343 042.072.047 2310 SHLD LSTC CLEAR SECTORS USED COUNT
045.346 315.244.060 2311 CALL $MOVE
045.351 011.000.277 2312 DW 9,S,DATE,LSTG1 SET DATE IN HEADING
2313
2314 * CRACK DESTINATION FILE NAMES
2315
000.000 2316 IF .FIP.
045.357 315.264.053 2317 CALL DDF DECODE DEST FILE NAME
045.362 332.265.051 2318 JC ERROR FILE NAME ERROR
045.365 247 2319 ANA A
045.366 076.205 2320 MVI A,PEC,IOW ILLEGAL USE OF WILDCARD IN DEST
045.370 312.265.051 2321 JZ ERROR
2322 ENDF
2323
2324 * BUILD LIST OF SPECIFICATIONS
2325

```

```

045.373 315 254 047 2326 CALL RLS BUILD LIST OF SOURCE SPECS
045.376 332 265 051 2327 JC ERROR ERROR IN LIST
046.001 001 003 000 2328 LXI B,3
046.004 041 250 063 2329 LXI H,DIRNAM
046.007 315 252 030 2330 CALL $MOVE MOVE DEVICE CODE INTO DIRECT.SYS NAME
046.012 041 252 063 2331 LXI H,DIRNAM+2
046.015 176 2332 MOV A,M SEE IF UNIT NUMBER OMITTED
046.016 247 2333 ANA A
046.017 302 024 046 2334 JNZ LIST1.5 SPECIFIED
046.022 066 060 2335 MVI M,'0' DONT ALLOW NULL NUMBER
2336
2337 * GET ADDRESS OF DEVICE'S GRT
2338
046.024 041 250 063 2339 LIST1.5 LXI H,DIRNAM (HL) = # OF XXX:DIRECT.SYS (XXX = DEVICE)
046.027 001 074 047 2340 LXI B,LSTD (BC) = ADDRESS FOR RETURN INFO
046.032 377 053 2341 DB SYSCALL,,DECODE DECODE NAME
046.034 332 265 051 2342 JC ERROR UNKNOWN DEVICE
046.037 072 074 047 2343 LDA LSTD+0
046.042 346 001 2344 ANI DT,DD
046.044 076 005 2345 MVI A,EC.DNS
046.046 312 265 051 2346 JZ ERROR NOT DIRECTORY DEVICE
046.051 052 115 047 2347 LHLD LSTD+17 (HL) = DEV TBL ADDR
2348
046.054 315 301 057 2349 CALL $INDLR
046.057 007 000 2350 DW DEV.SPG
046.061 062 126 047 2351 STA LSTF SAVE SECTORS PER GROUP
2352
046.064 021 012 000 2353 LXI D,DEV.UNT
046.067 031 2354 DAD D
046.070 072 077 047 2355 LDA LSTD+3
046.073 315 027 041 2356 CALL S.GUP HL = UNIT TABLE POINTER
2357
046.076 315 234 030 2358 CALL $INDL
046.101 001 000 2359 DW UNT.GRT
046.103 353 2360 XCHG
046.104 042 124 047 2361 SHLD LSTE SAVE GRT ADDRESS
046.107 353 2362 XCHG
2363
2364 * OPEN DEVICE'S DIRECTORY
2365
046.110 041 250 063 2366 LXI H,DIRNAM
046.113 076 002 2367 MVI A,CN.DIR
046.115 377 042 2368 DB SYSCALL,,OPENR
046.117 076 200 2369 MVI A,PEC.DF DEVICE FORMAT ERROR
046.121 332 265 051 2370 JC ERROR CANT OPEN DIRECTORY
2371
2372
2373 * OPEN OUTPUT FILE
2374
000.000 2375 IF .PIP,
046.124 041 273 063 2376 LXI H,DESTFB
046.127 315 160 061 2377 CALL $FOPEW OPEN FOR WRITE
2378 ENDF
2379
2380 * GENERATE HEADING
2381

```

```

046.132 001 001 000 2382 LXI B,1 (BC) = TEXT COUNT
046.135 021 127 047 2383 LXI D,LSTG (DE) = TEXT ADDRESS
046.140 072 070 047 2384 LDA LSTA
046.143 247 2385 ANA A
046.144 302 151 046 2386 JNZ LIST2 IS SHORT
046.147 016 051 2387 MVI C,LSTGL PRINT FULL HEADING
000.000 2388 IF ,PYP.
046.151 315 311 061 2389 LIST2 CALL $FWRIB WRITE HEADING
2390 ELSE
2391 LIST2 MOV A,C
2392 XCHG (HL) = LINE ADDRESS
2393 CALL $TYPCC PRINT ON CONSOLE
2394 ENDF
2395
2396 * READ DIRECTORY BLOCKS, LOOKING FOR FILE MATCHES
2397
046.154 001 000 002 2398 LIST3 LXI B,512
046.157 315 063 056 2399 CALL GDWP DE = DIRECTORY WORKSPACE POINTER /79.11.GC/
046.162 076 002 2400 MVI A,CN.DIR
046.164 325 2401 PUSH D /79.11.GC/
046.165 377 004 2402 DE SYSCALL,READ
046.167 321 2403 POP D DE = DIRECOTRY WORKSPACE /79.11.GC/
046.170 332 342 046 2404 JC LIST9 ALL DONE
2405
2406 * CHECK NEXT ENTRY IN NAMTAB AGAINST DIRECTORY ENTRY.
2407 * (DE) = DIRECTORY BUFFER POINTER
2408
046.173 032 2409 LIST4 LDAX D (A) = FIRST CHARACTER OF NAME
046.174 247 2410 ANA A
046.175 312 154 046 2411 JZ LIST3 END OF THIS BUFFER
046.200 074 2412 INR A
000.000 2413 ERRNZ DF,EMP-377Q
046.201 312 274 046 2414 JZ LIST7 THIS ENTRY IS EMPTY
046.204 074 2415 INR A
046.205 312 342 046 2416 JZ LIST9 NO MORE ENTRYS IN DIRECTORY
046.210 353 2417 XCHG
046.211 315 173 053 2418 CALL CFE CHECK FILE ELIGIBILITY
046.214 353 2419 XCHG
046.215 302 274 046 2420 JNE LIST7 NOT ELIGIBLE
046.220 041 154 065 2421 LXI H,NAMTAB
2422
046.223 345 2423 LIST5 PUSH H
046.224 325 2424 PUSH D SAVE ADDRESS OF FILE AND PATTERN
046.225 315 356 053 2425 CALL CAD CONVERT ASCII NAMTAB ENTRY TO DIRECTORY FORMAT
046.230 021 364 064 2426 LXI D,PIO.DIR+DIR.NAM (DE) = NAMTAB PATTERN
046.233 341 2427 POP H
046.234 345 2428 PUSH H (HL) = DIRECTORY PATTERN
046.235 006 013 2429 MVI B,B+3 CHECK FOR MATCH
046.237 315 246 053 2430 CALL CWM CHECK FOR WILDCARD MATCH
046.242 321 2431 LIST6 POP D
046.243 341 2432 POP H
046.244 312 323 046 2433 JE LIST8 GOT FILE TO LIST
046.247 001 021 000 2434 LXI B,FB.NAML
046.252 011 2435 DAD B ADVANCE PAST ENTRY IN NAMTAB
2436
2437 * SEE IF AT END OF NAMTAB
  
```

```

.....
2438
046.253 325 2439 PUSH D
046.254 353 2440 XCHG (DE) = NEW ADDRESS
046.255 052 326 063 2441 LHLD NAMLEN
046.260 001 154 065 2442 LXI B,NAMTAB
046.263 011 2443 DAD B (HL) = LWA+1 OF TABLE
046.264 353 2444 XCHG
046.265 315 216 030 2445 CALL $CDEHL COMPARE
046.270 321 2446 POP D
046.271 302 223 046 2447 JNE LIST5 MORE IN TABLE
2448
2449 * FILE DOESNT MATCH ANY SELECTED FILE. PASS TO NEXT ONE
2450
046.274 353 2451 LIST7 XCHG (HL) = DIR BUFFER ADDRESS
2452
046.275 345 2453 PUSH H
046.276 315 071 056 2454 CALL GDWP; HL = DIRECTORY WORKSPACE PTR. /79.11.GC/
046.301 315 301 057 2455 CALL $INDLR A = DIR. ENTRY LENGTH /79.11.GC/
046.304 373 001 2456 IN DIS:ENL /79.11.GC/
046.306 341 2457 POP H /79.11.GC/
2458
046.307 315 101 030 2459 CALL $DABA. ADVANCE
046.312 176 2460 MOV A,M
046.313 247 2461 ANA A
046.314 353 2462 XCHG
046.315 302 173 046 2463 JNZ LIST4 TRY THIS ONE
046.320 303 154 046 2464 JMP LIST3 READ ANOTHER BLOCK
2465
2466 * HAVE FILE TO LIST
2467
046.323 325 2468 LIST8 PUSH D SAVE DIR POINTER
046.324 072 126 047 2469 LDA LSTF (A) = SECTORS PER GROUP THIS DEVICE
046.327 315 024 050 2470 CALL PFI PRINT FILE INFO
046.332 321 2471 POP D
046.333 041 071 047 2472 LXI H,LSTB
046.336 064 2473 INR M COUNT FILE
046.337 303 274 046 2474 JMP LIST7 ADVANCE TO NEXT FILE
2475
2476 * ALL DONE. CLOSE DIRECTORY FILE
2477
046.342 076 002 2478 LIST9 MVI A,CN:DIR
046.344 377 046 2479 DB SYSCALL, CLOSE CLOSE FILE
046.346 001 001 000 2480 LXI B,1 ASSUME SHORT FORM, JUST WRITE NL
046.351 072 070 047 2481 LDA LSTA (A) = FORM FLAG
046.354 247 2482 ANA A
046.355 302 045 047 2483 JNZ LIST10 IS SHORT, NO TRAILER
2484
2485 * PRINT SUMMARY:
2486 *
2487 * NNN FILES, USING XXX SECTORS, YYY FREE
2488
046.360 072 071 047 2489 LDA LSTB
046.363 117 2490 MOV C,A
046.364 006 000 2491 MVI B,0 (BC) = FILE COUNT
046.366 076 003 2492 MVI A,3
046.370 041 204 047 2493 LXI H,LSTH1
.....

```

```

046.373 315 171 060 2494 CALL $UDDN FILE COUNT
046.376 052 072 047 2495 LHLI LSTC
047.001 104 2496 MOV B,H
047.002 115 2497 MOV C,L (BC) = SECTOR COUNT
047.003 041 225 047 2498 LXI H,LSTH2
047.006 076 003 2499 MVI A,3
047.010 315 171 060 2500 CALL $UDDN USED COUNT
047.013 052 124 047 2501 LHLI LSTE
047.016 176 2502 MOV A,H
047.017 315 213 053 2503 CALL CFS FOLLOW GRT CHAIN
047.022 072 126 047 2504 LDA LSTF
047.025 315 007 031 2505 CALL $MUS6 (HL) = SECTORS FREE
047.030 104 2506 MOV B,H
047.031 115 2507 MOV C,L
047.032 041 242 047 2508 LXI H,LSTH3
047.035 076 003 2509 MVI A,3
047.037 315 171 060 2510 CALL $UDDN UNPACK FREE
047.042 001 054 000 2511 LXI B,LSTHL
047.045 021 200 047 2512 LIST10 LXI D,LSTH
047.050 072 246 063 2513 LDA SUPRES
047.053 247 2514 ANA A
000.000 2515 IF ,PIP:
047.054 041 273 063 2516 LXI H,DESTFB
047.057 302 177 062 2517 JNZ $FCLO CLOSE AND EXIT, SUMMARY SUPPRESSED
047.062 315 311 061 2518 CALL $FWRIB WRITE TRAILER
2519
2520 * ALL DONE, CLOSE OUTPUT FILE
2521
047.065 303 177 062 2522 JMP $FCLO CLOSE AND EXIT
2523 ELSE
2524 RNZ NOT TO SUMMARYIZE
2525 MOV A,R (A) = COUNT
2526 XCHG (HL) = ADDRESS
2527 JMP $TYECC TYPE TEXT AND EXIT
2528 ENDIF
2529
047.070 000 2530 LSTA DB 0 <>0 IFF SHORT FORM
2531
047.071 000 2532 LSTB DB 0 FILE COUNT
047.072 000 000 2533 LSTC DW 0 SECTORS USED
047.074 2534 LSTH DS 24 FILE NAME DECODE AREA
047.124 000 000 2535 LSTE DW 0 GRT ADDRESS
047.126 000 2536 LSTF DB 0 SECTORS PER GROUP FOR THIS DEVICE
047.127 012 116 141 2537 LSTG DB NL,'Name','TAB','Ext','TAB','Size','TAB','Date','TAB','TAB','Class','TAB
047.165 2538 LSTG1 DS 9 DATE
047.176 012 012 2539 DB NL,NL
000.051 2540 LSTGL EQU *-LSTG
2541
047.200 012 040 040 2542 LSTH DB NL,' ' FIRST CHARACTER MUST BE <NL>
047.204 116 116 116 2543 LSTH1 DB 'NNN Files, Using '
047.225 115 115 115 2544 LSTH2 DB 'MMM Sectors ('
047.242 130 130 130 2545 LSTH3 DB 'XXX Free)',NL
000.054 2546 LSTHL EQU *-LSTH

```



```

2548 ** BLS - BUILD LIST OF SOURCE FILES.
2549 *
2550 * BLS BUILDS A LIST OF SOURCE FILES INTO *NAMTAB*
2551 * NULL FIELDS ARE SET TO WILDCARDS. BLS REQUIRES THAT ALL
2552 * FILES SPECIFIED HAVE THE SAME DEVICE.
2553 *
2554 * IF THE COMMAND LINE CONTAINS NO FILES, BUT CONTAINS AT LEAST
2555 * ONE BLANK (AS WOULD BE THE CASE IN PROCESSING THE /LIST SWITCH, SINCE
2556 * THE "/LIST" IS REPLACED WITH BLANKS) A FILE NAME OF ????????.???
2557 * IS DECODED.
2558 * ENTRY NAMTAB EMPTY
2559 * EXIT 'C' CLEAR IF OK
2560 * (DE) = *BLSA = 3 CHARACTER DEVICE NAME
2561 * 'C' SET IF ERROR
2562 * (A) = ERROR MESSAGE
2563 * USES ALL
2564
2565
047.254 315 244 060 2566 BLS CALL $MOVE
047.257 003 000 017 2567 DW 3,BLSC,BLSA SET INITIAL DEFAULT DEVICE
047.265 041 000 000 2568 LXI H,0
047.270 042 326 063 2569 SHLD NAMTLEN CLEAR NAMTAB
047.273 076 377 2570 MVI A,3770
047.275 062 016 050 2571 STA BLSB FLAG PROCESSING OF FIRST FILE NAME
047.300 315 127 056 2572 CALL LSN LOCATE SOURCE NAMES
2573
2574 * CRACK THE NEXT NAME
2575
047.303 176 2576 BLS1 MOV A,M
047.304 021 010 050 2577 LXI D,BLSA (DE) = DEFAULT ADDRESS
047.307 247 2578 ANA A
047.310 310 2579 RZ NO MORE NAMES
047.311 315 150 057 2580 CALL $SOB SEE IF ALL NULL
047.314 176 2581 MOV A,M
047.315 247 2582 ANA A
047.316 302 324 047 2583 JNZ BLS2 NOT ALL NULL
047.321 041 017 050 2584 LXI H,BLSC USE DEFAULT DEVICE
047.324 315 362 053 2585 BLS2 CALL CAD, CONVERT ASCII NAME TO DIRECTORY FORMAT
047.327 330 2586 RC ERROR
2587
2588 * IF FIRST NAME, RECORD DEVICE
2589 * IF NOT FIRST, COMPARE DEVICE AGAINST FIRST DEVICE
2590
047.330 345 2591 PUSH H
047.331 021 361 064 2592 LXI D,FID.DEV
047.334 041 010 050 2593 LXI H,BLSA
047.337 061 003 000 2594 LXI B,3 SETUP COUNT, FROM AND TO
000.000 2595 IF ,PIP,
047.342 072 016 050 2596 LDA BLSB
047.345 247 2597 ANA A
047.348 362 363 047 2598 JP BLS3 NOT 1ST FILE
047.351 315 252 030 2599 CALL $MOVE MOVE IN REQUIRED DEVICE FOR REMAINING FILES
047.354 257 2600 XRA A
047.355 062 016 050 2601 STA BLSB FLAG 1ST NAME PROCESSED
047.360 303 376 047 2602 JMP BLS4
2603 ENDIF

```

```

.....
2604
047.363 315 060 030 2605 BLS3 CALL $COMP SEE IF THIS DEVICE SAME AS PREVIOUS
047.366 312 376 047 2606 JE BLS4 OK
047.371 076 201 2607 MVI A,PEC.DNC MULTIPLE DEVICES ARE ILLEGAL
047.373 067 2608 STC
047.374 341 2609 POP H
047.375 311 2610 RET RETURN WITH ERROR
2611
2612 * GOT NAME DECODED. ENTER IN NAMTAB
2613
047.376 315 307 052 2614 BLS4 CALL AEN ADD ENTRY TO NAMTAB
050.001 341 2615 POP H
050.002 315 316 056 2616 CALL SFS SKIP FILE SEPERATOR (BLANKS AND/OR COMMA)
050.005 303 303 047 2617 JMP BLS1 SEE IF MORE
2618
050.010 123 131 060 2619 BLSA DB 'SY0',200Q,200Q,200Q
050.016 000 2620 BLSB DB 0 FIRST FILE NAME FLAG
050.017 123 131 060 2621 BLSC DB 'SY0:',0 DEFAULT DEVICE
.....

```

```

.....
2623 ** PFI - PRINT FILE INFO.
2624 *
2625 * PFI DECODES A DIRECTORY ENTRY INTO A CODED LINE, THEN
2626 * WRITES IT TO 'DESTFB'.
2627 *
2628 * THE PRODUCED FORMAT DEPENDS UPON THE LISTING FORMAT FLAG,
2629 * LSTA.
2630 *
2631 * SHORT FORM:
2632 *
2633 * NAME .EXT (TAB)
2634 *
2635 * LONG FORM:
2636 *
2637 * NAME .EXT SIZE DATE FLAGS (NL)
2638 *
2639 * ENTRY (A) = SECTORS PER GROUP FOR THIS DEVICE
2640 * (DE) = DIRECTORY ENTRY POINTER
2641 * EXIT IF LONG FORM, SECTOR COUNT IS ACCUMULATED IN LSTC
2642 * USES ALL
2643
.....

```

```

.....
2644
050.024 062 372 050 2645 PFI STA PFIC SAVE SECTORS PER GROUP
050.027 041 310 050 2646 LXI H,PFIA
050.032 016 010 2647 MVI C,B
050.034 315 272 050 2648 CALL PFI20 COPY NAME
050.037 312 045 050 2649 JZ PFI1 ALL B CHARACTERS
050.042 066 011 2650 MVI M,TAB
050.044 043 2651 INX H
050.045 066 056 2652 PFI1 M,', '
050.047 043 2653 INX H
050.050 016 003 2654 MVI C,3
050.052 315 272 050 2655 CALL PFI20 COPY EXTENSION
050.055 066 011 2656 MVI M,TAB
.....

```

FFI

050.057	043		2657	INX	H	
050.060	072	070	047	2658	LDA	LSTA
050.063	247		2659	ANA	A	
050.064	312	111	050	2660	JZ	PF13
			2661			IS LONG FORM
			2662	*		IS SHORT FORM. SEE IF NEED TO END LINE
			2663			
050.067	074		2664	INR	A	
050.070	376	005	2665	CPI	S	
050.072	302	103	050	2666	JNE	PF12
			2667			NOT TIME YET
050.075	053		2667	DCX	H	
050.076	066	012	2668	MVI	M,NL	
050.100	043		2669	INX	H	TIME TO END LINE
050.101	076	001	2670	MVI	A,1	
050.103	062	070	047	2671	PF12	STA
			2672			LSTA
050.106	303	246	050	2672	JMP	PF16
			2673			OUTPUT TO FILE
			2674	*		IS LONG FORM.
			2675			
050.111	001	005	000	2676	PF13	LXI
			2677			B,DIR,FGN-DIR,EXT-3
050.114	353		2677	XCHG		(DE) = LINE ADDR, (HL) = #PIO.DIR+DIR,EXT+3
050.115	011		2678	DAD	B	(HL) = #DIR,FGN
050.116	176		2679	MOV	A,M	(A) = (DIR,FGN)
050.117	043		2680	INX	H	
050.120	043		2681	INX	H	
050.121	116		2682	MOV	C,M	(C) = DIR,LSI = SECTORS USED IN LAST GROUP
000.000			2683	ERRNZ		DIR,LSI-DIR,FGN-2
050.122	353		2684	XCHG		(DE) = ADDRESS OF LSI
050.123	325		2685	FUSH	D	SAVE #DIR,LSI
050.124	345		2686	FUSH	H	SAVE LINE ADDRESS
050.125	052	124	047	2687	LHLD	LSTE
050.130	157		2688	MOV	L,A	
050.131	176		2689	MOV	A,M	
050.132	315	213	053	2690	CALL	CFS
			2691			COMPUTE FILE ISZE
050.135	072	372	050	2691	LDA	PF1C
			2692			(A) = SECTORS PER GROUP
050.140	315	007	031	2692	CALL	\$MUS6
			2693			(HL) = SECTORS USED (EXCEPT FOR THOSE IN LAST GROUP)
050.143	006	000	2693	MVI	B,0	
050.145	011		2694	DAD	B	(HL) = SECTORS USED
050.146	104		2695	MOV	B,H	
050.147	115		2696	MOV	C,L	(BC) = SECTORS USED COUNT
050.150	052	072	047	2697	LHLD	LSTC
050.153	011		2698	DAD	B	
050.154	042	072	047	2699	SHLD	LSTC
			2700			ACCUMULATE COUNT OF SECTORS
050.157	341		2700	POP	H	(HL) = LINE ADDRESS
050.160	076	003	2701	MVI	A,3	3 DIGITS MAX
050.162	315	171	060	2702	CALL	\$UDDN
			2703			UNPACK COUNT
050.165	066	011	2703	MVI	M,TAB	
050.167	043		2704	INX	H	
050.170	321		2705	POP	D	(DE) = #DIR,LSI
			2706			
			2707	*		TYPE DATE
			2708			
050.171	353		2709	XCHG		
000.000			2710	ERRNZ		DIR,CRD-DIR,LSI-1
050.172	043		2711	INX	H	(HL) = #DIR,CRD
050.173	345		2712	FUSH	H	

