

DDDDDDDDDD		RRRRRRRRRR	IIIIIIIIII	VVV	VVV	EEEEEEEEEEEE	RRRRRRRRRR	
DDDDDDDDDD		RRRRRRRRRR	IIIIIIIIII	VVV	VVV	EEEEEEEEEEEE	RRRRRRRRRR	
DDDDDDDDDD		RRRRRRRRRR	IIIIIIIIII	VVV	VVV	EEEEEEEEEEEE	RRRRRRRRRR	
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDD	DDD	RRR	III	VVV	VVV	EEE	RRR	RRR
DDDDDDDDDD		RRR	IIIIIIIIII	VVV	VVV	EEEEEEEEEEEE	RRR	RRR
DDDDDDDDDD		RRR	IIIIIIIIII	VVV	VVV	EEEEEEEEEEEE	RRR	RRR
DDDDDDDDDD		RRR	IIIIIIIIII	VVV	VVV	EEEEEEEEEEEE	RRR	RRR

```

TTTTTTTTTT  SSSSSSSS  DDDDDDDD  RRRRRRRR  IIIIII  VV  VV  EEEEEEEEE  RRRRRRRR
TTTTTTTTTT  SSSSSSSS  DDDDDDDD  RRRRRRRR  IIIIII  VV  VV  EEEEEEEEE  RRRRRRRR
TT          SS          DD          RR          II          VV  VV  EE          RR          RR
TT          SS          DD          RR          II          VV  VV  EE          RR          RR
TT          SS          DD          RR          II          VV  VV  EE          RR          RR
TT          SS          DD          RR          II          VV  VV  EE          RR          RR
TT          SSSSSS     DD          RRRRRRRR  II          VV  VV  EEEEEEEEE  RRRRRRRR
TT          SSSSSS     DD          RRRRRRRR  II          VV  VV  EEEEEEEEE  RRRRRRRR
TT          SS          DD          RR          II          VV  VV  EE          RR          RR
TT          SS          DD          RR          II          VV  VV  EE          RR          RR
TT          SS          DD          RR          II          VV  VV  EE          RR          RR
TT          SSSSSSSS  DDDDDDDD  RR          RR          IIIIII  VV  VV  EEEEEEEEE  RR          RR
TT          SSSSSSSS  DDDDDDDD  RR          RR          IIIIII  VV  VV  EEEEEEEEE  RR          RR

```

```

LL          IIIIII  SSSSSSSS
LL          IIIIII  SSSSSSSS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SSSSSS
LL          II      SSSSSS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS

```

```

....
....
....

```

(1)	499	DRIVER TABLES
(1)	790	UNIT INITIALIZATION ROUTINE
(1)	935	TEST NBA (NEED BUFFER ADDRESS)
(1)	1034	START I/O OPERATION
(2)	1233	NOP AND SIMULATED FUNCTIONS
(2)	1268	READ HARDWARE FUNCTIONS
(2)	1353	WRITE FUNCTIONS
(2)	1420	POSITIONING FUNCTIONS
(2)	1563	FORMAT COMMANDS
(2)	1615	CONTROL COMMANDS
(2)	1641	INITIALIZE AND GET STATUS
(2)	1693	COMPLETION PROCESSING
(2)	1740	HARDWARE COMMAND EXECUTOR
(3)	2170	TS11/TS04 INTERRUPT SERVICE ROUTINE
(3)	2241	TIMEOUT HANDLER
(3)	2358	TS11/TS04 REGISTER DUMP ROUTINE

:MMD0331
-1

```

0000 1 .TITLE TSDRIVER - VAX/VMS TS11/TS04 MAGTAPE SUBSYSTEM DRIVER
0000 .1 .IDENT 'V04-001'
0000 3
0000 4
0000 5 *****
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
0000 9 * ALL RIGHTS RESERVED. *
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
0000 16 * TRANSFERRED. *
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
0000 20 * CORPORATION. *
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
0000 24 *
0000 25 *
0000 26 *****
0000 27
0000 .1 F. E. OUYANG 2-APR-79
0000 .2
0000 .3 MODIFIED BY:
0000 .4
0000 .5 V04-001 MMD0331 Meg Dumont, 29-Oct-1984 12:50
0000 .6 Delete the check for powerfail in the TEST_NBA routine.
0000 .7 Clear the number of bytes returned on a read block
0000 .8 when a tape mark is encountered
0000 32
0000 33 V03-017 MMD0317 Meg Dumont, 25-Jul-1984 11:13
0000 34 Add support for the UCBSL_MEDIA_ID field
0000 35
0000 36 V03-016 MMD0304 Meg Dumont, 27-Jun-1984 15:24
0000 37 Fix to 296 so that only READ REVERSE into BOT returns ENDOFFILE
0000 38
0000 39 V03-015 MMD0296 Meg Dumont, 3-May-1984 9:45
0000 40 Fix to fix MMD0265 we really must return SSS_NORMAL not
0000 41 anyother error code.
0000 42
0000 43 V03-014 ROW0355 Ralph O. Weber 30-APR-1984
0000 44 Modify processing of the IOSM_OPPOSITE modifier so that its
0000 45 use is limited to IOS_REREADN and IOS_REREADP functions by
0000 46 code, rather than by comments. This provides some protection
0000 47 against accidental misuse of the IOSM_CLSEREXCP bit which is
0000 48 relivant only for tape class driver tapes but which shares the
0000 49 same modifier bit as IOSM_OPPOSITE.
0000 50
0000 51 V03-013 RAS0300 Ron Schaefer 27-Apr-1984
0000 52 Add DEVSM_NNM characteristic to DECHAR2 so that these
0000 53 devices will have the 'node$' prefix.

```

:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
-4

0000	54	:	
0000	55	:	V03-012 MMD0265 Meg Dumont, 22-Mar-1984 15:28
0000	56	:	Fix so that reverse into BOT returns \$\$\$_ENDOFFILE Like other
0000	57	:	drivers.
0000	58	:	
0000	59	:	V03-011 MMD0225 Meg Dumont, 23-Jan-1984 11:27
0000	60	:	Deleted the check in the drivers' unit init routine which
0000	61	:	checked on powerfail to see if the TS SUBSYSTEM was ready
0000	62	:	before reloading registers etc.. This check was no
0000	63	:	longer necessary since Robert added the code TEST_NBA which
0000	64	:	makes sure the controller is available before we allow
0000	65	:	the QIO to start on the device.
0000	66	:	
0000	67	:	V03-010 MMD0219 Meg Dumont, 9-Jan-1984 13:59
0000	68	:	Instead of checking for powerfail at TS_INIT check
0000	69	:	for command buffer allocated. Fix for support of
0000	70	:	switchable unibus
0000	71	:	
0000	72	:	V03-009 ROW0258 Ralph O. Weber 17-NOV-1983
0000	73	:	The Paul Painter Memorial Enhancement
0000	74	:	Named for one of the unfortunate customers who suffered much
0000	75	:	to determine the great UCBSL_MT_RECORD secret while trying to
0000	76	:	create a user-written magtape driver, this change eliminates
0000	77	:	use of the device dependent field, UCBSL_MS_RECORD in favor of
0000	78	:	the device independent field, UCBSL_RECORD.
0000	79	:	
0000	80	:	V03-008 ROW0213 Ralph O. Weber 20-AUG-1983
0000	81	:	Change basing for device-dependent UCB from UCBSL_DP_LINK+4 to
0000	82	:	a field independent UCBSK_LCL_TAPE_LENGTH. This allows the
0000	83	:	device-independent UCB to be altered without having to edit
0000	84	:	this module.
0000	85	:	
0000	86	:	V03-007 BLS0234 Benn Schreiber 9-Aug-1983
0000	87	:	Use general addressing mode for EXE\$READ_TODR.
0000	88	:	
0000	89	:	V03-006 KDM0060 Kathleen D. Morse 14-Jul-1983
0000	90	:	Change references to IPR TODR to use cpu-dependent
0000	91	:	routine, EXE\$READ_TODR.
0000	92	:	
0000	93	:	V03-005 RLRDPATH1 Robert L. Rappaport 31-May-1983
0000	94	:	Allow UCB to include new DUAL PORT extension by
0000	95	:	changing base of where we begin the private TSDRIVER
0000	96	:	extension from UCBSL_DPC+4 to UCBSL_DP_LINK+4.
0000	97	:	
0000	98	:	V03-004 RLRTRACE Robert L. Rappaport 11-Feb-1983
0000	99	:	Add conditionally assembled trace facility.
0000	100	:	
0000	101	:	V03-003 RLR52135 Robert L. Rappaport 22-Dec-1982
0000	102	:	Prevent reverse into BOT from returning \$\$\$_OPINCOMPL.
0000	103	:	
0000	104	:	V03-002 RLR0001 Robert L. Rappaport 15-July-1982
0000	105	:	Prevent logging two errors for each soft retry.
0000	106	:	
0000	107	:	V03-001 KDM0002 Kathleen D. Morse 28-Jun-1982
0000	108	:	Added \$DCDEF, \$DEVDEF, \$DYNDEF, \$PRDEF and \$VADEF.
0000	109	:	
0000	110	:	

```

0000 111 : TS11/TS04 MAGTAPE DRIVER
0000 112 :
0000 113 : MACRO LIBRARY CALLS
0000 114 :
0000 115 :
0000 116 $CRBDEF ;DEFINE CRB OFFSETS
0000 117 $DCDEF ;DEFINE DEVICE TYPES
0000 118 $DDBDEF ;DEFINE DDB OFFSETS
0000 119 $DEVDEF ;DEFINE DEVICE TYPES
0000 120 $DPTDEF ;DEFINE DPT OFFSETS
0000 121 $DYNDEF ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000 122 $EMBDEF ;DEFINE EMB OFFSETS
0000 123 $IDBDEF ;DEFINE IDB OFFSETS
0000 124 $IODEF ;DEFINE I/O FUNCTION CODES
0000 125 $IRPDEF ;DEFINE IRP OFFSETS
0000 126 $MTDEF ;DEFINE MAGTAPE STATUS BITS
0000 127 $PRDEF ;DEFINE PROCESSOR REGISTERS
0000 128 $SSDEF ;DEFINE QIO STATUS RETURN CODES
0000 129 $UCBDEF ;DEFINE UCB OFFSETS
0000 130 $VADEF ;DEFINE VIRTUAL ADDRESS FIELDS
0000 131 $VECDEF ;DEFINE INTERRUPT DISPATCH VECTOR OFFSETS
0000 132 $WCBDEF ;DEFINE WCB OFFSETS
0000 133 :
0000 134 :
0000 135 : LOCAL MACROS
0000 136 :
0000 137 : EXECUTE HARDWARE COMMAND AND BRANCH ON RETRIABLE ERROR CONDITION
0000 138 :
0000 139 :
0000 140 .MACRO EXHC BDST,HC
0000 141 .IF NB HC
0000 142 MOVZBL #CD'HC,RO ;GET HARDWARE COMMAND INDEX
0000 143 .ENDC
0000 144 BSBW HCEX ;CALL HARDWARE COMMAND EXECUTION ROUTINE
0000 145 .WORD BDST--2 ;BRANCH ADDR. ON ERROR CONDITION
0000 146 .ENDM EXHC
0000 147 :
0000 148 : MACRO TO CALL G^IOC$LOADUBAMAPA
0000 149 :
0000 150 .MACRO LOADUBAA
0000 151 JSB G^IOC$LOADUBAMAPA
0000 152 .ENDM LOADUBAA
0000 153 :
0000 154 :
0000 155 :
0000 156 : GENERATE HARDWARE COMMAND TABLE ENTRY AND CASE TABLE INDEX SYMBOL
0000 157 :
0000 158 :
0000 159 .MACRO GENHC HC
0000 160 CD'HC=<.-HCTAB>/2 ;DEFINE TABLE INDEX SYMBOL
0000 161 .WORD HC ;HARDWARE COMMAND TABLE ENTRY
0000 162 .ENDM GENHC
0000 163 :
0000 164 :
0000 165 :
0000 166 : LOCAL SYMBOLS
0000 167 :

```

```

0000 168 :
0000 169 : TS11/TS04 COMMAND PACKET DEFINITION
0000 170 :
0000 171 :
0000 172 :
0000 173 $DEFINI MS
0000 174
0000 175
00000000 0000 176 =0 $DEF
0000 177 $DEF MS_CPHD .BLKW 1 ;RESET PC??????
0002 178 _VIELD MS_CPHD,0,<- ;COMMAND PACKET HEADER
0002 179 <COD,5>,- ;
0002 180 <,2>,- ;COMMAND CODE FIELD
0002 181 <IE,,M>,- ;B5-B6 ALWAYS 0 FOR TS04
0002 182 <MOD,4>,- ;INTERRUPT ENABLE
0002 183 - ;COMMAND MODE FIELD(B11-B8)
0002 184 <SWB,,M>,- ; B8=REVERSE & B9=RETRY
0002 185 <OPP,,M>,- ;SWAP BYTES BIT(B12)
0002 186 <CVC,,M>,- ;OPPOSITE BIT(B13)
0002 187 <ACK,,M>,- ;CLEAR VOLUME CHECK(B14)
0002 188 > ;ACKNOWLEDGE BIT(B15)
0002 189 $DEF MS_BACT .BLKW 1 ;BUS ADDRESS(B15-B0) OR COUNT
0004 190 $DEF MS_BA1 .BLKW 1 ;BUS ADDRESS B17-B16(RIGHT JUST)
0006 191 $DEF MS_CNT .BLKW 1 ;BYTE COUNT
0008 192 ;FOR WRITE CHARACTERISTIC DATA
0008 193 $DEF MS_MBA0 .BLKW 1 ;MESSAGE BUFFER ADDR. WRD 1
000A 194 $DEF MS_MBA1 .BLKW 1 ;MESSAGE BUFFER ADDR. WRD 2
000C 195 $DEF MS_LNTH .BLKW 1 ;MESSAGE BUFFER LENGTH(ALWAYS 14.)
000E 196 $DEF MS_CHWD .BLKW 1 ;CHARACTERISTIC WORD
0010 197 _VIELD MS_CHWD,4,<- ;
0010 198 <ERI,,M>,- ;ENABLE MESSAGE BUFFER RELEASE INTERRUPTS
0010 199 <EAI,,M>,- ;ENABLE ATTENTION INTERRUPTS
0010 200 <ENB,,M>,- ;USED WITH ESS BIT***
0010 201 <ESS,,M>,- ;ENABLE SKIP TAPE MARKS STOP
0010 202 >
0010 203
0010 204 :
0010 205 : TS11/TS04 MESSAGE PACKET DEFINITION
0010 206 :
0010 207 :
0010 208 :
0010 209 $DEF MS_MHD .BLKW 1 ;MESSAGE PACKET
0012 210 _VIELD MS_MHD,0,<- ;MESSAGE HEADER WORD
0012 211 <COD,5>,- ;MESSAGE CODE FIELD
0012 212 <FMT,3>,- ;FORMAT FIELD
0012 213 <CLS,4>,- ;CLASS CODE FIELD
0012 214 <RSR,3>,- ;RESERVED FIELD
0012 215 <ACK,,M>,- ;MESSAGE ACKNOWLEDGE BIT(B15)
0012 216 >
0012 217 $DEF MS_LNH .BLKW 1 ;MESSAGE LENGTH WORD
0014 218 ;HIGH BYTE=0,LOW BYTE=1010(LENGTH)
0014 219 $DEF MS_RBPC .BLKW 1 ;RESIDUAL BYTE/POSITION COUNT
0016 220 $DEF MS_XSRO .BLKW 1 ;EXTENDED STATUS REGISTER 0
0018 221 _VIELD MS_XSRO,0,<- ;
0018 222 <EOT,,M>,- ;END OF TAPE DETECTED(B0)
0018 223 <BOT,,M>,- ;BEGINNING OF TAPE(B1)
0018 224 <WLK,,M>,- ;WRITE LOCKED(B2)

```

0018	225		<PED,,M>,-	:PHASE ENCODED DRIVE(B3)
0018	226		<VCK,,M>,-	:VOLUME CHECK(B4)
0018	227		<IE,,M>,-	:INTERRUPT WAS ENABLED(B5)
0018	228		<ONL,,M>,-	:DEVICE ON-LINE(B6)
0018	229		<MOT,,M>,-	:TAPE MOVING ON LAST COMMAND(B7)
0018	230		<ILA,,M>,-	:ILLEGAL ADDRESS(B8)
0018	231		<ILC,,M>,-	:ILLEGAL COMMAND(B9)
0018	232		<NEF,,M>,-	:NON-EXECUTABLE FUNCTION(B10)
0018	233		<WLE,,M>,-	:WRITE LOCK ERROR(B11)
0018	234		<RLL,,M>,-	:RECORD LENGTH LONG(B12)
0018	235		<LET,,M>,-	:LOGICAL END OF TAPE(B13)
0018	236		<RLS,,M>,-	:RECORD LENGTH SHORT(B14)
0018	237		<TMK,,M>,-	:TAPE MARK DETECTED(B15)
0018	238		>	:
0018	239	\$DEF	MS_XSR1 .BLKW 1	:EXTENDED STATUS REGISTER 1
001A	240		-VIELD MS_XSR1,0,<-	:
001A	241		<MTE,,M>,-	: (PE) MULTI-TRACK ERROR
001A	242		-	: (NRZ) VERTICAL PARITY ERROR
001A	243		<UNC,,M>,-	: (PE) UNCORRECTABLE DATA ERROR(B1)
001A	244		-	: (NRZ) CYCLIC REDUNDANCY CHECK ERROR
001A	245		<POL,,M>,-	: (PE) POSTAMBLE LONG(B2)
001A	246		-	: (NRZ) LONGITUDINAL REDUNDANCY CHECK ERROR
001A	247		<POS,,M>,-	: (PE) POSTAMBLE SHORT(B3)
001A	248		-	: (NRZ) NOISE RECORD
001A	249		<IED,,M>,-	: (PE) INVALID END DATA(B4)
001A	250		-	: (NRZ) LRC WAS 0.
001A	251		<IPO,,M>,-	: (PE) INVALID POSTAMBLE(B5)
001A	252		-	: (NRZ) ILLEGAL TAPE MARK
001A	253		<SYN,,M>,-	: (PE) SYNCH ERROR(B6)
001A	254		-	: (NRZ) FRAME DROPOUT
001A	255		<IPR,,M>,-	: (PE) INVALID PREAMBLE(B7)
001A	256		<,1>,-	: RESERVED BIT
001A	257		<SCK,,M>,-	: SPEED CHECK(B9)
001A	258		<DBF,,M>,-	: (PE) DESKEW BUFFER FAIL(B10)
001A	259		-	: (NRZ) NRZ BOARD FIFO OVERFLOW
001A	260		<TIG,,M>,-	: TRASH IN GAP(B11)
001A	261		<CRS,,M>,-	: CREASE DETECTED(B12)
001A	262		<COR,,M>,-	: CORRECTABLE DATA(B13)
001A	263		<,1>,-	: UNUSED BIT(B14)
001A	264		<DLT,,M>,-	: DATA LATE(B15)
001A	265		>	:
001A	266	\$DEF	MS_XSR2 .BLKW 1	:EXTENDED STATUS REGISTER 2
001C	267		-VIELD MS_XSR2,0,<-	:
001C	268		<DTP,8>,-	: DEAD TRACK PARITY,B7-B0
001C	269		<XSK,,M>,-	: EXCESSIVE SKEW(B9)
001C	270		<WCF,,M>,-	: WRITE CLOCK FAIL(B10),BROKEN HARDWARE
001C	271		<,1>,-	: B11 NOT USED
001C	272		<CAF,,M>,-	: CAPSTAN ACCELERATION FAIL(B12)
001C	273		<BPE,,M>,-	: SERIAL BUS PARITY ERROR AT DRIVE(B13)
001C	274		<SIP,,M>,-	: SILO PARITY ERROR(B14)
001C	275		<OPM,,M>,-	: OPERATION IN PROGRESS(B15)
001C	276		>	:
001C	277	\$DEF	MS_XSR3 .BLKW 1	:EXTENDED STATUS REGISTER 3
001E	278		-VIELD MS_XSR3,0,<-	:
001E	279		<RTB,,M>,-	: REVERSE INTO BOT(B0)
001E	280		<LXS,,M>,-	: LIMIT EXCEEDED STATICALLY(B1)
001E	281		<NOI,,M>,-	: NOISE RECORD(B2)


```

001E 282 <DCK,,M>,- ;DENSITY CHECK(B3)
001E 283 <CRF,,M>,- ;CAPSTAN RESPONSE FAIL(B4)
001E 284 <REV,,M>,- ;TAPE MOVED BACKWARDS(B5)
001E 285 <OPI,,M>,- ;OPERATION IN COMPLETE(B6)
001E 286 <LMX,,M>,- ;TAPE LIMIT EXCEEDED(B7)
001E 287 <FEC,,B>,- ;B15-B8, FATAL ERROR CODE(U-DIAGNOSTIC)
001E 288 >
001E 289
001E 290 $DEFEND MS
0000 291
6CE9300B 0000 292 MEDIA_ID_TS11 = ^X<6CE9300B>
0000 293 :
0000 294 : TS11/TS04 TSSR TERMINATION CLASS CODES
0000 295 :
0000 296
00000000 0000 297 TCC_NML=0 ;NORAML TERMINATION
00000001 0000 298 TCC_ATN=1 ;ATTENTION CONDITION
00000002 0000 299 TCC_TSA=2 ;TAPE STATUS ALERT
00000003 0000 300 TCC_FNR=3 ;FUNCTION REJECT
00000004 0000 301 TCC_REM=4 ;RECOVERABLE ERROR(TAPE MOVED)
00000005 0000 302 TCC_REN=5 ;RECOVERABLE ERROR(TAPE NOT MOVED)
00000006 0000 303 TCC_UER=6 ;UNRECOVERABLE ERROR(TAPE POSI. LOST)
00000007 0000 304 TCC_FTL=7 ;FATAL CONTROLLER ERROR
0000 305
0000 306 :
0000 307 : FATAL CLASS (FC) CODES IN TSSR
0000 308 :
0000 309
00000000 0000 310 FCC_IDF=0 ;INTERNAL DIAG. FAILURE
00000001 0000 311 FCC_CPE=1 ;IO SEQUENCE CROM PARITY ERROR
00000002 0000 312 FCC_UPE=2 ;U-PROCESSOR CROM PARITY ERROR OR OTHER
00000003 0000 313 FCC_LAP=3 ;LOSS OF AC POWER DETECTED
0000 314
0000 315 :
0000 316 : TS11/TS04 MESSAGE CODES IN MS_MHD_COD
0000 317 :
0000 318
00000010 0000 319 MSG_END=^0020 ;END
00000011 0000 320 MSG_FAL=^0021 ;FAIL
00000012 0000 321 MSG_ERR=^0022 ;ERROR
00000013 0000 322 MSG_ATN=^0023 ;ATTENTION
00000014 0000 323 MSG_LOG=^0024 ;LOG (NOT USED)
0000 324
0000 325 :
0000 326 : CLASS CODE FOR MESSAGE CODES (MS_MHD_CLS VALUES)
0000 327 :
0000 328
0000 329 ;**WHEN MSG TYPE=ATTENTION**
00000000 0000 330 CLS_ONF=0 ;ON OR OFFLINE
00000001 0000 331 CLS_MDF=1 ;MICRO DIAG. FAILURE
0000 332 ;**WHEN MSG TYPE=FAIL**
00000000 0000 333 CLS_PT8=0 ;PACKET BAD(SERIAL BUS PARITY ERROR)
00000001 0000 334 CLS_OTHER=1 ;OTHERS
00000002 0000 335 CLS_WLN=2 ;WRITE LOCK ERROR OR NON-EXECUTABLE FUNCTION
00000003 0000 336 CLS_MDE=3 ;MICRO DIAGNOSTIC ERROR
0000 337
0000 338 :

