```
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

FILEID**ROLLUP

```
RRRRRRRR      000000    LL          LL          UU      UU  PPPPPPPP
RRRRRRRR      000000    LL          LL          UU      UU  PPPPPPPP
RR      RR  00      00  LL          LL          UU      UU  PP      PP
RR      RR  00      00  LL          LL          UU      UU  PP      PP
RR      RR  00      00  LL          LL          UU      UU  PP      PP
RR      RR  00      00  LL          LL          UU      UU  PP      PP
RRRRRRRR    00      00  LL          LL          UU      UU  PPPPPPPP
RRRRRRRR    00      00  LL          LL          UU      UU  PPPPPPPP
RR  RR      00      00  LL          LL          UU      UU  PP
RR  RR      00      00  LL          LL          UU      UU  PP
RR    RR    00      00  LL          LL          UU      UU  PP              ....
RR    RR    00      00  LL          LL          UU      UU  PP              ....
RR      RR    000000    LLLLLLLLLL  LLLLLLLLLL  UUUUUUUUUU  PP              ....
RR      RR    000000    LLLLLLLLLL  LLLLLLLLLL  UUUUUUUUUU  PP              ....

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
```

```
0001    C
0002    C Version:      'V04-000'
0003    C
0004    C*****************************************************************************
0005    C*                                                                           *
0006    C*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0007    C*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0008    C*  ALL RIGHTS RESERVED.                                                     *
0009    C*                                                                           *
0010    C*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0011    C*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0012    C*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0013    C*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0014    C*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0015    C*  TRANSFERRED.                                                             *
0016    C*                                                                           *
0017    C*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0018    C*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0019    C*  CORPORATION.                                                             *
0020    C*                                                                           *
0021    C*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0022    C*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0023    C*                                                                           *
0024    C*                                                                           *
0025    C*****************************************************************************
0026    C
0027    C               AUTHOR  BRIAN PORTER            CREATION DATE   10-SEP-1980
0028
0029    C++
0030    C       Functional description:
0031    C
0032    C       This module displays the device error roll-up report.
0033    C
0034    C       Modified by:
0035    C
0036    C       V03-004 SAR0257         Sharon A. Reynolds      25-Apr-1984
0037    C               Widened the output field for the device names to
0038    C               accomodate the node name being logged now.
0039    C
0040    C       V03-003 SAR0094         Sharon A. Reynolds,     20-Jun-1983
0041    C               Changed the carriage control in the 'format' statements
0042    C               for use with ERF.
0043    C
0044    C       V03-002 SAR009          Sharon Reynolds,        7-Apr-1983
0045    C               Removed search_sid and entry_type as call parameters.
0046    C               Added include statement for the 'msghdr' common which
0047    C               defines the field definitions for the entry header
0048    C               and equivalenced search_sid and entry_type to the
0049    C               appropriate definitions.
0050    C
0051    C       v03-001 BP0004          Brian Porter,           05-APR-1982
0052    C               Added stuff for mscp.
0053    C--
0054    C--
0055
0056    C++
0057    C
```

```
0058     C        +---------------------------------+
0059     C        |              flink1             |
0060     C        +---------------------------------+
0061     C        |              blink1             |
0062     C        +---------------------------------+
0063     C        |           logging sid           |
0064     C        +---------------------------------+
0065     C        |         root name flink         |
0066     C        +---------------------------------+
0067     C        |         root name blink         |
0068     C        +---------------------------------+
0069     C        |        name entry count         |
0070     C        +---------------------------------+
0071     C
0072     C
0073     C        +---------------------------------+
0074     C        |              flink2             |
0075     C        +---------------------------------+
0076     C        |              blink2             |
0077     C        +---------------------------------+
0078     C        |                                 |
0079     C        +--                             --+
0080     C        |        16 bytes for name        |
0081     C        +--                             --+
0082     C        |             string              |
0083     C        +--                             --+
0084     C        |                                 |
0085     C        +---------------------------------+
0086     C        |          root unit flink        |
0087     C        +---------------------------------+
0088     C        |          root unit blink        |
0089     C        +---------------------------------+
0090     C        |          unit entry count       |
0091     C        +---------------------------------+
0092     C
0093     C
0094     C        +---------------------------------+
0095     C        |              flink3             |
0096     C        +---------------------------------+
0097     C        |              blink3             |
0098     C        +---------------------------------+
0099     C        |          ucb unit number        |
0100     C        +---------------------------------+
0101     C        |     device hard error count     |
0102     C        +---------------------------------+
0103     C        |     device soft error count     |
0104     C        +---------------------------------+
0105     C        |     timeout hard error count    |
0106     C        +---------------------------------+
0107     C        |     timeout soft error count    |
0108     C        +---------------------------------+
0109     C        |    ucb unit operation count     |
0110     C        +---------------------------------+
0111     C        |     ucb unit error count        |
0112     C        +---------------------------------+
0113     C        |           pack count            |
0114     C        +---------------------------------+
```

