


```

RRRRRRRR      MM      MM      11      PPPPPPPP      UU      UU      TTTTTTTTTT      SSSSSSSS      EEEEEEEEEEE      TTTTTTTTTT
RRRRRRRR      MM      MM      11      PPPPPPPP      UU      UU      TTTTTTTTTT      SSSSSSSS      EEEEEEEEEEE      TTTTTTTTTT
RR      RR      MMMM      MMMM      1111      PP      PP      UU      UU      TT      SS      SSSSSSSS      EE      TT
RR      RR      MMMM      MMMM      1111      PP      PP      UU      UU      TT      SS      SSSSSSSS      EE      TT
RR      RR      MM      MM      11      PP      PP      UU      UU      TT      SS      SSSSSSSS      EE      TT
RR      RR      MM      MM      11      PPPPPPPP      UU      UU      TT      SS      SSSSSSSS      EEEEEEEEEEE      TT
RRRRRRRR      MM      MM      11      PPPPPPPP      UU      UU      TT      SS      SSSSSSSS      EEEEEEEEEEE      TT
RRRRRRRR      MM      MM      11      PPPPPPPP      UU      UU      TT      SS      SSSSSSSS      EEEEEEEEEEE      TT
RR      RR      MM      MM      11      PP      PP      UU      UU      TT      SS      SSSSSSSS      EE      TT
RR      RR      MM      MM      11      PP      PP      UU      UU      TT      SS      SSSSSSSS      EE      TT
RR      RR      MM      MM      11      PP      PP      UU      UU      TT      SS      SSSSSSSS      EE      TT
RR      RR      MM      MM      111111      PP      UU      UU      TT      SSSSSSSS      EEEEEEEEEEE      TT
RR      RR      MM      MM      111111      PP      UU      UU      TT      SSSSSSSS      EEEEEEEEEEE      TT
                                     ....
                                     ....
                                     ....

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