```

```

0000 339 ; TS11/TS04 HARDWARE COMMAND MODES/CODES
0000 340 ;
0000 341 ;
0000 342 ;
00000000 0000 343 HC_NOP=0 ; INTERRUPT ENABLE & ACKNOWLEDGE
00000000 0000 344 HC_PAK=HC_NOP ; SIMULATED NOP (REAL NO OPERATION)
00000000 0000 345 HC_WCK=HC_NOP ; SIMULATED PACK ACKNOWLEDGE
00000000 0000 346 HC_WKR=HC_NOP ; SIMULATED WRITE CHECK
00000000 0000 347 HC_RPS=HC_NOP ; SIMULATED WRITE CHECK REVERSE
00000000 0000 348 HC_SCH=HC_NOP ; SIMULATED READ IN PRESET
00000000 0000 349 ; SIMULATED SET CHARACTERISTICS
0000 350 ;
0000C081 0000 351 HC_RDN=*00001!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; * READ NEXT (FORWARD)
0000C181 0000 352 HC_RDP=*00401!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; * READ PREVIOUS (REVERSE)
0000C281 0000 353 HC_RRP=*01001!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; * REREAD PREVIOUS (SPACE RE
0000C381 0000 354 HC_RRN=*01401!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; * REREAD NEXT (SPACE FWD, R
0000C084 0000 355 HC_WRC=*00004!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; + WRITE CHARACTERISTICS
0000C085 0000 356 HC_WRD=*00005!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; * WRITE DAT
0000C285 0000 357 HC_WDR=*01005!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; * WRITE DATA RETRY (SPACE R
0000 358 ; WRITE DATA)
0000C086 0000 359 HC_WSM=*00006!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; * WRITE SUBSYSTEM MEMORY
0000C088 0000 360 HC_SRF=*00010!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; $ SPACE RECORDS FORWARD
0000C188 0000 361 HC_SRR=*00410!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; $ SPACE RECORDS REVERSE
0000 0000 362 ; HC_STF=*01010!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; $ SKIP TAPE MARKS FORWARD
0000 0000 363 ; HC_STR=*01410!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; $ SKIP TAPE MARKS REVERSE
0000 0000 364 ; **NOTE** SKIP TAPE MARK COMMANDS ARE SIMULATED BY SKIP RECORD COMMANDS**
0000C088 0000 365 HC_STF=*00010!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; $ SPACE TAPE MARK FORWARD
0000C188 0000 366 HC_STR=*00410!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; $ SPACE TAPE MARKS REVERSE
0000C488 0000 367 HC_RWD=*02010!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; $ REWIND
0000C089 0000 368 HC_WTM=*00011!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - WRITE TAPE MARK
0000C189 0000 369 HC_ERS=*00411!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - ERASE
0000C289 0000 370 HC_WTR=*01011!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - WRITE TAPE MARK RETRY (SP
0000 371 ; ERASE, WRITE TAPE MARK)
0000C08A 0000 372 HC_BRL=*00012!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - MESSAGE BUFFER RELEASE
0000C18A 0000 373 HC_UNL=*00412!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - REWIND AND UNLOAD
0000C28A 0000 374 HC_CLN=*01012!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - CLEAN
0000C08B 0000 375 HC_DRI=*00013!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - DRIVER INITIALIZE
0000C08F 0000 376 HC_GST=*00017!MS_CPHD_M_IE!MS_CPHD_M_CVC!MS_CPHD_M_ACK ; - GET STATUS IMMEDIATE
0000 377 ;
0000 378 ; **NOTE**
0000 379 ; * => DATA XFR
0000 380 ; + => (SPECIAL)
0000 381 ; $ => POSITION
0000 382 ; - => FORMAT, CONTROL, INITIALIZE, & STATUS
0000 383 ;
0000 384 ; DEFINE DEVICE DEPENDENT UNIT CONTROL BLOCK OFFSETS
0000 385 ;
0000 386 ;
0000 387 ;
0000 388 ;
0000 389 ;
0000 390 ;
0000 391 ;
0000 392 ;
0000 393 ;
0000 394 ;
0000 395 ;

$DEFINI UCB
$VIELD UCB,0,<- ; DEV. DEP. STATUS BIT'S IN UCBSW_DEVSTS
<MS_FEF,,M>,- ; TAPE IS PAST ONE TAPE MARK
<MS_SWAP,,M>,- ; *SWAP BYTES FOR COMPATIBILITY MODE
<MS_IWR,,M>,- ; *INHIBIT WRITE RETRIES
<MS_SER,,M>,- ; SELECT ERROR HAS OCCURRED????
<MS_RWD,,M>,- ; UNIT IS REWINDING
<MS_RDPR,,M>,- ; REQUEST DATAPATH FLAG

```

```

0000 396 <MS_SWE,,M>,- ;DOING SOFTWARE EMULATION
0000 397 <MS_NER,,M>,- ;NO ERROR RECOVERY
0000 398 <MS_UMD,,M>,- ;USER MODE DIAGNOSTIC REQUEST
0000 399 <MS_RSP,,M>,- ;REWIND/SPACE IN PROGRESS
0000 400 <MS_LBA,,M>,- ;LOADING BUFFER ADDR. INTO TS04
0000 401 <MS_RPI,,M>,- ;RPOSITIONING IN PROGRESS
0000 402 <MS_VCK,,M>,- ;VOLUME CHECK
0000 403 <MS_RIP,,M>,- ;RETRY IN PROGRESS FLAG
0000 404 >
0000 405
0000 406 ;** STATUS BITS DEFINED ELSEWHERE**
0000 407 ;** IN UCBSL_DEVDEPEND:
0000 408 ;** MTSM_PARITY=1, IF EVEN;0, IF ODD
0000 409 ;** MTSV_FORMAT=MTSK_DEFAULT/NORMAL11/CORDMP11/NORMAL15
0000 410 ;** MTSV_DENSITY=MTSR_PE 1600/MTSK_NRZI-800
0000 411 ;** MTSM_BOT=TAPE IS AT BOT
0000 412 ;** MTSM_EOF=TAPE AT EOF
0000 413 ;** MTSM_EOT=TAPE AT EOT
0000 414 ;** MTSM_HWL=HARDWARE WRITE LOCKED
0000 415 ;** MTSM_LOST=TAPE POSITION LOST
0000 416 ;** IN UCBSW_STS:
0000 417 ;** UCBSM_TIM=TIMEOUT ENABLED
0000 418 ;** UCBSM_INT=INTERRUPT EXPECTED
0000 419 ;** UCBSM_ERLOGIP=ERRORLOG IN PROGRESS
0000 420 ;** UCBSM_CANCEL=CANCEL I/O
0000 421 ;** UCBSM_ONLINE=UNIT ONLINE
0000 422 ;** UCBSM_POWER=POWER FAILED WHILE UNIT BUSY
0000 423 ;** UCBSM_TIMEOUT=UNIT TIME OUT
0000 424 ;** UCBSM_INTTYPE=RECEIVER INTERRUPT,IF SET
0000 425 ;** UCBSM_BSY=UNIT IS BUSY
0000 426 ;** UCBSM_MOUNTING=DEVICE IS BEING MOUNTED
0000 427 ;** UCBSM_DEADMO=DEALLOCATE AT DISMOUNT
0000 428 ;** UCBSM_VALID=VOLUME IS SOFTWARE VALID
0000 429 ;** UCBSM_UNLOAD=UNLOAD VOLUME AT DISMOUNT
0000 430 ;** IN UCBSL_DEVCHAR:
0000 431 ;** .....
0000 432 ;** DEVSM_SWL=SOFTWARE WRITE LOCKED
0000 433 ;** .....
0000 434 :
0000 435 : NEW EXTENSION TO UCB FOR TS11/TS04
0000 436 :
0000 437 :
000000B4 0000 438 .=UCBSK_LCL TAPE_LENGTH
00B4 439 $DEF UCBSW_MS_SPACNT .BLKW 1 ;SPACING COUNT
00B6 440 $DEF UCBSL_MS_TSPT1 .BLKL 1 ;PTR. TO TS04 BUFFER IN
00BA 441 ;NON-PAGED POOL
00BA 442 $DEF UCBSL_MS_TSPT2 .BLKL 1 ;CORRESPONDING UNIBUS ADDR.
00BE 443 $DEF UCBSW_MS_TSPT3 .BLKW 1 ;COMMAND PTR FOR TS11/TS04
00C0 444 $DEF UCBSW_MS_TSBA .BLKW 1 ;TS11/TS04 DEVICE REGISTER(TSBA)
00C2 445 $DEF UCBSW_MS_TSSR .BLKW 1 ;TS11/TS04 DEVICE REGISTER(TSSR)
00C4 446 _VIELD MS_TSSR,1,<- ;TS11/TS04 STATUS REGISTER(B0 UNUSED)
00C4 447 <TCC,3>,- ;TERMINATION CLASS CODE FIELD
00C4 448 <FC,2>,- ;FATAL ERROR CLASS CODE FIELD
00C4 449 <OFL,,M>,- ;DEVICE IS OFF-LINE(B6)
00C4 450 <SSR,,M>,- ;SUBSYSTEM READY(B7)
00C4 451 <A16,,M>,- ;BUFFER ADDRESS BIT 16
00C4 452 <A17,,M>,- ;BUFFER ADDRESS BIT 17

```

```

00C4 453 <NBA,,M>,- ;NEED BUFFER ADDRESS(B10)
00C4 454 <NXM,,M>,- ;NON-EXISTENT MEMORY(B11)
00C4 455 <RMR,,M>,- ;REGISTER MODIFICATION REFUSED(B12)
00C4 456 <SPE,,M>,- ;SERIAL BUS PARITY ERROR(B13)
00C4 457 <UPE,,M>,- ;UNIBUS PARITY ERROR(B14)
00C4 458 <SC,,M>,- ;SPECIAL CONDITION(B15)
00C4 459 >
00C4 460 ;**FATAL ERROR CONDITION: UPE!SPE!NXM!NBA
00C4 461 $DEF UCBSW_MS_XC .BLKW 1 ;BYTES XFERRED OR RECORDS/FILES SKIPPED
00C6 462 $DEF UCBSB_MS_DPN .BLKB 1 ;DATA PATH NUMBER
00C7 463 $DEF UCBSB_MS_PER .BLKB 1 ;PURGE ERROR IF BIT 0 SET
00C8 464 $DEF UCBSL_MS_DPR .BLKL 1 ;DATA PATH REGISTER USED
00CC 465 $DEF UCBSL_MS_FMPR .BLKL 1 ;FINAL MAP REGISTER
00D0 466 $DEF UCBSL_MS_PMPR .BLKL 1 ;FINAL-1(PREVIOUS) MAP REGISTER
00D4 467 ;**NOTE**LAST 1 LONGWORD IS USED DURING
00D4 468 ;****POWERFAIL REPOSITIONING
00D4 469 $DEF UCBSL_MS_NMPR .BLKL 1 ;FINAL+1(NEXT) MAP REGISTER
00D8 470 $DEF UCBSL_MS_OMP .BLKL 1 ;COPY OF VEC$W_MAPREG(LONGWORD IN CRB)
00DC 471 $DEF UCBSL_MS_TIMEOUT .BLKL 1 ;Timeout value for function in progress
00E0 472 $DEF UCBSQ_MS_TMP1 .BLKQ 1 ;TEMP FOR UCBSW_BCNT,BOFF, and SVAPTE
00E8 473 $DEF UCBSL_MS_TMP2 .BLKL 1 ;TEMP. FOR CRB$C_INTD+VEC$W_MAPREG
00EC 474 $DEF UCBSQ_MS_BUF SVAPTE ;AREA TO SAVE PARAMETERS TO MAP MESSAGE
000000F4 00EC 475 .BLKQ 1 ; BUFFER IN UNIBUS SPACE
00F4 476 $DEF UCBSL_MS_TPOSITN .BLKL 1 ;TAPE POSITION AT POWERFAIL
00F8 477 $DEF UCBSW_MS_MHD .BLKW 1 ;MESSAGE PACKET**COPY IN UCB**
00FA 478 $DEF UCBSW_MS_LNH .BLKW 1 ;MESSAGE LENGTH WORD
00FC 479 $DEF UCBSW_MS_RBPC .BLKW 1 ;RESIDUAL BYTE/POSITOIN COUNT
00FE 480 $DEF UCBSW_MS_XSRO .BLKW 1 ;EXTENDED STATUS REGISTER 0
0100 481 $DEF UCBSW_MS_XSR1 .BLKW 1 ;EXTENDED STATUS REGISTER 1
0102 482 $DEF UCBSW_MS_XSR2 .BLKW 1 ;EXTENDED STATUS REGISTER 2
0104 483 $DEF UCBSW_MS_XSR3 .BLKW 1 ;EXTENDED STATUS REGISTER 3
0106 484
0106 485 .IF DF TS_TRACE
0106 486
0106 487 $DEF UCBSW_TRACESTS .BLKW 1 ; Status of trace.
0106 488 $DEF UCBSL_TRACEBEG .BLKL 1 ; Pointer to beginning of trace ring.
0106 489 $DEF UCBSL_TRACEPTR .BLKL 1 ; Pointer to next available slot.
0106 490 $DEF UCBSL_TRACEND .BLKL 1 ; Pointer to beyond trace ring.
0106 491
0106 492 TRACE_V_ACTIVE=0
0106 493 TRACE_M_ACTIVE=1
0106 494
00000106 0106 495 .ENDC
0106 496 UCBSK_MS_LENGTH=.
0106 497 $DEFEND UCB

```

```

0000 499 .SBTTL DRIVER TABLES
0000 500
0000 501 :
0000 502 : DRIVER PROLOGUE TABLE
0000 503 :
0000 504
0000 505 DPTAB - ;DEFINE DRIVER PROLOGUE TABLE
0000 506 END=TS END,- ;END OF DRIVER
0000 507 ADAPTER=UBA,- ;UNIBUS ADAPTER
0000 508 UCBSIZE=UCBS$K_MS_LENGTH,-
0000 509 NAME=TSDRIVER ;DRIVER NAME
0038 510 DPT_STORE INIT ;CONTROL BLOCK INIT VALUE
0038 511 DPT_STORE DDB,DBB$_ACPD,L,<^A\MTA\> ;DEFAULT ACP NAME
003F 512 DPT_STORE UCB,UCBS$_FIPL,B,8 ;FORK IPL
0043 513 DPT_STORE UCB,UCBS$_DEVCHAR,L,- ;DEVICE CHARACTERISTICS
0043 514 <DEVSM_FOD- ;FILES ORIENTED
0043 515 !DEVSM_DIR- ;DIRECTORY STRUCTURED
0043 516 !DEVSM_AVL- ;AVAILABLE
0043 517 !DEVSM_ELG- ;ERROR LOGGING ENABLED
0043 518 !DEVSM_IDV- ;INPUT DEVICE
0043 519 !DEVSM_ODV- ;OUTPUT DEVICE
0043 520 !DEVSM_SDI- ;SINGLE DIRECTORY DEVICE
0043 521 !DEVSM_SQD> ;SEQUENTIAL DEVICE
004A 522 DPT_STORE UCB,UCBS$_DEVCHAR2,L,- ;DEVICE CHARACTERISTICS
004A 523 <DEVSM_NNM> ;PREFIX NAME WITH 'node$'
0051 524 DPT_STORE UCB,UCBS$_DEVCLASS,B,DC$_TAPE ;DEVICE CLASS
0055 525 DPT_STORE UCB,UCBS$_DEVTYPE,B,DT$_TS11 ;DEVICE TYPE
0059 526 DPT_STORE UCB,UCBS$_MEDIA_ID,L,MEDIA_ID TS11 ;DEVICE MEDIA ID
0060 527 DPT_STORE UCB,UCBS$_DEVBUFSIZ,W,2048 ;DEFAULT BUFFER SIZE
0065 528 DPT_STORE UCB,UCBS$_DEVDEPEND,W,<^X4C0> ;DEFAULT TAPE PARAMETERS
006A 529 ;FORMAT=NORMAL11,DENSITY=1600BPI
006A 530 DPT_STORE UCB,UCBS$_DIPL,B,21 ;DEVICE IPL
006E 531 DPT_STORE UCB,UCBS$_ERTCNT,B,16 ;ERROR RETRY COUNT
0072 532 DPT_STORE UCB,UCBS$_ERTMAX,B,16 ;MAX ERROR RETRY COUNT
0076 533 DPT_STORE REINIT ;CONTROL BLOCK RE-INIT VALUES
0076 534 DPT_STORE CRB,CRB$_INTD+4,D,TSS$INT ;INTERRUPT SERVICE ROUTINE ADDR.
007B 535 DPT_STORE CRB,CRB$_INTD+VEC$_UNITINIT,D,TS INIT ;UNIT INIT
0080 536 DPT_STORE DDB,DBB$_DDT,D,MS$DDT ;DDT ADDRESS
0085 537 DPT_STORE END ;
00000001 0000 538 .MDELETE DPT_STORE
0000 539
0000 540 :
0000 541 : DRIVER DISPATCH TABLE
0000 542 :
0000 543 :
0000 544 DDTAB MS,- ;(MS=GENERIC NAME)DRIVER DISPATCH TABLE
0000 545 TS_STARTIO,- ;START I/O OPERATION
0000 546 0,- ;UNSOLICITED INTERRUPT
0000 547 TS_FUNC$TABLE,- ;FUNCTION DECISION TABLE
0000 548 +IOC$CANCELIO,- ;CANCEL I/O ENTRY POINT(STANDARD)
0000 549 TS_REGDUMP,- ;REGISTER DUMP ROUTINE
0000 550 <8*4+<1+23>*4>,- ;DIAG. BUFFER SIZE
0000 551 <<1+23>*4+EMB$_DV_REGS$AV> ;ERROR BUFFER SIZE
0038 552
0038 553 :
0038 554 : HARDWARE COMMAND TABLE - MODES/CODES
0038 555 :
  
```

0038	556			
0038	557	HCTAB:		
0038	558	GENHC	HC_NOP	:SIMULATED NOP
003A	559	GENHC	HC_UNL	:REWIND & UNLOAD
003C	560	GENHC	HC_STF	:SKIP TAPE MARK FORWARD(SPACE FILE)
003E	561	GENHC	HC_RWD	:REWIND
0040	562	GENHC	HC_DRI	:DRIVE INITIALIZE(DRIVE CLEAR)
0042	563	GENHC	HC_STR	:SKIP TAPE MARK REVERSE
0044	564	GENHC	HC_ERS	:ERASE
0046	565	GENHC	HC_SRR	:SKIP RECORD REVERSE
0048	566	GENHC	HC_PAK	:SIMULATED PACK ACKNOWLEDGE
004A	567	GENHC	HC_SRF	:SKIP RECORD FORWARD
004C	568	GENHC	HC_WCK	:SIMULATED WRITECHECK
004E	569	GENHC	HC_WRD	:WRITE DATA(WRITEPBLK)
0050	570	GENHC	HC_RDN	:READ DATA NEXT(READPBLK)
0052	571	GENHC	HC_WKR	:SIMULATED WRITECHECK REV.
0054	572	GENHC	HC_WRD	:WRITE DATA(NO WRITEPBLK REV.)
0056	573	GENHC	HC_RDP	:READ DATA PREVIOUS
0058	574	GENHC	HC_RRN	:REREAD DATA NEXT
005A	575	GENHC	HC_RRP	:REREAD DATA PREVIOUS
005C	576	GENHC	HC_WDR	:WRITE DATA RETRY
005E	577	GENHC	HC_RPS	:SIMULATED READ PRESET
0060	578	GENHC	HC_SCH	:SIMULATED SET CHARACTERISTIC
0062	579	GENHC	HC_GST	:GET STATUS IMMEDIATE
0064	580	GENHC	HC_WTM	:WRITE TAPE MARK
0066	581	GENHC	HC_WTR	:WRITE TAPE MARK RETRY
0068	582	GENHC	HC_CLN	:CLEAN
006A	583	GENHC	HC_BRL	:MESSAGE BUFFER RELEASE
006C	584	GENHC	HC_WSM	:WRITE SUBSYSTEM MEMORY
006E	585	GENHC	HC_WRC	:WRITE CHARACTERISTIC
0070	586			
0070	587			

0078	646	SPACEFILE,-	:SPACE FILES
0078	647	WRITEMARK,-	:WRITE TAPE MARK
0078	648	WRTTMKR,-	:**NEW**WRITE TAPE MARK RETRY
0078	649	CLEAN,-	:**NEW**CLEAN TAPE
0078	650	SENSEMODE,-	:SENSE MODE
0078	651	SETMODE,-	:SET MODE
0078	652	REWIND,-	:REWIND
0078	653	REWINDOFF,-	:REWIND AND UNLOAD
0078	654	SKIPRECORD,-	:SKIP RECORDS
0078	655	SKIPFILE,-	:SKIP FILES
0078	656	WRITEOF,-	:WRITE END OF FILE
0078	657	ACCESS,-	:ACCESS FILE AND/OR FIND DIRECTORY ENTRY
0078	658	ACPCONTROL,-	:ACP CONTROL FUNCTION
0078	659	CREATE,-	:CREATE FILE AND/OR CREATE DIRECTORY ENTRY
0078	660	DEACCESS,-	:DEACCESS FILE
0078	661	DELETE,-	:DELETE FILE AND/OR DIRECTORY ENTRY
0078	662	MODIFY,-	:MODIFY FILE ATTRIBUTES
0078	663	MOUNT>	:MOUNT VOLUME
0080	664	FUNCTAB +ACPSREADBLK,-	:READ FUNCTIONS
0080	665	<READLBLK,-	:READ LOGICAL BLOCK FORWARD
0080	666	READPBLK,-	:READ PHYSICAL BLOCK FORWARD
0080	667	REREADN,-	:*NEW*REREAD NEXT
0080	668	REREADP,-	:*NEW*REREAD PREVIOUS
0080	669	READVBLK>	:READ VIRTUAL BLOCK
008C	670	FUNCTAB +ACPSWRITEBLK,-	:WRITE FUNCTIONS
008C	671	<WRITECHECK,-	:WRITE CHECK FORWARD
008C	672	WRITELBLK,-	:WRITE LOGICAL BLOCK
008C	673	WRITEPBLK,-	:WRITE PHYSICAL BLOCK
008C	674	WRITERET,-	:*NEW*WRITE RETRY
008C	675	WRITEVBLK>	:WRITE VIRTUAL BLOCK
0098	676	FUNCTAB +ACPSACCESS,<ACCESS,CREATE>	:ACCESS AND CREATE FILE OR DIRECTORY
00A4	677	FUNCTAB +ACPSDEACCESS,<DEACCESS>	:DEACCESS FILE
00B0	678	FUNCTAB +ACPSMODIFY,-	:
00B0	679	<ACPCONTROL,-	:ACP CONTROL FUNCTION
00B0	680	DELETE,-	:DELETE FILE OR DIRECTORY ENTRY
00B0	681	MODIFY>	:MODIFY FILE ATTRIBUTES
00BC	682	FUNCTAB +ACPSMOUNT,<MOUNT>	:MOUNT VOLUME
00C8	683	FUNCTAB +MTSCHECK ACCESS,-	:MAGTAPE CHECK ACCESS FUNCITONS
00C8	684	<ERASETAPE,-	:ERASE TAPE
00C8	685	CLEAN,-	:**NEW**CLEAN TAPE
00C8	686	WRITEMARK,-	:WRITE TAPE MARK
00C8	687	WRTTMKR,-	:*NEW*WRITE TAPE MARK RETRY
00C8	688	WRITEOF>	:WRITE END OF FILE
00D4	689	FUNCTAB +EXESZEROPARM,-	:ZERO PARAMETER FUNCTIONS
00D4	690	<NOP,-	:NO OPERATION
00D4	691	UNLOAD,-	:UNLOAD VOLUME
00D4	692	RECAL,-	:RECALIBRATE (REWIND)
00D4	693	REWIND,-	:REWIND
00D4	694	REWINDOFF,-	:REWIND AND SET OFFLINE
00D4	695	DRVCLR,-	:DRIVE CLEAR
00D4	696	READPRESET,-	:READ IN PRESET
00D4	697	PACKACK,-	:PACK ACKNOWLEDGE
00D4	698	ERASETAPE,-	:ERASE TAPE
00D4	699	CLEAN,-	:**NEW**CLEAN TAPE
00D4	700	SENSECHAR,-	:SENSE TAPE CHARACTERISTICS
00D4	701	SENSEMODE,-	:SENSE TAPE MODE
00D4	702	AVAILABLE,-	:AVAILABLE (REWIND/NOWAIT CLEAR VALID)