```

050.174 315 211 030 2713 CALL $HLIHL
050.177 353 2714 XCHG
050.200 315 004 060 2715 CALL $DAD DECODE AUGUSTAN DATE
2716
2717 * CODE FLAGS
2718
050.203 353 2719 XCHG (DE) = LINE ADDRESS
050.204 341 2720 POP H (HL) = $DIR.CRD
050.205 001 373 377 2721 LXI B,DIR.FLG-DIR.CRD
050.210 011 2722 DAD B (HL) = ADDRESS OF DIRFLG
050.211 176 2723 MOV A,H (A) = FLAGS
050.212 353 2724 XCHG (HL) = LINE ADDRESS
050.213 247 2725 ANA A
050.214 312 243 050 2726 JZ PF15.5 NO FLAGS
050.217 066 011 2727 MVI M,TAB TAB BEFORE FLAGS
050.221 043 2728 INX H
050.222 021 362 050 2729 LXI D,PF1B
050.225 207 2730 POP A
050.226 322 236 050 2731 JNC PF15 NOT SET
050.231 365 2732 PUSH PSW SAVE FLAGS
050.232 032 2733 LIAX D
050.233 167 2734 MOV M,A
050.234 361 2735 POP PSW RESTORE FLAGS
050.235 043 2736 INX H
050.236 023 2737 PF15 INX D SET FLAG
050.237 247 2738 ANA A
050.240 302 225 050 2739 JNZ PF14 MORE FLAGS SET
050.243 066 012 2740 PF15.5 MVI M,NL
050.245 043 2741 INX H
2742
2743 * LINE ALL BUILT. WRITE TO DESTFB
2744
050.246 021 070 327 2745 PF16 LXI D,-PF1A
050.251 031 2746 DAD D
000.000 2747 IF .PIF.
050.252 104 2748 MOV B,H
050.253 115 2749 MOV C,L (BC) = LEN
050.254 021 310 050 2750 LXI D,PF1A (DE) = DATA FWA
050.257 041 273 063 2751 LXI H,DESTFB
050.262 303 311 061 2752 JMP $FWRIB WRITE AND EXIT
2753 ELSE
2754 MOV A,L (A) = COUNT
2755 LXI H,PF1A
2756 JMP $TYPCC TYPE LINE AND EXIT
2757 ENDF

```

```

2759 ** PFI20 - COPY FILE NAME.
2760 *
2761 * PFI20 COPIES A NAME FILED FROM THE DIRECTORY ENTRY TO A CODED
2762 * LINE
2763 *
2764 * EENTRY (DE) = DIRECTORY ADDRESS
2765 * (C) = NAME LENGTH
2766 * (HL) = LINE ADDRESS

```

```

2767 *      EXIT      (DE) = (DE) + (C)
2768 *      'Z' SET IF MAX CHARACTERS COPIED
2769 *      USES      A,F;C;D;E;H;L
2770
2771
050.265 167 2772 PFI19 MOV      M;A      COPY
050.266 043 2773      INX      H
050.267 023 2774      INX      D
050.270 015 2775      DCR      C
050.271 310 2776      RZ              ALL COPIED
050.272 032 2777 PFI20 LDAX     D
050.273 247 2778      ANA      A
050.274 302 265 050 2779      JNZ      PFI19      GOT CHAR
2780
2781 *      NO NAME. (C) = COUNT LEFT
2782
050.277 173 2783      MOV      A;E
050.300 201 2784      ADD      C
050.301 137 2785      MOV      E;A
050.302 172 2786      MOV      A;D
050.303 316 000 2787      ACI      0
050.305 127 2788      MOV      D;A
050.306 263 2789      ORA      E      CLEAR 'Z'
050.307 311 2790      RET
2791
050.310 2792 PFIA  DS      0      BUFFER AREA FOR LINE BUILD
050.310 130 130 130 2793      DB      'XXXXXXXX.YYY'  NNN DD-MMM-YY
050.342 011 011 106 2794      DB      '          '  FLAGS
050.362 123 114 127 2795 PFIB  DB      'SLW'      CODES
050.365 040 061 062 2796 PFIB1 DB      '1234'      ('C' FOR CONTIGUOUS IS OPTIONAL)
000.000 2797      ERRNZ  DIF.SYS-200Q
000.000 2798      ERRNZ  DIF.LOC-100Q
000.000 2799      ERRNZ  DIF.WP-40Q
000.000 2800      ERRNZ  DIF.CNT-20Q
050.372 000 2801 PFIC  DB      0      SECTORS PER GROUP FOR THIS DEVICE
  
```

```
2804 ***   VERSN - PIP VERSION INFORMATION
2805 *
2806 *   DEST=/V(VERSION)
2807 *
2808 *   PRINT THE PIP VERSION INFORMATION TO THE 'DEST' FILE.
2809 *
2810
050,373    2811 VERSN EQU *
2812
050,373 315 231 053 2813 CALL CTS CHECK FOR TARGET FILE SPECIFICATION
050,376 067 2814 STC
050,377 302 265 051 2815 JNZ ERROR TARGET FILE SPECIFICATION ILLEGAL
051,002 041 034 065 2816 LXI H,LINE
051,005 315 150 057 2817 CALL $SOB SKIP OVER ALL THE BLANKS ($DRS TURNS SWITCHES
051,010 176 2818 MOV A,M TO BLANKS)
051,011 247 2819 ANA A
051,012 076 207 2820 MUI A,PEC.SFI SOURCE FILE ILLEGAL
051,014 067 2821 STC
051,015 302 265 051 2822 JNZ ERROR ONLY ALLOW SWITCH ON LINE
051,020 315 136 031 2823 CALL $TYPTX
2824
000,000 2825 IF :PIP,
051,023 120 111 120 2826 DB 'PIP'
2827 ELSE
2828 DB 'ONECOPY'
2829 ENDIF
2830
051,026 011 126 145 2831 DB TAB,'Version: '
051,041 061 056 066 2832 DB VERS/16+'0',',',',',VERS:00001111B+'0'
051,044 212 2833 DB ENL
2834
051,045 311 2835 RET
```

2838 \*\* ERROR PROCESSING ROUTINES  
 2839 \*

2841 \*\*\* NAMERR - FILE TYPE ERROR, OCCURRED ON FILE WHOSE NAME  
 2842 \* IS NEXT UP IN NAMTAB.  
 2843 \*

2844 \* PROCESS VIA \$FERROR  
 2845

000.000				2846	IF	.PIP.
051.046	041	142	065	2847	NAMERR LXI	H,NAMTAB-FB.NAM
051.051	303	161	063	2848	JMP	\$FERROR
				2849	ELSE	
				2850	NAMERR LHLD	NAMTPTR
				2851	LXI	B,-FB.NAM
				2852	DAD	B
				2853	JMP	\$FERROR
				2854	DESTERR SPACE	4,10
				2855	**	ERROR ON FILE IN DESTFB
				2856		
				2857	DESTERR LXI	H,DESTFB
				2858	JMP	\$FERROR
				2859	ENDIF	

2861 \*\* INTERNAL ERRORS, SHOULD NOT OCCUR.  
 2862

051.054	076	061		2863	IERR1 MVI	A,'1'
051.056	303	073	051	2864	JMP	INTERR
				2865		
051.061	076	062		2866	IERR2 MVI	A,'2'
051.063	303	073	051	2867	JMP	INTERR
051.066	076	063		2868	IERR3 MVI	A,'3'
051.070	303	073	051	2869	JMP	INTERR

				2870		
				2871		
051.073	365			2872	INTERR PUSH	PSW SAVE CODE
051.074	315	136	031	2873	CALL	\$TYPTX
051.077	007	012	120	2874	DB	BELL,NL,'PIP INTERNAL ERROR ','\$'+2000
051.125	361			2875	POP	PSW
051.126	315	275	060	2876	CALL	\$WCHAR
051.131	315	136	031	2877	CALL	\$TYPTX
051.134	012	124	110	2878	DB	NL,'THIS ERROR SHOULD NOT OCCUR, CONTACT HEATH TECHNICAL'
051.221	012	103	117	2879	DB	NL,'CORRESPONDENCE FOR ASSISTANCE.',NL
051.261	076	001		2880	MVI	A,1
051.263	377	000		2881	DB	SYSCALL,'EXIT ABORT

```

2883 **      ERROR - GENERAL AND SYNTAX ERRORS NOT DIRECTLY ASSOCIATED
2884 *      WITH A VALID FILE NAME.
2885
2886
051.265 365 2887 ERROR PUSH PSW SAVE CODE
051.266 315 136 031 2888 CALL $TYPTX
051.271 007 105 122 2889 DB BELL,'ERROR -',',','+2000
051.302 361 2890 POP PSW
051.303 247 2891 ANA A
051.304 372 316 051 2892 JM ERROR1 IS PRODUCT ERROR
051.307 046 012 2893 HVI H,NL USE NL AS MESSAGE TRAIL CHAR
051.311 377 057 2894 DB SYSCALL,'.ERROR LOOK UP SYSTEM ERROR
051.313 303 200 042 2895 JMP RESTART
2896
2897 *      IS PRODUCT ERROR
2898
051.316 041 333 051 2899 ERROR1 LXI H,ERRORA
051.321 276 2900 ERROR2 CMP M
051.322 043 2901 INX H
051.323 302 321 051 2902 JNE ERROR2 FIND ERROR MESSAGE
000.001 2903 IF DNECOPY
2904 CALL $TYPTX
2905 DB BELL,'DNECOPY Error #',',','+2000
2906 ENDF
051.326 377 003 2907 DB SYSCALL,'.PRINT PRINT MESSAGE
051.330 303 200 042 2908 JMP RESTART
2909
051.333 2910 ERRORA DS 0 ERROR MESSAGES
000.000 2911 IF .PIP.
051.333 200 104 145 2912 DB PEC.DF,'Device Format Error',ENL
051.360 201 101 154 2913 DB PEC.DNC,'All Files Must Reside on the Same Device',ENL
052.032 203 104 145 2914 DB PEC.TFI,'Destination File Specification is Illegal',ENL
052.105 204 103 157 2915 DB PEC.CS,'Contradictory Switches Specified',ENL
052.147 205 111 154 2916 DB PEC.IUW,'Illegal Use of Wildcard',ENL
052.200 206 111 154 2917 DB PEC.IDF,'Illegal Destination File Format',ENL
052.241 207 123 157 2918 DB PEC.SFI,'Source File Specification is Illegal',ENL
2919 ELSE
2920 DB PEC.DF,'01',ENL
2921 DB PEC.DNC,'02',ENL
2922 DB PEC.TFI,'03',ENL
2923 DB PEC.CS,'04',ENL
2924 DB PEC.IUW,'05',ENL
2925 DB PEC.IDF,'06',ENL
2926 DB PEC.SFI,'07',ENL
2927 DB PEC.FCI,'08',ENL
2928 ENDF

```



```

2932 **      AEN - ADD ENTRY TO 'NAMTAB'
2933 *
2934 *      AEN EXPANDS THE FILE INFO IN PIO.XXX INTO A FILE DESCRIPTOR
2935 *      AND ENTERS IT IN THE NAMTAB TABLE.
2936 *
2937 *      ENTRY  NONE
2938 *      EXIT   'C' SET IF WILDCARD
2939 *      USES   ALL
2940
2941
052.307 041 361 052 2942 AEN  LXI   H,AENA
052.312 315 057 058 2943      CALL  CDA           CONVERT DIRECTORY FORMAT TO ASCII FORMAT
052.315 326 001      2944      SUI   I           'C' SET IF WILDCARD
052.317 365      2945      PUSH  PSW          SAVE FLAG
052.320 052 326 063 2946      LHLD  NAMTLEN
052.323 001 021 000 2947      LXI   B,FB,NAML
052.326 011      2948      DAD   B           INCREASE SIZE
052.327 042 326 063 2949      SHLD  NAMTLEN
052.332 353 2950      XCHG  (DE) = NEW LENGTH
052.333 052 330 063 2951      LHLD  NAMTMAX
052.336 175 2952      MOV   A,L           SEE IF WILL OVERFLOW
052.337 223 2953      SUB   E
052.340 174 2954      MOV   A,H
052.341 232 2955      SRB   D
052.342 334 075 056 2956      CC   INA           INCREASE NAMTAB ALLOCATION
052.345 041 133 065 2957      LXI   H,NAMTAB-FB,NAML
052.350 031 2958      DAD   D           (HL) = *TO* ADDRESS
052.351 021 361 052 2959      LXI   D,AENA        (DE) = *FROM* ADDRESS
052.354 315 252 030 2960      CALL  $MOVE        MOVE ENTRY IN
052.357 361 2961      POP  PSW          (PSW) = WILDCARD FLAG
052.360 311 2962      RET
2963
052.361      2964 AENA  DS    FB,NAML

```

```

2966 **      BSL - BUILD SOURCE FILE LIST.
2967 *
2968 *      BSL CRACKS THE LIST OF THE SOURCE FILES FROM THE COMMAND LINE AND
2969 *      BUILDS THEM INTO THE NAMTAB MANAGED TABLE.
2970 *      WILD CARDS ENCOUNTERED ARE EXPANDED.
2971 *
2972 *      ENTRY  (A) <= 0 IF TO ASK ABOUT '*,*' USE
2973 *      EXIT   'C' CLEAR IF OK
2974 *           'C' SET IF ERROR
2975 *           (A) = CODE
2976 *      USES   ALL
2977
2978
053.002 062 053 053 2979 BSL  STA   BSLA        SAVE ASK FLAG
053.005 315 127 056 2980      CALL  LSN          LOCATE SOURCE NAME
2981
2982 *      GO THROUGH SOURCE LIST CRACKING NAMES
2983
053.010 176      2984 BSL1  MOV   A,M

```

053.011	247		2985	ANA	A	
053.012	310		2986	RZ		ALL NONE
053.013	021 332 063		2987	LXI	D,DEFAULT	
053.016	315 356 053		2988	CALL	CAD	CONVERT ASCII NAME TO DIRECTORY FORMAT
053.021	330		2989	RC		ERROR
053.022	315 333 056		2990	CALL	SND	SET NEW DEFAULTS
053.025	345		2991	PUSH	H	SAVE LINE ADDRESS
053.026	072 053 053		2992	LDA	BSLA	
053.031	247		2993	ANA	A	
053.032	304 054 053		2994	CNZ	CCW	CHECK FOR COMPLETE WILDCARD (*.*)
053.035	332 200 042		2995	JC	RESTART	USER CHICKENED OUT /79.12.6C/
053.040	315 150 055		2996	CALL	EWS	EXPAND WILDCARD SPECIFICATION
053.043	341		2997	POP	H	RESTORE LINE ADDRESS
053.044	330		2998	RC		USER REFUSED *.*
053.045	315 316 056		2999	CALL	SFS	SKIP FILE SEPERATOR (BLANKS AND/OR COMMA)
053.050	303 010 053		3000	JMP	BSL1	DO MORE
			3001			
053.053	000		3002	BSLA	DB 0	<0 IF TO CHECK FOR *.*

```

3004 **      CCW - CHECK FOR COMPLETE WILDCARD.
3005 *
3006 *      CCW IS CALLED WITH A NAME CRACKED INTO PIO,XXX, TO SEE IF
3007 *      IT IS A *.* SPECIFICATION.
3008 *
3009 *      IF SO, CCW ASKS,
3010 *
3011 *      DELETE ALL FILES ON DEV: ?!? (Y/N)
3012 *
3013 *      THE USER REPLY IS ACCEPTED AND DECODED.
3014 *
3015 *      ENTRY  NONE
3016 *      EXIT  'C' CLEAR IF NOT *.* , OR 'Y' REPLIED
3017 *           'C' SET IF *.* AND NOT 'Y'
3018 *      USES  A:F,B:H,L
3019 *
3020

```

053.054	041 364 064		3021	CCW	LXI	H,PIO.DIR+DIR.NAM
000.000			3022		IF	.PIF,
053.057	006 013		3023		MVI	B,B+3
053.061	076 200		3024		MVI	A,2000
053.063	246		3025	CCW1	ANA	M
053.064	043		3026		INX	H
053.065	005		3027		DCR	B
053.066	302 063 053		3028		JNZ	CCW1
053.071	247		3029		ANA	A
053.072	360		3030		RF	NOT *.*
			3031			
			3032	*	IS	*.*
			3033			
053.073	315 136 031		3034		CALL	\$TYPTX
053.076	007 041 077		3035		DB	BELL,'?!' DELETE ALL FILES ON',''+2000
053.127	041 361 064		3036		LXI	H,PIO.DEV
053.132	076 003		3037		MVI	A,3

```

053.134 315 008 057 3038 CALL $TYPCC TYPE DEVICE NAME
053.137 315 136 031 3039 CALL $TYPTX
053.142 072 040 050 3040 DB 'Y(Y/N)?';'+2000
053.153 041 361 063 3041 LXI H,DESTBUF
053.156 315 103 057 3042 CALL $RTL; READ REPLY
053.161 072 361 063 3043 LDA DESTBUF
053.164 378 131 3044 CPI 'Y'
053.166 310 3045 RE IS OK
053.167 067 3046 STC
053.170 076 205 3047 MVI A,PEC,IUW FLAG ILLEGAL USE OF WILDCARD
3048 ENDDIF
053.172 311 3049 RET FORGET IT
  
```

3051 \*\* CFE - CHECK FILE ELIGIBILITY.

```

3052 *
3053 * CFE CHECKS TO SEE IF A WILDCARD-SELECTED FILE IS ELIGIBLE
3054 * FOR PROCESSING, IF THE FILE IS FLAGGED SYSTEM, AND /S IS NOT
3055 * SPECIFIED, THE FILE IS NOT ELIGIBLE.
3056 *
3057 * ENTRY (HL) = DIRECTORY ENTRY POINTER
3058 * EXIT 'Z' SET IF ELIGIBLE
3059 * USES A,F
  
```

```

053.173 345 3061 CFE
053.174 076 016 3062 PUSH H
053.176 315 101 030 3063 MVI A,DIR.FLG
053.201 176 3064 CALL $DADA.
053.202 346 200 3065 MOV A,H (A) = FLAG
053.204 341 3066 ANI DIF.SYS
053.205 310 3067 POP H
053.206 072 247 063 3068 RZ ELIGIBLE
053.211 247 3069 LDA SYSTEM CHECK /S FLAG
053.212 311 3070 ANA A
3071 RET
  
```

3073 \*\* CFS - COMPUTE FILE SIZE

```

3074 *
3075 * CFS COMPUTES THE SIZE OF A FILE. THE DEVICE'S GRT MUST BE IN
3076 * THE 'GRT' BUFFER.
3077 *
3078 * ENTRY (A) = FIRST GROUP NUMBER
3079 * EXIT (DE) = SIZE
3080 * USES ALL
  
```

```

053.213 052 124 047 3081 CFS
053.216 021 000 000 3082 CFS
053.221 247 3083 LHLB LSTE
053.222 310 3084 LXI D,0
053.223 157 3085 ANA A
3086 RZ ALL DONE
3087 MOV L,A
  
```

053.224 176 3088 MOV A,M (A) = NEXT GRT  
 053.225 023 3089 INX D  
 053.226 303 221 053 3090 JMP CFS1 TRY AGAIN

3092 \*\* CTS - CHECK TARGET FILE SPECIFICATION  
 3093 \*  
 3094 \* CTS CHECKS FOR A TARGET FILE SPECIFICATION  
 3095 \*  
 3096 \*  
 3097 \* ENTRY NONE  
 3098 \*  
 3099 \* EXIT (PSW) = 'Z' SET IF NO TARGET FILE  
 3100 \* = 'Z' CLEAR IF TARGET FILE  
 3101 \* (A) = PEC.TFI ERROR CODE  
 3102 \*  
 3103 \* USES (PSW),(HL)  
 3104 \*  
 3105 \*

053.231 315 127 056 3106 CTS CALL LSN (HL) = ADDRESS OF FIRST SOURCE NAME  
 053.234 021 344 312 3107 LXI D,-LINE  
 053.237 031 3108 DAD D (HL) == 0 IF NO '=' IN COMMAND LINE  
 053.240 175 3109 MOV A,L  
 053.241 264 3110 ORA H  
 053.242 310 3111 RZ NO TARGET FILE  
 053.243 076 203 3112 MVI A,PEC,TFI TARGET FILE ILLEGAL  
 053.245 311 3113 RET TARGET FILE SPECIFIED

3115 \*\* CWM - CHECK WILDCARD MATCH  
 3116 \*  
 3117 \* CWM CHECKS TO SEE IF A WILDCARDED FIELD MATCHES A NON-WILDCARDED  
 3118 \* FIELD.  
 3119 \*  
 3120 \* ENTRY (DE) = ADDRESS OF WC NAME  
 3121 \* (HL) = ADDRESS OF NON/WC NAME  
 3122 \* (B) = NUMBER OF CHARACTERS TO CHECK  
 3123 \* EXIT 'Z' SET IF MATCH  
 3124 \* (HL) = (HL)+(B)  
 3125 \* (DE) = (DE) = (B)  
 3126 \* 'Z' CLEAR IF NO MATCH  
 3127 \* USES A,F,B,D,E,H,L  
 3128 \*  
 3129 \*

053.246 032 3130 CWM LDAX D  
 053.247 247 3131 ANA A  
 053.250 372 255 053 3132 JM CWM1 IS MATCH  
 053.253 276 3133 CMP M  
 053.254 300 3134 RNE NO MATCH  
 053.255 023 3135 CWM1 INX D  
 053.256 043 3136 INX H ADVANCE ADDRESSES  
 053.257 005 3137 DCR B

CWM

053.260 302 246 053 3138  
 053.263 311 3139

JNZ CWM  
 RET

GO FOR MORE  
 GOT MATCH

```

3141 **   DDF = DECODE DESTINATION FILE.
3142 *
3143 *   DDF DECODES THE DESTINATION FILE NAME FROM THE COMMAND LINE.
3144 *
3145 *   IF NO DESTINATION NAME IS SPECIFIED, IT DEFAULTS TO
3146 *
3147 *   KB:PIFDEST.JGL
3148 *
3149 *   ENTRY  NONE
3150 *   EXIT   'C' CLEAR IF OK
3151 *         (A) = 0 IF NAME HAS WILDCARDS
3152 *         (A) = 1 IF NO WILDCARD USED
3153 *         DESTFB+FB.NAM CONTAINS A COMPLETE DESTINATION FILE NAME
3154 *         (HL) = COMMAND LINE POINTER UPDATED
3155 *         'C' SET IF ERROR
3156 *         (A) = CODE
3157 *   USES  ALL
3158
3159
053.264 021 034 065 3160 DDF  LXI  D,LINE
053.267 142 3161      MOV  H,D
053.270 153 3162      MOV  L,E           (HL) = COMMAND POINTER
053.271 032 3163 DDF1  LDAX  D
053.272 023 3164      INX  D
053.273 376 075 3165      CPI  '='
053.275 312 307 053 3166      JE   DDF2           HAVE A SOURCE FILE
053.300 247 3167      ANA  A
053.301 302 271 053 3168      JNZ  DDF1           MORE TO CHECK
053.304 041 336 053 3169 DDF1.0 LXI  H,DDFA           USE DEFAULT
3170
3171 *   (HL) = ADDRESS FOR NAME
3172
053.307 021 332 063 3173 DDF2  LXI  D,DEFAULT
053.312 315 356 053 3174      CALL CAD           CONVERT ASCII NAME TO DIRECTORY FORMAT
053.315 330 3175      RC   ERROR
053.316 312 304 053 3176      JZ   DDF1.0       NO FILE NAME SPECIFIED, USE DEFAULT
053.321 176 3177      MOV  A,M
053.322 376 075 3178      CPI  '='
053.324 076 206 3179      MVI  A,PEC.IDF    ASSUME ILLEGAL DESTINATION FORMAT
053.326 067 3180      STC
053.327 300 3181      RNE           MUST HAVE '='
3182
3183 *   HAVE NAME DECODED. EXPAND INTO DESTFB+FB.NAM
3184
053.330 041 308 063 3185      LXI  H,DESTFB+FB.NAM
000.000 3186      IF  .PIF.
053.333 303 057 055 3187      JMP  CDA           CONVERT DIRECTORY FORMAT TO ASCII FORMAT
3188      ELSE  ONECOPY
3189      CALL CDA           CONVERT DIRECTORY FORMAT TO ASCII FORMAT
3190      PUSH PSW           SAVE CODE
  
```