```
0115    C       |                                   |
0116    C       +--                               --+
0117    C       |          current label            |
0118    C       +--        previous error         --+
0119    C       |                                   |
0120    C       +-----------------------------------+
0121    C--
0122
0123
0124            subroutine rollup (search_name_length,search_name_string,
0125            1 search_unit,iosb,ucb$l_opcnt,ucb$w_errcnt)
0126
0127
0128            Include 'SRC$:MSGHDR.FOR /NOLIST'       ! EMB entry header definitions
0187
0188
0189            byte            lun
0190
0191            integer*4       buffer0(3)
0192
0193            integer*4       root_logging_sid_flink
0194
0195            integer*4       root_logging_sid_blink
0196
0197            integer*4       logging_sid_entry_count
0198
0199            Equivalence     (emb$l_hd_sid,search_sid)
0200
0201            Equivalence     (emb$w_hd_entry,entry_type)
0202
0203            equivalence     (buffer0(1),root_logging_sid_flink)
0204
0205            equivalence     (buffer0(2),root_logging_sid_blink)
0206
0207            equivalence     (buffer0(3),logging_sid_entry_count)
0208
0209            integer*4       buffer1(6)
0210
0211            integer*4       flink1
0212
0213            integer*4       blink1
0214
0215            integer*4       logging_sid
0216
0217            integer*4       root_name_flink
0218
0219            integer*4       root_name_blink
0220
0221            integer*4       name_entry_count
0222
0223            equivalence     (buffer1(1),flink1)
0224
0225            equivalence     (buffer1(2),blink1)
0226
0227            equivalence     (buffer1(3),logging_sid)
0228
0229            equivalence     (buffer1(4),root_name_flink)
```

```
0230
0231            equivalence     (buffer1(5),root_name_blink)
0232
0233            equivalence     (buffer1(6),name_entry_count)
0234
0235            integer*4       buffer2(9)
0236
0237            integer*4       flink2
0238
0239            integer*4       blink2
0240
0241            byte            name_string_array(0:15)
0242
0243            byte            name_string_length
0244
0245            character*15    name_string
0246
0247            equivalence     (name_string_array(0),name_string_length)
0248
0249            equivalence     (name_string_array(1),name_string)
0250
0251            integer*4       root_unit_flink
0252
0253            integer*4       root_unit_blink
0254
0255            integer*4       unit_entry_count
0256
0257            equivalence     (buffer2(1),flink2)
0258
0259            equivalence     (buffer2(2),blink2)
0260
0261            equivalence     (buffer2(3),name_string_array)
0262
0263            equivalence     (buffer2(7),root_unit_flink)
0264
0265            equivalence     (buffer2(8),root_unit_blink)
0266
0267            equivalence     (buffer2(9),unit_entry_count)
0268
0269            integer*4       buffer3(13)
0270
0271            integer*4       flink3
0272
0273            integer*4       blink3
0274
0275            integer*4       ucb_unit_number
0276
0277            integer*4       device_hard_error_count
0278
0279            integer*4       device_soft_error_count
0280
0281            integer*4       timeout_hard_error_count
0282
0283            integer*4       timeout_soft_error_count
0284
0285            integer*4       ucb_unit_operation_count
0286
```