```


(3) 85
(4) 111
(6) 298

DECLARATIONS
RM\$PUTSETUP1
RM\$PROBEREAD - PROBE BUFFER READABILITY


```

0000 1          $BEGIN RM1PUTSET,000,RMSRMS1,<SETUP FOR $PUT/$UPDATE SEQUENTIAL>
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 28 :++
0000 29 : Facility: rms32
0000 30 :
0000 31 :
0000 32 : Abstract:    this module performs various setups for
0000 33 :                $put and $update on the sequential file organization.
0000 34 :
0000 35 : Environment:
0000 36 :                star processor running starlet exec.
0000 37 :
0000 38 : Author:       L F Laverdure,    creation date: 17-FEB-1977
0000 39 :
0000 40 : Modified By:
0000 41 :
0000 42 :     V03-005 DAS0001      David Solomon      08-Feb-1984
0000 43 :                Performance enhancement for variable length records.
0000 44 :
0000 45 :     V03-004 RAS0115      Ron Schaefer        10-Jan-1983
0000 46 :                Check for total record size negative (i.e. size > 32767).
0000 47 :
0000 48 :     V03-003 KBT0416      Keith B. Thompson   30-Nov-1982
0000 49 :                Change ifb$w_devbufsiz to ifb$l_devbufsiz
0000 50 :
0000 51 :     V03-002 KBT0146      Keith B. Thompson   20-Aug-1982
0000 52 :                Reorganize psects
0000 53 :
0000 54 :     V03-001 KBT0087      Keith B. Thompson   13-Jul-1982
0000 55 :                Clean up psects
0000 56 :
0000 57 :     V02-011 RAS0049      Ron Schaefer        15-Dec-1981
0000 58 :                Fix stm terminator check.
0000 59 :
0000 60 :     V02-010 RAS0028      Ron Schaefer        20-AUG-1981
0000 61 :                Change FAB$C_STM11 to FAB$C_STM.
0000 62 :
0000 63 :     V02-009 RAS0016      Ron Schaefer        31-Jul-1981
0000 64 :                Add support for stream format.
0000 65 :
0000 66 :     V02-008 MCN0003      Maria del C. Nasr    11-Nov-1980
0000 67 :                Check that variable length records written to an ANSI tape
0000 68 :                are not longer than to 9999 bytes, since this is the biggest
0000 69 :                number that fits in the record control word.
0000 70 :
0000 71 :     V02-007 REFORMAT     Maria del C. Nasr    24-Jul-1980
0000 72 :
0000 73 :     V0006  CDS0001      C D Saether      11-MAR-1980
0000 74 :                don't calculate record overhead for unit record devices
0000 75 :
0000 76 : Revision History:
0000 77 :
0000 78 :     L F Laverdure,      14-AUG-1978  15:39
0000 79 :                long probe fix
0000 80 :
0000 81 : --
0000 82 :
0000 83 :
```



```
0000 85 .SBTTL DECLARATIONS
0000 86
0000 87 :
0000 88 : Include Files:
0000 89 :
0000 90 :
0000 91 :
0000 92 : Macros:
0000 93 :
0000 94 :
0000 95 $DEVDEF
0000 96 $IFBDEF
0000 97 $IRBDEF
0000 98 $FABDEF
0000 99 $RABDEF
0000 100 $RMSDEF
0000 101
0000 102 :
0000 103 : Equated Symbols:
0000 104 :
0000 105 :
0000 106 :
0000 107 : Own Storage:
0000 108 :
0000 109
```



```
0000 111 .SBTTL RM$PUTSETUP1
0000 112
0000 113 :++
0000 114 RM$PUTSETUP1 - This module makes user input and operation
0000 115 valid checks and calculates record overhead size.
0000 116
0000 117 Calling sequence:
0000 118
0000 119 bsbw rm$putsetup1
0000 120
0000 121 Input Parameters:
0000 122
0000 123 r10 ifab addr
0000 124 r9 irab addr
0000 125 r8 rab addr
0000 126
0000 127 Implicit Inputs:
0000 128
0000 129 the contents of the rab and related irab and ifab.
0000 130
0000 131 Output Parameters:
0000 132
0000 133
0000 134 r6 record data length in bytes
0000 135 r5 record address
0000 136 r1 total record length including overhead bytes
0000 137 r0 status code
0000 138
0000 139 Implicit Outputs:
0000 140
0000 141 sequential file org temp.
0000 142 irb$w_rovhdsz: record overhead size in bytes
0000 143 irb$w_rtotlsz: total record length including overhead bytes
0000 144
0000 145 Completion Codes:
0000 146
0000 147 standard rms, in particular, either suc, rbf, or rsz.
0000 148
0000 149 Side Effects:
0000 150
0000 151 none
0000 152 --
0000 153
0000 154
0000 155 RM$PUTSETUP1::
0000 156 $STPT PUTSET1
0006 157
0006 158
0006 159 : get the user record address & size and validate
0006 160 :
0006 161
0006 162 MOVL RAB$L_RBF(R8),R5 ; get record address
000A 163 CVTBL RAB$W_RSZ(R8),R6 ; get record length
000E 164 BLSS ERRRSZ ; negative size invalid
0010 165 BEQL CHKSIZ ; no need to probe a null rec
0012 166 CMPW R6,#512 ; long probe needed?
0017 167 BGTRU LONG_PROBE ; branch if yes
```

