

```

RRRRRRRRRRRRRRR      MMM      MMM      SSSSSSSSSSSSSS
RRRRRRRRRRRRRRR      MMM      MMM      SSSSSSSSSSSSSS
RRRRRRRRRRRRRRR      MMM      MMM      SSSSSSSSSSSSSS
RRR      RRR      MMMMMM      MMMMMM      SSS
RRR      RRR      MMMMMM      MMMMMM      SSS
RRR      RRR      MMMMMM      MMMMMM      SSS
RRR      RRR      MMM      MMM      MMM      SSS
RRR      RRR      MMM      MMM      MMM      SSS
RRR      RRR      MMM      MMM      MMM      SSS
RRRRRRRRRRRRRRR      MMM      MMM      SSSSSSSSSS
RRRRRRRRRRRRRRR      MMM      MMM      SSSSSSSSSS
RRRRRRRRRRRRRRR      MMM      MMM      SSSSSSSSSS
RRR      RRR      MMM      MMM      MMM      SSS
RRR      RRR      MMM      MMM      MMM      SSS
RRR      RRR      MMM      MMM      MMM      SSS
RRR      RRR      MMM      MMM      MMM      SSS
RRR      RRR      MMM      MMM      MMM      SSS
RRR      RRR      MMM      MMM      SSSSSSSSSSSSSS
RRR      RRR      MMM      MMM      SSSSSSSSSSSSSS
RRR      RRR      MMM      MMM      SSSSSSSSSSSSSS

```

— 52

Syn

NTS

NTS
NTS

NTS

NTS
NTS

NTS

NTS

NTS
NTS

NTS

NTS
NTSNTS
NTS

NTS

NTS
NTS

NTS

NTS
NTS

NTS

NTS
NTS

NTS

NTS
NTSNTS
NTS

NTS

NTS

NTA

NTS
NTSNTS
NTS

NTS

NTS
NTS

NTS

10

10

NTC

NT
NT

NTS

NT
NT

NT

PI

10

100

100

10

1

11. *Journal of the American Medical Association*, 1997; 277: 1039-1043.

1

```
RRRRRRRR MM MM SSSSSSSS 000000 RRRRRRRR UU UU HH HH NN NN DDDDDDDD
RRRRRRRR MM MM SSSSSSSS 000000 RRRRRRRR UU UU HH HH NN NN DDDDDDDD
RR RR RR MMMM MMMM SS SS 00 00 00 RR RR UU UU HH HH NN NN DD DD
RR RR RR MMMM MMMM SS SS 00 00 00 RR RR UU UU HH HH NN NN DD DD
RR RR RR MM MM MM SS SS 00 00 00 RR RR UU UU HH HH NN NN DD DD
RRRRRRRR MM MM SSSSSS 00 00 00 RRRRRRRR UU UU HHHHHHHHHH NN NN NN DD DD
RRRRRRRR MM MM SSSSSS 00 00 00 RRRRRRRR UU UU HHHHHHHHHH NN NN NN DD DD
RR RR MM MM MM SS SS 0000 00 00 RR RR UU UU HH HH NN NNNN DD DD
RR RR MM MM MM SS SS 0000 00 00 RR RR UU UU HH HH NN NNNN DD DD
RR RR MM MM MM SS SS 00 00 00 RR RR UU UU HH HH NN NN DD DD
RR RR MM MM MM SSSSSSSS 000000 RRRRRRRR UU UU HH HH NN NN DDDDDDDD
RR RR MM MM SSSSSSSS 000000 RRRRRRRR UU UU HH HH NN NN DDDDDDDD
```

```
LL LL IIIIII SSSSSSSS
LL LL IIIIII SSSSSSSS
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
```