```

3191          MVI    C,3
3192          LXI    D,DDFB
3193          LXI    H,DESTFB+FB.NAM
3194          CALL   #COMP      SEE IF DEVICE IS SYO
3195          JNE    DDF3      IS ERROR
3196          POP    PSW
3197          RET
3198
3199 DDF3      POP    PSW      ERROR, ILLEGAL DEVICE CODE
3200          MVI    A,EC.DNS
3201          STC
3202          RET
3203
3204 DDFA      DB     'SYO!*.*=' ,0   DEFAULT TARGET FOR ONECOPY
3205 DDFB      DB     'SYO'          REQUIRED DEVICE SPECIFICATION FOR ONECOPY
3206          ELSE
3207
053.336 124 124 072 3208 DDFA      DB     'TT:PIPDEST,JGL=' ,0
3209          ENDIF

3211 **      CAD - CONVERT ASCII FILE NAME INTO DIRECTORY FORMAT.
3212 *
3213 *      CAD CRACKS AN ALPHANUMERIC FILE DESCRIPTION, OF THE FORM
3214 *
3215 *      DEV:NAME.EXT
3216 *
3217 *      INTO THE PIO.XXX FIELDS.
3218 *
3219 *      THE DEFAULT BLOCK DETERMINES THE VALUES FOR THE DEVICE AND EXTENSION
3220 *      FIELDS, IF THEY ARE UNSPECIFIED. IF *CAD* IS ENTERED
3221 *      AT *CAD*, AN UNSPECIFIED NAME FIELD IS RETURNED AS ZERO BYTES.
3222 *      IF ENTERED AT *CAD.*, AN UNSPECIFIED NAME FIELD IS
3223 *      RETURNED AS 2000 (MATCH-ONE) BYTES.
3224 *
3225 *      ENTRY (DE) = POINT TO DEFAULT BLOCK
3226 *           (HL) = POINTER TO TEXT
3227 *      EXIT  'C' SET IF ERROR
3228 *           (A) = ERROR CODE
3229 *           'C' CLEAR IF OK
3230 *           (HL) = POINTS PAST FILE NAME
3231 *           'Z' SET IF NULL NAME
3232 *           'Z' CLEAR IF NON-NULL
3233 *           PIO.DIR.NAM = NAME
3234 *           PIO.DIR.EXT = EXTENSION
3235 *           PIO.DEV = DEVICE CODE
3236 *           PIO.UNI = UNIT NUMBER (ASCII DIGIT)
3237 *      USES  ALL
3238
3239
053.356 257 3240 CAD      XRA    A          SET TO NULLS
053.357 303 364 053 3241      JMP    CAD0
3242
053.362 076 200 3243 CAD,    MVI    A,2000

```

```

.....
053.364 345          3244 CAD0  PUSH  H
053.365 062 230 054 3245  STA   CADA          SAVE DEFAULT VALUE
.....
3246
3247 *              SET DEFAULTS IN PIO,XXX
3248
053.370 041 361 064 3249  LXI   H,PIO.DEV
053.373 001 003 000 3250  LXI   B,3
053.376 315 252 030 3251  CALL  $MOVE          SET DEFALUT DEVICE
054.001 001 003 000 3252  LXI   B,3
054.004 041 374 064 3253  LXI   H,PIO.DIR+DIR.EXT
054.007 315 252 030 3254  CALL  $MOVE          SET DEFAULT EXTENSION
054.012 341          3255  POP   H
054.013 315 150 057 3256  CALL  $SOB          SKIP BLANKS
054.016 006 000          3257  MVI   B,0
054.020 376 077          3258  CFI   '?'
054.022 312 051 054 3259  JE    CAD1          IS '?'
054.025 376 052          3260  CFI   '*'
054.027 312 051 054 3261  JE    CAD1          IS '*'
054.032 376 056          3262  CFI   ':'
054.034 312 051 054 3263  JE    CAD1          IS ':'
054.037 376 101          3264  CFI   'A'
054.041 332 211 054 3265  JC    CAD4          NOT NAME
054.044 376 133          3266  CFI   'Z'+1
054.046 322 211 054 3267  JNC   CAD4          NOT NAME
.....
3268
3269 *              HAVE ALPHA STRING. CRACK IT
3270
054.051 315 231 054 3271  CAD1  CALL  DNT          DECODE NEXT TOKEN
054.054 332 224 054 3272  JC    CAD5          ERROR
054.057 376 072          3273  CFI   ':'
054.061 302 114 054 3274  JNE   CAD2          NOT DEVICE
.....
3275
3276 *              HAVE EXPLICIT DEVICE
3277
054.064 043          3278  INX   H          SKIP '?'
054.065 076 003          3279  MVI   A,3
054.067 271          3280  CMP   C
054.070 332 224 054 3281  JC    CAD5          TOO MANY CHARACTERS
054.073 001 003 000 3282  LXI   B,3
054.076 345          3283  PUSH  H          SAVE (HL)
054.077 041 361 064 3284  LXI   H,PIO.DEV
054.102 315 252 030 3285  CALL  $MOVE          SET EXPLICIT DEVICE
054.105 341          3286  POP   H
054.106 315 231 054 3287  CALL  DNT          DECODE NEXT TOKEN
054.111 332 224 054 3288  JC    CAD5          ERROR
.....
3289
3290 *              DECODE NAME
3291
054.114 001 010 000 3292  CAD2  LXI   B,B          (BC) = COUNT
054.117 345          3293  PUSH  H          SAVE TEXT ADDR
.....
3294
3295 *              SEE IF NAME IS UNSPECIFIED
3296
054.120 041 364 064 3297  LXI   H,PIO.DIR+DIR.NAM
054.123 345          3298  PUSH  H          SAVE ADDRESS OF DIR.NAM
054.124 315 252 030 3299  CALL  $MOVE          MOVE IN NAME
.....

```

```

054.127 341      3300      FOP      H      (HL) = #PIO.DIR+DIR.NAM
054.130 176      3301      MOV      A,M
054.131 247      3302      ANA      A
054.132 302 150 054 3303      JNZ      CAD2.6      IS SPECIFIED
054.135 072 230 054 3304      LDA      CADA      (A) = FILL CHARACTER
054.140 016 010      3305      MVI      C,B      (C) = COUNT
054.142 167      3306      MOV      M,A      CAD2.4
054.143 043      3307      INX      H
054.144 015      3308      INCR     C
054.145 302 142 054 3309      JNZ      CAD2.4
054.150 341      3310      FOP      H      CAD2.6
054.151 176      3311      MOV      A,M      (A) = DELIMITER
054.152 376 056      3312      CPI      ' '
054.154 302 207 054 3313      JNE      CAD3      NOT EXTENSION
3314
3315 *      HAVE EXPLICIT EXTENSION
3316
054.157 043      3317      INX      H
054.160 315 231 054 3318      CALL     DNT
054.163 332 224 054 3319      JC       CAD5      ERROR
054.166 076 003      3320      MVI      A,3
054.170 271      3321      CMP      C
054.171 332 224 054 3322      JC       CAD5      TOO LONG
054.174 001 003 000 3323      LXI      B,3
054.177 345      3324      PUSH     H      SAVE TEXT POINTER
054.200 041 374 064 3325      LXI      H,PIO.DIR+DIR.EXT
054.203 315 252 030 3326      CALL     $MOVE      MOVE EXTENSION
054.206 341      3327      POP      H
3328
3329 *      DONE WITH NAME. MUST HAVE LEGIT DELIMITER
3330
054.207 006 001      3331      CAD3     MVI      B,1      (B) = NAME PRESENT FLAG
3332
3333 *      END OF NAME. EXIT
3334 *      (B) = 0 IF NULL, (B) <> 0 IF NON-NULL
3335
054.211 315 150 057 3336      CAD4     CALL     $SOB      SKIP BLANKS
054.214 176      3337      MOV      A,M      (A) = NEXT CHARACTER
054.215 315 363 056 3338      CALL     $CFD      CHECK FILE NAME DELIMITER
054.220 330      3339      RC       ERROR
054.221 170      3340      MOV      A,B
054.222 247      3341      ANA      A      SET 'Z' IF NULL
054.223 311      3342      RET
3343
3344 *      ERROR
3345
054.224 076 007      3346      CAD5     MVI      A,EC.IFN      ILLEGAL FILE NAME
054.226 067      3347      STC
054.227 311      3348      RET
3349
054.230 000      3350      CADA     DR       0      FILL CHARACTER FOR OMITTED NAME FIELD

```



```

3352 **      DNT - DECODE NEXT TOKEN.
3353 *
3354 *      DNT COPIES THE NEXT ALPHANUMERIC FIELD INTO A ZERO-FILLED WORK AREA.
3355 *
3356 *      ENTRY (HL) = TEXT POINTER
3357 *      EXIT  'C' SET IF ERROR
3358 *          'C' CLEAR IF OK
3359 *          (A) = DELIMITER CHARACTER
3360 *          (HL) UPDATED TO DELIMITER CHARACTER
3361 *          (DNTA) = STRING
3362 *          (C) = LENGTH
3363 *          (DE) = #DNTA
3364 *      USES  ALL
3365
054.231 021 343 054 3367 DNT LXI  D,DNTA
054.234 016 011 3368 DNT HVI  C,9          (C) = SIZE OF DNTA
054.236 101 3369 MOV  B,C          (B) = MAX ALLOWED +1
054.237 257 3370 XRA  A
054.240 022 3371 DNT1 STAX D          ZERO BUFFER
054.241 023 3372 INX  D
054.242 015 3373 DCR  C
054.243 302 240 054 3374 JNZ  DNT1
054.246 021 343 054 3375 LXI  D,DNTA
3376
3377 *      COPY CHARACTERS
3378
054.251 176 3379 DNT2 MOV  A,M
054.252 376 077 3380 CPI  '?'
054.254 076 200 3381 MVI  A,200D
054.256 312 313 054 3382 JE   DNT3          IS MATCHONE
054.261 176 3383 MOV  A,M
054.262 376 052 3384 CPI  '*'
054.264 312 325 054 3385 JE   DNT5          IS WILDCARD
054.267 376 060 3386 CPI  '0'
054.271 332 336 054 3387 JC   DNT4          NOT ALPHANUMERIC
054.274 376 072 3388 CPI  '9'+1
054.276 332 313 054 3389 JC   DNT3          NUMERIC
054.301 376 101 3390 CPI  'A'
054.303 332 336 054 3391 JC   DNT4          DELIMITER
054.306 376 133 3392 CPI  'Z'+1
054.310 322 336 054 3393 JNC  DNT4          DELIMITER
3394
3395 *      HAVE GOOD CHARACTER
3396
054.313 022 3397 DNT3 STAX D          STORE CHAR
054.314 023 3398 INX  D
054.315 043 3399 INX  H
054.316 014 3400 INR  C          COUNT
054.317 005 3401 DCR  B          LIMIT DECREMENT
054.320 302 251 054 3402 JNZ  DNT2          NOT OVERFLOW
3403
3404 *      OVERFLOW
3405
054.323 067 3406 STC          FLAG ERR
054.324 311 3407 RET

```

```

3408
3409 *      IS '*' WILDCARD
3410
054.325 076 200 3411 DNT5 MVI      A,2000
054.327 022 3412 STAX     D
054.330 023 3413 INX     B
054.331 005 3414 DCR     B
054.332 302 325 054 3415 JNZ     DNT5      FILL WITH MATCH ONE
054.335 043 3416 INX     H      SKIP '*'
3417
3418 *      END OF STRING
3419
054.336 247 3420 DNT4 ANA     A      CLEAR 'C'
054.337 021 343 054 3421 LXI     D,DNTA    SET POINTER
054.342 311 3422 RET
3423
054.343 3424 DNTA DS     9      WORK AREA

3426 **     ERM - EXPAND BUFFER TO MAXIMUM.
3427 *
3428 *      ERM IS CALLED TO EXPAND THE BUFFER 'BUF' TO THE MAXIMUM SIZE.
3429 *      WHICH DOES NOT REQUIRE THE OVERLAYING OF THE SYSTEM.
3430 *
3431 *      ENTRY NONE
3432 *      EXIT (BUFSIZ) = BUFFER SIZE (MULTIPLE OF 256)
3433 *      USES ALL
3434
054.354 052 320 040 3436 ERM LHL D S.SYSM
054.357 345 3437 PUSH  H
054.360 052 350 040 3438 LHL D S.DFWA
054.363 021 006 000 3439 LXI D,DVLO*OVL.ENS+OVL.FLB
054.366 031 3440 DAD D (HL) = ADDR. OF DVLO OVL.FLB ENTRY
054.367 076 002 3441 MVI A,OVL.RES
054.371 246 3442 ANA M
054.372 021 010 000 3443 LXI D,OVL.ENS
054.375 031 3444 DAD D (HL) = ADDR. OF OVL1 OVL.FLB ENTRY
000.000 3445 ERNZ OVL1-OVL0-1
054.376 246 3446 ANA M
054.377 302 014 055 3447 JNZ EBH1 OVL0 AND OVL1 ARE PERM. RESIDENT
055.002 052 324 040 3448 LHL D S.OMAX
055.005 315 224 030 3449 CALL $CHL
055.010 353 3450 XCHG
055.011 341 3451 POP H
055.012 031 3452 DAD D (HL) = NEW ADDRESS SOUGHT
055.013 345 3453 PUSH H
3454
055.014 341 3455 EBH1 POP H
055.015 021 372 377 3456 LXI D,-6
055.020 031 3457 DAD D (HL) = NEW ADDRESS SOUGHT
055.021 377 052 3458 DB SYSCALL,SETTP
055.023 332 054 051 3459 JC IERR1 INTERNAL ERROR 1
055.026 052 322 040 3460 LHL D S.USRM

```

```

000.000          3461      IF      .PIP.
055.031  353          3462      XCHG
055.032  052 267 063 3463      LHL  D, BUFTR
055.035  315 224 030 3464      CALL #CHL      (HL) = - BUFFER FWA
055.040  031          3465      DAD  D
055.041  056 000          3466      MVI  L, 0
055.043  042 271 063 3467      SHLD BUFSIZ
055.046  076 001          3468      MVI  A, BUFMINL/256-1
055.050  274          3469      CMP  H
055.051  330          3470      RC      IF OK
055.052  076 021          3471      MVI  A, EC.NEM
055.054  303 265 051 3472      JMP  ERROR      NOT ENOUGH MEMORY
3473
3474      ELSE
3475
3476      MOV  A, H      (A) = .LIMIT/256
3477      STA  OBUFLIM   SET LIMIT
3478      RET
3479      ENDF

```

  

```

3481 **      CDA = CONVERT DIRECTORY FORMAT TO ASCII.
3482 *
3483 *      CDA COPIES A DIRECTORY ENTRY FROM PIO,XXX TO A TARGET FIELD.
3484 *      THE DEVICE SPECIFICATION (IN PIO.DEV AND PIO.UNI) IS ALSO ENCODED.
3485 *      THE TARGET FIELD IS LEFT IN THE FORM:
3486 *
3487 *      DEV;NAME,XXX.<00>
3488 *
3489 *      ENTRY (HL) = FWA NAME FIELD
3490 *      EXIT (A) = 0, HAVE WILDCARD
3491 *           = 1, NO WILDCARDS USED
3492 *      'C' CLEAR
3493 *      USES  ALL
3494
3495
055.057  001 000 003 3496 CDA  LXI  B, 3*256      (B) = CHARACTER COUNT, (C) = WILDCARD FLAG
055.062  021 361 064 3497      LXI  D, PIO.DEV
055.065  315 123 055 3498      CALL CDAS      COPY IT
055.070  066 072          3499      MVI  M, ' '
055.072  043          3500      INX  H
055.073  006 010          3501      MVI  B, B
055.075  021 364 064 3502      LXI  D, PIO.DIR+DIR.NAM
055.100  315 123 055 3503      CALL CDAS      COPY IT
055.103  066 056          3504      MVI  M, ' '
055.105  043          3505      INX  H
055.106  006 003          3506      MVI  B, 3
000.000          3507      ERRNZ DIR.EXT-DIR.NAM-8
055.110  315 123 055 3508      CALL CDAS      COPY IT
055.113  066 000          3509      MVI  M, 0      FLAG END OF NAME
055.115  171          3510      MOV  A, C      (A) (BIT 7) = 1 IF WILDCARDS
055.116  007          3511      RLC
055.117  057          3512      CMA
055.120  346 001          3513      ANI  1      =0 IF WILDCARD

```

```

055.122 311          3514      RET
                                3516  **   CDA5 - CONVERT DIRECTORY FIELD TO ASCII.
                                3517  *
                                3518  *   ZEROS ARE IGNORED; 2000 WILDCARDS ARE MAPPED TO '?'
                                3519  *
                                3520  *   ENTRY  (DE) = FROM
                                3521  *           (HL) = TO
                                3522  *           (B) = COUNT
                                3523  *           (C) = ORA ACCUMULATOR
                                3524  *   EXIT  (DE) ADVANCED
                                3525  *           (HL) = (HL)+(B)
                                3526  *           (C) = (C) .OR. (FROM CHARACTERS PROCESSED)
                                3527  *   USES  ALL
                                3528  *
                                3529
055.123 032          3530  CDA5  LDAX  D           (A) = CHARACTER
055.124 261          3531          ORA  C
055.125 117          3532          MOV  C,A
055.126 032          3533          LDAX  D
055.127 023          3534          INX  D
055.130 247          3535          ANA  A
055.131 312 143 055 3536          JZ   CDA7          IS 00
055.134 362 141 055 3537          JF  CDA6          NOT 2000
055.137 076 077    3538          MVI  A,'?'
055.141 167          3539  CDA6  MOV  M,A
055.142 043          3540          INX  H           INCREMENT TO
055.143 005          3541  CDA7  DCR  B
055.144 302 123 055 3542          JNZ  CDA5          IF MORE TO GO
055.147 311          3543          RET

                                3545  **   EWS - EXPAND WILDCARD SPECIFICATION
                                3546  *
                                3547  *   EWS ENTERS THE FILE NAME IN PIO,XXX INTO THE MANAGED TABLE
                                3548  *   NAMTAB. IF THE FILE NAME CONTAINS WILDCARDS, THE DIRECTORY
                                3549  *   IS READ FOR ELIGIBLE FILES.
                                3550  *
                                3551  *   ENTRY  PIO,XXX = FILE NAME
                                3552  *   EXIT  'C' CLEAR IF OK
                                3553  *           'C' SET IF ERROR
                                3554  *   USES  ALL
                                3555  *
                                3556
055.150 315 307 052 3557  EWS   CALL  AEN           TRY TO ENTER IT
055.153 320          3558          RNC           NO WILDCARDS, AM DONE
                                3559
                                3560  *   IS WILDCARD. LOOK UP DEVICE TYPE
                                3561
055.154 052 326 063 3562          LHLD  NAMLLEN
055.157 021 133 065 3563          LXI  D,NAMTAB-FB,NAML
055.162 031          3564          DAD  D           (HL) = ADDRESS OF LAST ENTRY
055.163 315 356 053 3565          CALL  CAD           CONVERT ASCII NAME TO DIRECTORY FORMAT

```

PIP - PERIPHERAL INTERCHANGE PROGRAM  
SUBROUTINES

HEATH HBASH V1.4 01/20/78  
14:41:00 16-MAY-80

PAGE 77

.....						
055.166	330		3566	RC		ERROR
055.167	052 326 063		3567	LHLD	NAMTLEN	
055.172	021 357 377		3568	LXI	D,-FB,NAML	
055.175	031		3569	DAD	D	
055.176	042 326 063		3570	SHLD	NAMTLEN	REMOVE WILDCARD FROM TABLE
055.201	315 244 060		3571	CALL	*MOVEL	
055.204	003 000 361		3572	DW	3,PID,DEV,DIRNAM	SET DIRECTORY NAME IN XXX:DIRECT.SYS
055.212	315 244 060		3573	CALL	*MOVEL	
055.215	013 000 364		3574	DW	8+3,PID,DIR+DIR.NAM,EWSC	SAVE WILDCARD PATTERN
055.223	001 012 056		3575	LXI	B,EWSB	
055.226	041 250 063		3576	LXI	H,DIRNAM	
055.231	377 053		3577	DB	SYSCALL,,DECODE	GET INFORMATION ABOUT DEVICE
055.233	330		3578	RC		ERROR
055.234	072 012 056		3579	LDA	EWSB	SEE IF A DIRECTORY DEVICE
055.237	346 001		3580	ANI	DT,DD	
055.241	076 005		3581	MVI	A,EC,INS	ASSUME DEVICE NOT SUITABLE
055.243	067		3582	STC		
055.244	310		3583	RZ		ERROR
			3584			
			3585	*	IS DIRECTORY DEVICE,, OPEN DIRECTORY	
			3586			
055.245	041 250 063		3587	LXI	H,DIRNAM	
055.250	076 002		3588	MVI	A,CN,DIR	
055.252	377 042		3589	DB	SYSCALL,,OPENR	
055.254	076 200		3590	MVI	A,PEC,DF	
055.256	330		3591	RC		DEVICE FORMAT FAILURE
			3592			
			3593	*	READ DIRECTORY ENTRYS FOR MATCH	
			3594			
055.257	315 063 056		3595	EWS1	CALL	GDWP DE = DIRECTORY WORKSPACE PTR /79.11.GC/
055.262	001 000 002		3596	LXI	R,512	
055.265	076 002		3597	MVI	A,CN,DIR	
055.267	325		3598	PUSH	D	SAVE ADDRESS
055.270	377 004		3599	DB	SYSCALL,,READ	READ BLOCK
055.272	341		3600	POP	H	(HL) = DIRECTORY ADDRESS
055.273	332 377 055		3601	JC	EWS7	ALL DONE
			3602			
			3603	*	LOOK AT DIRECTORY BLOCK FOR MATCHES	
			3604			
055.276	345		3605	PUSH	H	
055.277	315 071 056		3606	CALL	GDWP,	/79.11.GC/
055.302	315 301 057		3607	CALL	*INDLR	/79.11.GC/
055.305	373 001		3608	DW	DIS.ENL	A = DIRECTORY ENTRY LENGTH /79.11.GC/
055.307	341		3609	POP	H	/79.11.GC/
			3610			
055.310	117		3611	MOV	C,A	(C) = LENGTH
			3612			
			3613	*	CHECK NEXT ENTRY	
			3614			
055.311	176		3615	EWS3	MOV	A,M (A) = 1ST CHAR THIS ENTRY
055.312	247		3616	ANA	A	
055.313	312 257 055		3617	JZ	EWS1	END OF BLOCK
000.000			3618	ERRNZ	DF,EMP-377Q	
055.316	074		3619	INR	A	
055.317	312 371 055		3620	JZ	EWS6	ENTRY EMPTY
000.000			3621	ERRNZ	DF,CLR-376Q	
.....						