```
0287          integer*4       ucb_unit_error_count
0288
0289          integer*4       pack_count
0290
0291          byte            previous_label_at_error_array(12)
0292
0293          character*12    previous_label_at_error
0294
0295          equivalence     (previous_label_at_error_array,previous_label_at_error)
0296
0297          equivalence     (buffer3(1),flink3)
0298
0299          equivalence     (buffer3(2),blink3)
0300
0301          equivalence     (buffer3(3),ucb_unit_number)
0302
0303          equivalence     (buffer3(4),device_hard_error_count)
0304
0305          equivalence     (buffer3(5),device_soft_error_count)
0306
0307          equivalence     (buffer3(6),timeout_hard_error_count)
0308
0309          equivalence     (buffer3(7),timeout_soft_error_count)
0310
0311          equivalence     (buffer3(8),ucb_unit_operation_count)
0312
0313          equivalence     (buffer3(9),ucb_unit_error_count)
0314
0315          equivalence     (buffer3(10),pack_count)
0316
0317          equivalence     (buffer3(11),previous_label_at_error_array)
0318
0319          integer*4       logging_sid_entry_address
0320
0321          integer*4       name_entry_address
0322
0323          integer*4       unit_entry_address
0324
0325          character*15    search_name
0326
0327          character*12    current_label_at_error
0328
0329          logical*1       lib$extzv
0330
0331          logical*1       lib$get_vm
0332
0333          integer*4       search_sid
0334
0335          byte            search_name_length
0336
0337          character*15    search_name_string
0338
0339          integer*2       search_unit
0340
0341          integer*2       entry_type
0342
0343          integer*4       compress4
```

```
0344
0345            integer*4       column
0346
0347            integer*4       iosb
0348
0349            integer*4       ucb$l_opcnt
0350
0351            integer*2       ucb$w_errcnt
0352
0353            integer*4       device_count
0354
0355            integer*4       insert_blink
0356
0357
0358
0359            call movc5 (%val(search_name_length),%ref(search_name_string),%val(42),
0360            1 %val(15),%ref(search_name))
0361
0362            logging_sid_entry_address = root_logging_sid_flink
0363
0364            do 50,i = 1,logging_sid_entry_count
0365
0366            call movc3 (%val(24),%val(logging_sid_entry_address),buffer1)
0367
0368    5       if (search_sid .eq. logging_sid) then
0369
0370            name_entry_address = root_name_flink
0371
0372            do 40,j = 1,name_entry_count
0373
0374            call movc3 (%val(36),%val(name_entry_address),buffer2)
0375
0376    10      if (search_name .eq. name_string) then
0377
0378            unit_entry_address = root_unit_flink
0379
0380            do 30,k = 1,unit_entry_count
0381
0382            call movc3 (%val(52),%val(unit_entry_address),buffer3)
0383
0384    15      if (search_unit .eq. ucb_unit_number) then
0385
0386            if (
0387            1 ucb$l_opcnt .ne. -1
0388            1 .and.
0389            1 ucb$w_errcnt .ne. -1
0390            1 ) then
0391
0392            if (
0393            1 ucb_unit_operation_count .eq. -1
0394            1 .and.
0395            1 ucb_unit_error_count .eq. -1
0396            1 ) then
0397
0398            ucb_unit_operation_count = 0
0399
0400            ucb_unit_error_count = 0
```

```
0401            endif
0402
0403            ucb_unit_operation_count = ucb$l_opcnt
0404
0405            ucb_unit_error_count = ucb$w_errcnt
0406            endif
0407
0408            if (iosb .ne. -1) then
0409
0410            if (
0411            1 entry_type .eq. 1
0412            1 .or.
0413            1 entry_type .eq. 98
0414            1 .or.
0415            1 entry_type .eq. 99
0416            1 .or.
0417            1 entry_type .eq. 100
0418            1 ) then
0419
0420            if (lib$extzv(0,1,iosb) .eq. 1) then
0421
0422            if (device_soft_error_count .eq. -1) then
0423
0424            device_soft_error_count = 0
0425            endif
0426
0427            device_soft_error_count = device_soft_error_count + 1
0428            else
0429
0430            if (device_hard_error_count .eq. -1) then
0431
0432            device_hard_error_count = 0
0433            endif
0434
0435            device_hard_error_count = device_hard_error_count + 1
0436            endif
0437
0438            else if (entry_type .eq. 96) then
0439
0440            if (lib$extzv(0,1,iosb) .eq. 1) then
0441
0442            if (timeout_soft_error_count .eq. -1) then
0443
0444            timeout_soft_error_count = 0
0445            endif
0446
0447            timeout_soft_error_count = timeout_soft_error_count + 1
0448            else
0449
0450            if (timeout_hard_error_count .eq. -1) then
0451
0452            timeout_hard_error_count = 0
0453            endif
0454
0455            timeout_hard_error_count = timeout_hard_error_count + 1
0456            endif
0457            endif
```