55	28	A8	D0	0006	162	MOVL	RAB\$L_RBF(R8),R5	; get record address
56	22	A8	32	000A	163	CVTBL	RAB\$W_RSZ(R8),R6	; get record length
		2F	19	000E	164	BLSS	ERRRSZ	; negative size invalid
		0D	13	0010	165	BEQL	CHKSIZ	; no need to probe a null rec
0200	8F	56	B1	0012	166	CMPW	R6,#512	; long probe needed?
		52	1A	0017	167	BGTRU	LONG_PROBE	; branch if yes


```
0019 168          IFNORD R6,(R5),ERRRBF          : probe buffer
001F 169
001F 170 :
001F 171 : make record size checks and compute overhead based on
001F 172 : record format type
001F 173 :
001F 174
51  D4 001F 175 CHKSIZ: CLRL R1          : compute record overhead in r1
02  91 0021 176          CMPB #FAB$C VAR,-      : is it var record format?
50 AA 0023 177          IFB$B RFMORG(R10)
24  12 0025 178          BNEQ NOTVAR          : no, go handle others
0027 179
0027 180 :
0027 181 : variable length - add in size of record length bytes
0027 182 :
0027 183
0A 6A E8 0027 184 VARLEN: ASSUME DEV$V_REC EQ 0
0027 185          BLBS IFB$L_PRIM_DEV(R10), -
002A 186          LENCHK          : no size for unit record devices
51  02 A0 002A 187          ADDW2 #2,R1          : normally 2 bytes
03 6A 26 E1 002D 188          BBC IFB$V_ANSI_D,(R10), -
0031 189          LENCHK          : all set unless ansi d
51  02 A0 0031 190          ADDW2 #2,R1          : in which case it's 4
60 AA B5 0034 191 LENCHK: TSTW IFB$W_MRS(R10)      : omit check if limit is 0
6E  13 0037 192          BEQL CHKBLK
60 AA 56 B1 0039 193          CMPW R6,IFB$W_MRS(R10) : check record length
68  1B 003D 194          BLEQU CHKBLK          : and branch if le max. allowed
003F 195
003F 196 :
003F 197 : record size too big
003F 198 :
003F 199
003F 200 ERRRSZ:
003F 201          RMSERR RSZ
05 0044 202          RSB
0045 203
0045 204 ERRRBF:
0045 205          RMSERR RBF          : bad user buffer
05 004A 206          RSB
004B 207
004B 208 :
004B 209 : Dispatch for other record formats.
004B 210 :
004B 211
004B 212 NOTVAR: CASE LIMIT=#FAB$C UDF,-      : based on record format
004B 213          SRC=IFB$B RFMORG(R10),-
004B 214          TYPE=B,DISPLIST=-
004B 215          <UDFLEN,-          : FAB$C_UDF
004B 216          FIXEDLEN,-          : FAB$C_FIX
004B 217          10$,-              : FAB$C_VAR
004B 218          VFLEN,-            : FAB$C_VFC
004B 219          STMLEN,-           : FAB$C_STM
004B 220          STMLEN,-           : FAB$C_STMLF
004B 221          STMLEN>           : FAB$C_STMCR
005E 222
005E 223 10$: RMSTBUG -99          : bugcheck if we fall through
0065 224
```

RM1PUTSET
V04-000

SETUP FOR \$PUT/\$UPDATE SEQUENTIAL L 8
RM\$PUTSETUP1

16-SEP-1984 00:55:00 VAX/VMS Macro V04-00
5-SEP-1984 16:23:42 [RMS.SRC]RM1PUTSET.MAR;1

Page 6
(4)

51	5F	AA	90	0065	225	VFLEN:	MOVB	IFB\$B	FSZ(R10),R1	:	get fixed header size for vfc
		BC	11	0069	226		BRB	VARLEN		:	pick up in VAR code


```
006B 228 :  
006B 229 : long probe needed subroutine  
006B 230 :  
006B 231 :  
006B 232 LONG_PROBE:  
AF 50 6A 10 006B 233 BSBB RMS$PROBEREAD : check readability  
E8 006D 234 BLBS RO,CHKSIZ : all is ok continue  
05 0070 235 RSB : else exit  
0071 236 :  
0071 237 :  
0071 238 : stream format record - check whether a DFT must be added  
0071 239 :  
0071 240 $TMLEN:  
0071 241 ASSUME DEV$V_REC EQ 0  
24 6A E8 0071 242 BLBS IFB$L_PRIM_DEV(R10),10$ : no additional chars for unit-record  
56 B5 0074 243 TSTW R6 : zero len record  
16 13 0076 244 BEQL 5$ : needs a terminator  
22 BB 0078 245 PUSHF #^M<R1,R5> : save size and record addr  
51 FF A5 46 9E 007A 246 MOVAB -1(R5)[R6],R1 : setup for term check  
50 01 D0 007F 247 MOVL #1,R0 : check only last char  
54 50 AA 9A 0082 248 MOVZBL IFB$B_RFMORG(R10),R4 : get format type  
FF 77 30 0086 249 BSBW RMS$STM_TERM : check for terminator  
22 BA 0089 250 POPR #^M<R1,R5> : restore regs  
0A 50 E8 008B 251 BLBS R0,10$ : already have a terminator  
51 B6 008E 252 5$: INCW R1 : add in the DFT size  
50 AA 91 0090 253 CMPB IFB$B_RFMORG(R10),- : STM format?  
04 0093 254 #FAB$C_STM  
02 12 0094 255 BNEQ 10$ : nope  
51 B6 0096 256 INCW R1 : STM's DFT is 2 bytes long  
03 6A 26 E1 0098 257 10$: BBC #IFB$V_ANSI_D,(R10),20$ : need count field for ANSI  
51 04 A0 009C 258 ADDW2 #4,R1  
93 11 009F 259 20$: BRB LENCHK  
00A1 260 :  
00A1 261 :  
00A1 262 : fixed length record - check its size  
00A1 263 :  
00A1 264 :  
52 AA 56 B1 00A1 265 FIXEDLEN:  
98 12 00A5 266 CMPW R6,IFB$W_LRL(R10) : compare against fixed size  
00A7 267 BNEQ ERRRSZ : branch if not equal  
00A7 268 :  
00A7 269 :  
00A7 270 : if blk bit set (records can't cross block boundaries) check  
00A7 271 : that total record size is less than a block  
00A7 272 :  
64 A9 51 B0 00A7 273 $DLEN:  
51 56 A0 00AB 274 CHKBLK: MOVW R1,IRB$W_ROVHDSZ(R9) : save overhead size  
8F 19 00AE 275 ADDW2 R6,R1 : compute total record size  
1A 51 AA 03 E1 00B0 276 BLSS ERRRSZ : bad if neg (=> size>32767)  
00B5 277 BBC #FAB$V_BLK,IFB$B_RAT(R10),10$ : branch if no boundary  
48 AA 51 B1 00B5 278 : restriction  
84 1A 00B9 279 CMPW R1,IFB$L_DEVBUFSIZ(R10) : compare against block size  
00BB 280 BGTRU ERRRSZ : and branch if too big  
00BB 281 :  
00BB 282 :  
00BB 283 : If ANSI_D record (variable length in ANSI magtape) make sure that  
00BB 284 : record is not bigger than 9999 bytes.
```