RMS
Psc

PS

RMS
SA

Ph

In
Cor
Pas
Syl
Pas
Syl
Psc
Cro
Ass

The
370
The
350
26

Ma
-S
-S
-S
TO

79

Th
MA

(2)	89	DECLARATIONS
(3)	117	RMS\$RMSRUHNDLR - Dispatch to Recovery Unit Event Routines
(4)	212	Action Routines
(4)	213	STARTRU - Start a recovery Unit
(5)	234	ENDRU
(6)	267	UNLOCK - Recovery Unit Complete
(7)	287	RUSCAN - Search for files that are RU journaled.
(8)	340	SYNCH - Wait for asynch operations to complete

```
0000 1          $BEGIN RMSORUHND,000,RMSRMS,<RMS Recovery Unit Handler>
0000 2
0000 3
0000 4 :*****
0000 5 :*
0000 6 :*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 :*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 :*  ALL RIGHTS RESERVED.
0000 9 :*
0000 10 :*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 :*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 :*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 :*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 :*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 :*  TRANSFERRED.
0000 16 :*
0000 17 :*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 :*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 :*  CORPORATION.
0000 20 :*
0000 21 :*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 :*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23 :*
0000 24 :*
0000 25 :*****
0000 26 :
0000 27 :++
0000 28 : Facility:      RMS-32
0000 29 :
0000 30 : Abstract:
0000 31 :   This routine serves as a recovery unit handler for RMS and handles
0000 32 :   events concerned with the initiation, termination, etc. of recovery
0000 33 :   units.
0000 34 :
0000 35 : Environment:
0000 36 :   VAX/VMS Operating System
0000 37 :
0000 38 : Author:      Jeffrey W. Horn          Creation Date: 22-Jul-1982
0000 39 :
0000 40 : Modified By:
0000 41 :
0000 42 :   V03-010 JWT0159      Jim Teague      29-Feb-1984
0000 43 :   Use the new entry point, RMS$FLUSH_ALT, which will
0000 44 :   invalidate the cache.
0000 45 :
0000 46 :   V03-009 RAS0222      Ron Schaefer      9-Dec-1983
0000 47 :   Re-enable exec mode ASTs if necessary on exit from
0000 48 :   the handler.
0000 49 :
0000 50 :   V03-008 KPL0002      Peter Lieberwirth 11-Oct-1983
0000 51 :   MARKPOINT must flush buffers and generally bring the file
0000 52 :   to a quiet, recoverable state. Clear the flag IMP$V_RUH
0000 53 :   while waiting for operations to quiet in the routine SYNCH.
0000 54 :   If his flag is not clear, STALL does an RU stall for a
0000 55 :   user operation. This may occur at EXEC AST level,
0000 56 :   resulting in a hang.
0000 57 :
```


0000 58 :
0000 59 :
0000 60 :
0000 61 :
0000 62 :
0000 63 :
0000 64 :
0000 65 :
0000 66 :
0000 67 :
0000 68 :
0000 69 :
0000 70 :
0000 71 :
0000 72 :
0000 73 :
0000 74 :
0000 75 :
0000 76 :
0000 77 :
0000 78 :
0000 79 :
0000 80 :
0000 81 :
0000 82 :
0000 83 :
0000 84 :
0000 85 :
0000 86 :--
0000 87

Since this handler synchronizes with the completion of user-initiated RMS requests, a flag must be used to indicate to exit_rms that the asynch EFN must be set at completion. Since RUH cannot be used for the above reason, use RUH_SYNCH.

V03-007 KPL0001 Peter Lieberwirth 20-Jun-1983
Change some references to JNLFLG to JNLFLG2.

V03-006 TSK0001 Tamar Krichevsky 12-Jun-1983
Fix broken branch to journaling routine.

V03-005 ADE9003 Alan D. Eldridge 29-May-1983
Added RUF\$C_PHASE2 to case dispatch vector.

V03-004 JWH0178 Jeffrey W. Horn 02-Feb-1983
Back out JWH0171.

V03-003 JWH0176 Jeffrey W. Horn 31-Jan-1983
Provide a FAB in R8 for call to RMSMAPJNL_RU.

V03-002 JWH0172 Jeffrey W. Horn 21-Jan-1983
Fix mistype in JWH0171.

V03-001 JWH0171 Jeffrey W. Horn 18-Jan-1983
If IFB\$V_RU_RLK is set then turn on record locking for duration of recovery unit.


```

0000 89      .SBTTL DECLARATIONS
0000 90
0000 91
0000 92 :
0000 93 : Include Files:
0000 94 :
0000 95      $FABDEF
0000 96      $IFBDEF
0000 97      $IMPDEF
0000 98      $IRBDEF
0000 99      $PIODEF
0000 100     $PSLDEF
0000 101     $RUFDEF
0000 102
0000 103 :
0000 104 : Macros:
0000 105 :
0000 106
0000 107 :
0000 108 : Equated Symbols:
0000 109 :
0000 110
0000 111 :
0000 112 : Own Storage:
0000 113 :
0000 114
0000 115