```

055.322 074 3622 INR A
055.323 312 377 055 3623 JZ EWS7 END OF LIST
055.326 315 173 053 3624 CALL CFE CHECK FOR FILE ELIGIBILITY
055.331 302 371 055 3625 JNZ EWS6 NOT TO PROCESS
055.334 345 3626 PUSH H
055.335 021 050 056 3627 LXI D,EWSC
055.340 006 013 3628 MVI B,8+3
055.342 315 246 053 3629 CALL CWM CHECK WILDCARD MATCH
055.345 302 370 055 3630 JNZ EWS4 NO MATCH
3631
3632 * HAVE MATCH, ADD TO LSIT
3633
055.350 321 3634 POP D (DE) = FROM
055.351 325 3635 PUSH D
055.352 305 3636 PUSH B SAVE (C)
055.353 001 013 000 3637 LXI B,8+3
055.356 041 364 064 3638 LXI H,PIB.DIR+DIR.NAM
055.361 315 252 030 3639 CALL $MOVE
055.364 315 307 052 3640 CALL AEN ADD TO TABLE
055.367 301 3641 POP B RESTORE (C)
3642
3643 * LOOKUP NEXT ENTRY
3644
055.370 341 3645 EWS4 POP H
055.371 006 000 3646 EWS6 MVI B,0
055.373 011 3647 DAD B POINT TO NEXT
055.374 303 311 055 3648 JMP EWS3
3649
3650 * ALL DONE, CLOSE DIRECTORY FILE
3651
055.377 076 002 3652 EWS7 MVI A,CN.DIR
056.001 377 046 3653 DB SYSCALL,CLOSE
056.003 311 3654 RET
3655
056.004 123 131 060 3656 EWSA DB 'SY0',200Q,200Q,200Q
3657
056.012 3658 EWSB DS 30
3659
056.050 3660 EWSC DS 8+3 WILDCARD PATTERN FOR DIRECTORY SEARCH

3662 ** GDWP - GET DIRECTORY WORKSPACE POINTER /79.11.GC/
3663 *
3664 * GDWP GETS THE DIRECTORY WORKSPACE POINTER
3665 *
3666 * ENTRY: NONE
3667 *
3668 * EXIT: DE = DIRECTORY WORKSPACE POINTER
3669 *
3670 * USES: DE
3671 *
3672
056.063 353 3673 GDWF XCHG
056.064 315 071 056 3674 CALL GDWF, HL = DIRECTORY WORKSPACE POINTER

```

```

056.067 353      3675      XCHG
056.070 311      3676      RET
                   3677
056.071 052 120 041 3678 GDWF.  LHL D  S,SCR      HL = SYSTEM SCRATCH
056.074 311      3679      RET
  
```

```

3681 **      INA - INCREASE NAMTAB ALLOCATION.
3682 *
3683 *      INA IS CALLED TO INCREASE THE NAMTAB ALLOCATION. THE
3684 *      BUFFER AREA IS MOVED UP TO MAKE ROOM.
3685 *
3686 *      ENTRY  NONE
3687 *      EXIT  NONE
3688 *      USES  A,F,H,L
3689
  
```

```

056.075 041 331 063 3690 INA  LXI  H,NAMTMAX+1
056.100 064      3691      INR  M      INCREMENT LENGTH
056.101 041 270 063 3692      LXI  H,BUFFTR+1
056.104 064      3693      INR  M      MOVE BUFFER
056.105 052 271 063 3694      LHL D  BUFSIZ
056.110 174      3695      MOV  A,H
056.111 265      3696      ORA  C
056.112 076 021  3697      MVI  A,EC,NEM      FLAG OUT OF MEMORY IF BUFFER NOT EMPTY
056.114 302 265 051 3698      JNZ  ERROR
056.117 305      3699      PUSH B
056.120 325      3700      FUSH D
056.121 315 250 056 3701      CALL SBE      NOTIFY SYSTEM
056.124 321      3702      POP  D
056.125 301      3703      POP  B
056.126 311      3704      RET
  
```

```

3706 **      LSN - LOCATE SOURCE NAME
3707 *
3708 *      LSN SCANS THE COMMAND LINE FOR THE FIRST SOURCE FILE NAME.
3709 *
3710 *      ENTRY  NONE
3711 *      EXIT  (HL) = 1ST FILE NAME FWA
3712 *      USES  A,F,H,L
3713
  
```

```

056.127 041 034 065 3714 LSN  LXI  H,LINE
056.132 176      3715 LSN1  MOV  A,M
056.133 043      3716      INX  H
056.134 376 075  3717      CPI  '='
056.136 310      3718      RE   GOT IT
056.137 247      3719      ANA  A
056.140 302 132 056 3720      JNZ  LSN1      MORE LINE
056.143 041 034 065 3721      LXI  H,LINE IS NO =
056.146 311      3722      RET
  
```

```

3724 **      MWN - MERGE WILDCARD NAMES.
3725 *
3726 *      MWN MERGES A COMPLETELY SPECIFIED FILENAME WITH A WILDCARDED COMPLETELY
3727 *      SPECIFIED FILE NAME.
3728 *
3729 *      BOTH FILE NAMES SHOULD HAVE THE SAME DEVICE SPECIFICATION.
3730 *
3731 *      FILE NAME FORMAT:
3732 *
3733 *      DEV:NAMEXXXX,EXT 00
3734 *
3735 *      ENTRY      (BC) = ADDRESS OF WILDCARDED ASCII NAME
3736 *                (DE) = ADDRESS OF NON-WC ASCII NAME
3737 *                (HL) = ADDRESS FOR RESULTANT ASCII NAME
3738 *      EXIT      NONE
3739 *      USES      ALL
3740
3741
056.147 345      3742 MWN      PUSH      H          SAVE TARGET ADDRESS
056.150 305      3743          PUSH      B          SAVE WC PATTERN
056.151 353      3744          XCHG          (HL) = MASTER NAME
056.152 315 356 053 3745          CALL      CAD          CONVERT TO DIRECTORY FORMAT
056.155 315 244 060 3746          CALL      $MOVE1
056.160 013 000 364 3747          DW      8+3,PI0,DIR,MWNA      (MWNA) = DECODED MASTER
056.166 341      3748          POP       H          (HL) = WC PATTERN
056.167 315 356 053 3749          CALL      CAD          (PI0,DIR) = WC PATTERN
056.172 021 340 063 3750          LXI      D,MWNA      (DE) = MASTER PATTERN
056.175 041 364 064 3751          LXI      H,PI0,DIR      (DE) = WC PATTERN ADDRESS
056.200 016 013      3752          MVI      C,8+3      MERGE NAME AND EXTENSION
3753
3754 *      MERGE NAMES
3755
056.202 176      3756 MWN1     MOV       A,M          (A) = WC PATTERN
056.203 247      3757          ANA      A
056.204 362 210 056 3758          JF       MWN2     USE THIS
056.207 032      3759          LDAX   D          IS MATCH CHARACTER, USE MASTER INSTEAD
056.210 167      3760 MWN2     MOV       M,A          STORE CHARACTER
056.211 023      3761          INX      D
056.212 043      3762          INX      H
056.213 015      3763          ICR      C
056.214 302 202 056 3764          JNZ      MWN1     MERGE TILL DONE
056.217 341      3765          POP       H          (HL) = TARGET ADDRESS
056.220 303 057 055 3766          JMP      CIA          CONVERT DIRECTORY FORMAT TO ASCII

```

```

3768 **      REN - REMOVE ENTRY FROM *NAMTAB*
3769 *
3770 *      REN REMOVES THE FIRST 'FB.NAML' BYTES FROM NAMTAB.
3771 *
3772 *      THE AMOUNT (FB.NAML) IS REMOVED FROM THE SIZE OF THE TABLE, THE
3773 *      TABLE IS NOT CHECKED FOR UNDERFLOW, THE CALLER MUST GUARANTEE THE
3774 *      PRESENCE OF AT LEAST FB.NAML BYTES IN NAMTAB.
3775 *
3776 *      ENTRY      NONE

```



```

3777 *   EXIT   NONE
3778 *   USES   ALL
3779
3780
056.223 052 326 063 3781 REN  LHL  NAMLEN
056.224 021 357 377 3782   LXI  D,-FB,NAML
056.231 031          3783   DAD  D          REMOVE COUNT FROM LEN
056.232 042 326 063 3784   SHLD NAMLEN
056.235 104          3785   MOV  B,H
056.236 115          3786   MOV  C,L          (BC) = REMAINING LENGTH
056.237 021 175 065 3787   LXI  D,NAMTAB+FB,NAML      (DE) = START OF 2ND ENTRY
056.242 041 154 065 3788   LXI  H,NAMTAB
056.245 303 252 030 3789   JMP  $MOVE          MOVE DOWN AND RETURN

3791 **   SBE - SET BUFFER EMPTY.
3792 *
3793 *   THE SYSTEM IS NOTIFIED.
3794 *
3795 *   ENTRY   NONE
3796 *   EXIT   NONE
3797 *   USES   ALL
3798
3799
056.250 041 000 000 3800 SBE  LXI  H,0
056.253 042 271 063 3801   SHLD BUFSIZ
056.256 052 267 063 3802   LHL  BUFFTR          (HL) = BUFFER FWA (AND LWAT)
056.261 043          3803   INX  H
056.262 043          3804   INX  H
056.263 377 052      3805   DB   SYSCALL,SETTP
056.265 320          3806   RNC  DK
056.266 303 265 051 3807   JMP  ERROR          NOT ENOUGH ROOM

3809 **   SDD - SET DEFAULT DEFAULT.
3810 *
3811 *   SDD IS CALLED TO SETUP THE CURRENT DEFAULT DEVICE
3812 *   AND EXTENSION TO 'SY0' AND <NULL>, RESPECTIVELY.
3813 *
3814 *   ENTRY   NONE
3815 *   EXIT   NONE
3816 *   USES   NONE
3817
3818
056.271 315 054 031 3819 SDD  CALL  $SAVALL
056.274 315 244 060 3820   CALL $MOVE1
056.277 006 000 310 3821   DW  6,SDDA,DEFAULT  SET DEFAULT DEFAULT
056.305 303 047 031 3822   JMP  $RSTALL        RESTORE AND RETURN
3823
056.310 123 131 060 3824 SDDA DB   'SY0',0,0,0  DEFAULT DEFAULT VALUES
  
```

```

3826 **      SFS - SKIP FILE SEPERATOR.
3827 *
3828 *      SFS IS CALLED TO SKIP OVER THE CHARACTERS SEPERATING ONE
3829 *      FILE NAME FROM ANOTHER ON THE LINE. THE FILES MAY BE SEPERATED
3830 *      BY BLANKS OR A COMMA ALONE, OR BY BLANKS WITH A COMMA. THE
3831 *      SYNTAX IS
3832 *
3833 *      <BLANKS> <,> <BLANKS>
3834 *
3835 *      ONE, TWO OR ALL THREE FIELDS MAY BE PRESENT.
3836 *
3837 *      ENTRY (HL) = POINT TO START OF SEP FIELD
3838 *      EXIT (HL) ADVANCED PAST SEPERATOR FIELD
3839 *      USES A,F,H,L
3840
3841
056.316 315 150 057 3842 SFS CALL $SOB SKIP BLANKS
056.321 176 3843 MOV A,H
056.322 376 054 3844 CPI ','
056.324 302 330 056 3845 JNE SFS1 NOT ,
056.327 043 3846 INX H SKIP ,
056.330 303 150 057 3847 SFS1 JMP $SOB GET ANY MORE BLANKS AND EXIT
  
```

```

3849 **      SND - SET NEW DEFAULTS.
3850 *
3851 *      SND IS CALLED TO SET A NEW DEFAULT DEVICE AND EXTENSION
3852 *      IN THE 'DEFAULT' AREA.
3853 *
3854 *      ENTRY PIO.DEV = DEVICE CODE
3855 *      PIO.UNI = UNIT #
3856 *      PIO.DIR+DIR.EXT = EXTENSION
3857 *      EXIT NONE
3858 *      USES NONE
3859
3860
056.333 315 054 031 3861 SND CALL $SAVALL SAVE REGS
000.000 3862 ERNZ PIO.UNI-PIO.DEV-2
056.336 315 244 060 3863 CALL $MOVE1
056.341 003 000 3864 DW 3
056.343 361 064 3865 DW PIO.DEV
056.345 332 063 3866 DW DEFAULT
056.347 315 244 060 3867 CALL $MOVE1
056.352 003 000 3868 DW 3
056.354 374 064 3869 DW PIO.DIR+DIR.EXT
056.356 335 063 3870 DW DEFAULT+3
056.360 303 047 031 3871 JMP $RSTALL RETURN
  
```

056.363

3874

XTEXT CFD

3876X \*\* \*CFD - CHECK FILE DELIMITER.  
3877X \*  
3878X \* \*CFD CHECKS AN ASCII CHARACTER TO SEE IF IT IS A LEGAL FILE  
3879X \* NAME DELIMITER. LEGAL DELIMITERS ARE  
3880X \*  
3881X \* , = / <BLANK> <00>  
3882X \*  
3883X \* ENTRY (A) = CHARACTER  
3884X \* EXIT (C) CLEAR IF OK  
3885X \* (C) SET IF ERROR  
3886X \* (A) = ERROR CODE  
3887X \* USES A,F  
3888X  
3889X

056.363	247		3890X	%CFD	ANA	A		
056.364	310		3891X		RZ		IS	00
056.365	376	054	3892X		CPI	'/'		
056.367	310		3893X		RE		IS	'
056.370	376	075	3894X		CPI	'='		
056.372	310		3895X		RE		IS	=
056.373	376	057	3896X		CPI	'/'		
056.375	310		3897X		RE		IS	/
056.376	376	040	3898X		CPI	'/'		
057.000	310		3899X		RE		IS	'
057.001	076	007	3900X		MVI	A,EC,IFN		ILLEGAL FILE NAME
057.003	067		3901X		STC			
057.004	311		3902X		RET			
057.005			3903		XTEXT	TYPCC		

3905X \*\* \*TYPCC - TYPE A CHARACTER STRING BY COUNT.  
3906X \*  
3907X \* \*TYPCC TYPES A STRING OF CHARACTERS. THE CALLER SUPPLIES  
3908X \* THE CHARACTER ADDRESS AND COUNT.  
3909X \*  
3910X \* ENTRY (HL) = ADDRESS  
3911X \* (A) = COUNT  
3912X \* EXIT (HL) = LAST CHARACTER ADDRESS+1  
3913X \* USES A,F,H,L

057.005			3914X					
			3915X					
057.005			3916X	%TYPCC	EQU	*		
057.005	247		3917X		ANA	A		
057.006	310		3918X		RZ			NOTHING TO TYPE
057.007	365		3919X		PUSH	PSW		SAVE COUNT
057.010	176		3920X		MOV	A,M		(A) = CHARACTER
057.011	043		3921X		INX	H		
057.012	377	002	3922X		DB	SYSCALL,SCOUT		
057.014	361		3923X		PDF	PSW		

057.015 075 3924X DCR A  
057.016 303 005 057 3925X JMP \$TYPCC  
057.021 3926 XTEXT WER

3928X \*\* \$WER - WRITE ENABLE RAM.  
3929X \*  
3930X \* \$WER IS CALLED TO ENABLE WRITING TO THE H17 CONTROLLER'S  
3931X \* RAM AREA.  
3932X \*  
3933X \* ENTRY NONE  
3934X \* EXIT NONE  
3935X \* USES NONE

031.241 3936X  
3937X  
3938X \$WER EQU 31241A IN H17 ROM

3940X \*\* \$WDR - WRITE DISABLE RAM.  
3941X \*  
3942X \* \$WDR IS CALLED TO DISABLE WRITING TO THE H17 CONTROLLER'S  
3943X \* RAM AREA.  
3944X \*  
3945X \* ENTRY NONE  
3946X \* EXIT NONE  
3947X \* USES NONE

031.222 3948X  
3949X  
057.021 3950X \$WDR EQU 31222A IN H17 ROM  
3951 XTEXT ZERO

3953X \*\* \$ZERO - ZERO MEMORY  
3954X \*  
3955X \* \$ZERO ZEROS A BLOCK OF MEMORY.  
3956X \*  
3957X \* ENTRY (HL) = ADDRESS  
3958X \* (B) = COUNT  
3959X \* EXIT (A) = 0  
3960X \* USES A,B,F,H,L

031.212 3961X  
3962X  
057.021 3963X \$ZERO EQU 31212A IN H17 ROM  
3964 XTEXT MUB6

```

3966X **      $MUB6 - MULTIPLY 8X18 UNSIGNED.
3967X *
3968X *      $MUB6 MULTIPLIES A 18 BIT VALUE BY A 8
3969X *      BIT VALUE.
3970X *
3971X *      ENTRY   (A) = MULTIPLIER
3972X *             (DE) = MULTIPLICAND
3973X *      EXIT    (HL) = RESULT
3974X *             'Z' SET IF NOT OVERFLOW
3975X *      USES   A,F,H,L
3976X *
3977X *
031.007      3978X $MUB6 EQU   31007A      IN H17 ROM
057.021      3979      XTEXT  CCO
    
```

```

3981X **      $CCO - CLEAR CONTROL-0
3982X *
3983X *      $CCO IS CALLED TO CLEAR THE EFFECT OF THE CTL-0 CHARACTER.
3984X *
3985X *      ENTRY   NONE
3986X *      EXIT    NONE
3987X *      USES   NONE
3988X *
057.021      3989X $CCO CALL  $SAVALL      SAVE REGISTERS
057.024      3990X      MVI   A,I,CONFL
057.026      3991X      LXI   B,CO,FLG      CLEAR CO,FLG
057.031      3992X      DB    SYSCALL,CONSL
057.033      3993X      JMP   $RSTALL      RESTORE REGISTERS AND RETURN
057.036      3994X      XTEXT  GNL
3995X
    
```

```

3997X **      $GNL - GUARANTEE NEW LINE.
3998X *
3999X *      $GNL GUARANTEES THE START OF A NEW LINE BY ISSUING A CRLF
4000X *      IF THE CURSOR IS NOT AT COLUMN 1..
4001X *
4002X *      ENTRY   NONE
4003X *      EXIT    NONE
4004X *      USES   ALL
4005X *
057.036      4006X $GNL MVI   A,I,CUSOR
057.040      4007X      LXI   B,0
057.043      4008X      DB    SYSCALL,CONSL      READ CURSOR
057.045      4009X      ICR   A
057.046      4010X      RZ
057.047      4011X      JMP   $CALF      AT COLUMN 1
057.052      4012X      XTEXT  MLU      NEW LINE
4013X
    
```

```

4015X **      MLU - MAP LOWER CASE LINE TO UPPER CASE.
4016X *
4017X *      MLU MAPS THE LOWER CASE ALPHABETICS IN A LINE TO UPPER CASE.
4018X *
4019X *      ENTRY (HL) = LINE FWA
4020X *      EXIT NONE
4021X *      USES NONE
4022X
4023X
057.052 365 4024X $MLU PUSH PSW SAVE (PSW)
057.053 345 4025X PUSH H SAVE FWA
057.054 053 4026X DCX H ANTICIPATE INX H
057.055 043 4027X $MLUI INX H
057.056 176 4028X MOV A,M (A)= CHARACTER
057.057 315 072 057 4029X CALL $MCU MAP CHAR TO UPPER
057.062 167 4030X MOV M,A
057.063 247 4031X ANA A
057.064 302 055 057 4032X JNZ $MLUI MORE TO GO
057.067 341 4033X POP H RESTORE (HL)
057.070 361 4034X POP PSW RESTORE (PSW)
057.071 311 4035X RET
057.072 4036 XTEXT MCU

```

```

4038X **      MCU - MAP LOWER CASE TO UPPER CASE.
4039X *
4040X *      MCU MAPS A LOWER CASE ALPHABETIC TO UPPER
4041X *      CASE.
4042X *
4043X *      ENTRY (A) = CHARACTER
4044X *      EXIT (A) = CHARACTER RESULT
4045X *      USES A,F
4046X
4047X
057.072 376 141 4048X $MCU CPI 'a'
057.074 330 4049X RC NOT LOWER CASE
057.075 376 173 4050X CPI 'z'+1
057.077 320 4051X RNC NOT LOWER CASE
057.100 326 040 4052X SUI 'a'-'A'
057.102 311 4053X RET
057.103 4054 XTEXT RTL

```

```

4054X **      $RTL - READ TEXT LINE.
4057X *
4058X *      $RTL READS A LINE FROM THE TERMINAL.
4059X *
4060X *      CHARACTER ARE ACCEPTED FROM THE TERMINAL, RUBOUT AND BACKSPACE
4061X *      CHARACTERS ARE PROCESSED, WHEN A CARRIAGE RETURN IS ENTERED,
4062X *      $RTL RETURNS.
4063X *
4064X *      ENTRY (HL) = BUFFER FWA

```

```

4085X *      EXIT      'C' CLEAR IF OK
4066X *      DATA IN BUFFER
4067X *      (A) = TEXT LENGTH
4068X *      'C' SET IF CTL-D STRUCK
4069X *      USES     A,F
4070X
4071X
057.103 315 112 057 4072X $RTL CALL $RTL      *RTL IN UPPER CASE
057.106 330          4073X RC              CTL-D
057.107 303 052 057 4074X JMP $HLU      MAP LINE TO UPPER CASE
4075X
057.112          4076X $RTL EQU *
057.112 345          4077X PUSH H          SAVE FWA
057.113 315 267 060 4078X $RTL1 CALL $RCHAR
057.116 376 004          4079X CPI CTLD
057.120 312 145 057 4080X JE $RTL2      CTL-D STRUCK
057.123 167          4081X MOV M,A
057.124 043          4082X INX H
057.125 376 012          4083X CPI NL
057.127 302 113 057 4084X JNE $RTL1
057.132 053          4085X DCX H
057.133 066 000          4086X MVI M,0
057.135 043          4087X INX H
4088X
4089X *      ALL DONE, COMPUTE LENGTH
4090X
057.136 353          4091X XCHG      (DE) = LWA+1
057.137 343          4092X XTHL      (HL) = FWA
057.140 173          4093X MOV A,E
057.141 225          4094X SUB L      (A) = LENGTH
057.142 247          4095X ANA A      CLEAR CARRY
057.143 321          4096X POP D      RESTORE (DE)
057.144 311          4097X RET
4098X
4099X *      CTL-D STRUCK
4100X
057.145 341          4101X $RTL2 POP H      (HL) = FWA
057.146 067          4102X STC
057.147 311          4103X RET
057.150          4104 XTEXT MOVE

```

```

4106X **     $MOVE - MOVE DATA
4107X *
4108X *     $MOVE MOVES A BLOCK OF BYTES TO A NEW MEMORY ADDRESS.
4109X *     IF THE MOVE IS TO A LOWER ADDRESS, THE BYTES ARE MOVED FROM
4110X *     FIRST TO LAST.
4111X *
4112X *     IF THE MOVE IS TO A HIGHER ADDRESS, THE BYTES ARE MOVED FROM
4113X *     LAST TO FIRST.
4114X *
4115X *     THIS IS DONE SO THAT AN OVERLAPED MOVE WILL NOT 'RIPPLE'.
4116X *
4117X *     ENTRY (BC) = COUNT

```

```

4118X *      (DE) = FROM
4119X *      (HL) = TO
4120X *      EXIT  MOVED
4121X *      (DE) = ADDRESS OF NEXT FROM BYTE
4122X *      (HL) = ADDRESS OF NEXT *TO* BYTE
4123X *      C/ CLEAR
4124X *      USES  ALL
4125X
4126X
030.252     4127X $MOVE EQU 30252A      IN H17 ROM
057.150     4128      XTEXT CHL
    
```

```

4130X **     $CHL - COMPLEMENT (HL).
4131X *
4132X *      (HL) = -(HL)          TWO'S COMPLEMENT
4133X *
4134X *      ENTRY  NONE
4135X *      EXIT  NONE
4136X *      USES  A,F,H,L
4137X
4138X
030.224     4139X $CHL EQU 30224A      IN H17 ROM
057.150     4140      XTEXT SOB
    