10 6A 26	E1	00BB	285 ;	BBC	#IFB\$V ANSI_D,(R10),10\$; branch, if not ANSI_D
01 50 AA	91	00BB	286	CMPB	IFB\$B_RFMORG(R10),#FAB\$C_FIX ; branch, if fixed length
	13	00BF	287	BEQL	10\$
270F 8F 51	B1	00C3	288	CMPW	R1,#9999 ; within maximum size?
	1B	00C5	289	BLEQU	10\$; yes
	31	00CA	290	BRW	ERRRSZ ; error if not
66 A9 FF70 51	B0	00CC	291	MOVW	R1,IRB\$W_RTOTLSZ(R9) ; save total record size
		00CF	292 10\$:	RMSSUC	
	05	00D3	293	RSB	
		00D6	294		
		00D7	295		
			296		


```
00D7 298      .SBTTL RM$PROBEREAD - PROBE BUFFER READABILITY
00D7 299
00D7 300      :++
00D7 301      : RM$PROBEREAD - This routine probes the caller's buffer
00D7 302      : for readability.
00D7 303
00D7 304      : Calling sequence:
00D7 305
00D7 306      :     bsbw      rm$proberead
00D7 307
00D7 308      : Input Parameters:
00D7 309
00D7 310      :     r10      ifab addr
00D7 311      :     r9       irab addr
00D7 312      :     r6       size of buffer
00D7 313      :     r5       addr of buffer
00D7 314
00D7 315      : Implicit Inputs:
00D7 316
00D7 317      :     irb$b_mode
00D7 318
00D7 319      : outputs:
00D7 320
00D7 321      :     r0       status code
00D7 322
00D7 323      : Implicit Outputs:
00D7 324
00D7 325      :     none
00D7 326
00D7 327      : condition codes:
00D7 328
00D7 329      :     standard rms, in particular, rbf or suc.
00D7 330
00D7 331      : Side Effects:
00D7 332
00D7 333      :     none
00D7 334
00D7 335      :--
00D7 336
00D7 337
00D7 338      RM$PROBEREAD::
00D7 339      MOVQ    R5,-(SP)          ; save r5,r6
00D7 340      PUSHL   #1              ; anticipate success
00D7 341      TSTL    R6              ; zero buffer size?
00D7 342      BEQL    EXIT1          ; omit probe if so
00E0 343
00E0 344      :
00E0 345      : probe all pages
00E0 346      :
00E0 347
00E0 348      50      FE00 8F 32      CVTWL    #-512,R0          ; get address calc constant
00E5 349      10$: IFNORD    R6,(R5),ERRRBF1,-          ;
00E5 350      :     IRB$b_MODE(R9)          ;
00EC 351      :     SUBL2    R0,R5          ; branch if not readable
00EF 352      :     MOVAW    (R6)[R0],R6      ; get address of next page
00F3 353      :     BGTR     10$             ; calculate new length
00F5 354      :     SUBL2    R0,R6          ; continue probing if positive
00F5 354      :     need to handle last page?
```



```

0061 EB 14 00F8 355      BGTR 10$      ; branch if yes
      BF BA 00FA 356 EXIT1: POPR #^M<R0,R5,R6> ; restore buffer desc. & status
      05 00FE 357      RSB
          00FF 358
          00FF 359
          00FF 360 ; probe failure - set error code
          00FF 361
          00FF 362
          00FF 363 ERRRBF1:
      F4 11 00FF 364      RMSERR RBF,(SP)
          0104 365      BRB EXIT1
          0106 366
          0106 367      .END
  