```



```
0000 117 .SBTTL RMS$RMSRUHNDLR - Dispatch to Recovery Unit Event Routines
0000 118
0000 119 :++
0000 120 : RMS$RUHNDLR - Dispatch to Recovery Unit Event Routines
0000 121 :
0000 122 : This is the entry point for this service. It dispatches to one of
0000 123 : several event routines depending on the action code.
0000 124 :
0000 125 :
0000 126 : Calling sequence:
0000 127 :
0000 128 : Entered from Exec as a result of RUF calling SYS$RMSRUHNDLR in
0000 129 : EXEC mode.
0000 130 :
0000 131 : Input Parameters:
0000 132 :
0000 133 : AP Address of RUF argument list
0000 134 :
0000 135 : Implicit Inputs:
0000 136 :
0000 137 : RUF$$_RUCODE(AP) Address of action code
0000 138 :
0000 139 : Output Parameters:
0000 140 : R0,R1 Destroyed
0000 141 :
0000 142 : Implicit Outputs:
0000 143 : None
0000 144 :
0000 145 : Completion Codes:
0000 146 : None
0000 147 :
0000 148 : Side Effects:
0000 149 : See descriptions of dispatched routines
0000 150 :
0000 151 :--
0000 152 :
0000 153 :
0000 154 : $ENTRY RMS$RMSRUHNDLR
0000 155 :
0000 156 :
0000 157 : Verify this is actually call from RUF by checking previous
0000 158 : mode against current mode.
0000 159 :
0000 160 :
51 50 02 50 DC 0000 161 MOVPSL R0 ; get PSL
50 50 02 18 EF 0002 162 EXTZV #PSL$_CURMOD,#PSL$_CURMOD,R0,R1 ; get current mode
50 50 02 16 EF 0007 163 EXTZV #PSL$_PRVMOD,#PSL$_PRVMOD,R0,R0 ; get previous mode
51 50 D1 000C 164 CMPL R0,R1 ; are they the same?
38 12 000F 165 BNEQ ERRMOD ; branch if not
00000000'9F 01 A8 0011 166 BISW2 #1@PIO$_INHAST,@#PIO$_STATUS ; set AST disable
00000037'EF DF 0018 167 PUSHAL NULL ; set return point
001E 168
001E 169 :
001E 170 : Dispatch to action routine based on code from RUF.
001E 171 :
001E 172 :
0A 01 0C BC AF 001E 173 CASEW @RUF$_RUCODE(AP),#RUF$_MIN_CODE,#RUF$_MAX_CODE
```



```
0023 174
0023 175 ASSUME RUF$C_START EQ RUF$C_MIN_CODE
0023 176 ASSUME RUF$C_PHASE1 EQ 1+RUF$C_START
0023 177 ASSUME RUF$C_PHASE1_END EQ 1+RUF$C_PHASE1
0023 178 ASSUME RUF$C_PHASE2 EQ 1+RUF$C_PHASE1_END
0023 179 ASSUME RUF$C_PHASE2_END EQ 1+RUF$C_PHASE2
0023 180 ASSUME RUF$C_MARKPOINT EQ 1+RUF$C_PHASE2_END
0023 181 ASSUME RUF$C_RESET EQ 1+RUF$C_MARKPOINT
0023 182 ASSUME RUF$C_RESET_END EQ 1+RUF$C_RESET
0023 183 ASSUME RUF$C_CANCEL EQ 1+RUF$C_RESET_END
0023 184 ASSUME RUF$C_CANCEL_END EQ 1+RUF$C_CANCEL
0023 185 ASSUME RUF$C_CANCEL_END EQ RUF$C_MAX_CODE
0023 186
0023 187 DISP:
002C' 0023 188 .WORD STARTRU - DISP ; RUF$C_START
0047' 0025 189 .WORD ENDRU - DISP ; RUF$C_PHASE1
0014' 0027 190 .WORD NULL - DISP ; RUF$C_PHASE1_END
0014' 0029 191 .WORD NULL - DISP ; RUF$C_PHASE2
0080' 002B 192 .WORD UNLOCK - DISP ; RUF$C_PHASE2_END
0047' 002D 193 .WORD ENDRU - DISP ; RUF$C_MARKPOINT
0047' 002F 194 .WORD ENDRU - DISP ; RUF$C_RESET
0014' 0031 195 .WORD NULL - DISP ; RUF$C_RESET_END