```
0458            endif
0459
0460            call get_current_label(3,search_sid,search_name_length,
0461            1 search_name_string,search_unit,%ref(current_label_at_error),*20)
0462
0463            if (current_label_at_error .ne. previous_label_at_error) then
0464
0465            pack_count = pack_count + 1
0466
0467            previous_label_at_error = current_label_at_error
0468            endif
0469
0470    20      call movc3 (%val('44),ucb_unit_number,%val(unit_entry_address + 8))
0471
0472            return
0473            endif
0474
0475            insert_blink = blink3
0476
0477            if (ucb_unit_number .gt. search_unit) goto 35
0478
0479            unit_entry_address = flink3
0480
0481    30      continue
0482
0483            insert_blink = root_unit_blink
0484
0485    35      call movc5 (%val(0),,%val(0),%val(52),buffer3)
0486
0487            if (lib$get_vm(((52+7)/8)*8,unit_entry_address)) then
0488
0489            call insque (%val(unit_entry_address),%val(insert_blink))
0490
0491            ucb_unit_number = search_unit
0492
0493            device_hard_error_count = -1
0494
0495            device_soft_error_count = -1
0496
0497            timeout_hard_error_count = -1
0498
0499            timeout_soft_error_count = -1
0500
0501            ucb_unit_operation_count = -1
0502
0503            ucb_unit_error_count = -1
0504
0505            unit_entry_count = unit_entry_count + 1
0506
0507            call movl (unit_entry_count,%val(name_entry_address + 32))
0508
0509            goto 15
0510            endif
0511
0512            return
0513            endif
0514
```

```
0515            name_entry_address = flink2
0516
0517    40      continue
0518
0519            call movc5 (%val(0),,%val(0),%val(36),buffer2)
0520
0521            if (lib$get_vm(((36+7)/8)*8,name_entry_address)) then
0522
0523            call insque (%val(name_entry_address),%val(root_name_blink))
0524
0525            name_string_length = search_name_length
0526
0527            name_string = search_name
0528
0529            root_unit_flink = name_entry_address + 24
0530
0531            root_unit_blink = root_unit_flink
0532
0533            call movc3 (%val(28),name_string_length,%val(name_entry_address + 8))
0534
0535            name_entry_count = name_entry_count + 1
0536
0537            call movl (name_entry_count,%val(logging_sid_entry_address + 20))
0538
0539            goto 10
0540            endif
0541
0542            return
0543            endif
0544
0545            logging_sid_entry_address = flink1
0546
0547    50      continue
0548
0549            call movc5 (%val(0),,%val(0),%val(24),buffer1)
0550
0551            if (lib$get_vm(((24+7)/8)*8,logging_sid_entry_address)) then
0552
0553            if (logging_sid_entry_count .eq. 0) then
0554
0555            root_logging_sid_flink = %loc(root_logging_sid_flink)
0556
0557            root_logging_sid_blink = root_logging_sid_flink
0558            endif
0559
0560            call insque (%val(logging_sid_entry_address),
0561            1 %val(root_logging_sid_blink))
0562
0563            logging_sid = search_sid
0564
0565            root_name_flink = logging_sid_entry_address + 12
0566
0567            root_name_blink = root_name_flink
0568
0569            call movc3 (%val(16),logging_sid,%val(logging_sid_entry_address + 8))
0570
0571            logging_sid_entry_count = logging_sid_entry_count + 1
```