00D4	703	WRITEMARK,-	:WRITE TAPE MARK
00D4	704	WRITMKR,-	:*NEW*WRITE TAPE MARK RETRY
00D4	705	WRITEOF>	:WRITE END OF FILE
00E0	706	FUNCTAB +EXESONEPARM,-	:ONE PARAMETER FUNCTIONS
00E0	707	<SPACERECORD,-	:SPACE RECORDS
00E0	708	SPACEFILE,-	:SPACE FILES
00E0	709	SKIPRECORD,-	:SKIP RECORDS
00E0	710	SKIPFILE>	:SKIP FILES
00EC	711	FUNCTAB +EXESSETMODE,-	:SET TAPE CHARACTERISTICS
00EC	712	<SETCHAR,-	:
00EC	713	SETMODE>	:
00F8	714		:

```

00F8 716      .IF      DF      TS TRACE
00F8 717      .SBTTL  +      TRACE_IRP and TRACE_STATUS
00F8 718
00F8 719      : Routines to record IRP and I/O status contents in the trace table.
00F8 720      : Trace table entries are 96 bytes long so that they line up nicely in
00F8 721      : a dump.
00F8 722
00F8 723      : TRACE_IRP
00F8 724
00F8 725      : Inputs:
00F8 726      :      R3 => IRP
00F8 727      :      R5 => UCB
00F8 728
00F8 729      TRACE_IRP:
00F8 730
00F8 731      BBC      #TRACE_V_ACTIVE,-      : If trace table not intialized,
00F8 732      UCBSW_TRACESTS(R5),20$      : branch around.
00F8 733      MOVQ     R0,-(SP)      : Save R0 and R1.
00F8 734      MOVL    R3,R0      : R0 => IRP to trace.
00F8 735      CMPL   UCBSL_TRACEND(R5),-      : See if we should circle back to start
00F8 736      UCBSL_TRACEPTR(R5)      : of trace table.
00F8 737      BGTR    10$      : GTR implies NO.
00F8 738      MOVL   UCBSL_TRACEBEG(R5),-      : TRACE_PTR => base of trace table.
00F8 739      UCBSL_TRACEPTR(R5)
00F8 740      10$:
00F8 741      MOVL   UCBSL_TRACEPTR(R5),R1      : R1 => area in trace table to use.
00F8 742
00F8 743      MOVQ   (R0)+,(R1)+      : Twelve quad words are 96 bytes.
00F8 744      MOVQ   (R0)+,(R1)+
00F8 745      MOVQ   (R0)+,(R1)+
00F8 746      MOVQ   (R0)+,(R1)+
00F8 747      MOVQ   (R0)+,(R1)+
00F8 748      MOVQ   (R0)+,(R1)+
00F8 749      MOVQ   (R0)+,(R1)+
00F8 750      MOVQ   (R0)+,(R1)+
00F8 751      MOVQ   (R0)+,(R1)+
00F8 752      MOVQ   (R0)+,(R1)+
00F8 753      MOVQ   (R0)+,(R1)+
00F8 754      MOVQ   (R0)+,(R1)+
00F8 755
00F8 756      MOVL   UCBSL_TRACEPTR(R5),R1      : R1 => area in trace table to use.
00F8 757      MOVL   R3,(R1)      : Trace entry => IRP.
00F8 758      MNEGL  #1,4(R1)      : Init flag field.
00F8 759      MNEGL  #1,IRPSL_ARB(R1)      : Init field for I/O Status #1.
00F8 760      MNEGL  #1,IRPSL_ARB+4(R1)      : Init field for I/O Status #2.
00F8 761      MOVQ   (SP)+,R0      : Restore R0 and R1.
00F8 762      20$:
00F8 763      RSB
00F8 764
00F8 765      : TRACE_STATUS
00F8 766
00F8 767      : Inputs:
00F8 768
00F8 769      :      R0 = I/O status value #1.
00F8 770      :      R5 => UCB
00F8 771      :      UCBSL_DEVDEPEND = I/O status #2.
00F8 772

```

```
00F8 773 TRACE_STATUS:
00F8 774
00F8 775      BBC      #TRACE_V_ACTIVE,-      ; If Trace table not active, branch.
00F8 776      UCBSW_TRACESTS(R5),30$
00F8 777      PUSHL   R2      ; Save register.
00F8 778      MOVL    UCBSL_TRACEPTR(R5),R2 ; R2 => area in trace table to use.
00F8 779
00F8 780      MOVL    R0,IRPSL_ARB(R2)      ; Save I/O status.
00F8 781      MOVL    UCBSL_DEVDEPEND(R5),- ;
00F8 782      IRPSL_  IRPSL_+4(R2)
00F8 783      POPL    R2      ; Restore register.
00F8 784      ADDL    #96,UCBSL_TRACEPTR(R5) ; Point to next entry.
00F8 785 30$:
00F8 786      RSB
00F8 787
00F8 788      .ENDC
```

```

00F8 790 .SBTTL UNIT INITIALIZATION ROUTINE
00F8 791 :+
00F8 792 : THIS ROUTINE IS CALLED WHEN THE DRIVER IS LOADED OR ON POWERFAIL
00F8 793 : RECOVERY.
00F8 794 :
00F8 795 : CALLING SEQUENCE:
00F8 796 :     JSB     TS_INIT
00F8 797 :
00F8 798 : INPUT:
00F8 799 :     R5 = UCB ADDRESS
00F8 800 :     R4 = EQUIVALENT CSR FOR TS11
00F8 801 :
00F8 802 : OUTPUT:
00F8 803 :
00F8 804 :
00F8 805 TS_INIT:
10  A8 00F8 806 BISW #UCB$M_ONLINE,- ; Always mark TS11 as online since
64 A5 00FA 807 UCBSW_STS(R5) ; interrupts are not normally enabled
00FC 808 ; we have no method to set it on
00FC 809 ; dynamically.
05 11 00FC 810 BRB 50$ ; Go to allocate buffer and load registers
00FE 811 15$:
10 AA 00FE 812 BICW #UCB$M_ONLINE,- ; Only reason for marking TS11 offline
64 A5 0100 813 UCBSW_STS(R5) ; is lack of pool space for PACKET.
0102 814 20$:
05 0102 815 RSB ;RETURN
0103 816 50$:
0103 817 .IF DF TS_TRACE
0103 818 BBS #TRACE_V_ACTIVE,- ; If trace table already intialized,
0103 819 UCBSW_TRACESTS(R5),52$ ; branch around.
0103 820 MOVL #50*96+16,R1 ; Allocate trace table for 50 entries.
0103 821 PUSHL G^EXESGL_NONPAGED ; Save nonpaged IPL.
0103 822 MFPR #PR$ IPL,G^EXESGL_NONPAGED ; Use current IPL.
0103 823 JSB G^EXESALONONPAGED ; Get from non-paged memory.
0103 824 POPL G^EXESGL_NONPAGED ; Restore nonpaged IPL.
0103 825 BLBC R0,52$ ; Space not available, branch around.
0103 826
0103 827 CLRQ (R2)+ ; Initialize trace table header for SDA.
0103 828 MOVW R1,(R2)+ ; Save size.
0103 829 MOVW #DYN$C_SCS,(R2)+ ; Type.
0103 830 CLRL (R2)+ ; Round header upto 16 byte boundary.
0103 831 MOVL R2,UCB$S_TRACEBEG(R5) ; Save pointer to base of trace tabl
0103 832 MOVL R2,UCB$S_TRACEPTR(R5) ; Pointer to next area to use.
0103 833 ADDL3 #50*96,R2,- ; Pointer to beyond end of trace
0103 834 UCBSL_TRACEEND(R5) ; table.
0103 835 BISW #TRACE_M_ACTIVE,- ; Indicate Trace table initied.
0103 836 UCBSW_TRACESTS(R5)
0103 837 52$:
0103 838 .ENDC
51 24 A5 D0 0103 839 MOVL UCBSL_CRB(R5),R1 ;GET POINTER TO CRB
34 A1 D0 0107 840 MOVL CRBSL_INTD+VECSW_MAPREG(R1),-
00E8 C5 010A 841 UCBSL_MS_TMP2(R5) ;SAVE CURRENT UBA MAP CONTEXT.
50 2C A1 D0 010D 842 MOVL CRBSL_INTD+VECSL_IDB(R1),R0 ;GET POINTER TO IDB
04 A0 55 D0 0111 843 MOVL R5,IDBSL_OWNER(R0) ;MAKE UCB OWNER OF IDB
51 20 D0 0115 844 MOVL #32,R1 ;SIZE OF WORK BUFFER FOR TS11(=32.)
52 00B6 C5 D0 0118 845 MOVL UCBSL_MS_TSPT1(R5),R2 ;IF THE BUFFER HAS ALREADY BEEN ALLOCATED
25 12 011D 846 BNEQ 60$ ;BRANCH AROUND ELSE ALLOCATE THE BUFFER

```

```

011F 847
011F 848 :
011F 849 : DRIVER LOAD
011F 850 :
011F 851 :
011F 852 55$:
00000000'GF DD 011F 853 PUSHL G^EXE$GL_NONPAGED ;SAVE NONPAGED IPL
      12 DB 0125 854 MFPR #PRS_IPL,G^EXE$GL_NONPAGED ;USE CURRENT IPL
00000000'GF 0127
00000000'GF 16 012C 855 JSB G^EXE$ALONONPAGED ;GET FROM NON-PAGED MEMORY
00000000'GF 8ED0 0132 856 POPL G^EXE$GL_NONPAGED ;RESTORE NONPAGED IPL
      03 50 E8 0139 857 BLBS R0,57$ ; Space available, branch aroundd.
      FFBF 31 013C 858 BRW 15$ ; Branch on allocation failure.
013F 859 57$:
013F 860 ;YES, R1=SIZE OF ALLOCATED BLOCK
013F 861 ; R2 HAS ADDR. OF THE BLOCK
00B6 C5 52 D0 013F 862 MOVL R2,UCB$MS_TSPT1(R5) ;STORE ADDR. IN UCB
0144 863 60$:
0144 864 ASSUME UCB$W_BOFF EQ UCB$S_SVAPTE+4
0144 865 ASSUME UCB$W_BCNT EQ UCB$W_BOFF+2
      78 A5 7D 0144 866 MOVQ UCB$S_SVAPTE(R5),-
00E0 C5 0147 867 UCB$Q_MS_TMP1(R5) ;SAVE UCB$S_SVAPTE, W_BCNT, W_BOFF.
014A 868
      7E A5 51 B0 014A 869 MOVW R1,UCB$W_BCNT(R5) ;LOAD BYTE COUNT INTO UCB
52 FE00 8F AB 014E 870 BICW3 #^XFE00,R2,UCB$W_BOFF(R5) ;LOAD BYTE OFFSET IN UCB
      7C A5 0153
52 15 09 EF 0155 871 EXTZV S^#VAS$VPN,S^#VASS_VPN,R2,R2 ;GET VIRTUAL PAGE #
      52 0159
00000000'GF D0 015A 872 MOVL G^MMG$GL_SPTBASE,R0 ;GET ADDR. OF SYS. PAGE TABLE
      50 0160
78 A5 6042 DE 0161 873 MOVAL (R0)[R2],UCB$S_SVAPTE(R5) ;STORE SVA OF PTE FOR WORK BUFFER
0166 874 ;LOADED BCNT,BOFF,&SVAPTE FOR WORK BUFFER
0166 875 ;DIRECT DATA PATH IS USED FOR COMMUNICATION
51 24 A5 D0 0166 876 MOVL UCB$S_CRB(R5),R1 ; R1 => CRB
00D8 C5 D0 016A 877 MOVL UCB$S_MS_OMPR(R5),CRB$S_INTD+VEC$W_MAPREG(R1) ;IF NEQ USE OLD MAP RE
      34 A1 016E
      13 12 0170 878 BNEQ 80$ ;GOTO LOAD MAP REGISTER
0172 879 70$:
      37 A1 94 0172 880 CLRB CRB$S_INTD+VEC$B_DATAPATH(R1) ;INSURE DIRECT DATA PATH(=0)
0175 881 REQMPR ;ALLOCATE MAP REGISTER(S) TO MAP UNIBUS
51 24 A5 D0 017B 882 MOVL UCB$S_CRB(R5),R1 ;GET POINTER TO CRB
      34 A1 D0 017F 883 MOVL CRB$S_INTD+VEC$W_MAPREG(R1),UCB$S_MS_OMPR(R5) ;SAVE OLD MAP REGISTER
00D8 C5 0182
      78 A5 7D 0185 884 80$:
00EC C5 0185 885 MOVQ UCB$S_SVAPTE(R5),- ; Save message buffer parameters to
0188 886 UCB$Q_MS_BUF$SVAPTE(R5) ; facilitate later remapping.
0188 887 ;TO SBI ADDRESSES
0188 888 ; THE NO. OF MAP REGISTER AND STARTING
0188 889 ; MAP REG. NO. ARE STORED IN CRB
LOADUBA ;LOAD MAP REG. TO BE USED
0191 891 :
0191 892 : CALCULATE UNIBUS ADDR. FOR COMMAND PACKET, STORE IT IN UCB
0191 893 :
50 7C A5 3C 0191 894 MOVZWL UCB$W_BOFF(R5),R0 ;GET BYTE OFFSET
00E0 C5 7D C195 895 MOVQ UCB$Q_MS_TMP1(R5),-
      78 A5 0199 896 UCB$S_SVAPTE(R5) ;RESTORE SVAPTE, BOFF, BCNT
019B 897

```

```

51 24 A5 D0 019B 898      MOVL  UCBSL_CRB(R5),R1      ;GET CRB
09 34 A1 F0 019F 899      INSV  CRBSL_INTD+VECSW_MAPREG(R1),#9,#9,R0 ; HGH 9 BITS
00BA 50 09 01A3
C5 50 D0 01A5 900      MOVL  R0,UCBSL_MS_TSPT2(R5) ;STORE IN UCB
00E8 C5 D0 01AA 901      MOVL  UCBSL_MS_TMP2(R5),-
34 A1 01AE 902      CRBSL_INTD+VECSW_MAPREG(R1) ;RESTORE UNIBUS MAPPING CONTEXT
51 D4 01B0 903      CLRL  R1                    ;CLEAR R1
50 FE 8F 78 01B2 904      ASHL  #-2,R0,R0            ;MODULO 4,SHIFT OUT 2 0'S
0E 02 50 F0 01B7 905      INSV  R0,#2,#14,R1        ;INSERT B2-B15
51 01BB
50 F2 8F 78 01BC 906      ASHL  #-14,R0,R0         ;SHIFT OUT B2-B15
02 00 50 F0 01C1 907      INSV  R0,#0,#2,R1        ;INSERT B16-B17
51 01C5
00BE C5 51 B0 01C6 908      MOVW  R1,UCBSW_MS_TSPT3(R5) ;STORE COMMAND PTR IN UCB
01CB 909
01CB 910
01CB 911
01CB 912
50 00B6 C5 D0 01CB 913      MOVL  UCBSL_MS_TSPT1(R5),R0 ;COMMAND PACKET ADDR. IN R0
60 C084 8F B0 01D0 914      MOVW  #<HC_0RC!MS_CPHD_M_ACK>,MS_CPHD(R0) ;GET COMMAND PACKET HEADER
00BA C5 D0 01D5 915      MOVL  UCBSL_MS_TSPT2(R5),MS_BACT(R0) ;STORE CHAR. BUFFER ADDR.
02 A0 08 C0 01DB 916      ADDL  #8,MS_BACT(R0)      ;POINT TO CHAR. BUFFER NOW
06 A0 08 B0 01DF 917      MOVW  #8,MS_CNT(R0)      ;STORE BYTE COUNT FOR CHAR. DATA
00BA C5 D0 01E3 918      MOVL  UCBSL_MS_TSPT2(R5),MS_MBA0(R0) ;STORE MESSAGE BUFFER ADDR.
08 A0 10 C0 01E9 919      ADDL  #16,MS_MBA0(R0)    ; AS CHAR. DATA
0C A0 0E B0 01ED 920      MOVW  #14,MS_LNTH(R0)   ;LENGTH OF CHAR. DATA=14.
0E A0 B4 01F1 921      CLRW  MS_CHWD(R0)       ;ZERO CHARACTERISTIC WORD
01F4 922      ;***=>NO MESSAGE BUFFER RELEASE INTERRUPT
01F4 923      ;** NO ATTENTION INTERRUPT, AND NO
01F4 924      ;** SKIP TAPE MARKS STOP
01F4 925
01F4 926
01F4 927
64 00BE C5 R0 01F4 928      MOVW  UCBSW_MS_TSPT3(R5),(R4) ;LOAD INTO TSDB
0400 8F A8 01F9 929      BISW  #UCBSW_MS_LBA,UCBSW_DEVSTS(R5) ;MARK LOADING MESSAGE BUFFER
68 A5 01FD
01FF 930      ; ADDR. INTO TS11.
05 01FF 931      RSB                    ;RETURN
0200 932
0200 933

```

```

0200 935 .SBTTL TEST NBA (NEED BUFFER ADDRESS)
0200 936
0200 937 : TEST_NBA - Subroutine called from STARTIO to determine if the TS11 has
0200 938 : a valid message buffer. If YES, then we merely return. If NOT, we
0200 939 : re-establish the message buffer obtained at SYSTEM INIT time.
0200 940
0200 941 : This routine assumes that the following UCB fields were initialized
0200 942 : at UNIT INIT time:
0200 943
0200 944 : UCBSQ_MS_BUFVAPTE
0200 945 : UCBSL_MS_OMPR
0200 946
0200 947 : INPUTS:
0200 948 : R5 => UCB
0200 949
0200 950 : OUTPUTS:
0200 951 : Message buffer established in TS11.
0200 952
0200 953
0200 954 TEST_NBA:
009C C5 8ED0 0200 955 : POPL UCBSL_DPC(R5) ; Pop return off stack in case.
0400 8F AA 0205 956 : BICW #UCBSM_MS_LBA,- ; This bit maybe left on from INIT if
68 A5 0209 957 : UCBSW_DEVSTS(R5) ; setting of switches inside drive so
020B 958 : ; dictate. We clear it here because
020B 959 : ; this is a convenient place.
020B 960
51 24 A5 DO 020B 961 : MOVL UCBSL_CRB(R5),R1 ; R1 => CRB
020F 962 : ASSUME IDB$CSR EQ 0
54 2C B1 DO 020F 963 : MOVL @CRB$INTD+VECSL_IDB(R1),R4 ; R4 => CSR.
50 02 A4 B0 0213 964 : MOVW 2(R4),R0 ; R0 contains TSSR register.
03 50 07 E0 0217 965 : BBS #MS_TSSR_V_SSR,R0,10$ ; Branch to continue if TS11 ready.
0095 31 021B 966 : BRW 60$ ; Branch to failure if NOT ready.
021E 967 10$:
07 50 0A E0 021E 968 : BBS #MS_TSSR_V_NBA,R0,20$ ; Branch around if we NEED to re-
0222 969 : ; establish message buffer address.
50 01 9A 0222 970 : MOVZBL S^#SS$NORMAL,R0 ; Else indicate success and
009C D5 17 0225 971 : JMP @UCBSL_DPC(R5) ; return to caller.
0229 972 20$:
34 A1 DO 0229 973 : MOVL CRB$INTD+VECSW_MAPREG(R1),-
00E8 C5 022C 974 : UCBSL_MS_TMP2(R5) ;SAVE CURRENT UBA MAP CONTEXT.
00D8 C5 DO 022F 975 : MOVL UCBSL_MS_OMPR(R5),- ; Setup to map UNIBUS just in case
34 A1 0233 976 : CRB$INTD+VECSW_MAPREG(R1)
0235 977
0235 978 : ASSUME UCBSW_BOFF EQ UCBSL_SVAPTE+4
0235 979 : ASSUME UCBSW_BCNT EQ UCBSW_BOFF+2
78 A5 7D 0235 980 : MOVQ UCBSL_SVAPTE(R5),-
00E0 C5 0238 981 : UCBSQ_MS_TMP1(R5) ;SAVE UCBSL_SVAPTE, W BCNT, W_BOFF.
00EC C5 7D 0238 982 : MOVQ UCBSQ_MS_BUFVAPTE(R5),- ; Restore parameters to remap message
78 A5 023F 983 : UCBSL_SVAPTE(R5) ; buffer in UNIBUS space.
0241 984
0241 985 : LOADUBA ; Reload UNIBUS map registers for
0247 986 : ; message buffer.
0247 987
0247 988 :
0247 989 : : ISSUE WRITE CHARACTERISTIC COMMAND TO TELL MESSAGE BUFFER ADDR. TO TS11
0247 990 :
50 00B6 C5 DO 0247 991 : MOVL UCBSL_MS_TSPT1(R5),R0 ; R0 => command packet

```

-1

```

0084 8F B0 024C 992      MOVW  #<HC_WRC!MS_CPHD_M_ACK>,-
      60      0250 993      MS_CPHD(R0)
      00BA C5 D0 0251 994      : Move command (WRITE CHARACTERISTICS)
      02 A0 08 C0 0255 996      : to 1ST word of command packet.
      06 A0 08 B0 0257 997      : Store UNIBUS address of packet in
      00BA C5 D0 0258 998      : packet.
      08 A0 10 C0 025B 999      : Update to point to CHARACTERISTICS
      0C A0 0E B0 025B 999      : buffer beyond packet.
      0E A0 B4 025F 1000      : Store byte count for char. data
      08 A0 10 C0 0263 1001      : Store UNIBUS address of PACKET
      0C A0 0E B0 0265 1002      : into the CHARACTERISTICS data.
      0E A0 B4 0269 1003      : Message BUFF is 16 beyond packet.
      0E A0 B4 0269 1004      : LENGTH OF CHAR. DATA=14.
      0E A0 B4 026D 1005      : ZERO CHARACTERISTIC WORD
      0270 1006      : **=>NO MESSAGE BUFFER RELEASE INTERRUPT
      0270 1007      : ** ,NO ATTENTION INTERRUPT, AND NO
      0270 1008      : ** SKIP TAPE MARKS STOP
      0270 1009      DSBINT
64 00BE C5 B0 0276 1011      MOVW  UCBSW_MS_TSPT3(R5),(R4) ;LOAD INTO TSDB
      027B 1012      WFIKPC 40$,#2 ; Wait for interrupt.
      0285 1013      IOFORK
      09 11 028B 1014      BRB 50$ ; Branch around powerfail branch.
      028D 1015 30$:      ENBINT
      0290 1016 40$:      SETIPL UCBSB_FIPL(R5) ; Lower IPL in case of TIMEOUT.
      0294 1017      BRB 60$ ; Branch if we had POWERFAIL.
      0296 1018
      0296 1019 50$:
51 24 A5 D0 0296 1020      MOVL  UCBSL_CRB(R5),R1 ; R1 => CRB.
      00E8 C5 D0 029A 1021      MOVL  UCBSL_MS_TMP2(R5),- ; Restore previous mapping context.
      34 A1 029E 1022      CRBSL_INTD+VEC$W_MAPREG(R1)
      00E0 C5 7D 02A0 1023      MOVQ  UCBSQ_MS_TMP1(R5),- ; And also transfer parameters.
      78 A5 02A4 1024      UCBSL_SVAPTE(R5)
      07 00C2 C5 E0 02A6 1025      BBS #MS_TSSR_V_NBA,- ; Test if all that had any effect
      50 01 9A 02A8 1026      UCBSW_MS_TSSR(R5),60$ ; by seeing if we still have NBA.
      009C D5 17 02AC 1027      MOVZBL S^#SS$_NORMAL,R0 ; Else indicate success and
      02AF 1028      JMP @UCBSL_DPC(R5) ; return to caller.
      02B3 1029 60$:
50 0084 8F 3C 02B3 1030      : Terminate the I/O function
      009C D5 17 02B3 1031      MOVZWL #SS$_DEVOFFLINE,R0 ; by returning the OFFLINE status and
      02B8 1032      JMP @UCBSL_DPC(R5) ; return to caller.