0047' 0033 196 .WORD ENDRU - DISP ; RUF$C_CANCEL
0080' 0035 197 .WORD UNLOCK - DISP ; RUF$C_CANCEL_END
0037 198
0037 199 :
0037 200 : Get here if no action needed for code, or code out of range.
0037 201 :
0037 202
09 00000000'9F 00 E4 0037 203 NULL: BBSC #PIO$V_INHAST,@#PIO$GW_STATUS,10$ ; re-enable ASTs
003F 204 $SETAST_S #1 ; if disabled
04 0048 205 10$: RET
0049 206
0049 207
0049 208
05 004E 209 ERRMOD: RMSERR RUM
210 RSB
```



```
004F 212 .SBTTL Action Routines
004F 213 .SBTTL STARTRU - Start a recovery Unit
004F 214
004F 215 ;++
004F 216 ; STARTRU - Start a recovery Unit
004F 217 ;
004F 218 ; This routine processes the RUF$C_START action code.
004F 219 ;
004F 220 ;--
004F 221
004F 222 STARTRU:
56 00000059'EF DE 004F 223 MOVAL RUINIT,R6 ; Action to be taken for
5F 10 0056 224 BSBB RUSCAN ; each file.
05 0058 225 RSB
0059 226
0059 227
0059 228 RUINIT: SSB #IFB$V RUP,IFB$B JNLFLG2(R9) ; indicate RU in prog
58 24 A9 D0 005F 229 MOVL IFB$L CAST FAB(R9),R8 ; get FAB address
00000000'EF 16 0063 230 JSB RMSMAPJNL_RU ; write out Mapping ent
05 0069 231 RSB
006A 232
```



```
006A 234 .SBTTL ENDRU
006A 235 :++
006A 236 : ENDRU - End a recovery Unit
006A 237 :
006A 238 : This routine processes the
006A 239 : RUF$C_CANCEL,
006A 240 : RUF$C_RESET,
006A 241 : RUF$C_PHASE1
006A 242 :
006A 243 :--
006A 244 :
006A 245 ENDRU:
006A 246 MOVAL RUEND,R6 ; action to be taken
DE 10 0071 247 BSBP RUSCAN ; for each file
05 0073 248 RSB
0074 249
0074 250
0074 251 RUEND:
009F 30 0074 252 BSBW SYNCH ; wait for asynch ops
5A 59 D0 0077 253 MOVL R9,R10 ; save IFB addr
59 1C A9 D0 007A 254 10$: MOVL IFB$L_IRAB_LNK(R9),R9 ; get IRB
17 13 007E 255 BEQL 20$ ; branch if none
0093 30 0080 256 BSBW SYNCH ; wait for asynch ops
58 24 A9 D0 0083 257 MOVL IRB$L_LAST_RAB(R9),R8 ; get RAB address
00C0 8F BB 0087 258 PUSHF #^M<R6,R7> ; save registers
00000000'EF 16 008B 259 JSB RMS$FLUSH_ALT ; flush buffers
00C0 8F BA 0091 260 POPR #^M<R6,R7> ; restore registers
E3 11 0095 261 BRB 10$ ; go get next IRB
59 5A D0 0097 262 20$: MOVL R10,R9 ; restore IFB addr
54 D4 009A 263 CLRL R4 ; force all journal entries
00000000'EF 16 009C 264 JSB RMS$FRCJNL ; call force
05 00A2 265 RSB
```



```

00A3 267 .SBTTL UNLOCK - Recovery Unit Complete
00A3 268 :++
00A3 269 : UNLOCK - Recovery Unit Complete
00A3 270 :
00A3 271 : This routine processes the RUF$C_CANCEL_END and the RUF$C_PHASE2
00A3 272 : action codes
00A3 273 :
00A3 274 :--
00A3 275
00A3 276 UNLOCK:
56 000000B0'EF DE 00A3 277 MOVAL KILLRU,R6 ; action to be taken
      OB 10 00AA 278 BSBB RUSCAN ; for each file.
      FF51' 30 00AC 279 BSBW RMSRU_UNLOCK ; release all RU locks
      05 00AF 280 RSB
      00B0 281
      00B0 282 KILLRU:
      00B0 283 CSB #IFB$V_RUP,IFB$B_JNLFLG2(R9) ; clear RU in prog
      05 00B6 284 RSB
      00B7 285