```

```

4142X **     $SOB - SKIP OVER BLANKS.
4143X *
4144X *      $SOB IS CALLED TO SKIP AN ARBITRARILY LONG STRING OF BLANKS AND TABS.
4145X *
4146X *      ENTRY  (HL) = FWA OF (POSSIBLE) BLANK STRING
4147X *      EXIT  (HL) = LWA+1 OF BLANK STRING (UNCHANGED IF NO BLANKS)
4148X *      (A) = FIRST NON-BLANK, NON-TAB CHARACTER EEN
4149X *      USES  A,F,H,L
4150X
4151X
057.150 053 4152X $SOB DCX H          PRE-DECREMENT
057.151 043 4153X $SOB1 INX H
057.152 176 4154X      MOV  A,M
057.153 376 040 4155X      CFI  / /
057.155 312 151 057 4156X      JE  $SOB1      GOT BLANK
057.160 376 011 4157X      CFI  TAB
057.162 312 151 057 4158X      JE  $SOB1      GOT TAB
057.165 311 4159X      RET
057.166     4160      XTEXT TBL5
    
```



\$TBL5

14:41:53 16-MAY-80

```

4162X ** $TBL5 - TABLE SEARCH
4163X *
4164X * TABLE FORMAT
4165X *
4166X * DB KEY1,VAL1,
4167X * .
4168X * .
4169X * DB KEYN,VALN
4170X * DB 0
4171X *
4172X * ENTRY (A) = PATTERN.
4173X * (H,L) = TABLE FWA
4174X * EXIT (A) = PATTERN IF FOUND
4175X * 'Z' SET IF FOUND
4176X * 'Z' CLEAR IF NOT FOUND OR PATTERN=0 /78.10.GC/
4177X * USES A,F,H,L
4178X
4179X
057.166 305 4180X $TBL5 PUSH B
057.167 376 000 4181X CPI 0 /78.10.GC/
057.171 312 213 057 4182X JZ TBL2 /78.10.GC/
057.174 107 4183X MOV B,A
057.175 176 4184X TBL1 MOV A,M (A) = CHARACTER
057.176 043 4185X INX H
057.177 270 4186X CMP B
057.200 312 215 057 4187X JZ TBL3 IF MATCH
057.203 247 4188X ANA A
057.204 043 4189X INX H SKIP PAST
057.205 302 175 057 4190X JNZ TBL1 IF NOT END OF TABLE
057.210 053 4191X DCX H
057.211 053 4192X DCX H
057.212 257 4193X XRA A SET TO ZERO FOR OLD USERS /78.10.GC/
057.213 376 001 4194X TBL2 CPI 1 CLEAR ZERO /78.10.GC/
4195X
4196X * DONE
4197X
057.215 301 4198X TBL3 POP B
057.216 311 4199X RET
057.217 4200 XTEXT DADA

```

```

4202X ** $DADA - PERFORM (H,L) = (H,L) + (0,A)
4203X *
4204X * ENTRY (H,L) = BEFORE VALUE
4205X * (A) = BEFORE VALUE
4206X * EXIT (H,L) = (H,L) + (0,A)
4207X * 'C' SET IF OVERFLOW
4208X * USES F,H,L
4209X
4210X
030.072 4211X $DADA EQU 30072A IN HI7 ROM
057.217 4212 XTEXT TJMP

```

```

4214X **      $TJMP - TABLE JUMP.
4215X *
4216X *      USAGE
4217X *
4218X *      CALL      $TJMP      (A) = INDEX
4219X *      DW        ADDR1
4220X *      .
4221X *      .
4222X *      .
4223X *      DW        ADDR2
4224X *
4225X *      ENTRY      (A) = INDEX
4226X *      EXIT      TO PROCESSOR
4227X *      (A) = INDEX*2
4228X *      USES      NONE.
4229X
4230X
031.061      4231X $TJMP EQU 31061A      IN H17 ROM, (A) = INDEX*2
4232X
031.062      4233X $TJMP EQU 31062A      IN H17 ROM
057.217      4234X XTEXT CRLF

```

```

4234X **      $CRLF - TYPE CARRIAGE RETURN/ LINE FEED.
4237X *
4238X *      $CRLF IS USED TO GENERATE PADDED CRLF'S.
4239X *
4240X *      ENTRY      NONE
4241X *      EXIT      (A) = 0
4242X *      USES      A,F
4243X
4244X
057.217 076 012 4245X $CRLF MVI A,NL
057.221 377 002 4246X DB SYSCALL,SCOUT
057.223 357 4247X XRA A
057.224 311 4248X RET
057.225 4249X XTEXT TYPCH

```

```

4251X **      $TYPCH - TYPE SINGLE CHARACTER.
4252X *
4253X *      ENTRY      (RET) = CHARACTER
4254X *      EXIT      TO (RET)+1
4255X *      (A) = CHARACTER TYPED
4256X
4257X
057.225 343 4258X $TYPCH XTHL      (HL) = RETURN ADDRESS
057.226 176 4259X MOV A,M      (A) = CHARACTER
057.227 043 4260X INX H
057.230 343 4261X XTHL      RESTORE ADVANCED EXIT ADDRESS
4262X
4263X **      $TYPC - TYPE SINGLE CHARACTER.

```

\*TYPCH

```

4264X *
4265X *      ENTRY (A) = CHARACTER
4266X *      EXIT  TO (RET)
4267X
057.231 377 002 4268X $TYPC, DB SYSCALL, SCOUT
057.233 311     4269X RET
000.001     4270 $CMP$ EQU 1
057.234     4271 XTEXT TYPLN

4273X **      $TYPLN - TYPE LINE.
4274X *
4275X *      $TYPLN IS CALLED TO TYPE A LINE OF TEXT. ZERO BYTES ARE
4276X *      TAKEN AS CRLF (WITH THE PROPER PADDING)
4277X *
4278X *      CALL $TYPLN
4279X *      DB N BYTE COUNT OF FOLLOWING MESSAGE
4280X *      DB 'N-CHARACTER MESSAGE'
4281X *
4282X *      ENTRY (RET) = TEXT COUNT
4283X *      (RET)+1 - (RET)+N = TEXT
4284X *      EXIT  TO (RET)+N+1
4285X *      USES A,F
4286X *
4287X
4288X
057.234 343 4289X $TYPLN, XTHL (HL) = COUNT ADDRESS
057.235 176 4290X MOV A,M (A) = COUNT
057.236 043 4291X INX H (HL) = TEXT ADDRESS
057.237 345 4292X PUSH H SAVE TEXT FWA
057.240 315 072 030 4293X CALL $DADA CALCULATE RETURN ADDRESS
057.243 343 4294X XTHL (HL) = TEXT ADDR
057.244 315 252 057 4295X CALL $TYPL, OUTPUT LINE
057.247 341 4296X POP H (HL) = RETURN ADDRESS
057.250 343 4297X XTHL RESTORE (HL); SET RETURN ADDRESS
057.251 311 4298X RET
4299X

4300X **      $TYPL. - TYPE LINE.
4301X *
4302X *      ENTRY (HL) = ADDRESS
4303X *      (A) = COUNT
4304X *      EXIT  NONE
4305X *      USES A,F,H,L
4306X
057.252 4307X $TYPL, EQU *
057.252 247 4308X ANA A
057.253 310 4309X RZ NOTHING TO TYPE
057.254 345 4310X PUSH PSW SAVE COUNT
057.255 176 4311X MOV A,M (A) = CHARACTER
057.256 043 4312X INX H
057.257 247 4313X ANA A
000.001 4314X IF $CMP$ IF HAVE COMPRESSED SPACES
4315X JH TPL2 IS COMPRESSED SPACE
4316X ENDIF

```

```

057.260 314 217 057 4317X CZ *CRLF
057.263 315 231 057 4318X CALL $TYFCH TYPE CHARACTER
057.266 361 4319X TPL1 POP PSW
057.267 075 4320X DCR A
057.270 302 252 057 4321X JNZ $TYPL
057.273 311 4322X RET
000.001 4323X IF *CMP$ IF COMPRESSED TEXT
4324X
4325X * HAVE COMPRESSED SPACE.
4326X
4327X TPL2 DCR A
4328X CF $TYFCH TYPE 00 IF CHARACTER WAS 2000
4329X DB 0
4330X ANA A SET CODES
4331X TPL3 JP TPL1 ALL EXPANDED
4332X PUSH PSW SAVE COUNT
4333X CALL $TYFCH
4334X DR
4335X POP PSW
4336X DCR A
4337X JMP TPL3
4338X ENDIF
057.274 4339 XTEXT TYPT2

```

```

4341X ** $TYPTX - TYPE TEXT.
4342X *
4343X * $TYPTX IS CALLED TO TYPE A BLOCK OF TEXT ON THE SYSTEM CONSOLE.
4344X *
4345X * IMPBEDDED ZERO BYTES INDICATE A CARRIAGE RETURN LINE FEED.
4346X * A BYTE WITH THE 2000 BIT SET IS THE LAST BYTE IN THE MESSAGE.
4347X *
4348X * ENTRY (RET) = TEXT
4349X * EXIT TO (RET+LENGTH)
4350X * USES A,F
4351X
4352X

```

```

031.136 4353X $TYPTX EQU 31136A IN H17 ROM
4354X
031.144 4355X $TYPTX EQU 31144A IN H17 ROM
057.274 4356 XTEXT COMP

```

```

4358X ** *COMP - COMPARE TWO CHARACTER STRINGS.
4359X *
4360X * *COMP COMPARES TWO BYTE STRINGS.
4361X *
4362X * ENTRY (C) = COMPARE COUNT
4363X * (DE) = FWA OF STRING #1
4364X * (HL) = FWA OF STRING #2
4365X * EXIT 'Z' CLEAR, IS MIS-MATCH
4366X * (C) = LENGTH REMAINING

```

```

4367X *      (DE) = ADDRESS OF MISMATCH IN STRING#1
4368X *      (HL) = ADDRESS OF MISMATCH IN STRING #2
4369X *      'C' SET, HAVE MATCH
4370X *      (C) = 0
4371X *      (DE) = (DE) + (OC)
4372X *      (HL) = (HL) + (OC)
4373X *      USES  A,F,C,D,E,H,L
4374X
4375X
030.060     4376X *COMP EQU 30060A      IN H17 ROM
057.274     4377      XTEXT SAVALL

```

```

4379X **     $RSTALL - RESTORE ALL REGISTERS.
4380X *
4381X *     $RSTALL RESTORES ALL THE REGISTERS OFF THE STACK, AND
4382X *     RETURNS TO THE PREVIOUS CALLER.
4383X *
4384X *     ENTRY  (SP) = PSW
4385X *           (SP+2) = BC
4386X *           (SP+4) = DE
4387X *           (SP+6) = HL
4388X *           (SP+8) = RET
4389X *     EXIT  TO *RET*, REGISTERS RESTORED
4390X *     USES  ALL
4391X
4392X
031.047     4393X *RSTALL EQU 31047A      IN H17 ROM

```

```

4395X **     $SAVALL - SAVE ALL REGISTERS ON STACK.
4396X *
4397X *     $SAVALL SAVES ALL THE REGISTERS ON THE STACK.
4398X *
4399X *     ENTRY  NONE
4400X *     EXIT  (SP) = PSW
4401X *           (SP+2) = BC
4402X *           (SP+4) = DE
4403X *           (SP+6) = HL
4404X *     USES  H,L
4405X
4406X
031.054     4407X *SAVALL EQU 31054A      IN H17 ROM
057.274     4408      XTEXT CDEHL

```

4410X \*\* \$CDEHL - COMPARE (DE) TO (HL)  
4411X \*  
4412X \* \$CDEHL COMPARES (DE) TO (HL) FOR EQUALITY.  
4413X \*  
4414X \* ENTRY NONE  
4415X \* EXIT 'Z' SET IF (DE) = (HL)  
4416X \* USES A,F  
4417X  
4418X  
030,216 4419X \$CDEHL EQU 30216A IN H17 ROM  
057,274 4420 XTEXT UDD

4422X \*\* \$UDD - UNPACK DECIMAL DIGITS.  
4423X \*  
4424X \* UDD CONVERTS A 16 BIT VALUE INTO A SPECIFIED NUMBER OF  
4425X \* DECIMAL DIGITS. THE RESULT IS ZERO FILLED.  
4426X \*  
4427X \* ENTRY (B,C) = ADDRESS VALUE  
4428X \* (A) = DIGIT COUNT  
4429X \* (H,L) = MEMORY ADDRESS  
4430X \* EXIT (HL) = (HL) f (A)  
4431X \* USES ALL  
4432X  
4433X  
031,157 4434X \$UDD EQU 31157A IN H17 ROM  
057,274 4435 XTEXT DU66

4437X \*\* \$DU66 - UNSIGNED 16 / 16 DIVIDE.  
4438X \*  
4439X \* (HL) = (BC)/(DE)  
4440X \*  
4441X \* ENTRY (BC), (DE) PRESET  
4442X \* EXIT (HL) = RESULT  
4443X \* (DE) = REMAINDER  
4444X \* USES ALL  
4445X  
4446X  
030,106 4447X \$DU66 EQU 30106A IN H17 ROM  
057,274 4448 XTEXT DADA2

4450X \*\* \$DADA - ADD (0,A) TO (H,L)  
4451X \*  
4452X \* ENTRY NONE  
4453X \* EXIT (HL) = (HL) + (0A)  
4454X \* USES A,F,H,L  
4455X  
4456X

\$DADA

030.101 4457X \$DADA EQU 30101A IN H17 ROM  
057.274 4458 XTEXT HLIHL

4460X \*\* \$HLIHL - LOAD HL INDIRECT THROUGH HL.  
4461X \*  
4462X \* (HL) = ((HL))  
4463X \*  
4464X \* ENTRY NONE  
4465X \* EXIT NONE  
4466X \* USES A,H,L  
4467X  
030.211 4468X \$HLIHL EQU 30211A IN H17 ROM  
057.274 4469 XTEXT ILDEHL

4471X \*\* ILDEHL - INDEXED LOAD OF DE FROM HL  
4472X \*  
4473X \* 'DE' GET THE FULL WORD VALUE POINTED TO BY 'HL', AND 'HL' IS  
4474X \* INCREMENTED BY TWO.  
4475X \*  
4476X \* ENTRY: HL = ADDRESS OF FULL WORD VALUE  
4477X \*  
4478X \* EXIT: DE = (HL)  
4479X \* HL = HL + 2  
4480X \*  
4481X \* USES: DE  
4482X \*  
4483X  
057.274 136 4484X ILDEHL MOV E,M  
057.275 043 4485X INX H  
057.276 126 4486X MOV D,M  
057.277 043 4487X INX H  
057.300 311 4488X RET  
057.301 4489 XTEXT INDL

4491X \*\* \$INDL - INDEXED LOAD.  
4492X \*  
4493X \* \$INDL LOADS DE WITH THE TWO BYTES AT (HL)+DISPLACEMENT  
4494X \*  
4495X \* THIS ACTS AS AN INDEXED FULL WORD LOAD.  
4496X \*  
4497X \* (DE) = ( (HL) + DSPLACEMENT )  
4498X \*  
4499X \* ENTRY ((RET)) = DISPLACEMENT (FULL WORD)  
4500X \* (HL) = TABLE ADDRESS  
4501X \* EXIT TO (RET+2)  
4502X \* USES A,F,D,E  
4503X

4504X  
030.234 4505X \*INDL EQU 30234A IN H17 ROM  
057.301 4506 XTEXT INDXX

4508X \*\* \*INDLB - INDEXED LOAD BYTE

4509X \*

4510X \* BYTE INDEXED LOAD PRIMITIVE

4511X \*

4512X \* ENTRY: HL = BASE ADDRESS

4513X \* (RET) = FULL WORD RELOCATION

4514X \*

4515X \* EXIT: A = ( HL + (RET) )

4516X \*

4517X \* USES: A

4518X \*

4519X \*

057.301 353 4520X \*INDLB XCHG DE = BASE

057.302 343 4521X XTHL SAVE .DE.

057.303 325 4522X PUSH D SAVE .BASE

057.304 305 4523X PUSH B SAVE .BC.

4524X \*

057.305 116 4525X MOV C,M

057.306 043 4526X INX H

057.307 106 4527X MOV B,M BC = OFFSET

057.310 043 4528X INX H HL = .RET.

4529X \*

057.311 353 4530X XCHG HL = BASE

057.312 011 4531X DAD B HL = BASE + OFFSET

057.313 176 4532X MOV A,M A = ( BASE + OFFSET )

057.314 353 4533X XCHG HL = .RET.

4534X \*

057.315 301 4535X POP B RESTORE .BC.

057.316 321 4536X POP D RESTORE .BASE

057.317 343 4537X XTHL HL = .DE. ; (SP) = .RET.

057.320 353 4538X XCHG DE = .DE. ; HL = BASE

057.321 311 4539X RET

4541X \*\* \*INDS - INDEXED STORE

4542X \*

4543X \* INDEXED STORE PRIMITIVE

4544X \*

4545X \* ENTRY: HL = BASE ADDRESS

4546X \* DE = VALUE TO STORE

4547X \*

4548X \* EXIT: ( HL + (RET) ) = DE

4549X \*

4550X \* USES: NONE

4551X \*

4552X \*

057.322 315 300 060 4553X \*INDS CALL XCHGBC



\$INDS

057.325	343	4554X	XTHL		SAVE ,BC.
057.326	325	4555X	PUSH	D	
057.327	315 274 057	4556X	CALL	ILDEHL	DE = OFFSET
057.332	315 300 060	4557X	CALL	XCHGBC	BC = ,RET.
057.335	353	4558X	XCHG		DE = BASE + HL = OFFSET
057.336	031	4559X	DAD	D	HL = BASE + OFFSET
057.337	353	4560X	XCHG		
057.340	343	4561X	XTHL		SAVE BASE
057.341	353	4562X	XCHG		DE = VALUE
057.342	315 377 057	4563X	CALL	ISDEHL	
057.345	341	4564X	POP	H	HL = BASE
057.346	315 300 060	4565X	CALL	XCHGBC	
057.351	343	4566X	XTHL		RESTORE ,BC.
057.352	315 300 060	4567X	CALL	XCHGBC	
057.355	311	4568X	RET		

4570X \*\* \$INDSB - INDEXED BYTE STORE  
4571X \*  
4572X \* INDEXED BYTE STORE.  
4573X \*  
4574X \* ENTRY: A = VALUE TO STORE  
4575X \* HL = BASE ADDRESS  
4576X \* (RET) = OFFSET  
4577X \*  
4578X \* EXIT: NONE  
4579X \*  
4580X \* USES: PSW  
4581X \*

057.356	353	4583X	\$INDSB XCHG		DE = BASE
057.357	343	4584X	XTHL		SAVE ,DE.
057.360	325	4585X	PUSH	D	SAVE BASE
057.361	305	4586X	PUSH	B	SAVE ,BC.
		4587X			
057.362	116	4588X	MOV	C,M	
057.363	043	4589X	INX	H	
057.364	106	4590X	MOV	B,M	BC = OFFSET
057.365	043	4591X	INX	H	HL = ,RET.
		4592X			
057.366	353	4593X	XCHG		HL = BASE
057.367	011	4594X	DAD	B	HL = BASE + OFFSET
057.370	167	4595X	MOV	M,A	( BASE + OFFSET ) = A
057.371	353	4596X	XCHG		
		4597X			
057.372	301	4598X	POP	B	RESTORE ,BC.
057.373	321	4599X	POP	D	RESTORE BASE
057.374	343	4600X	XTHL		HL = ,DE, ; (SP) = ,RET.
057.375	353	4601X	XCHG		DE = ,DE, ; HL = BASE
057.376	311	4602X	RET		
057.377		4603	XTEXT	ISDEHL	

```

4605X ** ISDEHL - INDEXED STORE OF DE AT HL
4606X *
4607X * STORE 'DE' AT THE ADDRESS POINTED TO BY 'HL', AND INCREMENT 'HL'
4608X * BY 2.
4609X *
4610X * ENTRY: DE = VALUE
4611X * HL = ADDRESS OF VALUE
4612X *
4613X * EXIT: (HL) = DE
4614X * HL = HL + 2
4615X *
4616X * USES: HL
4617X *
4618X
060.004 163 4619X ISDEHL MOV M,E
060.000 043 4620X INX H
060.001 162 4621X MOV M,D
060.002 043 4622X INX H
060.003 311 4623X RET
060.004 4624 XTEXT DAD

```

```

4626X ** $DAD - DECODE AUGUSTAN DATE.
4627X *
4628X * $DAD DECODES A 15 BIT DATE CODE OF THE FORMAT:
4629X *
4630X *
4631X * I 0 I 4 BITS I 4 BITS I 5 BITS I
4632X *
4633X * YEAR-70 MON DAY
4634X * 1-63 1-12 1-31
4635X *
4636X * TO THE FORM:
4637X *
4638X * DD-MMM-YY
4639X *
4640X * ENTRY (DE) = 15 BIT VALUE
4641X * (HL) = ADDRESS FOR DECODE
4642X * EXIT 'C' CLEAR IF OK
4643X * (DE) = (DE)+9
4644X * 'C' SET IF ERROR
4645X * USES ALL
4646X
4647X
060.004 102 4648X $DAD MOV B,D
060.005 113 4649X MOV C,E
060.006 021 040 000 4650X LXI D,32
060.011 345 4651X PUSH H SAVE ADDRESS
060.012 315 106 030 4652X CALL $DU66 (DE) = DAY, (HL) = YEAR & MONTH
060.015 343 4653X XTHL (HL) = ADDRESS
060.016 102 4654X MOV B,D
060.017 113 4655X MOV C,E
060.020 173 4656X MOV A,E
060.021 247 4657X ANA A

```

\$DAD

```

060.022 312 122 060 4658X JZ DAD1 BAD VALUE
060.025 076 002 4659X MVI A,2
060.027 315 157 031 4660X CALL $UDD UNPACK DAY
060.032 066 055 4661X MVI M,'-'
060.034 043 4662X INX H
060.035 301 4663X POP B (BC) = YEAR & MONTH
060.036 021 020 000 4664X LXI D,16
060.041 345 4665X PUSH H SAVE ADDRESS
060.042 315 106 030 4666X CALL $DD66 (HL) = ADDRESS, ((SP)) = YEAR
060.046 173 4668X MOV A,E
060.047 207 4669X ADD A (A) = 3*MONTH
060.050 203 4670X ADD E
060.051 312 122 060 4671X JZ DAD1 BAD VALUE
060.054 376 047 4672X CPI 13*3
060.056 322 122 060 4673X JNC DAD1 TOO LARGE
060.061 353 4674X XCHG (DE) = ADDRESS
060.062 041 122 060 4675X LXI H,DADB-3 (HL) = ADDRESS OF MONTH
060.065 315 101 030 4676X CALL $DADA
060.070 001 003 000 4677X LXI B,3 (HL) = BUFFER ADDR, (DE) = ADDR IN 'DADB'
060.073 353 4678X XCHG
060.074 315 252 030 4679X CALL $MOVE MOVE MONTH IN
060.077 066 055 4680X MVI M,'-'
060.101 043 4681X INX H
060.102 301 4682X POP B (BC) = YEAR
060.103 171 4683X MOV A,C
060.104 306 106 4684X ADI 70
060.106 376 144 4685X CPI 100
060.110 077 4686X CMC
060.111 330 4687X RC TOO LARGE
060.112 117 4688X MOV C,A (BC) = YEAR
060.113 076 002 4689X MVI A,2
060.115 315 157 031 4690X CALL $UDD UNPACK YEAR
060.120 247 4691X ANA A
060.121 311 4692X RET
4693X
4694X * ILLEGAL FORMAT, (NOT ALL ILLEGALS EXIT HERE!)
4695X
060.122 341 4696X DAD1 POP H RESTORE STACK
060.123 067 4697X STC FLAG ERROR
060.124 311 4698X RET
4699X
060.125 112 141 156 4700X DADB DB 'JanFebMarAprMayJunJulAugSepOctNovDec'
060.171 4701 XTEXT UDDN

```