```



```

02BC 1034 .SBTTL START I/O OPERATION
02BC 1035 :+
02BC 1036 : TS_STARTIO - START I/O OPERATION ON DEVICE
02BC 1037 :
02BC 1038 : THIS ENTRY POINT IS ENTERED TO START AN I/O OPERATION ON TS11/TS04
02BC 1039 :
02BC 1040 : INPUTS:
02BC 1041 :
02BC 1042 : R3 = ADDRESS OF I/O PACKET.
02BC 1043 : R5 = UCB ADDRESS OF DEVICE UNIT
02BC 1044 :
02BC 1045 : OUTPUT:
02BC 1046 :
02BC 1047 : FUNCTION DEPENDENT PARAMETERS ARE STORED INTO THE DEVICE UCB,
02BC 1048 : ERROR RETRY COUNT IS RESET, AND THE FUNCTION IS EXECUTED. AT
02BC 1049 : FUNCTION COMPLETION THE OPERATION IS TERMINATED THRU REQUEST COMPLETE.
02BC 1050 :
02BC 1051 :-
02BC 1052 :
02BC 1053 TS_STARTIO: ;START I/O OPERATION
02BC 1054 .IF DF TS_TRACE
02BC 1055 BSBW TRACE_IRP ; Trace this IRP.
02BC 1056 .ENDC
02BC 1057 BSBW TEST_NBA ; Assure that TS11 has valid MESSAGE
02BF 1058 ; BUFFER.
02BF 1059 BLBS RO,5$ ; LBS implies success. GOTO continue.
04E5 31 02C2 1060 BRW FCNEXT ; Else branch to terminate function.
02C5 1061 5$:
0081 C5 90 02C5 1062 MOVB UCBSB_ERTMAX(R5),UCBSB_ERTCNT(R5) ;INITIALIZE ERROR RETRY COUNT
0080 C5 02C9
20 A3 B0 02CC 1063 MOVW IRPSW_FUNC(R3),UCBSW_FUNC(R5) ;SAVE FUNCTION CODE & MODIFIER
009A C5 02CF
50 38 A3 D0 02D2 1064 MOVL IRPSL_MEDIA(R3),RO ;GET PARAMETER LONGWORD
02D6 1065
02D6 1066 :
02D6 1067 : MOVE FUNCTION DEPENDENT PARAMETERS TO UCB
02D6 1068 :
02D6 1069 :
06 00 EF 02D6 1070 EXTZV #IRPSV_FCODE,#IRPSS_FCODE,- ;EXTRACT I/O FUNCTION CODEE
51 20 A3 02D9 1071 IRPSW_FUNC(R3),R1
51 02 D1 02DC 1072 CMPL #IOS_SPACEFILE,R1 ;SPACE FILE FUNCTION?
51 2B 13 02DF 1073 BEQL 10$ ;IF EQL YES
51 09 D1 02E1 1074 CMPL #IOS_SPACERECORD,R1 ;SPACE RECORD FUNCTION?
51 32 13 02E4 1075 BEQL 20$ ;IF EQL YES
51 1A D1 02E6 1076 CMPL #IOS_SETCHAR,R1 ;SET CHARACTERISTICS FUNCTION?
51 46 13 02E9 1077 BEQL 50$ ;IF EQL YES
51 11 D1 02EB 1078 CMPL #IOS_AVAILABLE,R1 ;AVAILABLE function?
51 5A 13 02EE 1079 BEQL 75$ ;IF EQL YES
51 0D D1 02F0 1080 CMPL #IOS_READPBLK+1,R1 ;DISJOINT CODE?
6C 1A 02F3 1081 BGTRU 100$ ;IF GTRU NO
02F5 1082 CASE R1,- ;DISPATCH LOGICAL FUNCTIONS
02F5 1083 70$,- ;REWIND AND SET OFFLINE
02F5 1084 60$,- ;SET MODE
02F5 1085 80$,- ;REWIND
02F5 1086 10$,- ;SKIP FILE
02F5 1087 20$,- ;SKIP RECORD
02F5 1088 90$,- ;SENSE TAPE MODE

```

```

02F5 1089          90$ -          ;WRITE EOF
02F5 1090          > LIMIT=#IOS REWINDOFF ;
51  06  A2 0307 1091 SUBW          #IOS_READPRESET-IOS_READ;BLK-7,R1 ;CONVERT TO DENSE FUNCTION CODE
63  11  11 030A 1092 BRB          110$          ;**LAST LINE NEED BE ADJUSTED
030C 1093
030C 1094
030C 1095 : SPACE FILE FUNCTION - SET SPACE COUNT AND PROPER FUNCTION
030C 1096 :
030C 1097
51  02  3C 030C 1098 10$: MOVZWL #CDHC_STF,R1 ;SET SPACE FILE FORWARD
50  B5  030F 1099 TSTW RO ;SPACE FILE FORWARD?
12  14  0311 1100 BGTR 40$ ;IF GTR YES
51  05  9A 0313 1101 MOVZBL #CDHC_STR,R1 ;SET FOR SPACE FILE REVERSE
0A  11  0316 1102 BRB 30$ ;
0318 1103
0318 1104 : SPACE RECORD FUNCTION - SET SPACE COUNT AND PROPER HARDWARE COMMAND
0318 1105 :
0318 1106 :
0318 1107
51  09  9A 0318 1108 20$: MOVZBL #CDHC_SRF,R1 ;SET FOR SPACE RECORD FORWARD
50  B5  031B 1109 TSTW RO ;SPACE RECORD FORWARD?
06  14  031D 1110 BGTR 40$ ;IF GTR YES
51  07  9A 031F 1111 MOVZBL #CDHC_SRR,R1 ;SET FOR SPACE RECORD REVERSE
00B4 50  AE 0322 1112 30$: MNEGW RO,RO ;CONVERT TO POSITIVE COUNT
C5  50  B0 0325 1113 40$: MOVW RO,UCBSW_MS_SPACNT(R5) ;SET SPACE COUNT
43  12  032A 1114 BNEQ 110$ ;IF NEQ SPACING REQUIRED
51  00  9A 032C 1115 MOVZBL #CDHC_NOP,R1 ;SET FOR NO OPERATION
3E  11  032F 1116 BRB 110$ ;
0331 1117

```

```

0331 1119 :
0331 1120 : SET CHARACTERISTICS FUNCTION - STORE NEW TAPE CHARACTERISTICS
0331 1121 :
0331 1122 : ****TS11/TS04 HAS ONLY ONE CLASS AND TYPE****
0331 1123 :
40 A5 38 A3 B0 0331 1124 50$: MOVW IRP$L_MEDIA(R3),UCBSB_DEVCLASS(R5) ;SET NEW DEVICE CLASS AND TYPE
0336 1125 :
0336 1126 :
0336 1127 : SET MODE FUNCTION - STORE NEW TAPE MODE
0336 1128 :
0336 1129 :
42 A5 3A A3 B0 0336 1130 60$: MOVW IRP$L_MEDIA+2(R3),UCBSW_DEVBUFSIZ(R5) ;SET NEW DEFAULT BUFFER SIZE
7C A5 3C A3 B0 0338 1131 MOVW IRP$L_MEDIA+4(R3),UCBSW_BOFF(R5) ;SAVE NEW TAPE CONTROL PARAMETERS
51 14 9A 0340 1132 MOVZBL #CDHC_SCH,R1 ;SET DISPATCH INDEX
2A 11 0343 1133 BRB 110$ ;
0345 1134 :
0345 1135 :
0345 1136 : LOGICAL REWIND AND SET TAPE OFFLINE - CONVERT TO UNLOAD COMMAND
0345 1137 :
51 01 9A 0345 1138 70$: MOVZBL #CDHC_UNL,R1 ;SET FOR UNLOAD COMMAND
25 11 0348 1140 BRB 110$ ;
034A 1141 :
034A 1142 :
034A 1143 : AVAILABLE FUNCTION - Equivalent of REWIND(NOWAIT) and clear of UCBSM_VALID.
034A 1144 :
034A 1145 :
034A 1146 75$:
00A4 8F B0 034A 1147 MOVW #IOS_REWIND!IOSM_NOWAIT,-; Simulate a REWIND NOWAIT.
009A C5 034E 1148 UCBSW_FUNC(R5)
0800 8F AA 0351 1149 BICW #UCBSM_VALID,- ; And clear valid bit.
64 A5 0355 1150 UCBSW_STS(R5) ; and fall thru to rewind logic.
0357 1151 :
0357 1152 :
0357 1153 : LOGICAL REWIND FUNCTION - CONVERT TO PHYSICAL FUNCTION
0357 1154 :
0357 1155 :
51 03 9A 0357 1156 80$: MOVZBL #CDHC_RWD,R1 ;SET FOR REWIND
13 11 035A 1157 BRB 110$ ;
035C 1158 :
035C 1159 :
035C 1160 : LOGICAL WRITE EOF OR SENSE MODE FUNCTION - CONVERT TO PHYSICAL FUNCTION
035C 1161 :
035C 1162 :
51 12 A2 035C 1163 90$: SUBW #IOS_SENSEMODE-IOS_READPBLK-9,R1 ;CONVERT TO PHYSICAL****
0E 11 035F 1164 BRB 110$ ;
0361 1165 :
0361 1166 :
0361 1167 : DENSE FUNCTION CODE - CHECK FOR READ, WRITE, OR WRITECHECK FUNCTION
0361 1168 :
0361 1169 :
51 0A D1 0361 1170 100$: CMPL #IOS_WRITECHECK,R1 ;DATA TRANSFER FUNCTION?
09 1A 0364 1171 BGTRU 110$ ;IF GTRU NO
009A C5 06 E1 0366 1172 BBC #IOSV_REVERSE,UCBSW_FUNC(R5),110$ ;IF CLEAR,NOT REVERSE
03 03 0368 :
51 03 A0 036C 1173 ADDW #CDHC_WKR-CDHC_WCK,R1 ;CONVERT TO REVERSE FUNCTION
036F 1174 :

```

```

0092 C5 51 90 036F 1175 :
036F 1176 : FINISH PREPROCESSING
036F 1177 :
036F 1178 :
036F 1179 110$: MOVB R1,UCBSB_FEX(R5) :SAVE FUNCTION DISPATCH INDEX
0374 1180 :*****NOTE ABOUT BYTE FOR INDEX
0374 1181 :
0374 1182 :
0374 1183 :CENTRAL FUNCTION DISPATCH
0374 1184 :
0374 1185 :
0374 1186 FDISPATCH:
53 58 A5 DO 0374 1187 MOVL UCBSL_IRP(R5),R3 :RETRIEVE ADDR. OF I/O PACKET
OD 2A A3 08 EO 0378 1188 BBS #IRPSV_PHYSIO,IRPSW_STS(R3),10$ :IF SET, PHYSICAL I/O FUNCTION
08 64 A5 08 EO 037C 1189 BBS #UCBSV_VALID,UCBSW_STS(R5),10$ :IF SET, VOLUME SOFTWARE VALID
50 0254 8F 3C 0382 1190 MOVZWL #SS$ VOLINV,R0 :SET VOLUME INVALID STATUS
0420 31 0387 1191 BRW FCNEXT :**NO CHANGE ON UCBSV_VALID BIT HERE**
038A 1192 :
038A 1193 :
038A 1194 : UNIT IS SOFTWARE VALID OR FUNCTION IS PHYSICAL I/O
038A 1195 :
038A 1196 :
038A 1197 10$:
54 24 A5 DO 038A 1198 MOVL UCBSL_CRB(R5),R4 :GET CSR ADDRESS INTO R4...
54 2C B4 DO 038E 1199 MOVL @CRBSL_INTD+VECSL_IDB(R4),R4 :
50 0092 C5 9A 0392 1200 MOVZBL UCBSB_FEX(R5),R0 :GET DISPATCH INDEX
CASE RO,- :DISPATCH TO COMMAND HANDLING ROUTINE
0397 1201 NOP,- :NO OPERATION
0397 1202 UNLOAD,- :REWIND & UNLOAD
0397 1203 SPCFILFOR,- :SPACE FILE FORWARD
0397 1204 REWIND,- :REWIND
0397 1205 DRVCLR,- :DRIVE CLEAR
0397 1206 SPCFILREV,- :SPACE FILE REVERSE
0397 1207 ERASE,- :ERASE
0397 1208 SPCRECREV,- :SPACE RECORD REVERSE
0397 1209 PACKACK,- :PACK ACKNOWLEDGE
0397 1210 SPCRECFOR,- :SPACE RECORD FORWARD
0397 1211 WRITECHECK,- :SIMULATED WRITECHECK
0397 1212 WRITEDATA,- :WRITE DATA FORWARD
0397 1213 READDATA,- :READ DATA FORWARD
0397 1214 WRITECHECKR,- :WRITE CHECK REVERSE
0397 1215 WRITEDATA,- :WRITE DATA(NO REVERSE)
0397 1216 READDATAR,- :READ DATA REVERSE
0397 1217 REREADN,- :REREAD DATA NEXT
0397 1218 REREADP,- :REREAD DATA PREVIOUS
0397 1220 WRITERET,- :WRITE DATA RETRY
0397 1221 READPRESET,- :SIMULATED READ PRESET
0397 1222 SETCHAR,- :SIMULATED SET CHARACTERISTIC
0397 1223 GETSTS,- :GET STATUS IMMEDIATE(SENS CHAR.)
0397 1224 WRTTMK,- :WRITE TAPE MARK
0397 1225 WRTTMKR,- :WRITE TAPE MARK RETRY
0397 1226 CLEAN,- :CLEAN
0397 1227 MSGREL,- :MESSAGE BUFFER RELEASE
0397 1228 WRITESUBS,- :WRITE SUBSYSTEM MEMEORY
0397 1229 WRITECHAR,- :WRITE CHARACTERISTICS
0397 1230 >
03D3 1231 :*****NOTE INDEX OUT OF BOUND***

```

```

03D3 1233 .SBTTL NOP AND SIMULATED FUNCTIONS
03D3 1234
03D3 1235 :
03D3 1236 : SET CHARACTERISTIC FUNCTION
03D3 1237 :
03D3 1238
03D3 1239 SETCHAR: ;SET CHARACTERISTIC
68 A5 02 AA 03D3 1240 BICW #UCBSM_MS_SWAP,UCBSW_DEVSTS(R5) ;CLEAR SWAP BIT 1ST
04 EF 03D7 1241 EXTZV #MTSV_FORMAT,- ;GET FORMAT FIELD
04 03D9 1242 #MTSS_FORMAT,- ;
51 7C A5 03DA 1243 UCBSW_BOFF(R5),R1 ;
51 0E B1 03DD 1244 CMPW #MTSK_NORMAL15,R1 ;IS IT INDUSTRIAL COMPATIBLE?
04 12 03E0 1245 BNEQ 5$ ;BR IF NO
68 A5 02 AB 03E2 1246 BISW #UCBSM_MS_SWAP,UCBSW_DEVSTS(R5) ;SET SWAP BIT FOR
03E6 1247 ; SUBSEQUENT IO FUNCTIONS
03E6 1248 5$:
03E6 1249
03E6 1250 :
03E6 1251 : NO OPERATION AND SIMULATED NON-EXISTENT TS11/TS04 HARDWARE COMMAND
03E6 1252 :
03E6 1253
0800 8F AB 03E6 1254 PACKACK: ;PACK ACKNOWLEDGE
64 A5 03EA 1255 BISW #UCBSM_VALID,UCBSW_STS(R5) ; PACKACK implies set volume valid.
03EC 1256 NOP: ;NO OPERATION
03EC 1257 WRITECHECK: ;SIMULATED WRITECHECK
03EC 1258 WRITECHECKR: ;SIMULATED WRITECHECK REVERSE
03EC 1259 READPRESET: ;READ IN PRESET
03EC 1260
03EC 1261 EXHC 10$ ;EXECUTE HARDWARE COMMAND, IF ANY
03F1 1262 10$:
03B6 31 03F1 1263 BRW FCNEXT ;GOTO FUNCTION EXIT
03F4 1264 ;10$ AS RETRIABLE ERROR OCCURRED
03F4 1265 ;NO RETRIABLE ERROR, NOP ALWAYS SUCCESSFUL
03F4 1266
  
```

```
03F4 1268 .SBTTL READ HARDWARE FUNCTIONS
03F4 1269
03F4 1270 :
03F4 1271 : READ HARDWARE FUNCTIONS
03F4 1272 :
03F4 1273
03F4 1274 READDATA: ;READ DATA FORWARD
03F4 1275 EXHC 10$ ;EXECUTE HARDWARE COMMAND
00B0 C5 D6 03F9 1276 INCL UCBSL_RECORD(R5) ;INCREMENT RECORD COUNT
: 009F 31 03FD .1 BRW EXIT_READ_FCNEXT ;GOTO SUCCESSFUL RETURN
: ;10$ HANDLES RETRIABLE ERRORS
0400 1278
0400 1279 10$:
50 DD 0400 1280 PUSHL R0 ;SAVE R0 HAS TCC
00000000 GF 16 0402 1281 JSB G^ERL$DEVICERR ;LOG BEFORE RETRY
: 50 8ED0 0408 1282 POPL R0 ;RESTORE
:
50 04 D1 040B 1283 20$:
15 13 040B 1284 CMPL #TCC_REM,R0 ;DID TAPE MOVED
00B0 C5 97 040E 1285 BEQL 22$ ;YES BRANCH
: 24 19 0410 1286 DECB UCBSB_ERTCNT(R5) ;ANY RETRIES REMAINING?
: ;NO, GO AS FATAL
00B0 C5 D6 0416 1287 BLSS 30$ ;DO READ AGAIN
: 0087 31 0414 1288 EXHC 20$,HC RDN ;INCREMENT RECORD COUNT
: 041E 1289 INCL UCBSL_RECORD(R5) ;SUCCEED, RETURN
: 0422 .1 BRW EXIT_READ_RFCNEXT
00B0 C5 97 0425 .2 22$:
: 0F 19 0425 .3 DECB UCBSB_ERTCNT(R5) ;ANY RETRIES REMAINING?
: ;NO, GO AS FATAL ERROR
: 0429 .4 BLSS 30$ ;DO REREAD PREVIOUS
00B0 C5 D6 042B .5 EXHC 22$,HC RRP ;INCREMENT RECORD COUNT
: 0072 31 0433 .6 INCL UCBSL_RECORD(R5) ;SUCCEED, RETURN
: 0437 .7 BRW EXIT_READ_RFCNEXT
0000078F EF 17 043A 1297 30$:
: 043A 1298 JMP FATALERO
: 0440 1299
: 0440 1300 :
: 0440 1301 : REREAD PREVIOUS (SPACE REV,READ FWD)
: 0440 1302 :
: 0440 1303
: 0440 1304 REREADP: ;REREAD DATA PREVIOUS
: 0440 1305 EXHC 10$ ;
: 0057 31 0445 .1 BRW EXIT_READ_FCNEXT ;SUCCESS RETURN
: 0344 31 0448 1307 10$:
: 0448 1308 BRW FATALERO ;TREATED AS FATAL AS NOW
: 044B 1309
: 044B 1310 :
: 044B 1311 : READ PREVIOUS
: 044B 1312 :
: 044B 1313
: 044B 1314 READDATA: ;READ DATA REVERSE
: 00B0 C5 D7 044B 1315 EXHC 10$ ;
: 0048 31 0450 1316 DECL UCBSL_RECORD(R5) ;DECREMENT RECORD COUNT
: ;*NOTE*TMK PROBLEM???
: 0454 1317 BRW EXIT_READ_FCNEXT ;DO SUCCESSFUL RETURN
: 0457 1319 10$:
: 50 DD 0457 1320 PUSHL R0 ;RETRIABLE
: 00000000 GF 16 0459 1321 JSB G^ERL$DEVICERR ;SAVE R0 WHICH HAS TCC CODE
: 50 8ED0 045F 1322 POPL R0 ;LOG BEFORE RETRY
: ;RESTORE
50 04 D1 0462 1323 20$:
: 0462 1324 CMPL #TCC_REM,R0 ;TAPE MOVED?
```

:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
-7

```

0080 15 13 0465 1325 BEQL 22$ :YES
      C5 97 0467 1326 DECB UCBSB_ERTCNT(R5) :ANY RETRIES LEFT?
      24 19 046B 1327 BLSS 30$ :NO, AS FATAL ERROR
0080 C5 D7 046D 1328 EXHC 20$ HC RDP :DO READ DATA PREVIOUS AGAIN
      0030 31 0475 1329 DECL UCBSL_RECORD(R5) :DECREMENT RECORD COUNT
      0479 .1 BRW EXIT_READ_RFCNEXT :SUCCESS RETURN
      047C .2 22$:
0080 C5 97 047C .3 DECB UCBSB_ERTCNT(R5) :ANY RETRIES LEFT?
      OF 19 0480 .4 BLSS 30$ :NO, AS FATAL ERROR
      0482 .5 EXHC 22$ HC RRN :DO REPEAD DATA NEXT
0080 C5 D7 048A .6 DECL UCBSL_RECORD(R5) :DECREMENT RECORD COUNT
      001B 31 048E .7 BRW EXIT_READ_RFCNEXT :SUCCESS RETURN
      0491 1337 30$:
      02FB 31 0491 1338 BRW FATALERO
      0494 1339
      0494 1340
      0494 1341 :
      0494 1342 : REREAD DATA NEXT(SPACE FWD, READ REV)
      0494 1343 :
      0494 1344 :
      0494 1345 REREADN: :REREAD DATA NEXT
      0494 1346 EXHC 10$
0003 31 0499 .1 BRW EXIT_READ_FCNEXT :SUCCESS RETURN
      049C .2 10$:
      02F0 31 049C .3 BRW FATALERO :AS FATAL ERROR AS NOW
      049F .4
      049F .5 EXIT_READ_FCNEXT:
00FE C5 OF E1 049F .6 BBC #MS_XSRO_V_TMK,UCBSW_MS_XSRO(R5),10$ ; If not TM then branch
      04 04A4
00C4 C5 B4 04A5 .7 CLRW UCBSW_MS_XC(R5) ; Else clear the count returned
      02FE 31 04A9 .8 10$: BRW FCNEXT ; Branch to complete the I/O
      04AC .9
      04AC .10 EXIT_READ_RFCNEXT:
00FE C5 OF E1 04AC .11 BBC #MS_XSRO_V_TMK,UCBSW_MS_XSRO(R5),10$ ; If not TM then branch
      04 04B1
00C4 C5 B4 04B2 .12 CLRW UCBSW_MS_XC(R5) ; Else clear the count returned
      02F1 31 04B6 .13 10$: BRW RFCNEXT ; Branch to complete the I/O
      04B9 1350
      04B9 1351
  
```

:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331

:MMD0331
:MMD0331
:MMD0331
:MMD0331
:MMD0331

:MMD0331
:MMD0331
-3

```

04B9 1353 .SBTTL WRITE FUNCTIONS
04B9 1354
04B9 1355 : WRITE DATA
04B9 1356 :
04B9 1357 :
04B9 1358
04B9 1359 WRITEDATA: ;WRITE DATA FORWARD
46 A5 08 AA 04B9 1360 BICW #<MT%M_HWL>@-16,UCB$L_DEVDEPEND+2(R5) ;CLEAR
04BD 1361 ; HARDWARE WRITE LOCK BIT
04BD 1362 EXHC 10$ ;
00B0 C5 D6 04C2 1363 INCL UCB$L_RECORD(R5) ;INCREMENT RECORD COUNT
02E1 31 04C6 1364 BRW FCNEXT ;TAKE FUNCTION EXIT
04C9 1365 10$:
50 DD 04C9 1366 PUSHL R0 ;SAVE R0
00000000'GF 16 04CB 1367 JSB G^ERL$DEVICERR ;LOG BEFORE RETRY
50 BED0 04D1 1368 POPL R0 ;RESTORE
04D4 1369 20$:
50 04 D1 04D4 1370 CMLP #TCC_REM,R0 ;TAPE MOVED?
15 13 04D7 1371 BEQL 22$ ;YES
00B0 C5 97 04D9 1372 DECB UCB$B_ERTCNT(R5) ;ANY RETRIES LEFT?
24 19 04DD 1373 BLSS 30$ ;NO, /3 FATAL ERROR
04DF 1374 EXHC 20$,HC WRD ;YES, DO WRITE AGAIN
00B0 C5 D6 04E7 1375 INCL UCB$L_RECORD(R5) ;INCREMENT RECORD COUNT
02BC 31 04EB 1376 BRW RFCNEXT ;TAKE SUCCESS RETURN
04EE 1377 22$:
00B0 C5 97 04EE 1378 DECB UCB$B_ERTCNT(R5) ;ANY RETRIES LEFT?
OF 19 04F2 1379 BLSS 30$ ;NO, FATAL
04F4 1380 EXHC 22$,HC WDR ;DO WRITE DATA RETRY
00B0 C5 D6 04FC 1381 INCL UCB$L_RECORD(R5) ;INCREMENT RECORD COUNT
02A7 31 0500 1382 BRW RFCNEXT ;SUCCESS RETURN
0289 31 0503 1383 30$:
0503 1384 BRW FATALERO
0506 1385
0506 1386 :
0506 1387 : WRITE DATA RETRY(SPACE REV,ERASE,WRITE DATA)
0506 1388 :
0506 1389
0506 1390 WRITERET: ;WRITE DATA RETRY
029C 31 0506 1391 EXHC 10$ ;
050B 1392 BRW FCNEXT ;TAKE SUCCESS RETURN
050E 1393 10$:
027E 31 050E 1394 BRW FATALERO ;AS FATAL
0511 1395
0511 1396 :
0511 1397 : WRITE SUBSYSTEM MEMORY
0511 1398 :
0511 1399
0511 1400 WRITESUBS: ;WRITE SUBSYSTEM MEMORY
0291 31 0511 1401 EXHC 10$ ;
0516 1402 BRW FCNEXT ;SUCCESS RETURN
0519 1403 10$:
0519 1404
0273 31 0519 1405 BRW FATALERO
051C 1406
051C 1407 :
051C 1408 : WRITE CHARACTERISTICS
051C 1409 : USED TO TELL SUBSYSTEM MSG BUFFER ADDR. & SET CHARACTERISTIC WORD
  