```



```
00B7 287 .SBTTL RUSCAN - Search for files that are RU journaled.
00B7 288
00B7 289 :++
00B7 290 : RUSCAN - Search for files that are RU journaled.
00B7 291 :
00B7 292 : This subroutine searches for open files that have the recovery unit
00B7 293 : journal bit set. For each one found it sets up the registers
00B7 294 : as follows:
00B7 295 : R9 - Address of IFB
00B7 296 : R11 - Impure area address
00B7 297 : It then calls the routine specified by RUSCAN's caller at the
00B7 298 : address specified in R6.
00B7 299 :
00B7 300 : If the specified routine returns an error code, the scan is
00B7 301 : terminated.
00B7 302 :
00B7 303 :
00B7 304 :--
00B7 305
00B7 306
5B 00000000'9F DE 00B7 307 RUSCAN: MOVAL @#PIO$GW_PIOIMPA,R11 ; set impure area addr
00B7 308 BSBB SCAN ; go do scan
00B7 309 BLBC R0,10$ ; get out on error
5B 00000000'9F DE 00C3 310 MOVAL @#PIO$GW_IIOIMPA,R11 ; set impure area addr
01 00CA 311 BSBB SCAN ; go do scan
05 00CC 312 10$: RSB
00CD 313
00CD 314 SCAN: RMSSUC ; preset success
00D0 315 SSB #IMP$V_RUH_SYNCH,(R11) ; set handler synch with
00D4 316 ; rms exit
00D4 317 SSB #IMP$V_RUH,(R11) ; set handler in prog
57 18 AB D0 00D8 318 MOVL IMP$L_IFABTBL(R11),R7 ; get table address
7E 87 D0 00DC 319 10$: MOVL (R7)+,-(SP) ; get next seg addr
51 20 AB 3C 00DF 320 MOVZWL IMP$W_ENTPERSEG(R11),R1 ; get seg limit
7E 6741 DE 00E3 321 MOVAL (R7)[R1],-(SP) ; calculate end of seg
59 87 D0 00E7 322 20$: MOVL (R7)+,R9 ; get IFB address
15 13 00EA 323 BEQL 40$ ; branch if ifi not used
OF 00A0 C9 01 E1 00EC 324 BBC #IFB$V_RU,IFB$B_JNLFLG(R9),40$ ; branch if no RU
01 BB 00F2 325 PUSHF #^M<R0$ ; save current status
66 16 00F4 326 JSB (R6) ; call routine
06 50 E8 00F6 327 BLBS R0,30$ ; continue on success
03 6E E9 00F9 328 BLBC (SP),30$ ; branch if code already err
6E 50 D0 00FC 329 MOVL R0,(SP) ; substitute code
01 BA 00FF 330 30$: POPR #^M<R0> ; restore code
6E 57 D1 0101 331 40$: CMPL R7,(SP) ; at end of seg?
E1 15 0104 332 BLEQ 20$ ; branch if not
8E D5 0106 333 TSTL (SP)+ ; clean off seg end adr
57 8E D0 0108 334 MOVL (SP)+,R7 ; get next seg
CF 12 010B 335 BNEQ 10$ ; branch if one
010D 336 CSB #IMP$V_RUH,(R11) ; clear handler in prog
0111 337 CSB #IMP$V_RUH_SYNCH,(R11) ; clear handler in prog
05 0115 338 RSB
```



```
0116 340 .SBTTL SYNCH - Wait for asynch operations to complete
0116 341 :++
0116 342 : SYNCH - Wait for asynch operations to complete
0116 343 :--
0116 344
38 69 20 E1 0116 345 SYNCH: BBC #IRBSV_BUSY,(R9),20$ ; branch if not busy
34 69 3A E1 011A 346 BBC #IRBSV_RMS_STALL,(R9),20$ ; branch if not stalled
011E 347 CSB #IMPSV_RUH,(R11) ; don't do RUSTALL
09 00000000'9F 00 E4 0122 348 $CLREF_S #IMPSC_ASYEFN ; clear event flag
012B 349 BBSC #PIOSV_INHAST,@#PIOSGW_STATUS,10$ ; set enable ASTs
0133 350 $SETAST_S #1 ; enable ASTs
013C 351 10$: $WAITFR_S #IMPSC_ASYEFN ; wait for event
00000000'9F 01 A8 0145 352 BISW2 #1@PIOSV_INHAST,@#PIOSGW_STATUS ; set AST disable
014C 353 SSB #IMPSV_RUH,(R11) ; ok for RUSTALL again
C4 11 0150 354 BRB SYNCH ; make sure done
05 05 0152 355 20$: RSB
0153 356
0153 357 .END
```