```

4703X ** $UDDN - UNPACK DECIMAL DIGITS.
4704X *
4705X * UDDN CONVERTS A 16 BIT VALUE INTO A SPECIFIED NUMBER OF
4706X * DECIMAL DIGITS. THE RESULT IS NULL FILLED TO THE LEFT.
4707X *
4708X * ENTRY (B,C) = ADDRESS VALUE
4709X * (A) = DIGIT COUNT
4710X * (H,L) = MEMORY ADDRESS

```

\$UDDN

```

4711X *      EXIT      (HL) = (HL) + (A)
4712X *      USES     ALL
4713X
4714X
060.171      4715X $UDDN EQU      *
060.171 315 072 030 4716X CALL    $DATA
060.174 345      4717X PUSH    H      SAVE FINAL (H,L) VALUE
4718X
060.175 365      4719X UDDN1  PUSH   PSW
060.176 345      4720X        PUSH   H
060.177 021 012 000 4721X        LXI    D,10
060.202 315 106 030 4722X        CALL   $DU66      (H,L) = VALUE/10
060.205 104      4723X        MOV    B,H
060.206 115      4724X        MOV    C,L      (RC) = QUOTIENT
060.207 341      4725X        POP    H
060.210 076 060  4726X        MVI    A,'0'
060.212 203      4727X        ADD    E      ADD REMAINDER
060.213 053      4728X        DCX    H
060.214 167      4729X        MOV    M,A      STORE DIGIT
060.215 170      4730X        MOV    A,B
060.216 261      4731X        ORA    C
060.217 312 231 060 4732X        JZ     UDDN2      ALL ZEROS
060.222 361      4733X        POP    PSW
060.223 075      4734X        INR    A
060.224 302 175 060 4735X        JNZ   UDDN1      IF MORE TO GO
4736X
4737X *      ALL DONE, EXIT
4738X
060.227 341      4739X UDDN1.5 POP    H      RESTORE H
060.230 311      4740X        RET     RETURN
4741X
4742X *      DIGITS LEADING THIS ONE ARE ZERO, STORE NULLS INSTEAD.
4743X
060.231 361      4744X UDDN2  POP    PSW
060.232 075      4745X UDDN3  DCR    A
060.233 312 227 060 4746X        JE     UDDN1.5      ALL DONE
060.236 053      4747X        DCX    H
060.237 066 000  4748X        MVI    M,0
060.241 303 232 060 4749X        JMP    UDDN3
060.244      4750        XTEXT  MOVEL

4752X **      $MOVEL - MOVE DATA
4753X *
4754X *      $MOVEL MOVES A BLOCK OF BYTES TO A NEW MEMORY ADDRESS.
4755X *      IF THE MOVE IS TO A LOWER ADDRESS, THE BYTES ARE MOVED FROM
4756X *      FIRST TO LAST.
4757X *
4758X *      IF THE MOVE IS TO A HIGHER ADDRESS, THE BYTES ARE MOVED FROM
4759X *      LAST TO FIRST.
4760X *
4761X *      THIS IS DONE SO THAT AN OVERLAPED MOVE WILL NOT 'RIPPLE'.
4762X *
4763X *      CALL    $MOVEL

```

```

4764X *   DW   COUNT
4765X *   DW   FROM
4766X *   DW   TO
4767X *
4768X *   ENTRY  ((SP)) = RET
4769X *           (RET+0) = COUNT (WORD VALUE)
4770X *           (RET+2) = FROM
4771X *           (RET+4) = TO
4772X *   EXIT  TO (RET+6)
4773X *           (DE) = ADDRESS OF NEXT FROM BYTE
4774X *           (HL) = ADDRESS OF NEXT *TO* BYTE
4775X *           /C/ CLEAR
4776X *   USES  ALL
4777X
4778X
060.244 341 4779X $MOVEL POP   H           (HL) = RET
060.245 118 4780X      MOV   C,M
060.246 043 4781X      INX   H
060.247 106 4782X      MOV   B,M           (BC) = COUNT
060.250 043 4783X      INX   H
060.251 136 4784X      MOV   E,M
060.252 043 4785X      INX   H
060.253 126 4786X      MOV   D,M           (DE) = FROM
060.254 043 4787X      INX   H
060.255 325 4788X      PUSH  D           ((SP)) = FROM
060.256 136 4789X      MOV   E,M
060.257 043 4790X      INX   H
060.260 126 4791X      MOV   B,M           (DE) = TO
060.261 043 4792X      INX   H
060.262 343 4793X      XTHL
060.263 353 4794X      XCHG           ((SP)) = RET, (HL) = FROM
                                (DE) = FROM, (HL) = TO
060.264 303 252 030 4795X      JMP   $MOVEL MOVE IT
060.267      4796X      XTEXT  RCHAR

```

```

4798X **   $RCHAR - READ SINGLE CHARACTER FROM CONSOLE.
4799X *
4800X *   ENTRY  NONE
4801X *   EXIT  (A) = CHARACTER
4802X *   USES  A,F
4803X
4804X
060.267 377 001 4805X $RCHAR DB   SYSCALL, .SCIN
060.271 332 267 060 4806X      JC   $RCHAR NOT READY
060.274 311      4807X      RET
4808X
060.275 377 002 4809X $WCHAR DB   SYSCALL, .SCOUT
060.277 311      4810X      RET
060.300      4811X      XTEXT  XCHGBC

```

```

4813X **      XCHGBC = XCHG BC
4814X *
4815X *      EXCHANGE THE 'BC' REGISTER PAIR WITH THE 'HL' REGISTER PAIR.
4816X *
4817X *      ENTRY: BC      = ORIGINAL BC
4818X *      HL      = ORIGINAL HL
4819X *
4820X *      EXIT:  BC      = ORIGINAL HL
4821X *      HL      = ORIGINAL BC
4822X *
4823X *      USES:  BC,HL
4824X *
4825X
060.300 345   4826X XCHGBC PUSH  PSW
060.301 170   4827X      MOV   A,R
060.302 104   4828X      MOV   B,H
060.303 147   4829X      MOV   H,A
060.304 171   4830X      MOV   A,C
060.305 115   4831X      MOV   C,L
060.306 157   4832X      MOV   L,A
060.307 341   4833X      POP  PSW
060.310 311   4834X      RET
060.311       4835      XTEXT  DRS
    
```

```

4837X **      $DRS - DECODE AND REMOVE SWITCHES.
4838X *
4839X *      $DRS IS CALLED TO DECODE COMMAND SWITCHES FROM A LINE
4840X *      OF TEXT. SWITCHES TAKE THE FORM:
4841X *
4842X *      /XXXXX
4843X *
4844X *      AFTER A SWITCH HAS BEEN LOCATED, IT (AND THE PRECEDING '/')
4845X *      ARE REPLACED WITH BLANKS.
4846X *
4847X *      VALID SWITCH DESCRIPTIONS ARE ENCODED INTO A TABLE
4848X *      SUPPLIED BY THE CALLER, IN THE FORMAT:
4849X *
4850X *      DB      'X...X'      REQUIRED SWITCH CHARACTERS
4851X *      DB      'C'+200Q,...,'C'+200R      OPTIONAL CHARACTERS
4852X *      DB      2000          END OF CHARACTERS
4853X *      DW      ADDR          PROCESSOR ADDRESS (CALLED WHEN SWITCH DETECTED)
4854X *
4855X *      DB      'Y...Y'      NEXT SWITCH
4856X *      .
4857X *      .
4858X *      .
4859X *
4860X *      DB      0              FLAGS END OF TABLE
4861X *
4862X *      SWITCHES MUST BE FOLLOWED BY A '/' , A '/' (ANOTHER SWITCH)
4863X *      A ',', OR A 00 BYTE.
4864X *
4865X *      UPON DETECTION OF A VALID SWITCH, $DRS CALLS THE USER PROCESS
    
```

\*DRS

```

4866X * ROUTINE, UPON ENTRY;
4867X * (HL) = ADDRESS OF THE FIRST BYTE FOLLOWING THE SWITCH
4868X * 'Z' CLEAR IF CHARACTER = '???' OR '00'
4869X * 'Z' SET IF CHARACTER = '!'
4870X *
4871X * THE USER ROUTINE CAN DECODE SWITCH SUB-OPTIONS, IF DESIRED.
4872X * THE USER ROUTINE MAY USE ALL REGISTERS.
4873X *
4874X * ENTRY (DE) = SWITCH TABLE FWA
4875X * (HL) = LINE FWA
4876X * EXIT 'C' CLEAR IF OK
4877X * 'C' SET IF ERROR
4878X * (HL) = ADDRESS OF START OF BAD SWITCH
4879X * (A) = ERROR CODE
4880X * USES ALL
4881X
4882X
060,311 4883X *DRS EQU *
4884X
4885X * LOOK FOR SWITCHES.
4886X
060,311 176 4887X *DRS1 MOV A,M
060,312 247 4888X ANA A
060,313 310 4889X SZ END OF LINE
060,314 043 4890X INX H
060,315 376 057 4891X CPI '/'
060,317 302 311 060 4892X JNE $DRS1 NOT A SWITCH
060,322 042 106 061 4893X SHLD $DRSB ($DRSB) = SWITCH FWA (AFTER '/')
4894X
4895X * GOT A SWITCH, LOOK FOR A MATCH IN THE CALLER'S TABLE.
4896X
060,325 325 4897X PUSH D SAVE TABLE FWA
060,326 052 106 061 4898X *DRS2 LHLD $DRSB (HL) = SWITCH FWA
060,331 032 4899X *DRS3 LDAX D (A) = TABLE ENTRY
060,332 346 177 4900X ANI 177H
060,334 312 004 061 4901X JZ $DRS6 GOT A MATCH
060,337 276 4902X CMP M
060,340 302 350 060 4903X JNE $DRS4 NO MATCH
060,343 023 4904X INX D
060,344 043 4905X INX H
060,345 303 331 060 4906X JMP $DRS3 SEE IF MORE MATCH
4907X
4908X * HAVE MIS-MATCH, SEE IF THE MISSING CHARACTER IS SIGNIFICANT
4909X
060,350 176 4910X *DRS4 MOV A,M (A) = LINE CHARACTER WE COULDN'T MATCH
060,351 315 055 061 4911X CALL $DRS15 SEE IF OK TERMINATOR
060,354 302 364 060 4912X JNE $DRS4.5 NO MATCH ON THIS SWITCH
060,357 032 4913X LDAX D (A) = NEXT CHARACTER IN SWITCH PATTERN
060,360 247 4914X ANA A
060,361 372 004 061 4915X JM $DRS6 HAVE SUFFICIENT MATCH
060,364 315 070 061 4916X *DRS4.5 CALL $DRS20 SKIP TABLE ENTRY
060,367 032 4917X LDAX D
060,370 247 4918X ANA A
060,371 302 326 060 4919X JNZ $DRS2 MORE SWITCHES IN TABLE TO CHECK
4920X
4921X * BAD SWITCH

```

\$DRS

```

4922X
060.374 321 4923X $DRS5 POP D RESTORE STACK
060.375 052 196 061 4924X LHLI $DRSB POINT TO BAD SWITCH
061.000 067 4925X STC
061.001 076 032 4926X MVI A,EC.IS ILLEGAL SWITCH
061.003 311 4927X RET
4928X
4929X * HAVE SWITCH. CHECK IT'S FOLLOWING CHARACTER.
4930X
061.004 315 150 057 4931X $DRS6 CALL $SOB SKIP OVER BLANKS
061.007 176 4932X MOV A,M
061.010 315 055 061 4933X CALL $DRS15 CHECK CHARACTER
061.013 302 374 060 4934X JNE $DRS5 IN ERROR
061.016 315 070 061 4935X CALL $DRS20 GET PROCESSOR ADDRESS
061.021 021 033 061 4936X LXI D,$DRS7
061.024 345 4937X PUSH H SAVE (HL)
061.025 325 4938X PUSH D SET RETURN ADDRESS FOR TABLE CODE
061.026 305 4939X PUSH B SAVE PROCESSOR ADDRESS
061.027 176 4940X MOV A,M (A) = NEXT CHARACTER
061.030 376 072 4941X CPI '/' SET CONDITION CODES
061.032 311 4942X RET CALL USER PROCESS
4943X
4944X * USER PROCESS RETURNS HERE
4945X
061.033 321 4946X $DRS7 POP D (DE) = LAST CHARACTER OF SWITCH+1
061.034 052 196 061 4947X LHLI $DRSB (HL) = FIRST CHARACTER OF SWITCH AFTER /
061.037 053 4948X DCX H (HL) = ADDRESS OF '/'
4949X
4950X * REPLACE SWITCH WITH BLANKS
4951X
061.040 066 040 4952X $DRSB MVI M,' '
061.042 043 4953X INX H
061.043 315 216 030 4954X CALL $CDEHL
061.046 302 040 061 4955X JNE $DRSB NOT THERE YET
061.051 321 4956X POP D (DE) = SWITCH TABLE FWA
061.052 303 311 060 4957X JMP $DRS1 LOOK FOR MORE SWITCHES

4959X ** $DRS15 - CHECK FOR VALID DELIMITER CHARACTER.
4960X *
4961X * $DRS15 CHECKS THE NEXT TEXT CHARACTER TO SEE IF IT IS
4962X *
4963X * 00, '/', ',', '?'
4964X *
4965X * ENTRY (A) = CHARACTER
4966X * EXIT 'Z' SET IFF CHARACTER IS ONE OF THE ABOVE
4967X * USES F
4968X
061.055 247 4969X $DRS15 ANA A
061.056 310 4970X RZ IS 00
061.057 376 057 4971X CPI '?'
061.061 310 4972X RE
061.062 376 054 4973X CPI '/'
061.064 310 4974X RE
061.065 376 072 4975X CPI '?'
061.067 311 4976X RET

```



```

4978X **      $DRS20 - GET PROCESSOR ADDRESS.
4979X *
4980X *      $DRS20 IS CALLED TO GET THE PROCESSOR ADDRESS FIELD OUT OF
4981X *      AN ENTRY IN THE SWITCH TABLE. THE CALLER SUPPLIES A POINTER
4982X *      TO SOMEWHERE IN THE TEXT PART OF THE SWITCH DESCRIPTION.
4983X *      $DRS20 ADVANCES THE POINTER TO THE PROCESSOR ADDRESS.
4984X *
4985X *      ENTRY (DE) = POINTER TO TEXT PART OF SWITCH ENTRY
4986X *      EXIT (DE) = POINTER TO 1ST BYTE OF NEXT SWITCH TABLE ENTRY
4987X *      (BC) = PROCESSOR ADDRESS FROM TABLE
4988X *      USES A,F,B,C,D,E
4989X *
4990X
061.070 032 4991X $DRS20 LDAX D
061.071 023 4992X INX D
061.072 376 200 4993X CPI 200Q
061.074 302 070 061 4994X JNE $DRS20
061.077 032 4995X LDAX D (A) = LOW BYTE OF PROCESSOR ADDRESS
061.100 117 4996X MOV C,A
061.101 023 4997X INX D
061.102 032 4998X LBAX D (BC) = PROCESSOR ADDRESS
061.103 107 4999X MOV B,A
061.104 023 5000X INX D
061.105 311 5001X RET
5002X
061.106 000 000 5003X $DRSB DW 0 POINTER TO SWITCH BEING PROCESSED
000.000 5004 IF .PIP.
061.110 5005 XTEXT DTB

5007X **      $DTB - DELETE TRAILING BLANKS.
5008X *
5009X *      $DTB DELETES THE TRAILING BLANKS FROM A CODED LINE.
5010X *
5011X *      ENTRY (HL) = LINE FWA
5012X *      EXIT (A) = LENGTH OF RESULT (INCLUDING 00 TERMINATOR BYTE)
5013X *      USES A,F
5014X
5015X
061.110 325 5016X $DTB PUSH D SAVE (DE)
061.111 124 5017X MOV D,H
061.112 135 5018X MOV E,L (DE) = FWA
061.113 033 5019X DCX D (DE) = FWA-1
061.114 176 5020X $DTB1 MOV A,M
061.115 043 5021X INX H
061.116 247 5022X ANA A FIND END OF LINE
061.117 302 114 061 5023X JNZ $DTB1
061.122 053 5024X DCX H (HL) = ADDRESS OF TERMINATING ZERO BYTE
5025X
5026X *      GOT END OF LINE, DELETE TRAILING BLANKS
5027X
061.123 053 5028X $DTB2 DCX H BACKUP ONE CHARACTER
061.124 315 216 030 5029X CALL $CDEHL
061.127 312 140 061 5030X JE $DTB3 GONE PAST FRONT OF LINE, MUST BE ALL BLANKS

```

```

061.132 176          5031X      MOV      A,M
061.133 376 040     5032X      CPI
061.135 312 123 061 5033X      JE       $DTR2      GOT BLANK
                    5034X
                    5035X *      HAVE TRIMED LINE. COMPUTE LENGTH
                    5036X
061.140 043        5037X $DTR3  INX      H
061.141 066 000    5038X      MVI      M,0      TERMINATE LINE
061.143 175        5039X      MOV      A,L
061.144 223        5040X      SUB      E          (A) = LENGTH +1 (FOR OO BYTE)
061.145 353        5041X      XCHG
061.146 043        5042X      INX      H          (HL) = LINE FWA
061.147 321        5043X      POP      D          RESTORE (DE)
061.150 311        5044X      RET
061.151            5045      XTEXT  FOPE

```

```

5047X **          $FOPEX - OPEN FILE BLOCK FOR I/O
5048X *
5049X *          $FOPEX IS CALLED BEFORE ANY I/O IS DONE VIA A
5050X *          FILE BLOCK. $FOPEX SETS UP THE FILE BLOCK, AND OPENS
5051X *          THE FILE VIA *HDOS*.
5052X *
5053X *          ENTRY (DE) = ADDRESS OF DEFAULT BLOCK
5054X *          (HL) = ADDRESS OF FILE BLOCK
5055X *          EXIT TO $FERROR IF ERROR
5056X *          TO CALLER IF OK
5057X *          USES A,F,B,C,D,E
5058X
5059X

```

```

061.151 315 176 061 5060X $FOPER CALL $FOPER.
061.154 320          5061X      RNC
061.155 303 161 063 5062X      JMP      $FERROR      IN ERROR
                    5063X

```

```

061.160 315 201 061 5064X $FOPEW CALL $FOPEW.
061.163 320          5065X      RNC
061.164 303 161 063 5066X      JMP      $FERROR      IN ERROR
                    5067X

```

```

061.167 315 204 061 5068X $FOPEU CALL $FOPEU.
061.172 320          5069X      RNC
061.173 303 161 063 5070X      JMP      $FERROR      IN ERROR
                    5071X

```

```

                    5072X
061.176 076 002    5073X $FOPER. MVI A,FT,OR      FILE TYPE OF OPEN FOR READ
061.200 001        5074X      DB      0010      LXI,B TO SKIP NEXT MVI
061.201 076 004    5075X $FOPEW. MVI A,FT,OW      OPEN FOR WRITE
061.203 001        5076X      DB      0010      LXI,B TO SKIP NEXT MVI
061.204 076 006    5077X $FOPEU. MVI A,FT,OR+FT,OW
                    5078X

```

```

5079X *          (A) = FILE FLAGS
5080X

```

```

061.206 345        5081X      PUSH     H          SAVE FILE BLOCK ADDRESS
061.207 345        5082X      PUSH     PSW       SAVE NEW FLAGS
000.000            5083X      ERRNZ  FR,CHA

```

COMMON DECKS

%FOPE

14:43:07 16-MAY-80

```

061.210 106      5084X      MOV      B,M      (B) = CHANNEL NUMBER
061.211 305      5085X      PUSH     B          SAVE HANNEL NUMBER
000.000          5086X      ERRNZ   FB.FLG-FB.CHAN-1
061.212 043      5087X      INX      H
061.213 117      5088X      MOV      C,A      (C) = NEW FILE FLAGS
061.214 176      5089X      MOV      A,M      (A) = CURRENT TYPE
061.215 247      5090X      ANA      A
061.216 171      5091X      MOV      A,C      (A) = NEW FLAGS TO BE SET
061.217 312 231 061 5092X      JZ       %FOPE1    NOT ALREADY OPEN
                    5093X
                    5094X *      ALREADY OPEN. SQUACK
                    5095X
                    5096X      POP      B          RESTORE (BC)
061.222 301      5097X      POP      PSW       DISCARD NEW FLAGS
061.223 361      5098X      POP      H          (HL) = FB ADDRESS
061.224 341      5098X      POP      H
061.225 076 031 5099X      MVI     A,EC.FAQ   FILE ALREADY OPEN
061.227 067      5100X      STC
061.230 311      5101X      RET
                    5102X
000.000          5103X      ERRNZ   FB.FWA-FB.FLG-1
061.231 043      5104X      INX      H          (HL) = #FB.FWA
                    %FOPE1
061.232 116      5105X      MOV      C,M
061.233 043      5106X      INX      H
061.234 106      5107X      MOV      B,M      (BC) = FB.FWA
061.235 043      5108X      INX      H
000.000          5109X      ERRNZ   FB.PTR-FB.FWA-2
061.236 161      5110X      MOV      M,C      SET FB.PTR = FB.FWA
061.237 043      5111X      INX      H
061.240 160      5112X      MOV      M,B
061.241 043      5113X      INX      H
000.000          5114X      ERRNZ   FB.LIM-FB.PTR-2
061.242 161      5115X      MOV      M,C      SET FB.LIM = FB.FWA
061.243 043      5116X      INX      H
061.244 160      5117X      MOV      M,B
061.245 043      5118X      INX      H
000.000          5119X      ERRNZ   FB.NAM-FB.LIM-4
061.246 043      5120X      INX      H
061.247 043      5121X      INX      H          (HL) = #FB.NAM
                    5122X
                    5123X *      FILE BLOCK POINTERS SETUP. OPEN FILE
                    5124X
061.250 345      5125X      PUSH     H          SAVE NEW ADDRESS FOR NAME
061.251 041 302 061 5126X      LXI     H,%FOPEB
061.254 247      5127X      ANA      A
061.255 312 264 061 5128X      JZ       %FOPE2
000.000          5129X      ERRNZ   .EXIT
061.260 315 166 057 5130X      CALL    %BLS      FIND CODE
061.263 176      5131X      MOV      A,M
061.264 062 272 061 5132X      STA     %FOPEA    SET %SYSCALL CODE
                    %FOPE2
061.267 341      5133X      POP      H          (HL) = #FB.NAM
061.270 361      5134X      POP      PSW       (A) = CHANNEL NUMBER
061.271 377 000 5135X      DB     %SYSCALL,.EXIT
061.272          5136X      EQU     *-1        %SYSCALL CODE
                    %FOPEA
061.273 321      5137X      POP      D          (D) = NEW FLAG
061.274 341      5138X      POP      H          (HL) = FILE BLOCK ADDRESS
061.275 330      5139X      RC          EXIT IF ERROR
    