```



```
051C 1410 ;  
051C 1411  
051C 1412 WRITECHAR: ;WRITE CHARACTERISTICS  
051C 1413  
051C 1414  
0286 31 0521 1415 EXHC 10$  
BRW FCNEXT ;SUCCESS RETURN  
0268 31 0524 1416 10$:  
0524 1417 BRW FATALERO  
0527 1418 ;
```

```

0527 1420 .SBTTL POSITIONING FUNCTIONS
0527 1421
0527 1422 :
0527 1423 : SPACE FILE FORWARD
0527 1424 : NOTE: HARDWARE SKIPFILE COMMAND IS NOT USED.
0527 1425 : SKIPFILE IS SIMULATED BY A SERIES SKIPRECORD COMMANDS.
0527 1426 :
0527 1427 :
0527 1428 SPCFILFOR: ;SPACE FILE FORWARD
00B4 C5 B0 0527 1429 MOVW UCBSW_MS_SPACNT(R5),UCBSW_BOFF(R5) ;SAVE NO. OF
    7C A5 052B
00B4 C5 B0 052D 1430 MOVW UCBSW_MS_SPACNT(R5),UCBSW_BCNT(R5) ; TAPE MARKS TO SKIP
    7E A5 0531
00B0 C5 D0 0533 1431 MOVL UCBSL_RECORD(R5),UCBSL_MS_PMPR(R5) ;SAVE TAPE POSITION
00D0 C5 0537
0040 8F A8 053A 1432 BISW #UCBSM_MS_SWE,UCBSW_DEVSTS(R5) ;USE OLD TAPE POSITION IF POWERFAIL
    68 A5 053E
    0540 1433 ;**SOFTWARE EMULATED FUNCTION**
    0540 1434 0$:
68 A5 01 AA 0540 1435 BICW #UCBSM_MS_FEF,UCBSW_DEVSTS(R5) ;CLEAR FLAG FOR 1ST EOF SEEN
    7FFF 8F B0 0544 1436 1$:
    00B4 C5 0548 1437 MOVW #^X7FFF,UCBSW_MS_SPACNT(R5) ;SKIP 32,768 RECORDS INSTEAD
    054B 1438 EXHC 10$,HC STF ;DO IT
51 00C4 C5 3C 0553 1439 MOVZWL UCBSW_MS_XC(R5),R1 ;GET NO. OF RECORDS PASSED
00B0 C5 51 C0 0558 1440 ADDL R1,UCBSL_RECORD(R5) ;ADD IT
DE 44 A5 11 E1 055D 1441 BBC #MTSV_EOF,UCBSL_DEVDEPEND(R5),0$ ;BR IF DIDN'T SEE TAPE MARK
    7E A5 B7 0562 1442 DECW UCBSW_BCNT(R5) ;DECREMENT TAPE MARK PASSED
    13 E1 0565 1443 BBC #DEVSQ_MNT,- ;BR IF NOT MOUNTED
    05 38 A5 0567 1444 UCBSL_DEVCHAR(R5),2$
    18 E1 056A 1445 BBC #DEVSQ_FOR,- ;BR IF MOUNTED NOT FOREIGN
    34 38 A5 056C 1446 UCBSL_DEVCHAR(R5),5$
    056F 1447 2$:
2B 68 A5 00 E1 056F 1448 BBC #UCBSV_MS_FEF,UCBSW_DEVSTS(R5),4$ ;BR IF 1ST TMK
00C4 C5 01 B1 0574 1449 CMPW #1,UCBSW_MS_XC(R5) ;**1 RECORD=TAPE MARK??**
    28 12 0579 1450 BNEQ 5$ ;BR IF NO
00B4 C5 01 B0 057B 1451 MOVW #1,UCBSW_MS_SPACNT(R5) ;SKIP 1 TMK REVERSE
    7E A5 B6 0580 1452 EXHC 10$,HC STR
    00B0 C5 D7 0588 1453 INCW UCBSW_BCNT(R5) ;BACKUP 1 TMK PASSED
50 09A0 8F 3C 058B 1454 DECL UCBSL_RECORD(R5) ;UPDATE TAPE POSITION
7C A5 7E A5 A3 058F 1455 MOVZWL #SS$ ENDOFVOLUME,R0 ;YES, DOUBLE TMKS=ENDOFVOLUME
00C4 C5 0594 1456 SUBW3 UCBSW_BCNT(R5),UCBSW_BOFF(R5),UCBSW_MS_XC(R5) ;GET NO. OF
    0599
    020B 31 059C 1457 ; TAPE MARKS PASSED
    059C 1458 BRW FCNEXT ;GO EXIT
68 A5 01 A8 059F 1459 4$:
    05A3 1461 5$:
    7E A5 B5 05A3 1462 TSTW UCBSW_BCNT(R5) ;PASSED ALL TAPE MARKS
    9C 12 05A6 1463 BNEQ 1$ ;NO, GO BACK
    7C A5 B0 05A8 1464 MOVW UCBSW_BOFF(R5),UCBSW_MS_XC(R5) ;YES,COPY TMKS PASSED
00C4 C5 05AB
    01F9 31 05AE 1465 BRW FCNEXT ;GO EXIT
    01DB 31 05B1 1466 10$:
    05B1 1467 BRW FATALERO ;TAKE FAILURE RETURN
    05B4 1468
    05B4 1469 ;
    
```

```

05B4 1470 ; SPACEFILE REVERSE
05B4 1471 ;
05B4 1472 ;
05B4 1473 SPCFILREV: ;SPACE FILE REVERSE
00B4 C5 B0 05B4 1474 MOVW UCBSW_MS_SPACNT(R5),UCBSW_BOFF(R5) ;SAVE NO. OF
7C A5 05B8
00B4 C5 B0 05BA 1475 MOVW UCBSW_MS_SPACNT(R5),UCBSW_BCNT(R5) ; TAPE MARKS TO SKIP
7E A5 05BE
00B0 C5 D0 05C0 1476 MOVL UCBSL_RECORD(R5),UCBSL_MS_PMPR(R5) ;SAVE TAPE POSITION
00D0 C5 05C4
0040 8F A8 05C7 1477 BISW #UCBSM_MS_SWE,UCBSW_DEVSTS(R5) ;USE OLD TAPE POSITION IF POWERFAIL
68 A5 05CB
05CD 1478 ;**SOFTWARE EMULATED FUNCTION**
05CD 1479 1$: MOVW #^X7FFF,UCBSW_MS_SPACNT(R5) ;SKIP 32,768 RECORDS INSTEAD
00B4 C5 05D1
05D4 1481 EXHC 10$,HC STR ;DO IT
17 44 A5 E0 05DC 1482 BBS #MT$V_BOT,- ; If we ran into BOT, treat as if
51 00C4 C5 3C 05DE 1483 UCBSL_DEVDEPEND(R5),5$ ; we were done.
00B0 C5 51 C2 05E1 1484 MOVZWL UCBSW_MS_XC(R5),R1 ;GET NO. OF RECORDS PASSED
DD 44 A5 11 E1 05E6 1485 SUBL R1,UCBSL_RECORD(R5) ;SUBTRACT
7E A5 B7 05EB 1486 BBC #MT$V_EOF,UCBSL_DEVDEPEND(R5),1$ ;BR IF DIDN'T SEE TAPE MARK
7E A5 B5 05F0 1487 DECW UCBSW_BCNT(R5) ;DECREMENT TAPE MARK PASSED
D5 12 05F3 1488 TSTW UCBSW_BCNT(R5) ;PASSED ALL TAPE MARKS?
05F6 1489 BNEQ 1$ ;NO, BR BACK
7E A5 A3 05F8 1490 5$: SUBW3 UCBSW_BCNT(R5),- ; Calculate number of tape
7C A5 05FB 1491 UCBSW_BOFF(R5),- ; marks passed.
00C4 C5 05FD 1492 UCBSW_MS_XC(R5)
01A7 31 0600 1494 BRW FCNEXT ;GO EXIT
0189 31 0603 1495 10$: BRW FATALERO
0606 1497
0606 1498 ; SPACE RECORD FORWARD
0606 1499 ;
0606 1500 ;
0606 1501 ;
0606 1502 SPCRECFOR: ;SPACE REGRD FORWARD
5C 44 A5 11 E1 060B 1504 EXHC 10$ ;
00C4 C5 01 B1 0610 1505 BBC #MT$V_EOF,UCBSL_DEVDEPEND(R5),8$ ;BR IF NO TMK
55 12 0615 1506 CMPW #1,UCBSW_MS_XC(R5) ;**1 RECORD=TMK?**
05 38 A5 E1 0617 1507 BNEQ 8$ ;BR IF NO
4B 38 A5 E1 0619 1508 BBC #DEV$V_MNT,- ;BR IF NOT MOUNTED
061C 1509 BBC #DEV$V_FOR,- ;BR IF MOUNTED NOT FOREIGN
061E 1510 UCBSL_DEVCHAR(R5),2$ ;
0621 1511 2$: UCBSL_DEVCHAR(R5),8$ ;
00B0 C5 D5 0621 1512 TSTL UCBSL_RECORD(R5) ;WAS AT BOT?
00B4 C5 01 B0 0625 1513 BEQL 8$ ;BR IF YES
00B4 C5 01 B0 0627 1514 MOVW #1,UCBSW_MS_SPACNT(R5) ;SKIP 1 RECORD REVERSE
00B4 C5 01 B0 062C 1515 EXHC 10$,HC_SRR ;
00B4 C5 01 B0 0634 1516 MOVW #1,UCBSW_MS_SPACNT(R5) ;SKIP 1 RECORD REVERSE
0639 1517 EXHC 10$,HC_SRR ;
0641 1518 MOVW #1,UCBSW_MS_SPACNT(R5) ;SKIP 1 RECORD FORWARD
OC 44 A5 11 E1 0646 1519 EXHC 10$,HC_SRF ;
50 09A0 8F 3C 064E 1520 BBC #MT$V_EOF,UCBSL_DEVDEPEND(R5),6$ ;BR IF NO TMK
0653 1521 MOVZWL #$$$_ENDOFVOLUME,R0 ;WAS AT ENDOFVOLUME
    
```

```

00C4 C5 B4 0658 1522 CLRW UCBSW_MS_XC(R5) ;NO RESULTANT MOVEMENT
014B 31 065C 1523 BRW FCNEXT ;RETURN
00B4 C5 01 B0 065F 1524 6$:
065F 1525 MOVW #1,UCBSW_MS_SPACNT(R5) ;SKIP 1 RECORD FORWARD
0664 1526 EXHC 10$,HC_SRF ;
066C 1527 8$:
51 00C4 C5 3C 066C 1528 MOVZWL UCBSW_MS_XC(R5),R1 ;GET NO. OF RECORDS PASSED
00B0 C5 51 C0 0671 1529 ADDL R1,UCBSL_RECORD(R5) ;UPDATE
0131 31 0676 1530 BRW FCNEXT ;
0113 31 0679 1531 10$:
0679 1532 BRW FATALERO
067C 1533
067C 1534 ; SPACE RECORD REVERSE
067C 1535 ;
067C 1536 ;
067C 1537
067C 1538 SPCRECREV: ;SPACE RECORD REVERSE
067C 1539 EXHC 10$ ;
0681 1540 BBS #MT$V_BOT,- ; If we ran into BOT, treat as if
0683 1541 UCBSL_DEVDEPEND(R5),5$ ; we were done.
51 0A 44 A5 E0 0686 1542 MOVZWL UCBSW_MS_XC(R5),R1 ;GET NO. OF RECORDS PASSED
00B0 C5 51 C2 068B 1543 SUBL R1,UCBSL_RECORD(R5) ;UPDATE
0117 31 0690 1544 5$:
0690 1545 BRW FCNEXT ;
00F9 31 0693 1546 10$:
0693 1547 BRW FATALERO
0696 1548
0696 1549 ; REWIND
0696 1550 ;
0696 1551 ;
0696 1552
0696 1553 REWIND: ;REWIND
0696 1554 EXHC 10$
46 A5 01 A8 069B 1555 B!SW #<MT$M_BOTA-16>,UCBSL_DEVDEPEND+2(R5) ;MARK BOT
46 A5 10 AA 069F 1556 B!CW #<MT$M_LOSTA-16>,UCBSL_DEVDEPEND+2(R5) ;CLEAR POSITION-LOST
00B0 C5 D4 06A3 1557 CLRL UCBSL_RECORD(R5)
0100 31 06A7 1558 BRW FCNEXT
00E2 31 06AA 1559 10$:
06AA 1560 BRW FATALERO
06AD 1561
    
```

```

06AD 1563 .SBTTL FORMAT COMMANDS
06AD 1564
06AD 1565 :
06AD 1566 : WRITE TAPE MARK
06AD 1567 :
06AD 1568
06AD 1569 WRTTMK:
06AD 1570 BICW #<MTSM_HWL>@-16,UCBSL_DEV:WRITE TAPE MARK
06B1 1571 :VDEPEND+2(R5) :CLEAR
06B1 1572 : WRITE LOCK BIT FIRST
06B6 1573 EXHC 10$
06BA 1574 INCL UCBSL_RECORD(R5) :INCREMENT RECORD COUNT
06BD 1575 BRW FCNEXT :GOTO EXIT
10$:
06BD 1576 PUSHL R0 :SAVE R0
06BF 1577 JSB G*ERL$DEVICERR :LOG BEFORE RETRY
06C5 1578 POPL R0 :RESTORE
20$:
06C8 1579 CMPL #TCC_REM,R0 :TAPE MOVED?
06C8 1580 BEQL 22$ :YES
06CD 1582 DECB UCBSB_ERTCNT(R5) :ANY RETRIES LEFT?
06D1 1583 BLSS 30$ :NO, FATAL
06DB 1584 EXHC 20$,HC WTM :DO IT AGAIN
06DF 1586 INCL UCBSL_RECORD(R5) :INCREMENT RECORD COUNT
06E2 1587 BRW RFCNEXT :RETURN
22$:
06E2 1588 DECB UCBSB_ERTCNT(R5) :ANY RETRIES LEFT?
06E6 1589 BLSS 30$ :NO, FATAL
06E8 1590 EXHC 22$,HC WTR :DO WRITE TAPE MARK RETRY
06F0 1591 INCL UCBSL_RECORD(R5) :INCREMENT RECORD COUNT
06F4 1592 BRW RFCNEXT :
06F7 1593
06F7 1594 30$: BRW FATALERO :BRANCH FATAL ERROR
06FA 1595 :
06FA 1596 : WRITE TAPE MARK RETRY(SPACE REV,ERASE,WRITE TAPE MARK)
06FA 1597 :
06FA 1598
06FA 1599 WRTTMKR: :WRITE TAPE MARK RETRY
06FA 1600 EXHC 10$ :
06FF 1601 BRW FCNEXT :GO EXIT
10$:
0702 1602 BRW FATALERO :FATAL AS NOW
0705 1604 :
0705 1605 : ERASE
0705 1606 :
0705 1607
0705 1608 ERASE: :ERASE
0705 1609 EXHC 10$ :
070A 1610 BRW FCNEXT :
10$:
070D 1611 BRW FATALERO
0710 1612
0710 1613
  
```

```
0710 1615 .SBTTL CONTROL COMMANDS
0710 1616
0710 1617 :
0710 1618 : CONTROL COMMANDS
0710 1619 :
0710 1620
0710 1621 MSGREL: ;MESSAGE BUFFER RELEASE
0092 31 0710 1622 EXHC 10$
0715 1623 BRW FCNEXT
0718 1624 10$:
0718 1625
0074 31 0718 1626 BRW FATALERO
071B 1627
071B 1628 UNLOAD: ;
071B 1629 EXHC 10$
0087 31 0720 1630 BRW FCNEXT
0723 1631 10$:
0069 31 0723 1632 BRW FATALERO
0726 1633
0726 1634 CLEAN: ;CLEAN
0726 1635 EXHC 10$
007C 31 072B 1636 BRW FCNEXT
072E 1637 10$:
005E 31 072E 1638 BRW FATALERO
0731 1639
```

```

0731 1641 .SBTTL INITIALIZE AND GET STATUS
0731 1642
0731 1643 :
0731 1644 : DRIVE INITIALIZE
0731 1645 :
0731 1646
0731 1647 DRVCLR: ;DRIVE INITIALIZE
0731 1648
0071 31 0736 1649 EXHC 10$
BRW FCNEXT
0739 1650 10$:
0053 31 0739 1651 BRW FATALERO
073C 1652
073C 1653 :
073C 1654 : GET STATUS (END MESSAGE ONLY)
073C 1655 :
073C 1656
073C 1657 GETSTS:
073C 1658 EXHC 10$
0741 1659 BICW #<MTSM_BOT!- ;CLEAR BITS IN UCBSL_DEVDEPEND+2
0745 1660 MTSM_EOF!- ;
0745 1661 MTSM_EOT!- ;END OF TAPE
0745 1662 MTSM_HWL>@-16,UCBSL_DEVDEPEND+2(R5) ;
0745 1663 BBC #MS_XSRO_V BOT,- ;AT BOT?
0747 1664 UCBSW_MS_XSRO(R5),1$ ;BR IF NO
074B 1665 CLRL UCBSL_RECORD(R5) ;CLEAR RECORD COUNT
46 A5 01 AB 074F 1666 BISW #<MTSM_BOT@-16>,UCBSL_DEVDEPEND+2(R5) ;SET BOT
46 A5 10 AA 0753 1667 BICW #<MTSM_LOST@-16>,UCBSL_DEVDEPEND+2(R5) ;CLEAR LOST BIT
0757 1668 1$:
0757 1669 BBC #MS_XSRO_V TMK,- ;AT TAPE MARK
0759 1670 UCBSW_MS_XSRO(R5),2$ ;BR IF NO
46 A5 02 AB 075D 1671 BISW #<MTSM_EOF@-16>,UCBSL_DEVDEPEND+2(R5) ;SET EOF
0761 1672 2$:
0761 1673 BBC #MS_XSRO_V WLK,- ;WRITE-LOCKED?
0763 1674 UCBSW_MS_XSRO(R5),3$ ;BR IF NO
46 A5 08 AB 0767 1675 BISW #<MTSM_HWL@-16>,UCBSL_DEVDEPEND+2(R5) ;SET WRITE-LOCKED
076B 1676 3$:
076B 1677 BBC #MS_XSRO_V EOT,- ;END OF TAPE
076D 1678 UCBSW_MS_XSRO(R5),4$ ;BR IF NO
46 A5 10 AA 0771 1679 BICW #<MTSM_LOST@-16>,UCBSL_DEVDEPEND+2(R5) ;CLEAR POS.LOST
46 A5 04 AB 0775 1680 BISW #<MTSM_EOT@-16>,UCBSL_DEVDEPEND+2(R5) ;SET END OF TAPE
50 0878 8F 3C 0779 1681 MOVZWL #SS$_ENDOFTAPE,R0 ;PUT IN RETURN STATUS
077E 1682 4$:
077E 1683 BBS #MS_XSRO_V ONL,- ;CHECK IF ONLINE?
0780 1684 UCBSW_MS_XSRO(R5),6$ ;BR IF YES
0784 1685
50 01A4 8F 3C 0784 1686 MOVZWL #SS$_MEDOFL,R0 ;RETURN MEDIUM-OFFLINE
0789 1687 6$:
001E 31 0789 1688 BRW FCNEXT
078C 1689 10$:
0000 31 078C 1690 BRW FATALERO ;TREAT AS FATAL
078F 1691

```

```

078F 1693 .SBTTL COMPLETION PROCESSING
078F 1694 :
078F 1695 : FATALERR - FINISHING UP THE I/O REQUEST PROCESSING WHEN THE OPERATION
078F 1696 : ENDS WITH FATAL OR HARD ERROR.
078F 1697 : RO HAS THE FINAL STATUS CODE ALREADY.
078F 1698 :
078F 1699 :
50 008C 8F 3C 078F 1700 FATALERO: ;NO ERROR CODE IN RO
078F 1701 MOVZWL #SS$ _DRVERR,RO ;GIVE IT ONE FOR NOW
0794 1702 FATALERR:
0794 1703 CLRW UCBSW_MS_XC(R5) ;MAKE SURE NOTHING XFERRERD/SKIPPED
0798 1704
50 01A4 8F B1 0798 1705 CMPW #SS$ MEDOFL,RO ; See if error is MEDIA OFF LINE.
079D 1706 BEQL FCNEXT ; If so, then branch around logging error.
079F 1707 PUSHL RO ;SAVE FINAL STATUS
00000000 GF 16 07A1 1708 JSB G^ERL$DEVICERR ;LOG DEVICE ERROR
50 8ED0 07A7 1709 POPL RO ;
07AA 1710 RFCNEXT: ;SUCCESS RETURN AFTER RETRY
07AA 1711 FCNEXT:
0840 8F AA 07AA 1712 BICW #<UCBSM_MS_RPI!UCBSM_MS_SWE>,UCBSW_DEVSTS(R5) ;ASSURE FLAGS CLEARED
68 A5 07AE
00000000 GF 50 DD 07B0 1713 PUSHL RO ;SAVE FINAL STATUS
00C4 C5 B0 07B2 1714 JSB G^IOC$DIAGBUFILL ;FILL DIAGNOSTIC BUFFER IF PRESENT
02 AE 07B8 1715 MOVW UCBSW_MS_XC(R5),2(SP) ;SET BYTES XFERRERD OR RECORDS/FILES SKIPPED
36 6E E8 07BE 1716 BLBS (SP),70$ ;IF LBS SUCCESSFUL COMPLETION
54 58 A5 DO 07C1 1717 MOVL UCBSL_IRP(R5),R4 ;GET ADDRESS OF CURRENT I/O PACKET
2D 2A A4 04 E1 07C5 1718 BBC #IRPSV_VIRTUAL,IRPSW_STS(R4),70$ ;IF CLR, NOT VIRTUAL FUNCTION
54 18 A4 DO 07CA 1719 MOVL IRPSL_WIND(R4),R4 ;GET ADDRESS OF WINDOW BLOCK
16 A4 B4 07CE 1720 CLRW WCBSW_NMAP(R4) ;CLEAR NUMBER OF MAPPING POINTERS
54 34 A5 DO 07D1 1721 MOVL UCBSL_VCB(R5),R4 ;GET ADDRESS OF VCB LISTHEAD
52 4C A5 9E 07D5 1722 MOVAB UCBSL_IOQFL(R5),R2 ;GET ADDRESS OF I/O QUEUE
53 52 DO 07D9 1723 MOVL R2,R3 ;SET ADDRESS OF PREVIOUS ENTRY
53 63 DO 07DC 1724 60$: MOVL (R3),R3 ;GET ADDRESS OF NEXT ENTRY
52 53 D1 07DF 1725 CMPL R3,R2 ;END OF LIST?
13 13 07E2 1726 BEQL 70$ ;IF EQL YES
F3 2A A3 04 E1 07E4 1727 BBC #IRPSV_VIRTUAL,IRPSW_STS(R3),60$ ;IF CLR, NOT VIRTUAL FUNCTION
53 04 A3 DO 07E9 1728 MOVL 4(R3),R3 ;RETRIEVE ADDRESS OF PREVIOUS ENTRY
51 00 B3 OF 07ED 1729 REMQUE @ (R3),R1 ;REMOVE ENTRY FROM DRIVER QUEUE
04 B4 61 OE 07F1 1730 INSQUE (R1),@4(R4) ;INSERT ENTRY IN BLOCKED I/O LIST
E5 11 07F5 1731 BRB 60$
50 8ED0 07F7 1732 70$: POPL RO ;RETRIEVE FINAL STATUS
51 44 A5 DO 07FA 1733 MOVL UCBSL_DEVDEPEND(R5),R1 ;SET MAGTAPE STATUS AND CHARACTERISTIC
07FE 1734 .IF DF TS TRACE
07FE 1735 BSBW TRACE_STATUS . Trace final I/O status.
07FE 1736 .ENDC
07FE 1737 REQCOM ;COMPLETE REQUEST
0804 1738