```
0572
0573            goto 5
0574            endif
0575
0576            return
0577
0578
0579
0580
0581            entry display_rollup (lun)
0582
0583
0584
0585
0586            logging_sid_entry_address = root_logging_sid_flink
0587
0588            do 100,i = 1,logging_sid_entry_count
0589
0590            call movc3 (%val(24),%val(logging_sid_entry_address),buffer1)
0591
0592            call frctof (lun)
0593
0594            call linchk (lun,7)
0595
0596            write(lun,55) logging_sid
0597     55     format(/' ','DEVICE ROLLUP LOGGED BY SID ',z8.8,:a,/)
0598
0599            write(lun,60)
0600     60     format(/' ','DEVICE',t23,'ERROR BITS',t51,'ERRORS THIS',
0601            1 t64,'QIOS THIS',/,
0602            1 t26,'SET',t38,'QIO TIMEOUT',t53,'SESSION',t65,'SESSION')
0603
0604            write(lun,65)
0605     65     format(/' ',t21,'[HARD]',t28,'[SOFT]',
0606            1 t36,'[HARD]',t43,'[SOFT]')
0607
0608            device_count = 0
0609
0610            name_entry_address = root_name_flink
0611
0612            do 90,j = 1,name_entry_count
0613
0614            call movc3 (%val(36),%val(name_entry_address),buffer2)
0615
0616            unit_entry_address = root_unit_flink
0617
0618            do 80,k = 1,unit_entry_count
0619
0620            if (device_count .eq. 23) then
0621
0622            call frctof (lun)
0623
0624            device_count = 0
0625            endif
0626
0627            call movc3 (%val(44),%val(unit_entry_address),buffer3)
0628
```

```
0629            call linchk (lun,2)
0630
0631            write(lun,66)
0632    66      format(' ',:)
0633
0634            write(lun,67) name_string,ucb_unit_number
0635    67      format(' ',' ',a<name_string_length>,
0636            1 i<compress4 (ucb_unit_number)>,':')
0637
0638            column = name_string_length + (compress4 (ucb_unit_number)) + 2
0639
0640            if (device_hard_error_count .EQ. -1) then
0641            Device_hard_error_count = 0
0642            Endif
0643
0644            if (device_soft_error_count .EQ. -1) then
0645            Device_soft_error_count = 0
0646            Endif
0647
0648            if (timeout_hard_error_count .EQ. -1) then
0649            Timeout_hard_error_count = 0
0650            Endif
0651
0652            if (timeout_soft_error_count .EQ. -1) then
0653            Timeout_soft_error_count = 0
0654            Endif
0655
0656            if (ucb_unit_error_count .EQ. -1) then
0657            Ucb_unit_error_count = 0
0658            Endif
0659
0660            if (ucb_unit_operation_count .EQ. -1) then
0661            Ucb_unit_operation_count = 0
0662            Endif
0663
0664            write(lun,69) device_hard_error_count,device_soft_error_count,
0665            1 timeout_hard_error_count,timeout_soft_error_count,
0666            2 ucb_unit_error_count,ucb_unit_operation_count
0667
0668    69      format(T21,I5,'.',T28,I5,'.',
0669            1 T36,I5,'.',T43,I5,'.',
0670            2 T53,I6,'.',T64,I7,'.')
0671
0672            device_count = device_count + 1
0673
0674            unit_entry_address = flink3
0675
0676    80      continue
0677
0678            name_entry_address = flink2
0679
0680    90      continue
0681
0682            logging_sid_entry_address = flink1
0683
0684    100     continue
0685
```

```
                return

                end
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	1477	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	288	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	588	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	EMB	512	PIC OVR REL GBL SHR NOEXE RD WRT LONG
	Total Space Allocated	2865	

ENTRY POINTS

Address	Type	Name	Address	Type	Name
0-00000357		DISPLAY_ROLLUP	0-00000000		ROLLUP