```

78.10.GG/

\*FOFE

```

061.276 043 5140X INX H
000.000 5141X ERRNZ FB,FLG-1
061.277 162 5142X MOV M,D SET NEW FLAGS
061.300 053 5143X DCX H RESTORE (HL)
061.301 311 5144X RET
5145X
061.302 002 042 5146X $FOFEB DB FT,DR,.OPENR TABLE OF SYSCALL CODES
061.304 004 043 5147X DB FT,OW,.OPENW
061.306 006 044 5148X DB FT,DR+FT,OW,.OPENU
061.310 000 5149X DB 0 SHOULD NOT OCCUR
061.311 5150 XTEXT FWRIB
    
```

```

5152X ** $FWRIB - WRITE BYTES FROM FILE BUFFER.
5153X *
5154X * $FWRIB IS CALLED TO WRITE A NUMBER OF BYTES FROM A FILE BUFFER.
5155X *
5156X * ENTRY (BC) = BYTE COUNT
5157X * (DE) = FWA FOR BYTES
5158X * (HL) = ADDRESS OF FILE BUFFER
5159X * EXIT TO *FERROR* IF ERROR
5160X * TO CALLER IF OK
5161X * (DE) = ADDRESS OF FIRST UNWRITTEN BYTE
5162X * USES A,F,B,C,D,E
5163X
    
```

```

061.311 315 320 061 5165X $FWRIB CALL $FWRIB,
061.314 320 5166X RNC RETURN IF OK
061.315 303 181 063 5167X JMP $FERROR ERROR
    
```

```

061.320 5170X $FWRIB EQU *
061.320 345 5171X PUSH H
061.321 315 304 062 5172X CALL CRT COPY BUFFER POINTERS TO TEMP CELLS
    
```

```

5173X
5174X * COPY DATA FROM USER AREA TO BUFFER
5175X
5176X $WRIB2 PUSH D SAVE AREA ADDRESS
    
```

```

061.325 072 047 063 5177X LDA T,FLG
061.330 346 004 5178X ANI FT,OW SEE IF OPEN FOR WRITE
061.332 312 066 062 5179X JZ $WRIB3 FILE NOT OPEN FOR WRITE
061.335 170 5180X MOV A,B
061.336 261 5181X ORA C
061.337 312 066 062 5182X JZ $WRIB3 ALL DONE
    
```

```

5183X
5184X * COMPUTE MIN( ROOM IN BUFFER, WRITE COUNT REQUESTED)
5185X
    
```

```

061.342 052 052 063 5186X $WRIB3 LHLD T,PTR
061.345 353 5187X XCHG (DE) = (FB,PTR) + ADDRESS OF ROOM
061.346 052 056 063 5188X LHLD T,LWA (HL) = LIMIT ADDRESS
061.351 175 5189X MOV A,L
061.352 223 5190X SUB E
061.353 157 5191X MOV L,A
061.354 174 5192X MOV A,H
    
```

\$FWRIB

```

061.355 232      5193X      SBB      D
061.356 147      5194X      MOV      H,A      (HL) = BYTES OF ROOM IN BUFFER
061.357 171      5195X      MOV      A,C      COMPARE REQUESTED COUNT TO BUFFER ROOM
061.360 225      5196X      SUB      L
061.361 170      5197X      MOV      A,B
061.362 234      5198X      SBB      H
061.363 322 370 061 5199X      JNC      $WRIB4    MORE REQUESTED THEN ROOM
061.366 140      5200X      MOV      H,B
061.367 151      5201X      MOV      L,C      USE REQUESTED COUNT
061.370 174      5202X $WRIB4 MOV      A,H
061.371 265      5203X      ORA      L
061.372 302 032 062 5204X      JNZ      $WRIB6    SOME ROOM IN BUFFER
                    5205X
                    5206X *      BUFFER IS FULL, EMPTY IT.
                    5207X
061.375 305      5208X      PUSH     B      SAVE COUNT
061.376 052 050 063 5209X      LHL     T,FWA
062.001 042 052 063 5210X      SHLD   T,PTR    CLEAR REMOVAL POINTER
062.004 353      5211X      XCHG
062.005 052 054 063 5212X      LHL     T,LWA
062.010 175      5213X      MOV      A,L
062.011 223      5214X      SUB      E
062.012 117      5215X      MOV      C,A
062.013 174      5216X      MOV      A,H
062.014 232      5217X      SBB      D
062.015 107      5218X      MOV      B,A      (BC) = DATA IN BUFFER
062.016 072 046 063 5219X      LDA     T,CHA
062.021 377 005      5220X      DB      SYSCALL,WRITE WRITE BUFFER
062.023 301      5221X      POP     E      (BC) = DESIRED COUNT
062.024 322 342 061 5222X      JNC      $WRIB3    GOT THE DATA
                    5223X
                    5224X *      ERROR ON WRITE.
                    5225X
062.027 303 066 062 5226X      JMP     $WRIB5    HAVE ERROR
                    5227X
                    5228X *      GOT THE DATA, MOVE IT FROM BUFFER TO TARGET
                    5229X *
                    5230X *      (BC) = REQUEST COUNT
                    5231X *      (DE) = TO
                    5232X *      (HL) = COUNT
                    5233X *      ((SP)) = FROM
                    5234X
062.032 171      5235X $WRIB6 MOV      A,C
062.033 225      5236X      SUB      L
062.034 117      5237X      MOV      C,A
062.035 170      5238X      MOV      A,B
062.036 234      5239X      SBB      H
062.037 107      5240X      MOV      B,A      REMOVE BYTES ABOUT TO BE MOVED FROM REQUEST COUNT
062.040 305      5241X      PUSH     B
062.041 343      5242X      XTHL
062.042 301      5243X      POP     E      (BC) = COUNT FOR THIS COPY
062.043 343      5244X      XTHL      (HL) = TARGET ADDR, ((SP)) = REMAINING REQ. COUNT
062.044 176      5245X $WRIB7 MOV      A,M
062.045 022      5246X      STAX    R
062.046 023      5247X      INX     D
062.047 043      5248X      INX     H

```

```

062.050 013          5249X      DCX      B
062.051 170          5250X      MOV      A,B
062.052 261          5251X      ORA      C
062.053 302 044 062  5252X      JNZ      $WRIB7      MORE TO GO
062.056 353          5253X      XCHG
062.057 042 052 063  5254X      SHLD     T,PTR      UPDATE POINTER
062.062 301          5255X      POP      B          (BC) = REMAINING COUNT
062.063 303 324 061  5256X      JMF      $WRIB2     SEE IF MORE IN BUFFER
                    5257X
                    5258X *      WRITE COMPLETE,
                    5259X *
                    5260X *      (PSW) = COMPLETION FLAGS
                    5261X
062.066 321          5262X $WRIB8 POP      D          RESTORE TARGET ADDRESS
062.067 341          5263X      POP      H
062.070 303 332 062  5264X      JMF      CTR          COPY TEMP POINTERS BACK TO BLOCK, EXIT

```

```

5264X **      $FWBRK - BREAKOUTPUT /80.02.6C/
5267X *
5268X *      $FWBRK empties the specified buffer by filling it with NULLs
5269X *      and then writing it. Note this is used to insure that block
5270X *      mode I/O is output if it is not really a serial device (eg.
5271X *      writing to AT: from *EDIT*.
5272X *
5273X *
5274X *      ENTRY: HL = FILE BLOCK POINTER
5275X *
5276X *      EXIT: HL = FILE BLOCK POINTER
5277X *      TO $FERROR IF ERROR
5278X *
5279X *      USES: PSW,BC,DE
5280X *
5281X
062.073 315 102 062  5282X $FWBRK CALL   $FWBRK,
062.076 320          5283X      RNC          NO ERROR
                    5284X
062.077 303 161 063  5285X      JMF      $FERROR
                    5286X
062.102 345          5287X $FWBRK. PUSH   H
062.103 315 304 062  5288X      CALL   CBT      COPY BUFFER TO TEMPORARY
062.106 315 116 062  5289X      CALL   $FWBRK1
062.111 341          5290X      POP      H
062.112 315 332 062  5291X      CALL   CTR      COPY TEMPORARY TO BUFFER
062.115 311          5292X      RET
                    5293X
062.116 052 056 063  5294X $FWBRK1 LHLD   T,LWA
062.121 353          5295X      XCHG
062.122 052 052 063  5296X      LHLD   T,PTR     DE = BUFFER LWA
                    5297X      MOV      A,E     HL = BUFFER PTR
062.126 225          5298X      SUB      L
062.127 117          5299X      MOV      C,A
062.130 172          5300X      MOV      A,D
062.131 234          5301X      SBB      H

```

#FWBRK

```

062.132 107      5302X      MOV      B,A      BC = DE - HL
062.133 261      5303X      ORA      C
062.134 310      5304X      RZ              THE BUFFER IS ALREADY FLUSHED
                    5305X
                    5306X *      FILL THE BUFFER WITH NULLS
                    5307X
062.135 170      5308X FWBRK2  MOV      A,B
062.136 261      5309X      ORA      C
062.137 312 151 062 5310X      JZ       FWBRK3  NO MORE LEFT TO FILL
                    5311X
062.142 066 000   5312X      MVI      M,0
062.144 043      5313X      INX      H
062.145 013      5314X      DCX      B
062.146 303 135 062 5315X      JMP      FWBRK2
                    5316X
062.151 052 050 063 5317X FWBRK3  LHLD    T,FWA
062.154 042 052 063 5318X      SHLD   T,FTR
062.157 353      5319X      XCHG
062.160 052 056 063 5320X      LHLD    T,LWA      DE = BUFFER FWA
                                HL = BUFFER LWA
062.163 175      5321X      MOV      A,L
062.164 223      5322X      SUB      E
062.165 117      5323X      MOV      C,A
062.166 174      5324X      MOV      A,H
062.167 232      5325X      SBB      D
062.170 107      5326X      MOV      B,A      BC = HL - DE ( BC = COUNT )
062.171 072 046 063 5327X      LDA      T,CHA
062.174 377 005   5328X      DB      SYSCALL,WRITE
062.176 311      5329X      RET
062.177          5330      XTEXT   FCLO

```

```

                    5332X **      $FCLO - CLOSE FILE BLOCK,
                    5333X *
                    5334X *      $FCLO IS CALLED TO TERMINATE PROCESSING THROUGH A FILE
                    5335X *      BLOCK,
                    5336X *
                    5337X *      ENTRY (HL) = FILE BLOCK ADDRESS
                    5338X *      EXIT TO $FERROR IF ERROR
                                TO CALLER IF OK
                    5339X *
                    5340X *      USES A,F,R,C,D,E
                    5341X
                    5342X
062.177 315 206 062 5343X $FCLO  CALL   $FCLO,
062.202 320          5344X      RNC              NO ERROR
062.203 303 161 063 5345X      JMP      $FERROR
                    5346X
062.206 345      5347X $FCLO,  PUSH   H      SAVE FILE BLOCK ADDRESS
000.000          5348X      ERNZ   FB,FLG-1
062.207 043      5349X      INX      H      (HL) = $FB,FLG
062.210 176      5350X      MOV      A,M
062.211 066 000   5351X      MVI      M,0      CLEAR FLAG
062.213 247      5352X      ANA      A
062.214 312 302 062 5353X      JZ       $FCLO4  FILE NOT OPEN
062.217 346 004   5354X      ANI      FT,OW

```

\$FCLO

```

062.221 312 274 082 5355X JZ $FCLO3 NO WRITING; NO FLUSHING NEEDED
5356X
5357X * WAS OPEN FOR WRITE. SEE IF NEED FLUSH THE LAST SECTOR
5358X
062.224 315 234 030 5359X CALL $INDL
062.227 003 000 5360X DW FB, PTR-FB, FLG
062.231 325 5361X PUSH D SAVE (FB, PTR)
062.232 315 234 030 5362X CALL $INDL (DE) = (FB, FWA)
062.235 001 000 5363X DW FB, FWA-FB, FLG
062.237 341 5364X POP H (HL) = (FB, PTR)
062.240 175 5365X MOV A, L
062.241 223 5366X SUB E
062.242 117 5367X MOV C, A
062.243 174 5368X MOV A, H
062.244 232 5369X SBB B
062.245 107 5370X MOV B, A (BC) = AMOUNT IN BLOCK
062.246 261 5371X ORA C
062.247 312 274 062 5372X JZ $FCLO3 NONE TO FLUSH
5373X
5374X * NEED TO FLUSH BUFFER
5375X *
5376X * (BC) = DATA AMOUNT
5377X * (DE) = FWA
5378X * (HL) = LWA+1
5379X
062.252 171 5380X MOV A, C
062.253 247 5381X ANA A
062.254 312 267 062 5382X JZ $FCLO2 DONT HAVE PARTIAL SECTOR
5383X
5384X * ZERO FILL PARTIAL SECTOR
5385X
062.257 066 000 5386X $FCLO1 MVI M, 0
062.261 043 5387X INX H
062.262 014 5388X INR C
062.263 302 257 062 5389X JNZ $FCLO1
062.266 004 5390X INR B COUNT ANOTHER FULL SECTOR
062.267 341 5391X $FCLO2 POP H (HL) = FB, FWA
062.270 176 5392X MOV A, M (A) = CHANNEL NUMBER
000.000 5393X ERKNZ FB, CHA
062.271 345 5394X PUSH H
062.272 377 065 5395X DB $SYSCALL, WRITE FLUSH
5396X
5397X * READY TO CLOSE FILE.
5398X *
5399X * C SET IF ERROR
5400X * (A) = ERROR CODE
5401X
062.274 341 5402X $FCLO3 POP H (HL) = FILE BLOCK ADDRESS
062.275 330 5403X RC ERROR
000.000 5404X ERKNZ FB, CHA
062.276 176 5405X MOV A, M (A) = CHANNEL NUMBER
062.277 345 5406X PUSH H
062.300 377 048 5407X DB $SYSCALL, CLOSE CLOSE CHANNEL
062.302 341 5408X $FCLO4 POP H (HL) = FILE BLOCK ADDRESS
062.303 311 5409X RET
062.304 5410 XTEXT FUTIL
    
```



```

5412X ** $FUTIL - UTILITY ROUTINES FOR FILE BLOCK ROUTINES.
5413X
5414X ** CBT - COPY BLOCK POINTERS TO TEMP CELLS.
5415X *
5416X * ENTRY (HL) = FILE BLOCK FWA
5417X * EXIT NONE
5418X * USES A,F,H,L
5419X
062.304 325 5420X CBT PUSH D
062.305 305 5421X PUSH B SAVE REGISTERS
000.000 5422X ERRNZ TLEN-10 ASSUME 10 BYTES TO MOVE
062.306 021 046 063 5423X LXI D,T,CHA (DE) = TARGET FOR MOVE
062.311 006 005 5424X MVI B,10/2
062.313 176 5425X CBT1 MOV A,M COPY FILE BUFFER INTO WORK AREA
062.314 022 5426X STAX D
062.315 043 5427X INX H
062.316 023 5428X INX D
062.317 176 5429X MOV A,M
062.320 022 5430X STAX D
062.321 043 5431X INX H
062.322 023 5432X INX D
062.323 005 5433X DCR B
062.324 302 313 062 5434X JNZ CBT1 MORE TO GO
062.327 301 5435X POP B
062.330 321 5436X POP D (DE) = DATA TARGET ADDRESS
062.331 311 5437X RET
5438X
5439X
5440X ** CTB - COPY TEMP CELLS BACK TO FILE BLOCK.
5441X *
5442X * ENTRY (HL) = FILE BLOCK ADDRESS
5443X * EXIT NONE
5444X * USES NONE
5445X
062.332 365 5446X CTB PUSH PSW
062.333 325 5447X PUSH D
062.334 305 5448X PUSH B
062.335 345 5449X PUSH H SAVE REGISTERS
062.336 006 004 5450X MVI B,8/2
062.340 021 046 063 5451X LXI D,T,CHA
062.343 032 5452X CTB1 LDAX D
062.344 167 5453X MOV M,A
062.345 023 5454X INX D
062.346 043 5455X INX H
062.347 032 5456X LDAX D
062.350 167 5457X MOV M,A
062.351 023 5458X INX D
062.352 043 5459X INX H
062.353 005 5460X DCR B
062.354 302 343 062 5461X JNZ CTB1 RESTORE FILE BUFFER VALUES
062.357 341 5462X POP H
062.360 301 5463X POP B
062.361 321 5464X POP D
062.362 361 5465X POP PSW
062.363 311 5466X RET

```

```

5468X **      $FFB - FILE FILE BUFFER,
5469X *
5470X *      $FFB FILLS THE FILE BUFFER BY READING FROM THE FILE.
5471X *
5472X *      ENTRY   NONE
5473X *      EXIT    'C' SET IF READ INCOMPLETE
5474X *            'A' = ERROR CODE
5475X *            'C' CLEAR IF READ COMPLETEE
5476X *            DATA IN BUFFER
5477X *      USES   A,F,D,E,H,L
5478X
5479X
062.364 072 060 063 5480X $FFB LDA   EOFFLG
062.367 037          5481X RAR
062.370 330          5482X RC           EOF
5483X
5484X *      CAN READ MORE. DO SO
5485X
062.371 305          5486X PUSH   B           SAVE COUNT
062.372 052 050 063 5487X LHLD  T,FWA
062.375 042 052 063 5488X SHLD  T,FTR       CLEAR REMOVAL POINTER
063.000 353          5489X XCHG
063.001 052 056 063 5490X LHLD  T,LWA
063.004 042 054 063 5491X SHLD  T,LIM       SET DATA LIMIT
063.007 175          5492X MOV   A,L
063.010 223          5493X SUB   E
063.011 117          5494X MOV   C,A
063.012 174          5495X MOV   A,H
063.013 232          5496X SBB   D
063.014 107          5497X MOV   B,A         (BC) = ROOM IN BUFFER
063.015 072 046 063 5498X LDA   T,CHA
063.020 377 004      5499X DB   SYSCALL,,READ READ BUFFER
063.022 120          5500X MOV   D,B         (D) = SECTORS UNREAD
063.023 301          5501X POP   B           (BC) = DESIRED COUNT
063.024 320          5502X RNC           GOT THE DATA
5503X
5504X *      ERROR ON READ. SEE IF EOF
5505X
063.025 027          5506X RAL
063.026 062 060 063 5507X STA   EOFFLG     SET EOF, WE HOPE
063.031 376 003      5508X CPI   EC,EOF*2+1
063.033 037          5509X RAR
063.034 300          5510X RNE           IS NOT EOF, RETURN NOW!
063.035 072 055 063 5511X LDA   T,LIM+1
063.040 222          5512X SUB   D
063.041 062 055 063 5513X STA   T,LIM+1   SET AMOUNT OF DATA WE DID GET
063.044 247          5514X ANA   A
063.045 311          5515X RET           EXIT WITH DATA
5516X
5517X
5518X **      TEMP CELLS TO HOLD FILE BLOCK POINTERS DURING I/O
5519X
000.000          5520X ERRNZ  FB,CHA
063.046 000          5521X T,CHA DB   0           CHANNEL NUMBER
000.000          5522X ERRNZ  *-T,CHA-FB,FLG
063.047 000          5523X T,FLG DB   0           FLAG BYTE

```

```
000.000          5524X      ERRNZ  *-T.CHA-FB.FWA
063.050 000 000  5525X T.FWA  DW      0
000.000          5526X      ERRNZ  *-T.CHA-FB.PTR
063.052 000 000  5527X T.PTR  DW      0
000.000          5528X      ERRNZ  *-T.CHA-FB.LIN
063.054 000 000  5529X T.LIN  DW      0
000.000          5530X      ERRNZ  *-T.CHA-FB.LWA
063.056 000 000  5531X T.LWA  DW      0
000.012          5532X TLEN  ERU     *-T.CHA      LENGTH OF TEMP BELLS
          5533X
063.060 000      5534X EOFFLG DB      0
          5535      ELSE
          5536      XTEXT  BRP
          5537      ENDIF
```

063.061

5540 PATCH DS 64

PATCH AREA

000.001

```

5543 IF ONECOPY
5544
5545
5546 ** FDN - FILE DESCRIPTOR NODES.
5547 *
5548 * THESE NODES ARE USED TO KEEP TRACK OF FILES WHICH ARE BEING
5549 * HELD IN MEMORY WHILE TRANSFERING.
5550
5551 FDN DS 0 START OF TYPICAL NODE
5552 FDN.LNK EQU *-FDN LINK TO NEXT NODE IN CHAIN
5553 DS 1 ALL IN SAME PAGE, JUST KEEP PAGE INDEX
5554 FDN.STA EQU *-FDN STATUS BYTE
5555 ST.CNT EQU DIF.CNT IS CONTIGUOUS
5556 ST.OPR EQU 00000010B IS BEING READ
5557 ST.OPW EQU 00000001B OPEN FOR WRITE
5558 DS 1 STATUS BYTE
5559 FDN.SIZ EQU *-FDN TOTAL SIZE OF FILE (IF ST.CNT SET)
5560 DS 1 SIZE IN GROUPS
5561 FDN.AMR EQU *-FDN AMOUNT ALREADY READ
5562 DS 2 IN SECTORS
5563 FDN.AMW EQU *-FDN AMOUNT ALREADY WRITTEN
5564 DS 2 IN SECTORS
5565 FDN.ADR EQU *-FDN ADDRESS IN BUFFER
5566 DS 1 ADDRESS/256 (MUST BE EVEN PAGE)
5567 FDN.AIM EQU *-FDN AMOUNT IN MEMORY
5568 DS 1 IN SECTORS
5569 FDNLEN EQU *-FDN ENTRY LENGTH
5570 ORG FDN ORG BACK OVER DEFINITION AREA
5571
5572
5573
5574 ** TABLE, A LINK OF 0 IS A NULL LINK.
5575 *
5576 * THE ENTIRE GROUP OF NODES MUST RESIDE
5577 * IN THE SAME PAGE
5578
5579 FDNFWA EQU * START OF NODES
5580
5581 FDNFRE DB #FDN.1 START OF FREE CHAIN
5582 FDNHEAD DB 0 ACTIVE LIST NOW EMPTY
5583
5584 FDN.1 DS 0
5585 DB #FDN.2 FDN.LNK
5586 DB 0 FDN.STA
5587 DB 0 FDN.SIZ
5588 DW 0 FDN.AMR
5589 DW 0 FDN.AMW
5590 DB 0 FDN.ADR
5591 DB 0 FDN.AIM
5592
5593 FDN.2 DS 0
5594 DB #FDN.3 FDN.LNK
5595 DB 0 FDN.STA
5596 DB 0 FDN.SIZ
5597 DW 0 FDN.AMR
5598 DW 0 FDN.AMW
  
```

5599		DB	0	FDN.ADR
5600		DB	0	FDN.AIM
5601				
5602	FDN.3	DS	0	
5603		DB	#FDN.4	FDN.LNK
5604		DB	0	FDN.STA
5605		DB	0	FDN.SIZ
5606		DW	0	FDN.AMR
5607		DW	0	FDN.AMW
5608		DB	0	FDN.ADR
5609		DB	0	FDN.AIM
5610				
5611	FDN.4	DS	0	
5612		DB	#FDN.5	FDN.LNK
5613		DB	0	FDN.STA
5614		DB	0	FDN.SIZ
5615		DW	0	FDN.AMR
5616		DW	0	FDN.AMW
5617		DB	0	FDN.ADR
5618		DB	0	FDN.AIM
5619				
5620	FDN.5	DS	0	
5621		DB	#FDN.6	FDN.LNK
5622		DB	0	FDN.STA
5623		DB	0	FDN.SIZ
5624		DW	0	FDN.AMR
5625		DW	0	FDN.AMW
5626		DB	0	FDN.ADR
5627		DB	0	FDN.AIM
5628				
5629	FDN.6	DS	0	
5630		DB	#FDN.7	FDN.LNK
5631		DB	0	FDN.STA
5632		DB	0	FDN.SIZ
5633		DW	0	FDN.AMR
5634		DW	0	FDN.AMW
5635		DB	0	FDN.ADR
5636		DB	0	FDN.AIM
5637				
5638	FDN.7	DS	0	
5639		DB	#FDN.8	FDN.LNK
5640		DB	0	FDN.STA
5641		DB	0	FDN.SIZ
5642		DW	0	FDN.AMR
5643		DW	0	FDN.AMW
5644		DB	0	FDN.ADR
5645		DB	0	FDN.AIM
5646				
5647	FDN.8	DS	0	
5648		DB	0	FDN.LNK
5649		DB	0	FDN.STA
5650		DB	0	FDN.SIZ
5651		DW	0	FDN.AMR
5652		DW	0	FDN.AMW
5653		DB	0	FDN.ADR
5654		DB	0	FDN.AIM