```



```

0804 1740 .SBTTL  HARDWARE COMMAND EXECUTOR
0804 1741 :+
0804 1742 : HCEX - EXECUTES HARDWARE COMMAND
0804 1743 :
0804 1744 : THIS ROUTINE IS CALLED VIA A BSB WITH A WORD IMMEDIATELY FOLLOWING THAT
0804 1745 : SPECIFIES THE ADDRESS OF AN (RETRIABLE) ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE
0804 1746 : BEEN SET UP IN THE UCB BEFORE THE CALL. THE COMMAND PACKET IS APPROPRIATELY
0804 1747 : SETUP AND INITIATED BY LOADING THE ADDR. OF COMMAND PACKET INTO THE
0804 1748 : TS11/TS04 DEVICE REGISTER, TSDB. THEN, A WAITFOR INTERRUPT IS EXECUTED
0804 1749 : AND WHEN THE INTERRUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.
0804 1750 : THE ROUTINE MAINLY DEALS WITH THE HARDWARE INTERFACE.
0804 1751 :
0804 1752 : INPUTS:
0804 1753 : R0=HARDWARE COMMAND TABLE DISPATCH INDEX
0804 1754 : R4=EQUIVALENT CSR ADDR. FOR TS11/TS04
0804 1755 : R5=DEVICE UNIT UCB ADDRESS
0804 1756 :
0804 1757 : O0(SP) = RETURN ADDRESS OF CALLER
0804 1758 : O4(SP) = RETURN ADDRESS OF CALLER'S CALLER
0804 1759 :
0804 1760 : IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A WORD WHICH HAS
0804 1761 : A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE, IF APPROPRIATE.
0804 1762 : OUTPUTS:
0804 1763 : THE DRIVE STATUS, SUCH AS BOT, EOT, ETC, ARE RECORDED IN UCB.
0804 1764 :
0804 1765 : THERE ARE THREE EXITS FROM THIS ROUTINE:
0804 1766 : 1) NORMAL RETURN,
0804 1767 : 2) FATAL OR HARD ERROR EXIT, AND
0804 1768 : 3) RETRIABLE ERROR RETURN.
0804 1769 : WHEN EXITS, R0 HAS THE FINAL STATUS CODE IF NORMAL OR FATAL,
0804 1770 : R0 HAS TERMINATION CODE, 4 OR 5, IF RETRIABLE.
0804 1771 : THE DRIVE STATUS IS RECORDED INTO UCB WHILE PROCESSING TERMINATION CODE
0804 1772 :
0804 1773 :
0804 1774 HCEX:
0804 1775 : POPL  UCB$D_DPC(R5) ;SAVE DRIVER PC VALUE
0809 1776 : MOVZBL #20,UCB$D_MS_TIMEOUT(R5) ; Initialize timeout to 20 seconds.
080E 1777 : CLRW  UCB$B_MS_DPN(R5) ;CLEAR DATA PATH NO. & PURGE ERROR
0812 1778 : CLRW  UCB$B_MS_DPR(R5) ;ZERO DATAPATH REG. & FINAL MAP REG.
0816 1779 : CLRW  UCB$W_MS_XC(R5) ;INITIALIZE COUNT
081A 1780 : MOVB  R0,UCB$B_CEX(R5) ;SAVE CASE INDEX
081F 1781 : MOVL  UCB$D_MS_TSPT1(R5),R1 ;GET COMMAND PACKET POINTER
0824 1782 : MOVW  HCTAB[R0],MS_CPHD(R1) ;LOAD COMMAND PACKET HEAD WORD
009C C5 8ED0
00DC C5 14 9A
00C6 C5 B4
00C8 C5 7C
00C4 C5 B4
0093 C5 50 90
51 00B6 C5 D0
61 F80F CF40 B0
  
```

FFFF 8F	B0	082A	1784	MOVW	#^XFFFF,MS_MHD(R1)	;MARK MSG HEAD TO ENSURE MSG BUFFER RETURNED
10 A1		082E				
05 68 A5 OC	E5	0830	1785	BBCC	#UCBSV MS_VCK,UCBSW DEVSTS(R5),7\$;BR IF NOT VOLUME CHECK
61 4000 8F	A8	0835	1786	BISW	#MS_CPRD_M_CVC,MS_CPHD(R1)	;YES, FLAG TO CLEAR VOLUME CHECK
		083A	1787			7\$:
		083A	1788	CASE	RO,<-	;DISPATCH TO PROPER COMMAND ROUTINE
		083A	1789		PNOP,-	;NOP
		083A	1790		PMIS,-	;UNLOAD
		083A	1791		PPOS,-	;SPACE FILE FORWARD
		083A	1792		PPOS,-	;REWIND
		083A	1793		PMIS,-	;DRIVE CLEAR
		083A	1794		PPOS,-	;SPACE FILE REVERSE
		083A	1795		PMIS,-	;ERASE
		083A	1796		PPOS,-	;SPACE RECORD REVERSE
		083A	1797		PNOP,-	;SIMULATED PACK ACKNOWLEDGE
		083A	1798		PPOS,-	;SPACE RECORD FORWARD
		083A	1799		PNOP,-	;SIMULATED WRITECHECK
		083A	1800		PXFR,-	;WRITE DATA FORWARD
		083A	1801		PXFR,-	;READ DATA FORWARD
		083A	1802		PNOP,-	;SIMULATED WRITE CHECK REVERSE
		083A	1803		PXFR,-	;WRITE DATA (NO REVERSE)
		083A	1804		PXFRR,-	;READ DATA REVERSE
		083A	1805		PXFRRD,-	;REREAD DATA NEXT
		083A	1806		PXFRRD,-	;REREAD DATA PREVIOUS
		083A	1807		PXFR,-	;WRITE DATA RETRY
		083A	1808		PNOP,-	;SIMULATED READ PRESET
		083A	1809		PNOP,-	;SIMULATED SET CHARACTERISTIC
		083A	1810		PMIS,-	;GET STATUS IMMEDIATE(SENS CHAR.)
		083A	1811		PMIS,-	;WRITE TAPE MARK
		083A	1812		PMIS,-	;WRITE TAPE MARK RETRY
		083A	1813		PMIS,-	;CLEAN
		083A	1814		PMIS,-	;MESSAGE BUFFER RELEASE
		083A	1815		PXFR,-	;WRITE SUBSYSTEM MEMORY
		083A	1816		PWCH,-	;WRITE CHARACTERISTICS
		083A	1817		>	:
		0876	1818			:

```

0876 1820 :+PNOP - NO OPERATION ON HARDWARE FOR MANY SIMULATED FUNCTIONS
0876 1821 : THEY ARE: NOP,PACK ACKNOWLEDGE,WRITECHECK,WRITE CHECK REVERSE,
0876 1822 : READ IN PRESET,SET CHARACTERISTICS.
0876 1823 : THE ROUTINE SIMPLY RETURNS.
0876 1824 :
0876 1825 PNOP: ;
0876 1826 RET: ;
009C 50 01 3C 0876 1827 MOVZWL #SS$ NORMAL,R0 ;ALWAYS SUCCESS
C5 02 C0 0879 1828 ADDL #2,UCB$L_DPC(R5) ;ADJUST TO CORRECT RETURN
009C D5 17 087E 1829 JMP @UCB$L_DPC(R5) ;RETURN TO DRIVER
0882 1830
0882 1831
0882 1832 : PPOS - CONSTRUCT COMMAD PACKET FOR POSITIONING COMMANDS:
0882 1833 : SPACE RECORDS FORWARD, SPACE RECORDS REVERSE, SKIP TAPE MARKS FORWARD,
0882 1834 : SPACE TAPE MARKS REVERSE, AND REWIND.
0882 1835
0882 1836 PPOS: ;
50 03 91 0882 1837 CMPB #CDHC_RWD,R0 ;REWIND COMMAND?
2A 12 0885 1838 BNEQ 5$ ;NO
0887 1839 DSBINT ;YES, DISABLE INTERRUPTS
088D 1840 BBC #UCBSV POWER,-
03 64 A5 088F 1841 UCBSW_STS(R5),1$ ; Continue if NO powerfail
0090 31 0892 1842 BRW PWRFLT ; Branch around if we had POWERFAIL.
0895 1843 1$:
64 00BE C5 B0 0895 1844 MOVW UCBSW_MS_TSPT3(R5),(R4) ;LOAD COMMAND POINTER
089A 1845 WFIKPCB MSTMOT,#*D300 ;TIMEOUT FOR 5 MIN.'S FOR 2400 FEET TAPE
08A8 1846 IOFORK ;MAKE IT FORK FIRST
0193 31 08AE 1847 BRW XTC ;GO PROCESS TERMINATION CODE
08B1 1848 5$: ;NOT REWIND OR UNLOAD
00B4 C5 B0 08B1 1849 MOVW UCBSW_MS_SPACNT(R5),MS_BACT(R1) ;LOAD THE COUNT
02 A1 08B5
00B4 C5 B0 08B7 1850 MOVW UCBSW_MS_SPACNT(R5),UCBSW_MS_XC(R5) ;COPY COUNT
00C4 C5 08BB
08BE 1851 ;FALL INTO CODE TO LOAD COMMAND
00B4 C5 3C 08BE 1852 MOVZWL UCBSW_MS_SPACNT(R5),- ; Maximum size record takes one second
00DC C5 08C2 1853 UCBSL_MS_TIMEOUT(R5) ; to skip (approximately).
00000294 8F D1 08C5 1854 CML #11*60,- ; Compare time to skip entire tape to
00DC C5 08CB 1855 UCBSL_MS_TIMEOUT(R5) ; time to skip this # records.
09 18 08CE 1856 BGEQ 10$ ; GEQ implies skipping small # records
0294 8F 3C 08D0 1857 MOVZWL #11*60,- ; Else use maximum time to skip whole
00DC C5 08D4 1858 UCBSL_MS_TIMEOUT(R5) ; tape.
05 11 08D7 1859 BRB 20$ ; And branch around.
00DC C5 02 C0 08D9 1860 10$: ADDL #2,UCBSL_MS_TIMEOUT(R5) ; Add in fudge factor for small skips.
08DE 1861 20$:
08DE 1862
08DE 1863 :+PMIS - CONSTRUCT COMMAND PACKET FOR FORMAT COMMANDS: WRITE TAPE MARK,
08DE 1864 : ERASE,WRITE TAPE MARK RETRY; CONTROL COMMANDS: MESSAGE BUFFER RELEASE,
08DE 1865 : UNLOAD, & CLEAN; INITIALIZE COMMAND: DRIVE INITIALIZE; AND GET STATUS
08DE 1866 : COMMAND: GET STATUS IMMEDIATE.
08DE 1867
08DE 1868 PMIS: ;
001F 31 08DE 1869 BRW LDTSDB ;GO LOAD DEVICE REGISTER
08E1 1870
08E1 1871 : PWCH - CONSTRUCT COMMAND PACKET FOR WRITE CHARACTERISTIC COMMAND
08E1 1872
08E1 1873 PWCH: ; R1 POINTS TO COMMAND PACKET
00BA C5 D0 08E1 1874 MOVL UCBSL_MS_TSPT2(R5),MS_BACT(R1) ;STORE CHAR. BUFFER ADDR.

```

```

02 A1 02 A1 08 C0 08E5 1875 ADDL #8,MS_BACT(R1) ;POINT TO CHAR. BUFFER NOW
06 A1 08 B0 08E7 1876 MOVW #8,MS_CNT(R1) ;STORE BYTE COUNT FOR CHAR. DATA
00BA C5 D0 08EF 1877 MOVL UCBSL_MS_TSPT2(R5),MS_MBA0(R1) ;STORE MESSAGE BUFFER ADDR.
08 A1 08 A1 08F3
08 A1 10 C0 08F5 1878 ADDL #16,MS_MBA0(R1) ; AS CHAR. DATA
0C A1 0E B0 08F9 1879 MOVW #14,MS_LNTH(R1) ;LENGTH OF CHAR. DATA=14.
0E A1 B4 08FD 1880 CLRW MS_CHWD(R1) ;ZERO CHARACTERISTIC WORD
0900 1881
0900 1882 ; NOW, COMMAND PACKET IS SETUP, READY TO LOAD DEVICE REGISTER
0900 1883
0900 1884
0900 1885 LDTSDB: ;TS11/TS04 CSR EQUIVALENT=TSDB
0900 1886 DSBINT ;DISABLE INTERRUPTS
1A 64 A5 05 E0 0906 1887 BBS #UCBSV_POWER,UCBSW_STS(R5),PWRFL1 ;BR IF POWERFAILED
64 00BE C5 B0 090B 1888 MOVW UCBSW_MS_TSPT3(R5),(R4) ;LOAD THE COMMAND POINTER
0910 1889 WFIKPC MSTMOT,UCBSL_MS_TIMEOUT(R5)
091C 1890 IOFORK ;MAKE IT FORK FIRST
011F 31 0922 1891 BRW XTC ;PROCESS TERMINATION CODE
0925 1892
0925 1893 ;
0925 1894 ; HERE, TREAT POWERFAIL AS TIMEOUT
0925 1895 ;
0925 1896
0925 1897 PWRFL1:
0925 1898 ENBINT ;ENABLE INTERRUPTS
02DD 31 0928 1899 BRW MSTM01 ;GOTO TIMEOUT ROUTINE
092B 1900
092B 1901 ; PXFR - CONSTRUCT COMMAND PACKET FOR DATA TRANSFER COMMANDS:
092B 1902 ; READ NEXT(FORWARD), READ PREVIOUS(REVERSE), REREAD PREVIOUS(SPACE REV, READ
092B 1903 ; FWD), REREAD NEXT(SPACE FWD, READ REV), WRITE DATA, WRITE DATA RETRY,
092B 1904 ; AND WRITE SUBSYSTEM MEMORY.
092B 1905
092B 1906 PXFRRD: ; REREAD COMMANDS ENTER HERE.
009A C5 09 E1 092B 1907 BBC #IOSV_OPPOSITE, - ; Branch if opposite bit not set.
05 0930
0931 1908 UCBSW_FUNC(R5), 10$
61 2000 8F A8 0931 1909 BISW #MS_CPHD_M_OPP, - ; If its set, propogate it to the
0936 1910 MS_CPHD(R1) ; command header.
0A 11 0936 1911 10$: BRB PXFR ; Then rejoin common code.
0938 1912
0938 1913 PXFR:
68 A5 20 A8 0938 1914 BISW #UCBSM_MS_RDPR,UCBSW_DEVSTS(R5) ;FLAG BUFFERED DATAPATH
093C 1915 REQDPR ;REQUEST DATAPATH
0942 1916 PXFRR: ;*ENTRY PT FOR READ REVERSE*
0942 1917 ;**WHICH USES DIRECT DATA PATH**
0942 1918 REQMPR ;REQUEST MAP REGISTER
0948 1919 LOADUBAA ;LOAD MAP REGISTER
50 7C A5 3C 094E 1920 MOVZWL UCBSW_BOFF(R5),R0 ;GET BYTE OFFSET
51 24 A5 D0 0952 1921 MOVL UCBSL_CRB(R5),R1 ;GET CRB
09 34 A1 F0 0956 1922 INSV CRBSL_INTD+VECSW_MAPREG(R1),#9,#9,R0 ;INSERT HGH 9 BITS
095A
51 50 09 09 095A
51 00B6 C5 D0 095C 1923 MOVL UCBSL_MS_TSPT1(R5),R1 ;GET COMMAND PACKET ADDR.
02 A1 50 D0 0961 1924 MOVL R0,MS_BACT(R1) ;STORE XFR ADDR.
06 68 A5 01 E0 0965 1925 BBS #UCBSV_MS_SWAP,UCBSW_DEVSTS(R5),12$ ;BR IF INDUSTRI.COMP.
096A 1926 ; (SET BY SETCHAR COMMAND)
009A C5 08 E1 096A 1927 BBC #IOSV_SWAP,UCBSW_FUNC(R5),15$ ;SWAP BIT SET??

```

```

05 096F
61 1000 8F A8 0970 1928 12$: BISW #MS_CPHD_M_SWB,MS_CPHD(R1) ;YES, SET IT IN CMD HEADER
06 A1 7E A5 B0 0970 1929
06 A1 7E A5 B0 0975 1930 15$: MOVW UCBSW_BCNT(R5),MS_CNT(R1) ;STORE BYTE COUNT
00C4 C5 B0 097A 1932 MOVW UCBSW_BCNT(R5),UCBSW_MS_XC(R5) ;COPY BYTE COUNT
097D
4B 64 A5 05 E0 0980 1933 DSBINT ;DISABLE INTERRUPTS
64 00BE C5 B0 0986 1934 BBS #UCBSV_POWER,UCBSW_STS(R5),20$ ;BR IF POWERFAILED
098B 1935 MOVW UCBSW_MS_TSPT3(R5),(R4) ;LOAD THE COMMAND POINTER
0990 1936 WFIKPCB MSTMO,#^D20 ;TIMEOUT FOR 20. SEC.'S
099A 1937 IOFORK
00000000 GF 16 09A0 1938 JSB G^IOC$PURGDATAP ;PURGE DATAPATH
05 50 E8 09A6 1939 BLBS R0,18$ ;BR IF NO ERROR
00C7 C5 01 90 09A9 1940 MOVW #1,UCBSB_MS_PER(R5) ;MARK PURGE ERROR
26 64 A5 05 E0 09AE 1941 18$: BBS #UCBSV_POWER,UCBSW_STS(R5),21$ ;BR IF POWERFAILED
00FC C5 A2 09B3 1943 ;** WANT TO SAVE DATAPATH REG. & NO. & ETC. FOR ERRLOG
00C4 C5 09B3 1944 SUBW UCBSW_MS_RBPC(R5),UCBSW_MS_XC(R5) ;GET BYTES ACTUALLY XFERRED
00C8 C5 51 D0 09BA 1945 MOVL R1,UCBSL_MS_DPR(R5) ;GET DATA PATH REGISTER
37 A3 90 09BF 1946 MOVW CRBSL_INTD+VECSB_DATAPATH(R3),-
00C6 C5 09C2 1947 UCBSB_MS_DPN(R5) ;GET DATA PATH NO.
51 7C A5 3C 09C5 1948 MOVZWL UCBSW_BOFF(R5),R1 ;GET BYTE OFFSET
00C6 C5 95 09C9 1949 TSTB UCBSB_MS_DPN(R5) ;DATAPATH NO.=0(DIRECT)?
51 00FC C5 12 09CD 1950 BNEQ 22$ ;NO
0B 11 A0 09CF 1951 ADDW UCBSW_MS_RBPC(R5),R1 ;YES, REVERSE OPERATION
09D4 1952 BRB 24$ ;SKIP AROUND
09D6 1953
09D6 1954 ;
09D6 1955 ; HERE,TREAT POWERFAIL AS TIMEOUT
09D6 1956 ;
09D6 1957
09D6 1958 20$: ENBINT ;ENABLE INTERRUPTS
09D9 1960 21$: BRW MSTMO
0215 31 09D9 1961
09DC 1962
09DC 1963 22$: ADDW UCBSW_MS_XC(R5),R1 ;GET FINAL ADDR. OFFSET
51 00C4 C5 A0 09DC 1964
09E1 1965 24$: ASHL #<-9>,R1,R1 ;MODULO 512
51 F7 8F 78 09E1 1966
09E5
50 34 A3 3C 09E6 1967 MOVZWL CRBSL_INTD+VECSW_MAPREG(R3),R0 ;MOVE START MAP REG. NO.
51 50 C0 09EA 1968 ADDL R0,R1 ;R1 HAS FINAL MAP REG. NO.
09ED 1969 ;**R2 HAS START ADDR. OF 1ST MAP REGISTER
09ED 1970 ;**R3 HAS CRB ADDRESS
09ED 1971 ;** WHICH WERE GOTTEN FROM IOC$PURGDATAP
51 01EF 8F B1 09ED 1972 CMPW #495,R1 ;LEGAL MAP REG. NO.?
1A 14 09F2 1973 BGTR 25$ ;BR IF YES
51 01EF 8F 3C 09F4 1974 MOVZWL #495,R1 ;MAKE IT FINAL MAP REG. NO.=495
00CC C5 6241 D0 09F9 1975 MOVL (R2)[R1],UCBSL_MS_FMPR(R5) ;SAVE FINAL MAP REG
FC A241 D0 09FF 1976 MOVL -4(R2)[R1],UCBSL_MS_PMPR(R5) ;SAVE PREVIOUS MAP REG
00D0 C5 0A03
00D4 C5 6241 D0 0A06 1977 MOVL (R2)[R1],UCBSL_MS_NMPR(R5) ;SAVE NEXT MAP REG(=495)
1E 11 0A0C 1978 BRB 27$ ;GO AHEAD NOW
0A0E 1979 25$: ;LEGAL MAP REG. NO.

```

```

00CC C5 6241 D0 0A0E 1980      MOVL (R2)[R1],UCB$$_MS_FMPR(R5) ;SAVE FINAL MAP REG
00D0 C5 01   CE 0A14 1981      MNEGL #1,UCB$$_MS_PMPR(R5) ;ASSUME NO PREVIOUS MAP REG
      51 50   D1 0A19 1982      CMPL  R0,R1 ;WAS FINAL MAP REG. 1ST MAP REG
      07   D3 0A1C 1983      BEQL  26$ ;YES, THERE WAS NO PREVIOUS ONE
FC A241 D0 0A1E 1984      MOVL -4(R2)[R1],UCB$$_MS_PMPR(R5) ;OTHERWISE,SAVE PREVIOUS MAP REG
00D0 C5      0A22
      04 A241 D0 0A25 1985 26$: MOVL  4(R2)[R1],UCB$$_MS_NMPR(R5) ;SAVE NEXT MAP REG
00D4 C5      0A25 1986
      0A29
      00C6 C5 95 0A2C 1987 27$: TSTB  UCB$$_MS_DPN(R5) ;TEST IF DIRECT DATAPATH?
      0A   13 0A30 1989      BEQL  30$ ;YES
      0A32 1990      ;*RELEASE DATAPATH AND MAP REG.'S
      0A32 1991      RELDPR ;RELEASE DATA PATH
68 A5 20 AA 0A38 1992      BICW  #UCB$$_MS_RDPR,UCB$$_DEVSTS(R5) ;CLEAR FLAG
      0A3C 1993 30$:
      0A3C 1994      RELMPR ;RELEASE MAP REGISTERS
      0A42 1995
      OD 11 0A42 1996      BRB   XTC1 ;
      0A44 1997
      0A44 1998
      0A44 1999
      0A44 2000 : XTC - PROCESS TERMINATION CODE
      0A44 2001 : HERE, THE FINAL STATUS CODE IS PUT IN R0, &
      0A44 2002 : THE DRIVE STATUS IS RECORDED INTO UCB
      0A44 2003
      0A44 2004
      0A44 2005 XTC:
      00C4 C5 B5 0A44 2006      TSTW  UCB$$_MS_XC(R5) ;SHOULD ANYTHING XFERRED?
      07   13 0A48 2007      BEQL  XTC1 ;NO, BRANCH
00FC C5 A2 0A4A 2008      SUBW  UCB$$_MS_RBPC(R5),UCB$$_MS_XC(R5) ;GET ACTUALLY XFERRED
00C4 C5 0A4E
      04 00FE C5 E1 0A51 2009 XTC1: BBC  #MS_XSRO_V MOT,- ;DID TAPE MOVE?
      46 A5 07 AA 0A53 2011      UCB$$_MS_XSRO(R5),7$ ;BR IF NO
      0A57 2012      BICW  #<MTSM BOT!- ;CLEAR BOT, AND
      0A5B 2013      MTSM_EOF!- ;END OF FILE,
      0A5B 2014      MTSM_EOT>a-16,UCB$$_MS_DEVDEPEND+2(R5) ;END OF TAPE
      0A5B 2015 7$:
      50 03 01 EF 0A5B 2016      EXTZV #MS_TSSR_V TCC,#MS_TSSR_S_TCC,- ;EXTRACT TERMINATION CODE
      00C2 C5 0A5E 2017      UCB$$_MS_TSSR(R5),R0 ;INTO R0
      0A62 2018      CASE  R0,<- ;DISPATCH TO ROUTINES
      0A62 2019      100$,- ;NORMAL TERMINATION
      0A62 2020      110$,- ;ATTENTION CONDITION
      0A62 2021      120$,- ;TAPE STATUS SLERT
      0A62 2022      130$,- ;FUNCTION REJECT
      0A62 2023      140$,- ;RECOVERABLE ERROR(TAPE MOVED)
      0A62 2024      150$,- ;RECOVERABLE ERROR(TAPE NOT MOVED)
      0A62 2025      160$,- ;UNRECOVERABLE ERROR(TAPE POSI LOST)
      0A62 2026      170$,- ;FATAL CONTROLLER ERROR
      0A62 2027      >
      0A76 2028 :
      0A76 2029 : FATAL CONTROLLER ERROR(TCC=7)
      0A76 2030 :
      0A76 2031 :
      50 0054 8F 3C 0A76 2032 170$: MOVZWL #SS$$_CTRLERR,R0 ;PUT IN FINAL STATUS CODE
      0A76 2033

```

TSDRIVER V04-001

```

FD16 31 0A7B 2034 BRW FATALERR ;GOTO FATAL ERROR
      0A7E 2035
      0A7E 2036 :
      0A7E 2037 : UNRECOVERABLE ERROR(TAPE POSITION LOST)
      0A7E 2038 : (TCC=6)
      0A7E 2039
      0A7E 2040 160$:
46 A5 10 AB 0A7E 2041 BISW #<MTSM LOST@-16>,UCBSL_DEVDEPEND+2(R5) ;MARK POSITION LOST
50 00BC 8F 3C 0A82 2042 MOVZWL #SS$ DRVERR,R0 ;PUT IN FINAL STATUS CODE
      06 E0 0A87 2043 BBS #MS_XSRO_V_ONL,- ;CHECK IF ON-LINE
05 00FE C5 0A89 2044 UCBSW_MS_XSRO(R5),165$ ;BR IF YES
50 01A4 8F 3C 0A8D 2045 MOVZWL #SS$_MEDOFFL,R0 ;NO, RETURN MEDIUM OFFLINE
      FCFF 31 0A92 2046 165$:
      0A92 2047 BRW FATALERR ;
      0A95 2048
      0A95 2049 :
      0A95 2050 : NORMAL TERMINATION
      0A95 2051 : (TCC=0)
      0A95 2052
      50 01 3C 0A95 2053 100$: ;
      0A95 2054 MOVZWL #SS$_NORMAL,R0 ;PUT IN STATUS CODE
009C C5 02 C0 0A98 2055 101$:
009C D5 17 0A98 2056 ADDL #2,UCBSL_DPC(R5) ;ADJUST TO CORRECT RETURN ADDRESS
      0A9D 2057 JMP @UCBSL_DPC(R5) ;RETURN TO DRIVER
      0AA1 2058
      0AA1 2059 :
      0AA1 2060 : ATTENTION CONDITION
      0AA1 2061 : DRIVE HAS UNDERGONE STATUS CHANGE SUCH AS GOING OFFLINE OR COMING ONLINE
      0AA1 2062 : (TCC=1)
      0AA1 2063
      06 00FE 06 E1 0AA1 2064 110$:
      00FE C5 0A03 2065 BBC #MS_XSRO_V_ONL,- ;CHECK IF ONLINE?
50 05 3C 0AA7 2066 UCBSW_MS_XSRO(R5),112$ ;BR IF OFFLINE.
      00B3 31 0AAA 2067 MOVZWL #TCC_REN,R0 ;BECOME ONLINE, BUT
      0AAD 2068 ;SHOULDN'T HAVE BEEN OFFLINE
50 0093 C5 90 0AAD 2069 112$: ERW 150$ ;RETRY THE COMMAND
50 05 01 91 0AAB 2070
50 01A4 8F 13 0AAB 2071 POVB UCBSB_CEX(R5),R0 ; GET HARDWARE COMMAND INDEX
      FCD5 31 0AAB 2072 CMPB #CDHC_UNL,R0 ;WAS IT UNLOAD?
      0AB2 2073 BEQL 100$ ;YES, ITS OK
50 01A4 8F 3C 0AB7 2074 MOVZWL #SS$ MEDOFFL,R0 ;MARK AS MEDIUM OFFLINE
      FCD5 31 0ABC 2075 BRW FATALERR ;GOTO FATAL ERROR
      0ABF 2076
      0ABF 2077 :
      0ABF 2078 : TAPE STATUS ALERT
      0ABF 2079 : (BITS OF INTEREST: TMK, LET, RLS, EOT, RIB, AND RLL)
      0ABF 2080 : **LET BIT IS LOGICAL END OF TAPE FOR DOS, NOT USED FOR NOW**
      0ABF 2081 : (TCC=2)
      0ABF 2082
      0ABF 2083 : The reverse into BOT must return the status SS$ NORMAL because
      0ABF 2084 : at this time BACKUP depends on this fact and that is how the
      0ABF 2085 : other tape drivers work. This has been modified again. So that the
      0ABF 2086 : read reverse which BAKCUP doesn't depend on returns SS$_ENDOFFILE
      0ABF 2087 : when it encounters the BOT marker.
      0ABF 2088
      00 E1 0ABF 2089 120$:
      0ABF 2090 BBC #MS_XSR3_V_RIB,- ;REVERSE INTO BOT?
  
```

T
P

P
I
C
P
S
P
C
A
T
I
T
S

M
-
T
2
T
M

```

16 0104 C5      OAC1 2091      UCBSW MS_XSR3(R5),121$ ;
46 A5 01      AB  OAC5 2092      BISW  #<MTSM BOT@-16>,UCBSL_DEVDEPEND+2(R5) ;YES
0092 00B0 C5   D4  OAC9 2093      CLRL  UCBSL_RECORD(R5)
C5  OF 91  OACD 2094      CMPB  #CDHC_RDP,UCBSB_FEX(R5) ; Is this a read reverse?
50 0870 8F 3C  OAD2 2095      BNEQ  100$ ; If NEQ then return NORMAL code
      C1 12  OAD4 2096      MOVZWL #SS$_ENDOFFILE,RO ; Take error return.
      BD 11  OAD9 2097      BRB   101$
      JADB 2098
      OADB 2099 121$:
50 07 00FE C5   E1  OADB 2100      BBC   #MS_XSRO_V_RLL,- ;CHECK IF RECORD LENGTH LONG?
      0838 8F 3C  OADD 2101      UCBSW MS_XSRO(R5),122$ ;TAKE NORMAL RETURN, IF NOT
      BO 11  OAE1 2102      MOVZWL #SS$_DATAOVERUN,RO ;YES, ITS DATAOVERRUN
      OAE6 2103      BRB   101$ ;TAKE NORMAL RETURN
      OAE8 2104 122$:
      OF  E1  OAE8 2105      BBC   #MS_XSRO_V_TMK,- ;CHECK IF SEE TAPE MARK
21 00FE C5   OAEA 2106      UCBSW MS_XSRO(R5),125$ ; ??
46 A5 02      AB  OAEA 2107      BISW  #<MTSM EOF@-16>,UCBSL_DEVDEPEND+2(R5) ;YES
0092 C5 16 91  OAF2 2108      CMPB  #CDHC_QTM,UCBSB_FEX(R5) ;WAS IT WRITE TMK?
      16 13  OAF7 2109      BEQL  125$ ;YES, LOOK FOR EOT
0092 C5 05 91  OAF9 2110      CMPB  #CDHC_STR,UCBSB_FEX(R5) ;WAS IT SKIPFILE REVERSE?
      21 13  OAFE 2111      BEQL  128$ ;YES
0092 C5 02 91  OB00 2112      CMPB  #CDHC_STF,UCBSB_FEX(R5) ;WAS IT SKIPFILE FORWARD?
      1A 13  OB05 2113      BEQL  128$ ;YES
50 0870 8F 3C  OB07 2114      MOVZWL #SS$_ENDOFFILE,RO ;**NOTE UCBSL RECORD WAS ADJUSTED**
      FF89 31  OB0C 2115      BRW   101$ ;SET EOF
      OB0F 2116      ;TAKE NORMAL RETURN
      OB0F 2117 125$:
      OC 00FE C5   E1  OB0F 2118      BBC   #MS_XSRO_V_EOT,- ;CHECK IF AT EOT?
46 A5 04      AB  OB11 2119      UCBSW MS_XSRO(R5),128$ ;
50 0878 8F 3C  OB15 2120      BISW  #<MTSM EOT@-16>,UCBSL_DEVDEPEND+2(R5) ;YES, SET FLAG
      FF77 31  OB19 2121      MOVZWL #SS$_ENDOFFTAPE,RO ;WRITE ERROR INTO EOT
      OB1E 2122      BRW   101$ ;
      OB21 2123 128$:
      FF71 31  OB21 2124      BRW   100$ ;ANYTHING ELSE?
      OB24 2125      ;TAKE NORMAL RETURN**TEMP**
      OB24 2126 ;
      OB24 2127 ; FUNCTION REJECT
      OB24 2128 ; (BITS OF INTEREST:BOT,WLK,VCK,ONL,ILA,ILC,NEF,WLE)
      OB24 2129 ; (TCC=3)
      OB24 2130 ;
50 008C 8F 3C  OB24 2131 130$:
      01 E1  OB24 2132      MOVZWL #SS$_DRVERR,RO ;MARK AS DRIVE ERROR
      08 00FE C5   OB29 2133      BBC   #MS_XSRO_V_BOT,- ;CHECK IF AT BOT
46 A5 01      AB  OB2B 2134      UCBSW MS_XSRO(R5),132$ ;
      00B0 C5   D4  OB2F 2135      BISW  #<MTSM BOT@-16>,UCBSL_DEVDEPEND+2(R5) ;YES
      OB33 2136      CLRL  UCBSL_RECORD(R5)
      OB37 2137 132$:
      06 00FE C5   E1  OB37 2138      BBC   #MS_XSRO_V_VCK,- ;WAS VOLUME CHECK?
      1000 8F  AB  OB39 2139      UCBSW MS_XSRO(R5),134$ ;
      68 A5      OB3D 2140      BISW  #UCBSM_MS_VCK,UCBSW_DEVSTS(R5) ;YES,RECORD IT
      OB41 2141 134$:
      OB43 2142      BBC   #MS_XSRO_V_WLE,- ;CHECK IF WRITE LOCK ERROR
      09 00FE C5   E1  OB43 2143      UCBSW MS_XSRO(R5),136$ ;
46 A5 08      AB  OB45 2144      BISW  #<MTSM WOL@-16>,UCBSL_DEVDEPEND+2(R5) ;YES, SET FLAG
50 025C 8F 3C  OB49 2145      MOVZWL #SS$_WRITLCK,RO ;MARK AS WRITE-LOCKED ERROR
      OB52 2146 136$:

```



```

05 00FE 06  E0 0B52 2147      BBS      #MS_XSRO_V_ONL,-      ;CHECK IF ONLINE
50 01A4 8F  3C 0B54 2148      UCBSW_MS_XSRO(R5),138$ ;BR IF YES
      0B58 2149      MOVZWL  #SS$_MEDOFFL,R0    ;MARK MEDIUM OFFLINE
      FC34 31 0B5D 2150 138$: ;
      0B5D 2151      BRW      FATALERR      ;TAKE FATAL OR HARD ERROR RETURN
      0B60 2152      ;
      0B60 2153      ;
      0B60 2154      ; RECOVERABLE ERROR(TAPE MOVED)
      0B60 2155      ; RECOVERABLE ERROR(TAPE NOT MOVED)
      0B60 2156      ; (TCC=4 OR 5)
      0B60 2157      ;
      0B60 2158 140$:      ;
      0B60 2159      ;
      0B60 2160 150$:      ;
009A C5 0F  E0 0B60 2161      BBS      #IOSV_INHRETRY,UCBSW_FUNC(R5),155$ ;IF SET, RETRY INHIBITED
      13 0B65      ;
7E 009C D5 32 0B66 2162      CVTWL  @UCBSL_DPC(R5),-(SP) ;GET BRANCH DISPLACEMENT
009C C5 8E  C0 0B6B 2163      ADDL   (SP)+,UCBSL_DPC(R5) ;CALCULATE RETURN ADDRESS -2
009C C5 02  C0 0B70 2164      ADDL   #2,UCBSL_DPC(R5) ;ADJUST TO CORRECT RETURN ADDRESS
      009C D5 17 0B75 2165      JMP    @UCBSL_DPC(R5) ;RETURN TO DRIVER
      FEFA 31 0B79 2166      ;
      0B79 2167 155$: BRW 170$ ;RETURN AS FATAL
      0B7C 2168      ;
    
```

```

OB7C 2170 .SBTTL TS11/TS04 INTERRUPT SERVICE ROUTINE
OB7C 2171 :+
OB7C 2172 : TSSINT - TS11/TS04 MAGTAPE INTERRUPTS
OB7C 2173 :
OB7C 2174 : THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT OCCURS
OB7C 2175 : ON TS11/TS04 CONTROLLER. THE STATE OF THE STACK ON ENTRY IS:
OB7C 2176 :
OB7C 2177 :         00(SP) = ADDR. OF IDB ADDRESS
OB7C 2178 :         04-28(SP) = SAVED R0-R5
OB7C 2179 :         32(SP) = INTERRUPT PC
OB7C 2180 :         36(SP) = INTERRUPT PSL
OB7C 2181 :
OB7C 2182 : INTERRUPT DISPATCHING OCCURS AS FOLLOWS:
OB7C 2183 :
OB7C 2184 : (MUMBLE)
OB7C 2185 :
OB7C 2186 : -
OB7C 2187 :
OB7C 2188 TSSINT::
53 9E D0 OB7C 2189      MOVL    @ (SP)+,R3          ;GET ADDR. OF IDB
54 63 7D OB7F 2190      MOVQ    IDB$CSR(R3),R4        ;GET CONTROLLER CSR AND UCB ADDR.
50 00B6 C5 D0 OB82 2191      MOVL    UCB$MS_TSPT1(R5),R0      ;COMMAND PACKET ADDR. IN R0
00C0 C5 64 B0 OB87 2192      MOVW    (R4),UCB$MS_TSBA(R5) ;GET DEVICE REGISTER TSBA(TSDB)
      02 A4 B0 OB8C 2193      MOVW    2(R4),UCB$MS_TSSR(R5) ;GET TSSR INTO UCB
      00C2 C5 OB8F
23 64 A5 01 E5 OB92 2194      BBCC    #UCB$V_INT,UCB$MS_STS(R5),10$ ;IF CLR, INTERRUPT NOT EXPECTED
      10 A0 B0 OB97 2195      MOVW    MS_MHD(R0),UCB$MS_MHD(R5) ;SAVE MSG BUFFER IN UCB
      00F8 C5 OB9A
      12 A0 D0 OB9D 2196      MOVL    MS_LNH(R0),UCB$MS_LNH(R5) ;SAVE NEXT LONG WORD
      00FA C5 OBA0
      16 A0 7D OBA3 2197      MOVQ    MS_XSRO(R0),UCB$MS_XSRO(R5) ;SAVE REST OF MSG BUFFER
      00FE C5 OBA6
53 10 A5 D0 OBA9 2198      MOVL    UCB$FR3(R5),R3          ;RESTORE REMAINING DRIVER CONTEXT
      0C B5 16 OBAD 2199      JSB    @UCB$_FPC(R5)          ;CALL DRIVER
      50 8E 7D OBB0 2200 5$:      MOVQ    (SP)+,R0              ;RESTORE REGISTERS
      52 8E 7D OBB3 2202      MOVQ    (SP)+,R2              ;
      54 8E 7D OBB6 2203      MOVQ    (SP)+,R4              ;
      02 OB89 2204      REI                          ;RETURN FROM INTERRUPT
      OBBA 2205
      OBBA 2206 :
      OBBA 2207 : NON-QIO RESPONSE INTERRUPT
      OBBA 2208 :
      OBBA 2209
      OBBA 2210 10$:
02 68 A5 0A E4 OBBA 2211      BBSC    #UCB$V_MS_LBA,UCB$MS_DEVSTS(R5),20$ ;YES. LOADING BUFFER ADDR.?
      EF 11 OBBF 2212      BRB    5$                    ; Branch to dismiss interrupt.
      OBC1 2213
      OBC1 2214 :
      OBC1 2215 : HERE, WAS LOADING BUFFER ADDRESS
      OBC1 2216 :
      OBC1 2217 20$:
      OBC1 2218      BBS    #MS_TSSR_V_NBA,-
E9 00C2 C5 E0 OBC3 2219      UCB$MS_TSSR(R5),5$ ;FAIL TO LOAD BUFFER ADDR.
      OBC7 2220 :
      OBC7 2221 : BUFFER ADDRESS LOADED SUCCESSFULLY
      OBC7 2222 : DO RELEASE MESSAGE BUFFER TO TS11/TS04
    
```

```

                                OBC7 2223 ;
                                OBC7 2224 30$:
                                OBC7 2225 ;**MUST BE AT BOT
1A 46 A5 01 A8 OBC7 2226 BISW #<MTSM_BOT@-16>,UCB$L DEVDEPEND+2(R5) ;MARK IT
15 64 A5 05 E1 OBCB 2227 BBC #UCB$V_POWER,UCB$W_STS(R5),35$ ;BR IF NOT POWERFAIL
09 68 A5 0B E0 OBD0 2228 BBS #UCB$V_MS_RPI,UCB$W_DEVSTS(R5),35$ ;BR IF REPOSITION IN PROGRESS
00D0 C5 D0 OBDA 2229 BBC #UCB$V_MS_SWE,UCB$W_DEVSTS(R5),34$ ;BR IF NOT SOFTWARE EMULATION
00F4 C5 OBDE MOVL UCB$L_MS_PMPR(R5),UCB$L_MS_TPOSITN(R5) ;GET FROM ELSEWHERE
                                07 11 OBE1 2231 BRB 35$
                                OBE3 2232 34$:
00B0 C5 D0 OBE3 2233 MOVL UCB$L_RECORD(R5),UCB$L_MS_TPOSITN(R5) ;SAVE TAPE POSITION
00F4 C5 OBE7
                                OBEA 2234 35$:
00B0 C5 D4 OBEA 2235 CLRL UCB$L_RECORD(R5) ;
                                OBEE 2236 ; WHICH TELL TAPE POSITION
                                OBEE 2237 BRW 5$ ;DO RETURN FROM INTERRUPT
                                OBF1 2238
                                OBF1 2239
```

```

OBF1 2241 .SBTTL TIMEOUT HANDLER
OBF1 2242 :+MSTMO - HANDLES TIME-OUT WHEN TS11/TS04 DOES NOT INTERRUPT AFTER
OBF1 2243 : A HARDWARE COMMAND ISSUED FOR A SPECIFIED PERIOD OF TIME.
OBF1 2244 : THE ROUTINE DEALLOCATES DATA PATH AND MAP REGISTER IF IT'S DATA
OBF1 2245 : TRANSFER COMMAND, AND ABORTS THE I/O OPERATION.
OBF1 2246 : IF IT WAS DUE TO POWERFAIL, REPOSITIONING IS ATTEMPTED, AND
OBF1 2247 : THE TIME-OUTED IRP IS RE-ISSUED
OBF1 2248 :
OBF1 2249 : INPUT:
OBF1 2250 :
OBF1 2251 : OUTPUT:
OBF1 2252 :
OBF1 2253 .ENABL LSB ;ENABLE LOCAL SYMBOL
OBF1 2254 MSTMO: SETIPL UCBSB_FIPL(R5) ;LOWER IPL TO DEVICE FORK LEVEL
OBF1 2255 :**ASSUME NO PURGING OF DATAPATH FOR TIMEOUT**
OBF5 2256 BBCC #UCBSV_MS_RDPR,UCBSW_DEVSTS(R5),1$ ;BR IF DATAPATH NOT REQUESTED
06 68 A5 05 E5 OBF5 2257 RELDPR ;RELEASE DATA PATH
OBF1 2258 1$:
OC00 2259 RELMPR ;RELEASE MAP REGISTERS
04 11 OC06 2260 BRB 2$
OC08 2261 :
OC08 2262 : TIMEOUT FOR NON-I/O XFR OPERATION
OC08 2263 :
OC08 2264 :
OC08 2265 :
OC08 2266 MSTMO1:
OC08 2267 SETIPL UCBSB_FIPL(R5) ;LOWER IPL TO DEVICE FORK LEVEL
OC0C 2268 2$:
03 05 E4 OC0C 2269 BBSC #UCBSV POWER,- ; Branch around to reposition if
03 64 A5 00CF 31 OC0E 2270 UCBSW_STS(R5),5$ ; we had POWERFAIL.
OC11 2271 BRW 90$
OC14 2272 :
OC14 2273 : HERE, CHECK DRIVE OFF-LINE UNLOADED OR NOT
OC14 2274 :
OC14 2275 :
OC14 2276 5$:
F5E9 30 OC14 2277 BSBW TEST_NBA ; Test to assure we DON'T need TS11
OC17 2278 ; message buffer address loaded.
03 50 E8 OC17 2279 BLBS R0,6$ ; LBS implies TS11 READY and able.
FB77 31 OC1A 2280 BRW FATALERR ; If TS11 not ready, we can't even
OC1D 2281 ; try to reposition.
OC1D 2282 6$:
00000000'GF 16 OC1D 2283 JSB G^EXESREAD_TODR ;GET CURRENT TIME OF DAY
000005DC 8F C0 OC23 2284 ADDL #1500,R0 ;ADD 15 SEC. TO WAIT
OC29 2285
7C A5 50 D0 OC2A 2285 MOVL R0,UCBSW_BOFF(R5) ;STORE IT IN UCB
OC2E 2286 :
OC2E 2287 : HERE, GET TS11'S CSR EQUIVALENT INTO R4
OC2E 2288 :
OC2E 2289 :
OC2E 2290 7$:
OC2E 2291 DSBINT ;DISABLE INTERRUPTS
OC34 2292 WFIKPC 8$,#2 ;WAITFOR INTERRUPT OR TIMEOUT
OC3E 2293 IOFORK ;
OC44 2294 8$:
OC44 2295 SETIPL UCBSB_FIPL(R5) ;LOWER IPL TO FORK LEVEL
OC48 2296 EXHC FAIL,RC_RWD ;DO A REWIND
    
```

```

0020 31 0C50 2297 BRW 9$ ;BR IF SUCCESS=>DRIVE ONLINE
      0C53 2298 FAIL:
      0C53 2299
      0C53 2300 : HERE, TO SEND MESSAGE TO OPERATOR TO INFORM DRIVE OFFLINE
      0C53 2301
      0C53 2302
00000000'GF 16 0C53 2303 JSB G*EXESREAD TODR ;GET CURRENT TIME OF DAY
7C A5 50 D1 0C59 2304 CMLP R0,UCBSW_BOFF(R5) ;15 SEC. PASSED?
CF 1B 0C5D 2305 BLEQU 7$ ;NO , GO TRY AGAIN
      0C5F 2306
      0C5F 2307
54 00'8F 9A 0C5F 2308 MOVZBL #MSG$ DEVOFFLIN,R4 ;SET MESSAGE NUMBER
00000000'GF 9E 0C63 2309 MOVAB G*SY$SGL_OPRMBX,R3 ;GET ADDRESS OF OPERATOR MAILBOX
53
00000000'GF 16 0C6A 2310 JSB G*EXESSNDEVMSG ;SEND MESSAGE TO OPERATOR
FFAA 31 0C70 2311 BRW 6$
      0C73 2312
      0C73 2313
      0C73 2314 : OTHERWISE DO REPOSITIONING TAPE
      0C73 2315
      0C73 2316 9$:
0800 8F AB 0C73 2317 BISW #UCBSM_MS_RPI,UCBSW_DEVSTS(R5) ;FLAG REPOSITION IN PROGRESS
68 A5 0C77
      0C79 2318 EXHC 50$,HC_RWD ;DO REWIND 1ST
      0C81 2319 ; & CLEAR UCBSL_RECORD
      0C81 2320 10$:
00B0 C5 D1 0C81 2321 CMLP UCBSL_RECORD(R5),UCBSL_MS_TPOSITN(R5) ;CHECK REPOSITIONING
00F4 C5 0C85
46 13 0C88 2322 BEQL 80$ ;BR IF YES
00B0 C5 D1 0C8A 2323 CMLP UCBSL_RECORD(R5),UCBSL_MS_TPOSITN(R5) ;IS IT GTR THAN
00F4 C5 0C8E
33 14 0C91 2324 BGTR 50$ ;BR IF YES
00B0 C5 C3 0C93 2325 SUBL3 UCBSL_RECORD(R5),UCBSL_MS_TPOSITN(R5),R0 ;GET WHAT'S LEFT
50 00F4 C5 0C97
00007FFF 8F D1 0C9B 2326 CMLP #*X7FFF,R0 ;LESS THAN 32,768?
50 0CA1
09 14 0CA2 2327 BGTR 20$ ;BR IF YES
7FFF 8F B0 0CA4 2328 MOVW #*X7FFF,UCBSW_MS_SPACNT(R5) ;SKIP 32,768 RECORDS TILL DONE
00B4 C5 0CAB
05 11 0CAB 2329 BRB 30$
00B4 C5 50 B0 0CAD 2330 20$:
0CAD 2331 MOVW R0,UCBSW_MS_SPACNT(R5) ;SKIP WHAT'S LEFT
0CB2 2332 30$:
0CB2 2333 EXHC 50$,HC_SRF
51 00C4 C5 3C 0CBA 2334 MOVZWL UCBSW_MS_XC(R5),R1 ;GET NO. OF RECORDS PASSED
00B0 C5 51 C0 0CBF 2335 ADDL R1,UCBSL_RECORD(R5) ;UPDATE TAPE POSITION
BB 11 0CC4 2336 BRB 10$ ;GO BACK
0CC6 2337 50$:
0800 8F AA 0CC6 2338 BICW #UCBSM_MS_RPI,UCBSW_DEVSTS(R5) ;CLEAR FLAG, REPOSI. FAILED
68 A5 0CCA
FABF CF 17 0CCC 2339 JMP FATALERO
0CDO 2340
0CDO 2341 : HERE, GO AHEAD WITH THE CURRENT QIO
0CDO 2342
0CDO 2343 80$:
0800 8F AA 0CDO 2344 BICW #UCBSM_MS_RPI,UCBSW_DEVSTS(R5) ;CLEAR FLAG,REPOSITION DONE
68 A5 0CD4

```

```

53 58 A5 D0 OCD6 2345      MOVL   UCBSL_IRP(R5),R3      ;R3 HAS IRP ADDRESS
78 A5 2C A3 7D OCDA 2346      MOVQ   IRPSL_SVApte(R3),UCBSL_SVApte(R5) ;RESTORE XFER PARAMETERS
    F5D9 CF 17 OCDF 2347      JMP    TS_STARTIO
00000000 GF 16 OCE3 2348 90$: JSB    G*ERL$DEVICTMO      ;LOG TIMEOUT ERROR
50 022C BF 3C OCE9 2350      MOVZWL #SSS_TIMEOUT,R0      ;SET TIMEOUT STATUS
    OCEE 2351      .IF DF TS TRACE      ; Trace final I/O status.
    OCEE 2352      BSBW TRACE_STATUS
    OCEE 2353      .ENDC
    OCEE 2354      REQCOM      ;GO COMPLETE I/O REQUEST PROCESSING
    OCF4 2355
    OCF4 2356      .DSABL LSB
```

```

OCF4 2358 .SBTTL TS11/TS04 REGISTER DUMP ROUTINE
OCF4 2359 :+
OCF4 2360 : TS_REGDUMP - TS11/TS04 REGISTER DUMP ROUTINE
OCF4 2361 :
OCF4 2362 : THIS ROUTINE IS CALLED TO SAVE THE CONTROLLER AND DRIVE REGISTERS IN A
OCF4 2363 : SPECIFIED BUFFER. IT IS CALLED FROM THE DEVICE ERRORLOGGING ROUTINE AND
OCF4 2364 : FROM THE DIAGNOSTIC BUFFER FILL ROUTINE
OCF4 2365 :
OCF4 2366 : INPUT:
OCF4 2367 : R0 = ADDRESS OF REGISTER SAVE BUFFER
OCF4 2368 : R4 = ADDRESS OF CSR (EQUIVALENT)
OCF4 2369 : R5 = UCB ADDRESS
OCF4 2370 :
OCF4 2371 : OUTPUT:
OCF4 2372 :
OCF4 2373 : THE CONTROLLER AND DRIVE REGISTERS ARE SAVED IN THE SPECIFIED BUFFER
OCF4 2374 :-
OCF4 2375 :
OCF4 2376 TS_REGDUMP:
80 80 17 D0 OCF4 2377 MOVL #23,(R0)+ ;23 REGISTERS FOLLOW TO BE DUMPED
80 00C0 C5 3C OCF7 2378 MOVZWL UCBSW_MS_TSBA(R5),(R0)+ ;GET TSBA
80 00C2 C5 3C OCF8 2379 MOVZWL UCBSW_MS_TSSR(R5),(R0)+ ;GET TSSR
80 00C6 C5 9A OD01 2380 MOVZBL UCBSB_MS_DPN(R5),(R0)+ ;GET DATAPATH NO.
80 00C8 C5 D0 OD06 2381 MOVL UCBSL_MS_DPR(R5),(R0)+ ;GET DATAPATH REG.
80 00CC C5 D0 OD0B 2382 MOVL UCBSL_MS_FMPR(R5),(R0)+ ;GET FINAL MAP REGISTER
80 00D0 C5 D0 OD10 2383 MOVL UCBSL_MS_PMPR(R5),(R0)+ ;GET FINAL-1 MAP REGISTER
80 00D4 C5 D0 OD15 2384 MOVL UCBSL_MS_NMPR(R5),(R0)+ ;GET FINAL+1 MAP REGISTER
51 00B6 C5 D0 OD1A 2385 MOVL UCBSL_MS_TSPT1(R5),R1 ;GET MESSAGE BUFFER ADDR
52 0F 9A OD1F 2386 MOVZBL #15,R2 ;15 WORDS IN MSG BUFFER
80 80 81 3C OD22 2387 10$: MOVZWL (R1)+,(R0)+ ;COPY FROM MSG BUFFER
OD25 2388 ;**FROM MS_CPHD TO MS_XSR3, SEE $DEFINI MS**
80 FA 52 F5 OD25 2390 SOBGTR R2,10$ ;LOOP BACK
80 00C7 C5 9A OD28 2391 MOVZBL UCBSB_MS_PER(R5),(R0)+ ;GET PURGE ERROR INDICATOR
05 OD2D 2392 RSB ;
OD2E 2393
OD2E 2394 TS_END: ;ADDRESS OF LAST LOCATION IN DRIVER
OD2E 2395
OD2E 2396 .END
  
```

TSDRIVER
Symbol table

```

SSS = 00000020 R 02
SSOP = 00000002
ACPSACCESS ***** X 03
ACPSDEACCESS ***** X 03
ACPSMODIFY ***** X 03
ACPSMOUNT ***** X 03
ACPSREADBLK ***** X 03
ACPSWRITEBLK ***** X 03
ATS_UBA = 00000001
CDHC_BRL = 00000019
CDHC_CLN = 00000018
CDHC_DRI = 00000004
CDHC_ERS = 00000006
CDHC_GST = 00000015
CDHC_NOP = 00000000
CDHC_PAK = 00000008
CDHC_RDN = 0000000C
CDHC_RDP = 0000000F
CDHC_RPS = 00000013
CDHC_RRN = 00000010
CDHC_RRP = 00000011
CDHC_RWD = 00000003
CDHC_SCH = 00000014
CDHC_SRF = 00000009
CDHC_SRR = 00000007
CDHC_STF = 00000002
CDHC_STR = 00000005
CDHC_UNL = 00000001
CDHC_WCK = 0000000A
CDHC_WDR = 00000012
CDHC_WKR = 0000000D
CDHC_WRC = 0000001B
CDHC_WRD = 0000000E
CDHC_WSM = 0000001A
CDHC_WTM = 00000016
CDHC_WTR = 00000017
CLEAR = 0000726 R 03
CLS_MDE = 00000003
CLS_MDF = 00000001
CLS_ONF = 00000000
CLS_OTHER = 00000001
CLS_PTB = 00000000
CLS_WLN = 00000002
CRBSL_INTD = 00000024
DCS_TAPE = 00000002
DDBSL_ACPD = 00000010
DDBSL_DDT = 0000000C
DEVSM_AVL = 00040000
DEVSM_DIR = 00000008
DEVSM_ELG = 00400000
DEVSM_FOD = 00004000
DEVSM_IDV = 04000000
DEVSM_NNM = 00000200
DEVSM_ODV = 08000000
DEVSM_SDI = 00000010
DEVSM_SQD = 00000020
DEVSV_FOR = 00000018

```

```

DEVSV_MNT = 00000013
DPTSC_LENGTH = 00000038
DPTSC_VERSION = 00000004
DPTSINITAB 00000038 R 02
DPTSREINITAB 00000076 R R 02
DPTSTAB 00000000 R R 02
DRVCLR 00000731 R 03
DTS_TS11 = 00000004
DYN$C_CRB = 00000005
DYN$C_DDB = 00000006
DYN$C_DPT = 0000001E
DYN$C_UCB = 00000010
EMBSL_DV_REGSAV = 0000004E
ERASE 00000705 R 03
ERL$DEVICERR ***** X 03
ERL$DEVICTMO ***** X X 03
EXESALONONPAGED ***** X X 03
EXE$GL_NONPAGED ***** X X 03
EXESIOFORK ***** X X 03
EXESONEPARM ***** X X 03
EXESREAD TODR ***** X X 03
EXESSETMODE ***** X X 03
EXES$NDEVMSG ***** X X 03
EXESZEROPARM ***** X X 03
EXIT_READ_FCNEXT 0000049F R 03
EXIT_READ_RFCNEXT 000004AC R R 03
FAIL 00000C53 R R 03
FATALERO 0000078F R R 03
FATALERR 00000794 R 03
FCC_CPE = 00000001
FCC_IDF = 00000000
FCC_LAP = 00000003
FCC_UPE = 00000002
FCNEXT 000007AA R 03
FDISPATCH 00000374 R 03
FUNCTAB_LEN = 00000088
GETSTS 0000073C R R 03
HCEX 00000804 R R 03
HCTAB 00000038 R 03
HC_BRL = 0000C08A
HC_CLN = 0000C28A
HC_DRI = 0000C08B
HC_ERS = 0000C189
HC_GST = 0000C08F
HC_NOP = 00000000
HC_PAK = 00000000
HC_RDN = 0000C081
HC_RDP = 0000C181
HC_RPS = 00000000
HC_RRN = 0000C381
HC_RRP = 0000C281
HC_RWD = 0000C488
HC_SCH = 00000000
HC_SRF = 0000C088
HC_SRR = 0000C188
HC_STF = 0000C088
HC_STR = 0000C188

```


TSDRIVER
Symbol table

HC_UNL	= 0000C18A	IOCSDIAGBUFILL	*****	X	03
HC_WCK	= 00000000	IOCSLOADUBAMAP	*****	X	03
HC_WDR	= 0000C285	IOCSLOADUBAMAPA	*****	X	03
HC_WKR	= 00000000	IOCSMNTVER	*****	X	03
HC_WRC	= 0000C084	IOCSPURGDATAP	*****	X	03
HC_WRD	= 0000C085	IOCSRELDATAP	*****	X	03
HC_WSM	= 0000C086	IOCSRELMAPREG	*****	X	03
HC_WTM	= 0000C089	IOCSREQCOM	*****	X	03
HC_WTR	= 0000C289	IOCSREQDATAP	*****	X	03
IDBSL_CSR	= 00000000	IOCSREQMAPREG	*****	X	03
IDBSL_OWNER	= 00000004	IOCSRETURN	*****	X	03
IOSM_NOWAIT	= 00000080	IOCSWFIKPCH	*****	X	03
IOSV_INHRETRY	= 0000000F	IRPSL_MEDIA	= 00000038		
IOSV_OPPOSITE	= 00000009	IRPSL_SVAPE	= 0000002C		
IOSV_REVERSE	= 00000006	IRPSL_WIND	= 00000018		
IOSV_SWAP	= 00000008	IRPSS_FCODE	= 00000006		
IOS_ACCESS	= 00000032	IRPSV_FCODE	= 00000000		
IOS_ACPCONTROL	= 00000038	IRPSV_PHYSIO	= 00000008		
IOS_AVAILABLE	= 00000011	IRPSV_VIRTUAL	= 00000004		
IOS_CLEAN	= 0000001E	IRPSW_FUNC	= 00000020		
IOS_CREATE	= 00000033	IRPSW_STS	= 0000002A		
IOS_DEACCESS	= 00000034	LDTSD8	= 00000900	R	03
IOS_DELETE	= 00000035	MASKH	= 00000008		
IOS_DRVCLR	= 00000004	MASKL	= 04000000		
IOS_ERASETAPE	= 00000006	MEDIA_ID_TS11	= 6CE9300B		
IOS_MODIFY	= 00000036	MMG\$GC_SPTBASE	*****	X	03
IOS_MOUNT	= 00000039	MSSDDT	00000000	RG	03
IOS_NOP	= 00000000	MSG\$ DEVOFFLIN	*****	X	03
IOS_PACKACK	= 00000008	MSGREL	00000710	R	03
IOS_READBLK	= 00000021	MSG_ATN	= 00000013		
IOS_READPBLK	= 0000000C	MSG_END	= 00000010		
IOS_READPRESET	= 00000019	MSG_ERR	= 00000012		
IOS_READVBLK	= 00000031	MSG_FAL	= 00000011		
IOS_RECAL	= 00000003	MSG_LOG	= 00000014		
IOS_REREADN	= 00000016	MSTM0	00000BF1	R	03
IOS_REREADP	= 00000017	MSTMJ1	00000C08	R	03
IOS_REWIND	= 00000024	MS_BA1	00000004		
IOS_REWINDOFF	= 00000022	MS_BACT	00000002		
IOS_SENSECHAR	= 0000001B	MS_CHWD	0000000E		
IOS_SENSEMODE	= 00000027	MS_CNT	00000006		
IOS_SETCHAR	= 0000001A	MS_CPHD	00000000		
IOS_SETMODE	= 00000023	MS_CPHD_M_ACK	= 00008000		
IOS_SKIPFILE	= 00000025	MS_CPHD_M_CVC	= 00004000		
IOS_SKIPRECORD	= 00000026	MS_CPHD_M_IE	= 00000080		
IOS_SPACEFILE	= 00000002	MS_CPHD_M_OPP	= 00002000		
IOS_SPACERECORD	= 00000009	MS_CPHD_M_SWB	= 00001000		
IOS_UNLOAD	= 00000001	MS_LNH	00000012		
IOS_VIRTUAL	= 0000003F	MS_LNTH	0000000C		
IOS_WRITECHECK	= 0000000A	MS_MBA0	00000008		
IOS_WRITELBLK	= 00000020	MS_MBA1	0000000A		
IOS_WRITEMARK	= 0000001C	MS_MHD	00000010		
IOS_WRITEOF	= 00000028	MS_RBPC	00000014		
IOS_WRITEPBLK	= 0000000B	MS_TSSR_S_TCC	= 00000003		
IOS_WRITERET	= 00000018	MS_TSSR_V_NBA	= 0000000A		
IOS_WRITEVBLK	= 00000030	MS_TSSR_V_SSR	= 00000007		
IOS_WRTMKR	= 0000001D	MS_TSSR_V_TCC	= 00000001		
IOCSCANCELIO	*****	MS_XSRO	00000016		
	X				03

TSDRIVER
Symbol table

MS_XSRO_V_BOT	= 00000001			SS\$ NORMAL	= 00000001		
MS_XSRO_V_EOT	= 00000000			SS\$ TIMEOUT	= 0000022C		
MS_XSRO_V_MOT	= 00000007			SS\$ VOLINV	= 00000254		
MS_XSRO_V_ONL	= 00000006			SS\$ WRITLCK	= 0000025C		
MS_XSRO_V_RLL	= 0000000C			SYS\$GL_OPRMBX	*****	X	03
MS_XSRO_V_TMK	= 0000000F			TCC_ATN	= 00000001		
MS_XSRO_V_VCK	= 00000004			TCC_FNR	= 00000003		
MS_XSRO_V_WLE	= 0000000B			TCC_FTL	= 00000007		
MS_XSRO_V_WLK	= 00000002			TCC_NML	= 00000000		
MS_XSR1	00000018			TCC_REM	= 00000004		
MS_XSR2	0000001A			TCC_REN	= 00000005		
MS_XSR3	0000001C			TCC_TSA	= 00000002		
MS_XSR3_V_RIB	= 00000000			TCC_USER	= 00000006		
MT\$CHECKR_ACCESS	*****	X	03	TEST_NBA	00000200	R	03
MT\$K_NORMAL15	= 0000000E			TSS\$INT	00000B7C	RG	03
MTSM_BOT	= 00010000			TS_END	00000D2E	R	03
MTSM_EOF	= 00020000			TS_FUNCTABLE	00000070	R	03
MTSM_EOT	= 00040000			TS_INIT	000000F8	R	03
MTSM_HWL	= 00080000			TS_REGDUMP	00000CF4	R	03
MTSM_LOST	= 00100000			TS_STARTIO	000002BC	R	03
MTSS_FORMAT	= 00000004			UCB\$B_CEX	= 00000093		
MTSV_BOT	= 00000010			UCB\$B_DEVCLASS	= 00000040		
MTSV_EOF	= 00000011			UCB\$B_DEVTYPE	= 00000041		
MTSV_FORMAT	= 00000004			UCB\$B_DIPL	= 0000005E		
NOP	000003EC	R	03	UCB\$B_ERTCNT	= 00000080		
PACKACK	000003E6	R	03	UCB\$B_ERTMAX	= 00000081		
PMIS	000008DE	R	03	UCB\$B_FEX	= 00000092		
PNOP	00000876	R	03	UCB\$B_FIPL	= 0000000B		
PPOS	00000882	R	03	UCB\$B_MS_DPN	000000C6		
PR\$ IPL	= 00000012			UCB\$B_MS_PER	000000C7		
PWCH	000008E1	R	03	UCB\$K_LCC_TAPE_LENGTH	= 000000B4		
PWRFL1	00000925	R	03	UCB\$K_MS_CENGTR	= 00000106		
PXFR	00000938	R	03	UCB\$L_CRB	= 00000024		
PXFRR	00000942	R	03	UCB\$L_DEVCHAR	= 00000038		
PXFRRD	0000092B	R	03	UCB\$L_DEVCHAR2	= 0000003C		
READDATA	000003F4	R	03	UCB\$L_DEVDEPEND	= 00000044		
READDATAR	0000044B	R	03	UCB\$L_DPC	= 0000009C		
READPRESET	000003EC	R	03	UCB\$L_FPC	= 0000000C		
REREADN	00000494	R	03	UCB\$L_FR3	= 00000010		
REREADP	00000440	R	03	UCB\$L_IOQFL	= 0000004C		
RET	00000876	R	03	UCB\$L_IRP	= 00000058		
REWIND	00000696	R	03	UCB\$L_MEDIA_ID	= 0000008C		
RFCNEXT	000007AA	R	03	UCB\$L_MS_DPR	000000C8		
SETCHAR	000003D3	R	03	UCB\$L_MS_FMPR	000000CC		
SIZ...	= 00000001			UCB\$L_MS_NMPR	000000D4		
SPCFILFOR	00000527	R	03	UCB\$L_MS_OMPR	000000D8		
SPCFILREV	000005B4	R	03	UCB\$L_MS_PMPR	000000D0		
SPCRECFOR	00000606	R	03	UCB\$L_MS_TIMEOUT	000000DC		
SPCRECREV	0000067C	R	03	UCB\$L_MS_TMP2	000000E8		
SS\$ CTRLERR	= 00000054			UCB\$L_MS_TPOSITN	000000F4		
SS\$ DATAOVERUN	= 00000838			UCB\$L_MS_TSPT1	000000B6		
SS\$ DEVOFFLINE	= 00000084			UCB\$L_MS_TSPT2	000000BA		
SS\$ DRVERR	= 0000008C			UCB\$L_RECORD	= 000000B0		
SS\$ ENDOFFILE	= 00000870			UCB\$L_SVAPTE	= 00000078		
SS\$ ENDOFTAPE	= 00000878			UCB\$L_VCB	= 00000034		
SS\$ ENDOFVOLUME	= 000009A0			UCB\$M_MS_FEF	= 00000001		
SS\$ MEDOFL	= 000001A4			UCB\$M_MS_LBA	= 00000400		

TSDRIVER
Symbol table

UCBSM_MS_RDPR	=	00000020		
UCBSM_MS_RPI	=	00000800		
UCBSM_MS_SWAP	=	00000002		
UCBSM_MS_SWE	=	00000040		
UCBSM_MS_VCK	=	00001000		
UCBSM_ONLINE	=	00000010		
UCBSM_VALID	=	00000800		
UCBSQ_MS_BUF SVAPTE		000000EC		
UCBSQ_MS_TMP1		000000E0		
UCBSV_INT	=	00000001		
UCBSV_MS_FEF	=	00000000		
UCBSV_MS_LBA	=	0000000A		
UCBSV_MS_RDPR	=	00000005		
UCBSV_MS_RPI	=	0000000B		
UCBSV_MS_SWAP	=	00000001		
UCBSV_MS_SWE	=	00000006		
UCBSV_MS_VCK	=	0000C00C		
UCBSV_POWER	=	00000005		
UCBSV_VALID	=	0000000B		
UCBSW_BCNT	=	0000007E		
UCBSW_BOFF	=	0000007C		
UCBSW_DEVBUSIZ	=	00000042		
UCBSW_DEVSTS	=	00000068		
UCBSW_FUNC	=	0000009A		
UCBSW_MS_LNH		000000FA		
UCBSW_MS_MHD		000000F8		
UCBSW_MS_RBPC		000000FC		
UCBSW_MS_SPACNT		000000B4		
UCBSW_MS_TSBA		000000C0		
UCBSW_MS_TSPT3		000000BE		
UCBSW_MS_TSSR		000000C2		
UCBSW_MS_XC		000000C4		
UCBSW_MS_XSRO		000000FE		
UCBSW_MS_XSR1		00000100		
UCBSW_MS_XSR2		00000102		
UCBSW_MS_XSR3		00000104		
UCBSW_STS	=	00000064		
UNLOAD		0000071B	R	03
VASS_VPN	=	00000015		
VASV_VPN	=	00000009		
VECSB_DATAPATH	=	00000013		
VECSL_IDB	=	00000008		
VECSL_UNITINIT	=	00000018		
VECSW_MAPREG	=	00000010		
WCBSW_NMAP	=	00000016		
WRITECHAR		0000051C	R	03
WRITECHECK		000003EC	R	03
WRITECHECKR		000003EC	R	03
WRITEDATA		000004B9	R	03
WRITERET		00000506	R	03
WRITESUBS		00000511	R	03
WRTTMK		000006AD	R	03
WRTTMKR		000006FA	R	03
XTC		00000A44	R	03
XTC1		00000A51	R	03

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABS\$	00000106 (262.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$105_PROLOGUE	00000086 (134.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$115_DRIVER	00000D2E (3374.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	100	00:00:00.04	00:00:01.46
Command processing	115	00:00:00.15	00:00:02.56
Pass 1	698	00:00:10.78	00:00:40.78
Symbol table sort	0	00:00:01.14	00:00:05.50
Pass 2	413	00:00:02.56	00:00:13.54
Symbol table output	1	00:00:00.08	00:00:00.08
Psect synopsis output	0	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1329	00:00:14.76	00:01:03.93

The working set limit was 2700 pages.
180832 bytes (354 pages) of virtual memory were used to buffer the intermediate code.
There were 140 pages of symbol table space allocated to hold 2535 non-local and 149 local symbols.
2409 source lines were read in Pass 1, producing 27 object records in Pass 2.
51 pages of virtual memory were used to define 49 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA18:[SYS.OBJ]LIB.MLB;1	32
-\$255\$DUA18:[SYSLIB]STARLET.MLB;3	12
TOTALS (all libraries)	44

2529 GETS were required to define 44 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:TSDRIVER/OBJ=OBJ\$:TSDRIVER MSRC\$:TSDRIVER/UPDATE=(BUG\$:TSDRIVER)+EXECMLS/LIB