VARIABLES

Address	Type	Name	Address	Type	Name
2-0000005C	I*4	BLINK1	2-00000038	I*4	BLINK2
2-00000004	I*4	BLINK3	2-000000A4	I*4	COLUMN
2-0000008B	CHAR	CURRENT_LABEL_AT_ERROR	2-000000A8	I*4	DEVICE_COUNT
2-0000000C	I*4	DEVICE_HARD_ERROR_COUNT	2-00000010	I*4	DEVICE_SOFT_ERROR_COUNT
3-00000000	I*4	EMB$L_HD_SID	3-00000C04	I*2	EMB$W_HD_ENTRY
3-0000000E	I*2	EMB$W_HD_ERRSEQ	3-00000004	I*2	ENTRY_TYPE
2-00000058	I*4	FLINK1	2-00000034	I*4	FLINK2
2-00000000	I*4	FLINK3	2-000000B0	I*4	I
2-000000AC	I*4	INSERT_BLINK	AP-00000010@	I*4	IOSB
2-000000B4	I*4	J	2-000000B8	I*4	K
2-00000060	I*4	LOGGING_SID	2-00000098	I*4	LOGGING_SID_ENTRY_ADDRESS
2-00000078	I*4	LOGGING_SID_ENTRY_COUNT	AP-00000004@	L*1	LUN
2-0000009C	I*4	NAME_ENTRY_ADDRESS	2-0000006C	I*4	NAME_ENTRY_COUNT
2-0000003D	CHAR	NAME_STRING	2-0000003C	L*1	NAME_STRING_LENGTH
2-00000024	I*4	PACK_COUNT	2-00000028	CHAR	PREVIOUS_LABEL_AT_ERROR
2-00000074	I*4	ROOT_LOGGING_SID_BLINK	2-00000070	I*4	ROOT_LOGGING_SID_FLINK
2-00000068	I*4	ROOT_NAME_BLINK	2-00000064	I*4	ROOT_NAME_FLINK
2-00000050	I*4	ROOT_UNIT_BLINK	2-0000004C	I*4	ROOT_UNIT_FLINK
2-0000007C	CHAR	SEARCH_NAME	AP-00000004@	L*1	SEARCH_NAME_LENGTH
AP-00000008@	CHAR	SEARCH_NAME_STRING	3-00000000	I*4	SEARCH_SID
AP-0000000C@	I*2	SEARCH_UNIT	2-00000014	I*4	TIMEOUT_HARD_ERROR_COUNT
2-00000018	I*4	TIMEOUT_SOFT_ERROR_COUNT	AP-00000014@	I*4	UCB$L_OPCNT
AP-00000018@	I*2	UCB$W_ERRCNT	2-00000020	I*4	UCB_UNIT_ERROR_COUNT
2-00000008	I*4	UCB_UNIT_NUMBER	2-0000001C	I*4	UCB_UNIT_OPERATION_COUNT
2-000000A0	I*4	UNIT_ENTRY_ADDRESS	2-00000054	I*4	UNIT_ENTRY_COUNT

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000070	I*4	BUFFER0	12	(3)
2-00000058	I*4	BUFFER1	24	(6)
2-00000034	I*4	BUFFER2	36	(9)
2-00000000	I*4	BUFFER3	52	(13)
3-00000000	L*1	EMB	512	(0:511)
3-00000006	I*4	EMB$Q_HD_TIME	8	(2)
2-0000003C	L*1	NAME_STRING_ARRAY	16	(0:15)
2-00000028	L*1	PREVIOUS_LABEL_AT_ERROR_ARRAY	12	(12)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
0-0000004P	5	0-0000007C	10	0-000000AE	15	0-000001CB	20	**	30	0-00000204	35
**	40	**	50	1-00000020	55'	1-00000049	60'	1-000000AD	65'	1-000000DA	66'
1-000000DF	67'	1-000000F5	69'	**	80	**	90	**	100		

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name	Type	Name
I*4	COMPRESS4		FRCTOF		GET_CURRENT_LABEL
	INSQUE	L*1	LIB$EXTZV	L*1	LIB$GET_VM
	LINCHK		MOVC3		MOVC5
	MOVL				

COMMAND QUALIFIERS

FORTRAN /LIS=LIS$:ROLLUP/OBJ=OBJ$:ROLLUP MSRC$:ROLLUP

/CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
/DEBUG=(NOSYMBOLS,TRACEBACK)
/STANDARD=(NOSYNTAX,NOSOURCE_FORM)
/SHOW=(NOPREPROCESSOR,NOINCLUDE,MAP)
/F77 /NOG_FLOATING /I4 /OPTIMIZE /WARNINGS /NOD_LINES /NOCROSS_REFERENCE /NOMACHINE_CODE /CONTINUATIONS=19

COMPILATION STATISTICS

Run Time: 5.92 seconds
Elapsed Time: 14.57 seconds
Page Faults: 197
Dynamic Memory: 218 pages

0153 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RECSELECT
LIS

RLDISK
LIS

PUDRIVER
LIS

PCL11T
LIS

ROLLUP
LIS

RXDISK
LIS

PCL11R
LIS

SB11
LIS

RKDISK
LIS