RMSORUHND
Symbol table

RMS Recovery Unit Handler

K 15

16-SEP-1984 01:30:17 VAX/VMS Macro V04-00
5-SEP-1984 16:25:27 [RMS.SRC]RMSORUHND.MAR;1

Page 11
(8)

\$\$PSECT_EP	= 00000000		
\$\$RMSTEST	= 0000001A		
\$\$RMS_PBUGCHK	= 00000010		
\$\$RMS_TBUGCHK	= 00000008		
\$\$RMS_UMODE	= 00000004		
DISP	00000023	R	01
ENDRU	0000006A	R	01
ERRMOD	00000049	R	01
IFB\$B_JNLFLG	= 000000A0		
IFB\$B_JNLFLG2	= 000000A2		
IFB\$B_IRAB_LNK	= 0000001C		
IFB\$B_LAST_FAB	= 00000024		
IFB\$V_RU	= 00000001		
IFB\$V_RUP	= 00000002		
IMP\$C_ASYEFN	= 0000001E		
IMP\$C_IFABTBL	= 00000018		
IMP\$V_RUH	= 00000006		
IMP\$V_RUH_SYNCH	= 00000008		
IMP\$W_ENTPERSEG	= 00000020		
IRB\$B_LAST_RAB	= 00000024		
IRB\$V_BUSY	= 00000020		
IRB\$V_RMS_STALL	= 0000003A		
KILLR0	000000B0	R	01
NULL	00000037	R	01
PIO\$GW_IIOIMPA	*****	X	01
PIO\$GW_PIOIMPA	*****	X	01
PIO\$GW_STATUS	*****	X	01
PIO\$V_INHAST	= 00000000		
PSL\$S_CURMOD	= 00000002		
PSL\$S_PRVMOD	= 00000002		
PSL\$V_CURMOD	= 00000018		
PSL\$V_PRVMOD	= 00000016		
RMS\$FLUSH_ALT	*****	X	01
RMS\$FRCJNC	*****	X	01
RMS\$MAPJNL_RU	*****	X	01
RMS\$RU_UNLOCK	*****	X	01
RMS\$RMSRUHNDLR	= FFFFFFFE	RG	01
RMS\$RUM	*****	X	01
RUEND	00000074	R	01
RUF\$C_CANCEL	= 00000009		
RUF\$C_CANCEL_END	= 0000000A		
RUF\$C_MARKPOINT	= 00000006		
RUF\$C_MAX_CODE	= 0000000A		
RUF\$C_MIN_CODE	= 00000001		
RUF\$C_PHASE1	= 00000002		
RUF\$C_PHASE1_END	= 00000003		
RUF\$C_PHASE2	= 00000004		
RUF\$C_PHASE2_END	= 00000005		
RUF\$C_RESET	= 00000007		
RUF\$C_RESET_END	= 00000008		
RUF\$C_START	= 00000001		
RUF\$C_RUCODE	= 0000000C		
RUINIT	00000059	R	01
RUSCAN	000000B7	R	01
SCAN	000000CD	R	01
STARTRU	0000004F	R	01
SYNCH	00000116	R	01

SYSSCLREF
SYSSSETAST
SYSSWAITFR
UNLOCK

*****	GX	01
*****	GX	01
*****	GX	01
000000A3	R	01

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

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RMSRMS	00000153 (339.)	01 (1.)	PIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC BYTE
\$AB\$\$	00000000 (0.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.11	00:00:00.68
Command processing	110	00:00:00.70	00:00:04.53
Pass 1	254	00:00:07.04	00:00:18.86
Symbol table sort	0	00:00:00.84	00:00:01.29
Pass 2	75	00:00:01.47	00:00:04.01
Symbol table output	8	00:00:00.08	00:00:00.27
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	480	00:00:10.26	00:00:29.66

The working set limit was 1350 pages.
37653 bytes (74 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 663 non-local and 18 local symbols.
357 source lines were read in Pass 1, producing 14 object records in Pass 2.
26 pages of virtual memory were used to define 25 macros.

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

Macro library name	Macros defined
_\$255\$DUA28:[RMS.OBJ]RMS.MLB;1	11
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	1
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	9
TOTALS (all libraries)	21

791 GETS were required to define 21 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:RMSORUHND/OBJ=OBJ\$:RMSORUHND MSRC\$:RMSORUHND/UPDATE=(ENH\$:RMSORUHND)+EXECML\$/LIB+LIB\$:RMS/LIB

0330 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