```

5655
5656 FDNCNT EQU *-FDN.1/FDNELEN NUMBER OF NODES
5657
5658 SET */256
5659 ERRNZ FDNFWA/256- MUST BE ALL IN SAME PAGE
5660
5661 VOLFLAG DB 0 =0 IF READING FROM SOURCE, =3770 IF WRITING TO DEST
5662 VOLSER DB 0 SERIAL NUMBER OF CURRENT DISK
5663
5664 OBUFLIM DB 0 BUFFER LIMIT/256
5665 OBUFPTR DB 0 NEXT FREE PAGE IN BUFFER/256
5666
5667
5668 ENDIF
5669
063.161 5670 XTEXT FERROR APPEARS HERE TO ALLOW FDN. TO BE IN ONE PAGE
  
```

```

5672X ** $FERROR - PROCESS FILE ERRORS.
5673X *
5674X * $FERROR IS CALLED TO COMPLAIN ABOUT AN ERROR ENCOUNTERED
5675X * WHEN PROCESSING FILES.
5676X *
5677X * ENTRY (A) = ERROR CODE
5678X * (HL) = ADDRESS OF FILE NAME - FB.NAM
5679X * EXIT TO RESTART
5680X * USES ALL
5681X
5682X
063.161 365 5683X $FERROR PUSH PSW SAVE CODE
063.162 315 136 031 5684X CALL $TYPTX
063.165 012 007 105 5685X DB NL,BELL,'ERROR DN FILE',' '+2000
063.205 021 012 000 5686X LXI D,FB.NAM
063.210 031 5687X DAD D
5688X
5689X * PRINT FILE NAME
5690X
063.211 176 5691X $FERR1 MOV A,M
063.212 043 5692X INX H ADVANCE MESSAGE
063.213 247 5693X ANA A
063.214 312 225 063 5694X JZ $FERR2
063.217 315 275 060 5695X CALL $NCHAR
063.222 303 211 063 5696X JMP $FERR1
5697X
5698X * TYPE ERROR MESSAGE
5699X
063.225 315 136 031 5700X $FERR2 CALL $TYFTX.
063.230 040 055 240 5701X DB ' - ',' '+2000
063.233 046 012 5702X MVI H,NL
063.235 361 5703X POP PSW (A) = CODE
063.236 377 057 5704X DB SYSCALL,'ERROR
063.240 303 200 042 5705X JMP RESTART EXIT
  
```

```

063.243 000      5708 COMAND DB      0      COMMAND IN PROGRESS
063.244 000      5709 MODE  DB      0      <<0 IF LINE PASSED ON STACK
063.245 000      5710 JGL   DB      0      /JGL FLAG (<<0 IF /JGL SPECIFIED)
063.246 000      5711 SUPRES DB      0      /SUP FLAG (<<0 OF /SUP SPECIFIED)
063.247 001      5712 SYSTEM DB      1      /S FLAG (=0 IF /S SPECIFIED)
                    5713
063.250 130 130 130 5714 DIRNAM DB      'XXX:DIRECT.SYS',0  DIRECTORY FILE NAME
                    5715
063.267 154 065   5716 BUFFTR DW      BUFF  POINTER TO START OF BUFFER
063.271 000 000   5717 BUFSIZ DW      0      BUFFER LENGTH
  
```

5719 \*\* FILE BLOCKS

```

                    5720
000.000          5721          IF      ,FIP,
063.273          5722 DESTFB  DS      0      DESTINATION FILE BLOCK
063.273 001      5723          DB      CN.DES  CHANNEL NUMBER
063.274 000      5724          DB      0      FLAGS
063.275 361 063   5725          DW      DESTBUF
063.277 361 063   5726          DW      DESTBUF
063.301 361 063   5727          DW      DESTBUF
063.303 361 064   5728          DW      DESTBFE  END OF BLOCK
063.305          5729          DS      FB.NAML  NAME AREA
                    5730
                    5731 DESTFB  DS      0      DUMY BUFFER
                    5732          DB      200     ILLEGAL CHANNEL NUMBER
                    5733          DB      0      FLAGS
                    5734          DW      0
                    5735          DW      0
                    5736          DW      0
                    5737          DW      0      END OF BLOCK
                    5738          DS      FB.NAML  NAME AREA
                    5739          ENDIF
  
```

```

063.326 000 000   5741 NAMTLEN DW      0      NAME TABLE POINTER
063.330 000 000   5742 NAMTMAX DW      0      MAXIMUM SIZE OF NAME TABLE
000.001          5743          IF      ONECOPY
                    5744 NAMTPTR DW      0      POINTER TO ACTIVE ELEMENT IN NAMTAB
                    5745          ENDIF
                    5746
  
```



PRS - PRESET PROGRAM (OVERLAID BY BUFFERS),

PRS

14:43:30 16-MAY-80

```

5750 ***   PRS - PRESET PIP PROGRAM.
5751 *
5752 *   PRS IS CALLED TO PERFORM ONE-TIME-ONLY PRESETTING OF
5753 *   THE PROGRAM ENVIRONMENT.
5754 *
5755 *   THE CODE IS OVERLAID BY BUFFERS AND WORK AREAS WHEN PIP IS RUNNING.
000.000 5756 *   IF .PIP.
5757 *   BE CAREFUL NOT TO USE ANY OF THE BUFFERS AND WORK AREAS BEFORE
5758 *   THE AREA *LINE*.
5759 *   ELSE
5760 *   DO NOT USE ANY OF THE BUFFERS AND WORK AREAS IN *PRS*
5761 *   ENDIF
5762 *
5763 *
5764 *   ENTRY  NONE
5765 *
5766 *   EXIT   IF  CORRECT VERSION OF HDOS
5767 *         NONE
5768 *         ELSE
5769 *         EXIT TO HDOS
5770 *
5771 *   USES  ALL
5772 *
5773 *
063.332 5774 ENTRY  EQU   *   INITIAL ENTRY POINT
063.332 377 011 5775 PRS   DB   SYSCALL, .VERS
063.334 332 026 064 5776   JC   PRS1   ERROR IN GETTING VERSION
063.337 376 026 5777   CPI   VERS
063.341 302 026 064 5778   JNZ   PRS1   NOT CORRECT VERSION OF HDOS
063.344 041 154 065 5779   LXI   H, RMEML (HL) = RUN-TIME HIGH MEMORY
063.347 377 052 5780   DB   SYSCALL, .SETTP SET HI MEMORY
063.351 332 031 064 5781   JC   PRS2   IF ERROR
063.354 041 352 042 5782   LXI   H, CCHIT
063.357 076 003 5783   MVI   A, CTLC
063.361 377 041 5784   DB   SYSCALL, .CTLC SET CTL-C PROCESSING
063.363 076 377 5785   MVI   A, 3770
063.365 377 046 5786   DB   SYSCALL, .CLOSE CLOSE OVERLAY CHANNEL
000.000 5787   IF   .PIP.
5788 *
5789 *   SEE IF COMMAND LINE PASSED ON STACK
5790 *
063.367 041 000 000 5791   LXI   H, 0
063.372 071 5792   DAD   SP
063.373 353 5793   XCHG
063.374 076 200 5794   MVI   A, #STACK
063.376 223 5795   SUB   E
063.377 117 5796   MOV   C, A
064.000 076 042 5797   MVI   A, STACK/256
064.002 232 5798   SBB   D
064.003 107 5799   MOV   B, A (BC) = BYTES ON STACK
064.004 261 5800   ORA   C
064.005 062 244 063 5801   STA   MODE SET MODE <0 IF LINE ON STACK
064.010 312 207 042 5802   JZ   START NO LINE
5803 *
5804 *   HAVE LCOMMAND ON STACK. COPY INTO LINE BUFFER
5805 *   (BC) = COUNT

```

PRS - PRESET PROGRAM (OVERLAID BY BUFFERS).

PRS

14:43:31 16-MAY-80

```

5806 *      (DE) = FWA
5807
064.013 041 034 065 5808 LXI H,LINE
064.016 315 252 030 5809 CALL $MOVE COPY
064.021 066 000 5810 MVI M,0 ENSURE END
5811 ELSE ONECOPY
5812 CALL $DOS DISMOUNT OPERATING SYSTEM
5813 JC PRS2 IF ERROR
5814 CALL $TYPTX
5815 DB NL,TAB,TAB,TAB,' ', 'ONECOPY'
5816 DB NL,TAB,TAB,TAB,'Version: ',VERS/16,' ',VERS%OFHF'0'
5817 DB NL,TAB,TAB,' ', 'Issue: $50.05.00 '
5818 DB NL,NL,' ONECOPY is used to copy files for systems with only one
5819 DB NL,'floppy drive. Read the appropriate manual before using.'
5820 DB ENL
5821 CALL $TYPTX
5822 DB NL,'Insert the initial source disk. Hit RETURN when ready.', '12000
5823 CALL GDWP, /79.11.6C/
5824 CALL $RTL GET CR
5825
5826 *      READ NEW DISK'S LABEL
5827
5828 CALL GETLAB GET LABEL
5829 JC ERROR
5830 CALL MND MOUNT NEW DISK
5831 JC ERROR IF ERROR
5832 LDA LABEL+LAB.SER
5833 STA VOLSER SET CURRENT VOLUME NUMBER
5834 ENDIF
064.023 303 207 042 5835 JMP START START PROGRAM
5836
064.026 076 050 5837 PRS1 MVI A,EC.NCV NOT CORRECT VERSION
064.030 087 5838 STC
064.031 046 012 5839 PRS2 MVI H,NL
064.033 377 057 5840 DB SYSCALL,'ERROR'
064.035 303 347 042 5841 JMP EXIT
5842
000.001 5843 IF ONECOPY
5844 XTEXT DTB
5845 XTEXT DOS
5846 ENDIF
5847
064.040 5848 MEHL EQU * MEMORY LENGTH

```

```

5851 ** THE FOLLOWING BUFFERS AND AREAS OVERLAY THE PRS CODE.
5852 *
5853 * *PRS* MAY NOT USE ANY CELLS BELOW *LINE*, AT THE
5854 * RISK OF SMASHING ITSELF
5855
063.332 5856 ORG PRS
5857
063.332 5858 DEFALT DS 6 DEFAULT BLOCK
5859
063.340 5860 MWNA DS FB.NAML MUN WORK AREA
5861
000.000 5862 IF .PIP,
063.361 5863 DESTBUF DS 256 DESTINATION FILE BUFFER (ALSO USED BY *CCW*)
064.361 5864 DESTBFE EQU * END OF BUFFER
5865 ENDIF
5866
5867 ** * * NOTE * *
5868 * DIRWORK USES THE SYSTEM SCRATCH AREA, LABEL. DIRWORK WILL NOT
5869 * BE PRESERVED DURING A SYSCALL !!
5870
027.000 5871 LABEL EQU S.GRT2+256 USE EXTRA GRT TABLE AS BUFFER /79.12.GC/
5872
5873 *DIRWORK EQU SECSER USE SECTOR SCRATCH AREA /79.11.GC/

5875 ** PIO.XXX - IMAGE OF SYSTEM AIO.XXX AREA
5876 *
5877 * THESE CELLS MIRROR THE SYSTEM AIO.XXX AREA
5878
064.361 5880 PIO.DEV DS 2 DEVICE CODE
064.363 5881 PIO.UNI DS 1 UNIT NUMBER (0-9)
5882
064.364 5883 PIO.DIR DS DIRELEN DIRECTORY ENTRY
5884
065.013 5885 %FOPWRK DS FB.NAML WORK AREA FOR %FOPE
5886
000.000 5888 IF .PIP,
000.374 5889 ERRMI *-HEML FOLLOWING MUST NOT OVERLAY *PRS*
5890 ENDIF
065.034 5891 LINE DS 80 COMMAND BUFFER
5892
065.154 5894 NAMTAB DS 0 NAME TABLE
5895
002.000 5897 BUFMINL EQU 512 MINIMUM SIZE FOR BUFFER (WHEN IN USE)
065.154 5898 BUFF EQU * BUFFER AREA STARTS AFTER NAMTAB
5899
065.154 5900 RHEML EQU * INITIAL RUNNING MEMORY LENGTH
5901
5902
5903
065.154 5904 END
  
```

ASSEMBLY COMPLETE  
 5904 STATEMENTS  
 0 ERRORS DETECTED  
 8748 BYTES FREE







ABS.LDA	000002	867L						
ABS.LEN	000004	868L						
AC.DLY	000156	66E						
ACL	043274	920	1146L					
AEN	052307	2614	2942L	3557	3640			
AENA	052361	2942	2959	2984L				
AIO.CGN	041047	634L						
AIO.CHA	041116	649L						
AIO.GNT	041111	645L						
AIO.CSI	041050	635L						
AIO.IDA	041041	630E						
AIO.DES	041055	639L						
AIO.DEV	041057	640L						
AIO.BIR	041062	643L						
AIO.DTA	041053	638L						
AIO.EOF	041113	647L						
AIO.EOM	041112	646L						
AIO.FLG	041043	631L						
AIO.GRT	041044	632L						
AIO.LGN	041051	636L						
AIO.LSI	041052	637L						
AIO.SPG	041046	633L						
AIO.TFP	041114	648L						
AIO.UNI	041061	641L						
AIO.VEC	041040	629L						
BELL	000007	473E	1295	2874	2889	3035	5685	
BKSP	000010	475E						
BLS	047254	2326	2566L					
BLS1	047303	2576L	2617					
BLS2	047324	2583	2585L					
BLS3	047363	2598	2605L					
BLS4	047376	2602	2606	2614L				
BLSA	050010	2567	2577	2593	2619L			
BLSB	050016	2571	2596	2601	2620L			
BLSC	050017	2567	2584	2621L				
BOGT.F	000001	609E						
BRIEF	045332	943	2304L					
BSL	053002	1190	2187	2217	2979L			
BSL1	053010	2984L	3000					
BSL2	053043	2997L						
BSLA	053053	2979	2992	3002L				
BUFF	065154	912	5716	5898E				
BUFMINL	002000	3468	5897E					
BUFPTR	063267	913	1246	3463	3692	3802	5716L	
BUFSIZ	063271	909	1243	1259	3467	3694	3801	5717L
C.STX	000002	477E						
C.SYN	000026	476E						
CAD	053356	2425	2988	3174	3240L	3565	3745	3749
CAD.	053362	2585	3243L					
CAD0	053364	3241	3244L					
CAD1	054051	3259	3261	3263	3271L			
CAD2	054114	3274	3292L					
CAD2.4	054142	3306L	3309					
CAD2.6	054150	3303	3310L					
CAD3	054207	3313	3331L					
CAD4	054211	3265	3267	3336L				
CAD5	054224	3272	3281	3288	3319	3322	3346L	
CADA	054230	3245	3304	3350L				

CB.CLI	000100	747E	762						
CB.MTL	000040	746E							
CB.SPK	000200	748E							
CB.SSI	000020	745E							
CBT	062304	5172	5288	5420L					
CBT1	062313	5425L	5434						
GCHIT	042352	969L	5782						
CCW	053054	2994	3021L						
CCW1	053063	3025L	3028						
CDA	055057	2943	3187	3496L	3766				
CDA5	055123	3498	3503	3508	3530L	3542			
CDA6	055141	3537	3539L						
CDA7	055143	3536	3541L						
CDR.H84	000001	552E							
CDR.H85	000000	551E							
CFE	053173	2418	3062L	3624					
CFS	053213	2503	2690	3083L					
CFS.	053216	3084L							
CFS1	053221	3085L	3090						
CN.DES	000001	45E	1206	1235	1263	1280	1296	1306	5723
CN.DIR	000002	46E	2367	2400	2478	3588	3597	3652	
CN.SOU	000000	44E	1220	1248	1270	2243			
CO.FLG	000001	701E	3992						
COMMAND	063243	926	932	1063	1097	1105	1110	1119	5708L
COPY	043317	939	1182E						
COPY1	044003	1204	1213L	1279	1284				
COPY2	044070	1228	1242L						
COPY3	044073	1243L	1268						
COPY4	044127	1253	1255	1259L					
COPY5	044215	1216	1288L						
COPY6	044255	1290	1300L						
COPY7	044303	1304	1314L						
COPYA	044346	1187	1202	1226	1277	1302	1326L		
COPYC	044347	1184	1217	1288	1327L				
COPYD	044350	1195	1229	1328L	1329				
COPYDL	000021	1193	1329E						
COPYE	044324	1318	1322L						
CR	000015	469E							
CS.FLG	000200	702E							
CSL.CHR	000001	679E							
CSL.ECH	000200	677E							
CSL.WRF	000002	678E							
CTR	062332	5264	5291	5446L					
CTB1	062343	5452L	5461						
CTLA	000001	484E							
CTLB	000002	485E							
CTLC	000003	486E	5783						
CTLD	000004	487E	4079						
CTLQ	000017	488E							
CTLP	000020	489E							
CTLQ	000021	490E							
CTLS	000023	491E							
CTLZ	000032	492E							
CTP.25B	000010	687E							
CTP.BKM	000002	688E							
CTP.BNS	000200	684E							
CTP.MLI	000040	685E							
CTP.MLO	000020	686E							



CROSS REFERENCE TABLE

CTP.TAB	000001	689E						
CTS	053231	1392	2813	3106L				
CWM	053246	2430	3130L	3138	3629			
CWM1	053255	3132	3135L					
D.CON	040110	390L						
D.DLYHS	040244	510L						
D.DLYMO	040243	509L						
D.DRVTB	040251	515L						
D.DVCTL	040242	507L						
D.E.CHK	040267	526L						
D.E.HCK	040270	527L						
D.E.HSY	040266	525L						
D.E.MDS	040265	524L						
D.E.TRK	040272	529L						
D.E.VOL	040271	528L						
D.ERR	040265	523L						
D.ERRL	040273	530L						
D.HECNT	040261	517L						
D.DECNT	040264	519L						
D.OPR	040273	534L						
D.OPW	040275	535L						
D.RAM	040240	393L	502	537				
D.RAML	000037	537E						
D.SECNT	040262	518L						
D.TRKPT	040245	512L						
D.TS	040241	505L						
D.TT	040240	504L						
D.VEC	040130	392L						
D.VOLFT	040247	513L						
DAD1	060122	4658	4671	4673	4696L			
DADR	060125	4675	4700L					
DC.ABT	000007	724L						
DC.CLO	000006	723L						
DC.LOD	000011	726L						
DC.MAX	000012	727L						
DC.MOU	000010	725L						
DC.OPR	000003	720L						
DC.OPU	000005	722L						
DC.OPW	000004	721L						
DC.REA	000000	717L						
DC.RER	000002	719L						
DC.WRI	000001	718L						
DDF	053264	1185	2214	2317	3160L			
DDF.BOL	000011	824E						
DDF.BOO	000000	823L						
DDF.LAB	000011	825L						
DDF.RGT	000012	826L						
DDF.USR	000014	827L						
DDF1	053271	3163L	3168					
DDF1.0	053304	3169L	3176					
DDF2	053307	3166	3173L					
DDFA	053336	3169	3208L					
DEFALT	063332	2987	3173	3821	3866	3870	5858L	
DEL1	045103	2172L	2177					
DEL2	045123	2175	2186L					
DEL5	045133	2192L	2200					
DELETE	045100	950	2167E					
DESTBFE	064361	5728	5864E					

















## CROSS REFERENCE TABLE

TBL3	057215	4187	4198L
TLEN	000012	5422	5532E
TPL1	057266	4319L	
UC.2SB	000004	91E	
UC.5BW	000000	87E	
UC.6BW	000001	88E	
UC.7BW	000002	89E	
UC.8BW	000003	90E	
UC.RI	000020	110E	
UC.CTS	000020	119E	
UC.DCS	000001	115E	
UC.DDR	000002	116E	
UC.DLA	000200	96E	
UC.DR	000001	106E	
UC.DRL	000010	118E	
UC.DSR	000040	120E	
UC.DTR	000001	99E	
UC.EDA	000001	77E	
UC.EPS	000020	93E	
UC.FE	000010	109E	
UC.IID	000006	84E	
UC.IIF	000001	83E	
UC.LDD	000020	103E	
UC.MSI	000010	80E	
UC.OR	000002	107E	
UC.OU1	000004	101E	
UC.OU2	000010	102E	
UC.PE	000004	108E	
UC.FEN	000010	92E	
UC.RI	000100	121E	
UC.RLS	000200	122E	
UC.RSI	000004	79E	
UC.RTS	000002	100E	
UC.SB	000100	95E	
UC.SKP	000040	94E	
UC.TER	000004	117E	
UC.THE	000040	111E	
UC.TRE	000002	78E	
UC.TSE	000100	112E	
UCI.ER	000020	156E	
UCI.IE	000002	158E	
UCI.IR	000100	154E	
UCI.RE	000004	157E	
UCI.RO	000040	155E	
UCI.TE	000001	159E	
UDDN1	060175	4719L	4735
UDDN1.5	060227	4739L	4746
UDDN2	060231	4732	4744L
UDDN3	060232	4745L	4749
UDR	000000	131E	
UMI.16X	000002	149E	
UMI.1B	000100	139E	
UMI.1X	000001	148E	
UMI.2B	000300	141E	
UMI.64X	000003	150E	
UMI.HB	000200	140E	
UMI.L5	000000	144E	
UMI.L6	000004	145E	

## CROSS REFERENCE TABLE

UMI.L7	000010	146E				
UMI.LB	000014	147E				
UMI.PA	000020	143E				
UMI.FE	000040	142E				
UNT.DIS	000005	261L				
UNT.FLG	000000	258L				
UNT.GRT	000001	259L	2359			
UNT.GTS	000003	260L				
UNT.SIZ	000007	263E				
UD.CLK	000001	764E				
UD.DDU	000002	763E				
UD.HLT	000200	761E				
UD.NFR	000100	762E				
UR.DLL	000000	72E				
UR.DLM	000001	74E				
UR.IER	000001	76E				
UR.IIR	000002	82E				
UR.LCR	000003	86E				
UR.LSR	000005	105E				
UR.MCR	000004	98E				
UR.MSR	000006	114E				
UR.RBR	000000	68E				
UR.THR	000000	70E				
USERFWA	042200	402E	874	876	877	
USR	000001	132E				
USR.FE	000040	163E				
USR.DE	000020	164E				
USR.FE	000010	165E				
USR.RXR	000002	167E				
USR.TXE	000004	166E				
USR.TXR	000001	168E				
VERB	000026	409E	2832	2832	5777	
VERSN	050373	945	2811E			
XCHGBC	060300	4553	4557	4565	4567	4826L

13974 BYTES FREE