```


RM1PUTSET
Symbol table

SETUP FOR \$PUT/\$UPDATE SEQUENTIAL D 9

16-SEP-1984 00:55:00
5-SEP-1984 16:23:42

VAX/VMS Macro V04-00
[RMS.SRC]RM1PUTSET.MAR;1

Page 11
(6)

\$\$PSECT_EP	=	00000000		
\$\$RMSTEST	=	0000001A		
\$\$RMS_PBUGCHK	=	00000010		
\$\$RMS_TBUGCHK	=	00000008		
\$\$RMS_UMODE	=	00000004		
CHKBLK		000000A7	R	01
CHKSIZ		0000001F	R	01
DEV\$V_REC	=	00000000		
ERRRBF		00000045	R	01
ERRRBF1		000000FF	R	01
ERRRSZ		0000003F	R	01
EXIT1		000000FA	R	01
FAB\$C_FIX	=	00000001		
FAB\$C_STM	=	00000004		
FAB\$C_UDF	=	00000000		
FAB\$C_VAR	=	00000002		
FAB\$V_BLK	=	00000003		
FIXEDLEN		000000A1	R	01
IFB\$B_FSZ	=	0000005F		
IFB\$B_RAT	=	00000051		
IFB\$B_RFMORG	=	00000050		
IFB\$B_DEVBUFSIZ	=	00000048		
IFB\$B_PRIM_DEV	=	00000000		
IFB\$V_ANSI_D	=	00000026		
IFB\$W_LRL	=	00000052		
IFB\$W_MRS	=	00000060		
IRB\$B_MODE	=	0000000A		
IRB\$W_ROVHDSZ	=	00000064		
IRB\$W_RTOTLSZ	=	00000066		
LENCHR		00000034	R	01
LONG_PROBE		0000006B	R	01
NOTVAR		0000004B	R	01
PIOSA_TRACE		*****	X	01
RAB\$B_RBF	=	00000028		
RAB\$W_RSZ	=	00000022		
RM\$BUG		*****	X	01
RM\$PROBEREAD		000000D7	RG	01
RM\$PUTSETUP1		00000000	RG	01
RM\$STM_TERM		*****	X	01
RM\$B_RBF	=	00018654		
RM\$B_RSZ	=	000186A4		
STMLEN		00000071	R	01
TPT\$B_PUTSET1		*****	X	01
UDFLEN		000000A7	R	01
VARLEN		00000027	R	01
VFLEN		00000065	R	01

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

PSECT name

PSECT name	Allocation	PSECT No.	Attributes														
. ABS .	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
RM\$RMS1	00000106 (262.)	01 (1.)	PIC	USR	CON	REL	GBL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE				
\$AB\$B	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	40	00:00:00.11	00:00:00.97
Command processing	140	00:00:00.75	00:00:03.80
Pass 1	304	00:00:09.48	00:00:28.63
Symbol table sort	0	00:00:01.22	00:00:01.80
Pass 2	74	00:00:01.92	00:00:04.37
Symbol table output	6	00:00:00.09	00:00:00.22
Psect synopsis output	2	00:00:00.03	00:00:00.08
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	568	00:00:13.60	00:00:39.96

The working set limit was 1500 pages.
52606 bytes (103 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 1022 non-local and 8 local symbols.
367 source lines were read in Pass 1, producing 13 object records in Pass 2.
24 pages of virtual memory were used to define 23 macros.

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

Macro library name	Macros defined
_\$255\$DUA28:[RMS.OBJ]RMS.MLB;1	12
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	2
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	19

1152 GETS were required to define 19 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:RM1PUTSET/OBJ=OBJ\$:RM1PUTSET MSRC\$:RM1PUTSET/UPDATE=(ENH\$:RM1PUTSET)+EXECML\$/LIB+LIB\$:RMS/LIB

0322

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY