


```
RRRRRRRR  KK      KK  DDDDDDDD  IIIIII  SSSSSSSS  KK      KK
RRRRRRRR  KK      KK  DDDDDDDD  IIIIII  SSSSSSSS  KK      KK
RR      RR  KK      KK  DD      DD  II      SS      KK      KK
RR      RR  KK      KK  DD      DD  II      SS      KK      KK
RR      RR  KK      KK  DD      DD  II      SS      KK      KK
RRRRRRRR  KKKKKK  DD      DD  II      SSSSSS  KKKKKK  KK      KK
RRRRRRRR  KKKKKK  DD      DD  II      SSSSSS  KKKKKK  KK      KK
RR      RR  KK      KK  DD      DD  II      SS      KK      KK
RR      RR  KK      KK  DD      DD  II      SS      KK      KK
RR      RR  KK      KK  DD      DD  II      SS      KK      KK
RR      RR  KK      KK  DDDDDDDD  IIIIII  SSSSSSSS  KK      KK
RR      RR  KK      KK  DDDDDDDD  IIIIII  SSSSSSSS  KK      KK
```

```
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      SUBROUTINE RKDISK (LUN)
0002      C
0003      C Version:      'V04-000'
0004      C
0005      C*****
0006      C*
0007      C* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0008      C* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0009      C* ALL RIGHTS RESERVED.
0010      C*
0011      C* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0012      C* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0013      C* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0014      C* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0015      C* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0016      C* TRANSFERRED.
0017      C*
0018      C* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0019      C* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0020      C* CORPORATION.
0021      C*
0022      C* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0023      C* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0024      C*
0025      C*
0026      C*****
0027      C
0028      C
0029      C
0030      C      AUTHOR  BRIAN PORTER      CREATION DATE  31-MAR-1979
0031      C
0032      C
0033      C++
0034      C      Functional description:
0035      C
0036      C      This module displays entries made for the RK611 controller.
0037      C      The format of the error log packet after the 4 longword header
0038      C      is as follows.
0039      C
0040      C      +-----+
0041      C      |                rkcs1                |
0042      C      +-----+
0043      C      |                rkwc                |
0044      C      +-----+
0045      C      |                rkba                |
0046      C      +-----+
0047      C      |                rkda                |
0048      C      +-----+
0049      C      |                rkcs2               |
0050      C      +-----+
0051      C      |                rkds                |
0052      C      +-----+
0053      C      |                rker                |
0054      C      +-----+
0055      C      |                rkas                |
0056      C      +-----+
0057      C      |                rkdc                |
0057      C      +-----+
```



```

0058      c      +-----+
0059      c      |               |
0060      c      +-----+
0061      c      |               |
0062      c      +-----+
0063      c      |               |
0064      c      +-----+
0065      c      |               |
0066      c      +-----+
0067      c      |               |
0068      c      +-----+
0069      c      |   uba datapath number   |
0070      c      +-----+
0071      c      |   uba datapath register   |
0072      c      +-----+
0073      c      |   uba final mapping register   |
0074      c      +-----+
0075      c      |   uba previous mapping register   |
0076      c      +-----+
0077      c
0078      c      Modified by:
0079      c
0080      c      V03-003  SAR0231          Sharon A. Reynolds,      28-Mar-1984
0081      c      Changed the call to UCBSL_OWNUIC to ORBSL_OWNER.
0082      c
0083      c      V03-002  SAR0092          Sharon A. Reynolds,      20-Jun-1983
0084      c      Changed the carriage control in the 'format' statements
0085      c      for use with ERF.
0086      c
0087      c      V03-001  SAR0049          Sharon A. Reynolds,      13-Jun-1983
0088      c      Removed brief/cryptic support.
0089      c
0090      c      v02-005  BP0005          Brian Porter,             23-NOV-1981
0091      c      minor edit.
0092      c
0093      c      v02-004  BP0004          Brian Porter,             03-NOV-1981
0094      c      Added device attention support.
0095      c
0096      c      v02-003  BP0003          Brian Porter,             30-SEP-1981
0097      c      Corrected call to uba_mapping.
0098      c
0099      c      v02-002  BP0002          Brian Porter,             23-JUL-1981
0100      c      Added new uba handling routines.
0101      c
0102      c      v02-001  BP0001          Brian Porter,             29-JUN-1981
0103      c      Made the default register output 16-bit. Added call
0104      c      to new DHEAD and LOGGER modules. Removed call to
0105      c      UNUSED_BITS, ILLEGAL_BITS and REGCHK.
0106      c**
0107      c--
0108
0109      INCLUDE 'SRC$:MSGHDR.FOR /NOLIST'
0168      INCLUDE 'SRC$:DEVERR.FOR /NOLIST'
0269
0270      BYTE          LUN
0271
0272      INTEGER*4      RKCS1

```



```
0273 INTEGER*4 RKWC
0274 INTEGER*4 RKBA
0275 INTEGER*4 RKDA
0276 INTEGER*4 RKCS2
0277 INTEGER*4 RKDS
0278 INTEGER*4 RKER
0279 INTEGER*4 RKAS
0280 INTEGER*4 RKDC
0281 INTEGER*4 RKMR1
0282 INTEGER*4 RKEC1
0283 INTEGER*4 RKEC2
0284 INTEGER*4 RKMR2
0285 INTEGER*4 RKMR3
0286 INTEGER*4 UBA_REGS(4)
0287 INTEGER*4 FIELD
0288 INTEGER*4 DRIVE_FUNC
0289 INTEGER*4 COMPRESSC
0290 INTEGER*4 COMPRESS4
0291
0292 logical*1 diagnostic_mode
0293
0294 PARAMETER TIMEOUT = 96
0295 PARAMETER XFER_ERR = 32
0296 PARAMETER XFER_CMD = 8
0297 PARAMETER RK06 = 1
0298 PARAMETER RK07 = 2
0299 PARAMETER SEEK = 7
0300
0301 EQUIVALENCE (RKCS1,EMBSL_DV_REGSAV(0))
0302 EQUIVALENCE (RKWC,EMBSL_DV_REGSAV(1))
0303 EQUIVALENCE (RKBA,EMBSL_DV_REGSAV(2))
0304 EQUIVALENCE (RKDA,EMBSL_DV_REGSAV(3))
0305 EQUIVALENCE (RKCS2,EMBSL_DV_REGSAV(4))
0306 EQUIVALENCE (RKDS,EMBSL_DV_REGSAV(5))
0307 EQUIVALENCE (RKER,EMBSL_DV_REGSAV(6))
0308 EQUIVALENCE (RKAS,EMBSL_DV_REGSAV(7))
0309 EQUIVALENCE (RKDC,EMBSL_DV_REGSAV(8))
0310 EQUIVALENCE (RKMR1,EMBSL_DV_REGSAV(9))
0311 EQUIVALENCE (RKEC1,EMBSL_DV_REGSAV(10))
0312 EQUIVALENCE (RKEC2,EMBSL_DV_REGSAV(11))
0313 EQUIVALENCE (RKMR2,EMBSL_DV_REGSAV(12))
0314 EQUIVALENCE (RKMR3,EMBSL_DV_REGSAV(13))
0315 EQUIVALENCE (UBA_REGS,EMBSL_DV_REGSAV(14))
0316
0317 CHARACTER*8 OFFSET_DIR(0:1)
0318 DATA OFFSET_DIR(0) /'FORWARD*'/
0319 DATA OFFSET_DIR(1) /'REVERSE*'/
0320
0321 CHARACTER*5 RK06_OFFSET(1:5)
0322 DATA RK06_OFFSET(1) /'25*'/
0323 DATA RK06_OFFSET(2) /'200*'/
0324 DATA RK06_OFFSET(3) /'400*'/
0325 DATA RK06_OFFSET(4) /'800*'/
0326 DATA RK06_OFFSET(5) /'1200*'/
0327
0328 CHARACTER*5 RK07_OFFSET(1:5)
0329 DATA RK07_OFFSET(1) /'12.5*'/
```

```
0330 DATA RK07_OFFSET(2) /*100*/
0331 DATA RK07_OFFSET(3) /*200*/
0332 DATA RK07_OFFSET(4) /*400*/
0333 DATA RK07_OFFSET(5) /*600*/
0334
0335 CHARACTER*17 RK_FUNC(0:15)
0336 DATA RK_FUNC(0) /*SELECT DRIVE*/
0337 DATA RK_FUNC(1) /*PACK ACKNOWLEDGE*/
0338 DATA RK_FUNC(2) /*DRIVE CLEAR*/
0339 DATA RK_FUNC(3) /*UNLOAD*/
0340 DATA RK_FUNC(4) /*START SPINDLE*/
0341 DATA RK_FUNC(5) /*RECALIBRATE*/
0342 DATA RK_FUNC(6) /*OFFSET*/
0343 DATA RK_FUNC(7) /*SEEK*/
0344 DATA RK_FUNC(8) /*READ DATA*/
0345 DATA RK_FUNC(9) /*WRITE DATA*/
0346 DATA RK_FUNC(10) /*READ HEADER*/
0347 DATA RK_FUNC(11) /*WRITE HEADER*/
0348 DATA RK_FUNC(12) /*WRITE CHECK*/
0349 DATA RK_FUNC(13) /*ILLEGAL FUNCTION*/
0350 DATA RK_FUNC(14) /*ILLEGAL FUNCTION*/
0351 DATA RK_FUNC(15) /*ILLEGAL FUNCTION*/
0352
0353 CHARACTER*7 RKCS1_1(0:0)
0354 DATA RKCS1_1(0) /*GO BIT*/
0355
0356 CHARACTER*28 RKCS1_2(5:9)
0357 DATA RKCS1_2(5) /*TRANSFER ERROR (VMS)*/
0358 DATA RKCS1_2(6) /*INTERRUPT ENABLE*/
0359 DATA RKCS1_2(7) /*CONTROLLER READY*/
0360 DATA RKCS1_2(8) /*EXTENDED BUS ADDRESS BIT 16*/
0361 DATA RKCS1_2(9) /*EXTENDED BUS ADDRESS BIT 17*/
0362
0363 CHARACTER*5 RKCS1_10(0:1)
0364 DATA RKCS1_10(0) /*RK06*/
0365 DATA RKCS1_10(1) /*RK07*/
0366
0367 CHARACTER*20 RKCS1_3(11:11)
0368 DATA RKCS1_3(11) /*CONTROLLER TIME-OUT*/
0369
0370 CHARACTER*14 RK_FORMAT(0:1)
0371 DATA RK_FORMAT(0) /*16-BIT FORMAT*/
0372 DATA RK_FORMAT(1) /*18-BIT FORMAT*/
0373
0374 CHARACTER*33 RKCS1_4(13:15)
0375 DATA RKCS1_4(13) /*DRIVE-TO-CONTROLLER PARITY ERROR*/
0376 DATA RKCS1_4(14) /*DRIVE INTERRUPT*/
0377 DATA RKCS1_4(15) /*COMBINED ERROR*/
0378
0379 CHARACTER*30 RKCS2_1(3:15)
0380 DATA RKCS2_1(3) /*RELEASE*/
0381 DATA RKCS2_1(4) /*BUS ADDRESS INCREMENT INHIBIT*/
0382 DATA RKCS2_1(5) /*SUBSYSTEM CLEAR*/
0383 DATA RKCS2_1(6) /*INPUT READY*/
0384 DATA RKCS2_1(7) /*OUTPUT READY*/
0385 DATA RKCS2_1(8) /*UNIT FIELD ERROR*/
0386 DATA RKCS2_1(9) /*MULTIPLE DRIVE SELECT*/
```



```
0387 DATA RKCS2_1(10) /*PROGRAMMING ERROR*/
0388 DATA RKCS2_1(11) /*NON-EXISTENT MEMORY*/
0389 DATA RKCS2_1(12) /*NON-EXISTENT DRIVE*/
0390 DATA RKCS2_1(13) /*UNIBUS PARITY ERROR*/
0391 DATA RKCS2_1(14) /*WRITE CHECK ERROR*/
0392 DATA RKCS2_1(15) /*DATA LATE ERROR*/
0393
0394 CHARACTER*16 RKDS_1(0:0)
0395 DATA RKDS_1(0) /*DRIVE AVAILABLE*/
0396
0397 CHARACTER*16 RKDS_2(2:7)
0398 DATA RKDS_2(2) /*OFFSET MODE*/
0399 DATA RKDS_2(3) /*DRIVE AC LO*/
0400 DATA RKDS_2(4) /*SPEED LOSS*/
0401 DATA RKDS_2(5) /*DRIVE OFF TRACK*/
0402 DATA RKDS_2(6) /*VOLUME VALID*/
0403 DATA RKDS_2(7) /*DRIVE READY*/
0404
0405 CHARACTER*14 RK_DRIVETYPE(0:1)
0406 DATA RK_DRIVETYPE(0) /*DRIVE IS RK06*/
0407 DATA RK_DRIVETYPE(1) /*DRIVE IS RK07*/
0408
0409 CHARACTER*16 RKDS_3(11:11)
0410 DATA RKDS_3(11) /*WRITE PROTECTED*/
0411
0412 CHARACTER*24 RKDS_4(13:15)
0413 DATA RKDS_4(13) /*POSITIONING IN PROGRESS*/
0414 DATA RKDS_4(14) /*CURRENT DRIVE ATTENTION*/
0415 DATA RKDS_4(15) /*STATUS VALID*/
0416
0417 CHARACTER*33 RKER_1(0:15)
0418 DATA RKER_1(0) /*ILLEGAL FUNCTION*/
0419 DATA RKER_1(1) /*SEEK INCOMPLETE*/
0420 DATA RKER_1(2) /*NON-EXECUTABLE FUNCTION*/
0421 DATA RKER_1(3) /*CONTROLLER-TO-DRIVE PARITY ERROR*/
0422 DATA RKER_1(4) /*FORMAT ERROR*/
0423 DATA RKER_1(5) /*DRIVE TYPE ERROR*/
0424 DATA RKER_1(6) /*ERROR CORRECTION HARD*/
0425 DATA RKER_1(7) /*BAD SECTOR ERROR*/
0426 DATA RKER_1(8) /*HEADER VERTICAL CHECK ERROR*/
0427 DATA RKER_1(9) /*CYLINDER OVERFLOW ERROR*/
0428 DATA RKER_1(10) /*INVALID DISK ADDRESS ERROR*/
0429 DATA RKER_1(11) /*WRITE LOCK ERROR*/
0430 DATA RKER_1(12) /*DRIVE TIMING ERROR*/
0431 DATA RKER_1(13) /*OPERATION INCOMPLETE*/
0432 DATA RKER_1(14) /*DRIVE UNSAFE*/
0433 DATA RKER_1(15) /*DATA CHECK*/
0434
0435 CHARACTER*21 V1RKMR2(4:15)
0436 DATA V1RKMR2(4) /*SERVO SIGNAL PRESENT*/
0437 DATA V1RKMR2(5) /*HEADS HOME*/
0438 DATA V1RKMR2(6) /*BRUSHES HOME*/
0439 DATA V1RKMR2(7) /*DOOR LATCHED*/
0440 DATA V1RKMR2(8) /*CARTRIDGE PRESENT*/
0441 DATA V1RKMR2(9) /*SPEED OK*/
0442 DATA V1RKMR2(10) /*FORWARD*/
0443 DATA V1RKMR2(11) /*REVERSE*/
```

```

0444 DATA V1RKMR2(12) /*HEADS LOADING*/
0445 DATA V1RKMR2(13) /*RETURN TO ZERO*/
0446 DATA V1RKMR2(14) /*UNLOADING HEADS*/
0447 DATA V1RKMR2(15) /*ODD PARITY BIT*/
0448
0449 CHARACTER*29 V1RKMR3(4:15)
0450 DATA V1RKMR3(4) /*SECTOR ERROR*/
0451 DATA V1RKMR3(5) /*WRITE CURRENT, NO WRITE GATE*/
0452 DATA V1RKMR3(6) /*WRITE GATE, NO TRANSITIONS*/
0453 DATA V1RKMR3(7) /*HEAD FAULT*/
0454 DATA V1RKMR3(8) /*MULTIPLE HEAD SELECT*/
0455 DATA V1RKMR3(9) /*INDEX ERROR*/
0456 DATA V1RKMR3(10) /*TRIBIT ERROR*/
0457 DATA V1RKMR3(11) /*SERVO SIGNAL ERROR*/
0458 DATA V1RKMR3(12) /*SEEK AND NO MOTION*/
0459 DATA V1RKMR3(13) /*LIMIT DETECT ON SEEK*/
0460 DATA V1RKMR3(14) /*SERVO UNSAFE*/
0461 DATA V1RKMR3(15) /*ODD PARITY BIT*/
0462
0463 CALL FRCTOF (LUN)
0464
0465 call dhead1 (lun,'UBA RK611')
0466
0467 diagnostic_mode = .false.
0468
0469 if (lib$extzv(5,1,rkmr1) .eq. 1) diagnostic_mode = .true.
0470
0471 call linchk (lun,2)
0472
0473
0474 20 write(lun,20) rkcs1
0475 format(/' ',T8,'RKCS1',t24,z8.4)
0476
0477 if (.not. diagnostic_mode) then
0478
0479 CALL OUTPUT (LUN,RKCS1,RKCS1_1,0,0,0,'0')
0480
0481 DRIVE_FUNC=LIB$EXTZV(1,4,RKCS1)
0482
0483 CALL LINCHK (LUN,1)
0484
0485 30 WRITE(LUN,30) RK_FUNC(DRIVE_FUNC)
0486 format(' ',T40,A2COMPRESSC (RK_FUNC(DRIVE_FUNC)))>>
0487
0488 CALL OUTPUT (LUN,RKCS1,RKCS1_2,5,5,9,'0')
0489
0490 FIELD=LIB$EXTZV(10,1,RKCS1)
0491
0492 CALL LINCHK (LUN,1)
0493
0494 40 WRITE(LUN,40) RKCS1_10(FIELD)
0495 format(' ',T40,'CONTROLLER DRIVE TYPE ',
0496 1 A<COMPRESSC (RKCS1_10(FIELD)))>>
0497
0498 CALL OUTPUT (LUN,RKCS1,RKCS1_3,11,11,11,'0')
0499
0500 FIELD=LIB$EXTZV(12,1,RKCS1)

```



```
0501
0502      CALL LINCHK (LUN,1)
0503
0504      WRITE(LUN,44) RK_FORMAT(FIELD)
0505 44      FORMAT(' ',T40,AZCOMPRESSC (RK_FORMAT(FIELD)))>>
0506
0507      CALL OUTPUT (LUN,RKCS1,RKCS1_4,13,13,15,'0')
0508      endif
0509
0510      CALL LINCHK (LUN,1)
0511
0512      WRITE(LUN,50) RKWC
0513 50      FORMAT(' ',T8,'RKWC',T24,Z8.4)
0514
0515      CALL LINCHK (LUN,1)
0516
0517      WRITE(LUN,60) RKBA
0518 60      FORMAT(' ',T8,'RKBA',T24,Z8.4)
0519
0520      if (.not. diagnostic_mode) then
0521
0522      IF (DRIVE_FUNC .GE. XFER_CMD) THEN
0523
0524      CALL CALC_MAP (LUN,8,RKCS1,RKBA)
0525      ENDIF
0526      endif
0527
0528      CALL LINCHK (LUN,1)
0529
0530      WRITE(LUN,70) RKDA
0531 70      FORMAT(' ',T8,'RKDA',T24,Z8.4)
0532
0533      if (.not. diagnostic_mode) then
0534
0535      CALL LINCHK (LUN,2)
0536
0537      FIELD=LIB$EXTZV(0,5,RKDA)
0538
0539      WRITE(LUN,80) FIELD
0540 80      FORMAT(' ',T40,'SECTOR = ',I<COMPRESS4 (FIELD)>,>,'.')
0541
0542      FIELD=LIB$EXTZV(8,3,RKDA)
0543
0544      WRITE(LUN,90) FIELD
0545 90      FORMAT(' ',T40,'TRACK = ',I<COMPRESS4 (FIELD)>,>,'.')
0546      endif
0547
0548      CALL LINCHK (LUN,1)
0549
0550      WRITE(LUN,100) RKCS2
0551 100     FORMAT(' ',T8,'RKCS2',T24,Z8.4)
0552
0553      if (.not. diagnostic_mode) then
0554
0555      CALL LINCHK (LUN,1)
0556
0557      FIELD=LIB$EXTZV(0,3,RKCS2)
```

```
0558
0559      110  WRITE(LUN,110) FIELD
0560      FORMAT(' ',T40,'SELECTED DRIVE = ',I1,'.')
0561
0562      CALL OUTPUT (LUN,RKCS2,RKCS2_1,3,3,15,'0')
0563      endif
0564
0565      CALL LINCHK (LUN,1)
0566
0567      120  WRITE(LUN,120) RKDS
0568      FORMAT(' ',T8,'RKDS',T24,Z8.4)
0569
0570      if (.not. diagnostic_mode) then
0571
0572      CALL OUTPUT (LUN,RKDS,RKDS_1,0,0,0,'0')
0573
0574      CALL OUTPUT (LUN,RKDS,RKDS_2,2,2,7,'0')
0575
0576      CALL LINCHK (LUN,1)
0577
0578      FIELD=LIB$EXTZV(8,1,RKDS)
0579
0580      130  WRITE(LUN,130) RK_DRIVETYPE(FIELD)
0581      FORMAT(' ',T40,A<COMPRESSC (RK_DRIVETYPE(FIELD)))>>)
0582
0583      CALL OUTPUT (LUN,RKDS,RKDS_3,11,11,11,'0')
0584
0585      CALL OUTPUT (LUN,RKDS,RKDS_4,13,13,15,'0')
0586      endif
0587
0588      CALL LINCHK (LUN,1)
0589
0590      140  WRITE(LUN,140) RKER
0591      FORMAT(' ',T8,'RKER',T24,Z8.4)
0592
0593      if (.not. diagnostic_mode) then
0594
0595      IF (JIAND(RKDS,'8000'X) .NE. 0) THEN
0596
0597      CALL OUTPUT (LUN,RKER,RKER_1,0,0,15,'0')
0598      ENDIF
0599      endif
0600
0601      CALL LINCHK (LUN,1)
0602
0603      150  WRITE(LUN,150) RKAS
0604      FORMAT(' ',T8,'RKAS/OF',T24,Z8.4)
0605
0606      if (.not. diagnostic_mode) then
0607
0608      FIELD=LIB$EXTZV(0,6,RKAS)
0609
0610      IF (FIELD .NE. 0) THEN
0611
0612      CALL LINCHK (LUN,1)
0613
0614      IF (FIELD .NE. 1 .AND.
```



```
0615      1  FIELD .NE. 8 .AND.  
0616      2  FIELD .NE. 16 .AND.  
0617      3  FIELD .NE. 32 .AND.  
0618      4  FIELD .NE. 48) THEN  
0619  
0620      WRITE(LUN,155)  
0621 155   FORMAT(' ',T40,'INVALID OFFSET')  
0622  
0623      GOTO 185  
0624  
0625      ELSE IF (EMBSB_DV_TYPE .EQ. RK06) THEN  
0626  
0627      FIELD=FIELD/8  
0628  
0629      WRITE(LUN,160) RK06_OFFSET(FIELD)  
0630 160   FORMAT(' ',T40,'OFFSET = ',  
0631      1 A<COMPRESSC (RK06_OFFSET(FIELD))>,' MICRO INCHES')  
0632  
0633      ELSE IF (EMBSB_DV_TYPE .EQ. RK07) THEN  
0634  
0635      FIELD=FIELD/8  
0636  
0637      WRITE(LUN,165) RK07_OFFSET(FIELD)  
0638 165   FORMAT(' ',T40,'OFFSET = ',  
0639      1 A<COMPRESSC (RK07_OFFSET(FIELD))>,' MICRO INCHES')  
0640      ENDIF  
0641  
0642      FIELD=LIB$EXTZV(7,1,RKAS)  
0643  
0644      CALL LINCHK (LUN,1)  
0645  
0646      WRITE(LUN,170) OFFSET_DIR(FIELD)  
0647 170   FORMAT(' ',T40,'OFFSET DIRECTION = ',  
0648      1 A<COMPRESSC (OFFSET_DIR(FIELD))>'  
0649      ENDIF  
0650  
0651 185   DO 195 I=8,15  
0652  
0653      FIELD=LIB$EXTZV(I,1,RKAS)  
0654  
0655      IF (FIELD .NE. 0) THEN  
0656  
0657      CALL LINCHK (LUN,1)  
0658  
0659      WRITE(LUN,190) (I-8)  
0660 190   FORMAT(' ',T40,'ATTENTION DRIVE ',I1,'.')  
0661      ENDIF  
0662  
0663 195   CONTINUE  
0664      endif  
0665  
0666      CALL LINCHK (LUN,1)  
0667  
0668      WRITE(LUN,200) RKDC  
0669 200   FORMAT(' ',T8,'RKDC',T24,Z8.4)  
0670  
0671      if (.not. diagnostic_mode) then
```

```
0672 FIELD=LIB$EXTZV(0,10,RKDC)
0673
0674 IF (FIELD .NE. 0) THEN
0675
0676 CALL LINCHK (LUN,1)
0677
0678 WRITE(LUN,210) FIELD
0679 210 FORMAT(' ',T40,'DESIRED CYLINDER = ',I<COMPRESS4 (FIELD)>,'.')
0680
0681 ENDIF
0682 endif
0683
0684 CALL LINCHK (LUN,1)
0685
0686 WRITE(LUN,230) RKMR1
0687 230 FORMAT(' ',T8,'RKMR1',T24,Z8.4)
0688
0689 if (diagnostic_mode) then
0690
0691 call linchk (lun,1)
0692
0693 WRITE(LUN,235)
0694 235 FORMAT(' ',T40,'DIAGNOSTIC MODE')
0695 endif
0696
0697 CALL LINCHK (LUN,1)
0698
0699 WRITE(LUN,240) RKEC1
0700 240 FORMAT(' ',T8,'RKEC1',T24,Z8.4)
0701
0702 CALL LINCHK (LUN,1)
0703
0704 WRITE(LUN,250) RKEC2
0705 250 FORMAT(' ',T8,'RKEC2',T24,Z8.4)
0706
0707 CALL LINCHK (LUN,1)
0708
0709 WRITE(LUN,260) RKMR2
0710 260 FORMAT(' ',T8,'RKMR2',T24,Z8.4)
0711
0712 if (.not. diagnostic_mode) then
0713
0714 IF ((JIAND(RKMR1,'01'X) .NE. 0)
0715 1 .AND.
0716 2 (EMBSW_HD_ENTRY .EQ. TIMEOUT)
0717 3 .AND.
0718 4 (JIAND(RKMR3,'01'X) .NE. 0)) THEN
0719
0720 CALL LINCHK (LUN,1)
0721
0722 WRITE(LUN,261)
0723 261 FORMAT(' ',T40,'*** MESSAGE A1 ***')
0724
0725 FIELD = LIB$EXTZV(0,3,RKMR2)
0726
0727 CALL LINCHK (LUN,1)
0728
```



```
0729      WRITE(LUN,262) FIELD
0730 262    FORMAT(' ',T40,'SELECTED DRIVE = ',I<COMPRESS4 (FIELD)>,>,'.')
0731
0732      CALL OUTPUT (LUN,RKMR2,V1RKMR2,4,4,15,'0')
0733      ENDIF
0734    endif
0735
0736      CALL LINCHK (LUN,1)
0737
0738 270    WRITE(LUN,270) RKMR3
0739      FORMAT(' ',T8,'RKMR3',T24,Z8.4)
0740
0741      if (.not. diagnostic_mode) then
0742
0743        IF ((JIAND(RKMR1,'01'X) .NE. 0)
0744          1 .AND.
0745          2 (EMB$W_HD_ENTRY .EQ. TIMEOUT)
0746          3 .AND.
0747          4 (JIAND(RKMR3,'01'X) .NE. 0)
0748          5 .AND.
0749          6 (JIAND(RKMR3,'FFFC'X) .NE.0)) THEN
0750
0751        CALL LINCHK (LUN,1)
0752
0753 272    WRITE(LUN,272)
0754      FORMAT(' ',T40,'*** MESSAGE B1 ***')
0755
0756      CALL OUTPUT (LUN,RKMR3,V1RKMR3,4,4,15,'0')
0757      ENDIF
0758    endif
0759
0760    if (
0761      1 drive_func .ge. xfer_cmd
0762      1 .and.
0763      1 emb$w_hd_entry .ne. 96
0764      1 .and.
0765      1 emb$w_hd_entry .ne. 98
0766      1 ) then
0767
0768      if (uba_regs(1) .ne. 0) then
0769
0770        call uba_datapath (lun,uba_regs(1),uba_regs(2))
0771      endif
0772
0773      call calc_map2 (8,rkcs1,rkba,field)
0774
0775      call uba_mapping (lun,field,uba_regs(3))
0776
0777      if (
0778      1 lib$extzv(16,16,emb$l_dv_iosb1) .gt. 512
0779      1 .and.
0780      1 field .ne. 0
0781      1 ) then
0782
0783      call uba_mapping (lun,(field-1),uba_regs(4))
0784    endif
0785  endif
```

```
0786
0787
0788
0789      275      write(lun,275)
0790      format(' ',:)
0791
0792      if (emb$w_hd_entry .ne. 98) then
0793
0794      call ucb$b_ertcnt (lun,emb$b_dv_ertcnt)
0795
0796      call ucb$b_ertmax (lun,emb$b_dv_ertmax)
0797      endif
0798
0799      call orb$l_owner (lun,emb$l_dv_ownuic)
0800
0801      call ucb$l_char (lun,emb$l_dv_char)
0802
0803      call ucb$w_sts (lun,emb$w_dv_sts)
0804
0805      call ucb$l_opcnt (lun,emb$l_dv_opcnt)
0806
0807      call ucb$w_errcnt (lun,emb$w_dv_errcnt)
0808
0809      if (emb$w_hd_entry .ne. 98) then
0810
0811      call ucb$l_media (lun,emb$l_dv_media)
0812
0813      call linchk (lun,1)
0814
0815      write(lun,275)
0816
0817      call rkdisk_qio (lun,emb$w_dv_func)
0818
0819      call irp$w_bcnc (lun,emb$w_dv_bcnc)
0820
0821      call irp$w_boff (lun,emb$w_dv_boff)
0822
0823      call irp$l_pid (lun,emb$l_dv_rqid)
0824
0825      call irp$q_iosb (lun,emb$l_dv_iosb1)
0826      endif
0827
0828      RETURN
0829      END
```


PROGRAM SECTIONS

Name	Bytes	Attributes
0 \$CODE	2873	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 \$PDATA	789	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 \$LOCAL	3780	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3 EMB	512	PIC OVR REL GBL SHR NOEXE RD WRT LONG
Total Space Allocated	7954	

ENTRY POINTS

Address	Type	Name
0-00000000		RKDISK

VARIABLES

Address	Type	Name	Address	Type	Name
2-00000954	L*1	DIAGNOSTIC_MODE	2-0000095C	I*4	DRIVE_FUNC
3-0000001C	L*1	EMBSB_DV_CLASS	3-00000010	L*1	EMBSB_DV_ERTCNT
3-00000011	L*1	EMBSB_DV_ERTMAX	3-0000003E	L*1	EMBSB_DV_NAMLANG
3-0000003A	L*1	EMBSB_DV_SLAVE	3-0000001D	L*1	EMBSB_DV_TYPE
3-00000036	I*4	EMBSL_DV_CHAR	3-00000012	I*4	EMBSL_DV_IOSB1
3-00000016	I*4	EMBSL_DV_IOSB2	3-00000026	I*4	EMBSL_DV_MEDIA
3-0000004E	I*4	EMBSL_DV_NUMREG	3-0000002E	I*4	EMBSL_DV_OPCNT
3-00000032	I*4	EMBSL_DV_OWNUIC	3-0000001E	I*4	EMBSL_DV_RQPID
3-00000000	I*4	EMBSL_HD_SID	3-0000003F	CHAR	EMBST_DV_NAME
3-00000024	I*2	EMBSW_DV_BCNT	3-00000022	I*2	EMBSW_DV_BOFF
3-0000002C	I*2	EMBSW_DV_ERRCNT	3-0000003C	I*2	EMBSW_DV_FUNC
3-0000001A	I*2	EMBSW_DV_STS	3-0000002A	I*2	EMBSW_DV_UNIT
3-00000004	I*2	EMBSW_HD_ENTRY	3-0000000E	I*2	EMBSW_HD_ERRSEQ
2-00000958	I*4	FIELD	2-00000960	I*4	I
AP-00000004a	L*1	LUN	3-0000006E	I*4	RKAS
3-0000005A	I*4	RKBA	3-00000052	I*4	RKCS1
3-00000062	I*4	RKCS2	3-0000005E	I*4	RKDA
3-00000072	I*4	RKDC	3-00000066	I*4	RKDS
3-0000007A	I*4	RKEC1	3-0000007E	I*4	RKEC2
3-0000006A	I*4	RKER	3-00000076	I*4	RKMR1
3-00000082	I*4	RKMR2	3-00000086	I*4	RKMR3
3-00000056	I*4	RKWC			

ARRAYS

Address	Type	Name	Bytes	Dimensions
3-00000000	L*1	EMB	512	(0:511)
3-00000052	I*4	EMBSL_DV_REGSAV	420	(0:104)
3-00000006	I*4	EMBSQ_HD_TIME	8	(2)
2-00000000	CHAR	OFFSET DIR	16	(0:1)
2-00000010	CHAR	RK06_OFFSET	25	(5)
2-00000029	CHAR	RK07_OFFSET	25	(5)

2-00000152	CHAR	RKCS1-1	7	(0:0)
2-000001E5	CHAR	RKCS1-10	10	(0:1)
2-00000159	CHAR	RKCS1-2	140	(5:9)
2-000001EF	CHAR	RKCS1-3	20	(11:11)
2-0000021F	CHAR	RKCS1-4	99	(13:15)
2-00000282	CHAR	RKCS2-1	390	(3:15)
2-00000408	CHAR	RKDS-1	16	(0:0)
2-00000418	CHAR	RKDS-2	96	(2:7)
2-00000494	CHAR	RKDS-3	16	(11:11)
2-000004A4	CHAR	RKDS-4	72	(13:15)
2-000004EC	CHAR	RKER-1	528	(0:15)
2-00000478	CHAR	RK_DRIVE TYPE	28	(0:1)
2-00000203	CHAR	RK_FORMAT	28	(0:1)
2-00000042	CHAR	RK_FUNC	272	(0:15)
3-0000008A	I*4	UBA_REGS	16	(4)
2-000006FC	CHAR	V1RKMR2	252	(4:15)
2-000007F8	CHAR	V1RKMR3	348	(4:15)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-0000004C	20'	1-0000005F	30'	1-0000006B	40'	1-0000008F	44'	1-0000009B	50'	1-000000AC	60'
1-000000BD	70'	1-000000CE	80'	1-000000E8	90'	1-00000102	100'	1-00000114	110'	1-00000132	120'
1-00000143	130'	1-0000014F	140'	1-00000160	150'	1-00000174	155'	1-0000018A	160'	1-000001B0	165'
1-000001D6	170'	0-000005A9	185'	1-000001F7	190'	**	195'	1-00000214	200'	1-00000225	210'
1-00000249	230'	1-0000025B	235'	1-00000272	240'	1-00000284	250'	1-00000296	260'	1-000002A8	261'
1-000002C2	262'	1-000002E4	270'	1-000002F6	272'	1-00000310	275'				

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name	Type	Name	Type	Name	Type	Name	Type	Name
	CALC_MAP		CALC_MAP2	I*4	COMPRESS4	I*4	COMPRESSC		DHEAD1		FRCTOF
	IRP\$C_PID		IRP\$Q_IOSB		IRP\$W_BCNT		IRP\$W_BOFF	I*4	LIB\$EXTZV		LINCHK
	ORB\$C_OWNER		OUTPUT		RKDISK_QIO		UBA_DATAPATH		UBA_MAPPING		UCB\$B_ERTCNT
	UCB\$B_ERTMAX		UCB\$C_CHAR		UCB\$C_MEDIA		UCB\$C_OPNT		UCB\$W_ERRCNT		UCB\$W_STS


```
0001
0002
0003      Subroutine RKDISK_QIO (lun,emb$w_dv_func)
0004
0005      include 'src$:qiocommon.for /nolist'
0269
0270
0271      byte          lun
0272
0273      integer*2      emb$w_dv_func
0274
0275      integer*4      qiocode(0:1,0:63)
0276
0277
0278      if (qiocode(0,0) .eq. 0) then
0279
0280      qiocode(1,00) = %loc(io$_nop)
0281
0282      qiocode(1,01) = %loc(io$_unload)
0283
0284      qiocode(1,02) = %loc(io$_seek)
0285
0286      qiocode(1,03) = %loc(io$_recal)
0287
0288      qiocode(1,04) = %loc(io$_drvclr)
0289
0290      qiocode(1,05) = %loc(io$_release)
0291
0292      qiocode(1,06) = %loc(io$_offset)
0293
0294      qiocode(1,07) = %loc(io$_retcenter)
0295
0296      qiocode(1,08) = %loc(io$_packack)
0297
0298      qiocode(1,10) = %loc(io$_writecheck)
0299
0300      qiocode(1,11) = %loc(io$_writepblk)
0301
0302      qiocode(1,12) = %loc(io$_readpblk)
0303
0304      qiocode(1,13) = %loc(io$_writehead)
0305
0306      qiocode(1,14) = %loc(io$_readhead)
0307
0308      qiocode(1,25) = %loc(io$_startspndl)
0309
0310      qiocode(1,26) = %loc(io$_setchar)
0311
0312      qiocode(1,27) = %loc(io$_sensechar)
0313
0314      qiocode(1,32) = %loc(io$_writelblk)
0315
0316      qiocode(1,33) = %loc(io$_readlblk)
0317
0318      qiocode(1,35) = %loc(io$_setmode)
0319
0320      qiocode(1,39) = %loc(io$_sensemode)
```

```
0321      qiocode(1,48) = %loc(io$_writevblk)
0322
0323      qiocode(1,49) = %loc(io$_readvblk)
0324
0325      qiocode(1,50) = %loc(io$_access)
0326
0327      qiocode(1,51) = %loc(io$_create)
0328
0329      qiocode(1,52) = %loc(io$_deaccess)
0330
0331      qiocode(1,53) = %loc(io$_delete)
0332
0333      qiocode(1,54) = %loc(io$_modify)
0334
0335      qiocode(1,56) = %loc(io$_acpcontrol)
0336
0337      qiocode(1,57) = %loc(io$_mount)
0338
0339      do 10,i = 0,63
0340
0341      qiocode(0,i) = 33
0342
0343      if (qiocode(1,i) .eq. 0) then
0344
0345      qiocode(1,i) = %loc(qio_string)
0346      endif
0347
0348      10 continue
0349      endif
0350
0351      call irp$w_func (lun,emb$w_dv_func,
0352      1 qiocode(0,lib$extzv(0,6,emb$w_dv_func)))
0353
0354      return
0355
0356      end
0357
```


PROGRAM SECTIONS

Name	Bytes	Attributes
0 \$CODE	295	PIC CON REL LCL SHR EXE RD NOWRT LONG
1 \$PDATA	8	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2 \$LOCAL	548	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3 QIOCOMMON	1247	PIC OVR REL GBL SHR NOEXE RD WRT LONG
Total Space Allocated	2098	

ENTRY POINTS

Address	Type	Name
0-00000000		RKDISK_Q10

VARIABLES

Address	Type	Name	Address	Type	Name
AP-00000008a	I*2	EMBSW DV FUNC	2-00000200	I*4	I
3-00000442	CHAR	IOS_ABORT	3-0000034D	CHAR	IOS_ACCESS
3-000003C2	CHAR	IOS_ACPCONTROL	3-000004B3	CHAR	IOS_AVAILABLE
3-00000297	CHAR	IOS_CLEAN	3-00000369	CHAR	IOS_CREATE
3-00000385	CHAR	IOS_DEACCESS	3-00000393	CHAR	IOS_DELETE
3-0000026D	CHAR	IOS_DIAGNOSE	3-00000065	CHAR	IOS_DRVCLR
3-000004CB	CHAR	IOS_DSE	3-000000A9	CHAR	IOS_ERASETAPE
3-00000276	CHAR	IOS_FORMAT	3-00000071	CHAR	IOS_INITIALIZE
3-00000014	CHAR	IOS_LOADMCODE	3-000003A1	CHAR	IOS_MODIFY
3-000003E2	CHAR	IOS_MOUNT	3-00000000	CHAR	IOS_NOP
3-0000009D	CHAR	IOS_OFFSET	3-000000EB	CHAR	IOS_PACKACK
3-000000E0	CHAR	IOS_QSTOP	3-000003EF	CHAR	IOS_RDSTATS
3-00000421	CHAR	IOS_READCSR	3-00000169	CHAR	IOS_READHEAD
3-000002B6	CHAR	IOS_READLBLK	3-0000013F	CHAR	IOS_READPBLK
3-00000200	CHAR	IOS_READPRESET	3-00000195	CHAR	IOS_READTRACKD
3-0000033A	CHAR	IOS_READVBLK	3-0000045A	CHAR	IOS_READWTHBUF
3-00000484	CHAR	IOS_READWTHXBUF	3-0000004D	CHAR	IOS_RECAL
3-0000007C	CHAR	IOS_RELEASE	3-000001AB	CHAR	IOS_REREADN
3-000001B8	CHAR	IOS_REREADP	3-000000CA	CHAR	IOS_RETCENTER
3-000002E6	CHAR	IOS_REWIND	3-000002C9	CHAR	IOS_REWINDOFF
3-000000FC	CHAR	IOS_SEARCH	3-00000024	CHAR	IOS_SEEK
3-00000231	CHAR	IOS_SENSECHAR	3-00000309	CHAR	IOS_SENSEMODE
3-0000021D	CHAR	IOS_SETCCHAR	3-000003B8	CHAR	IOS_SETCLOCK
3-00000088	CHAR	IOS_SETCLOCKP	3-000002DD	CHAR	IOS_SETMODE
3-000002ED	CHAR	IOS_SKIPFILE	3-000002FA	CHAR	IOS_SKIPRECORD
3-00000029	CHAR	IOS_SPACEFILE	3-0000010E	CHAR	IOS_SPACERECORD
3-000003D7	CHAR	IOS_STARTDATA	3-000000B4	CHAR	IOS_STARTDATAP
3-00000037	CHAR	IOS_STARTMPROC	3-0000020F	CHAR	IOS_STARTSPNDL
3-00000059	CHAR	IOS_STOP	3-0000000D	CHAR	IOS_UNLOAD
3-00000468	CHAR	IOS_WRITEBUFNCRC	3-0000011E	CHAR	IOS_WRITECHECK
3-000001E4	CHAR	IOS_WRITECHECKH	3-000003FF	CHAR	IOS_WRITECSR
3-00000153	CHAR	IOS_WRITEHEAD	3-000002A2	CHAR	IOS_WritelBLK
3-00000247	CHAR	IOS_WRITEMARK	3-00000314	CHAR	IOS_WRITEOF
3-0000012A	CHAR	IOS_WRITEPBLK	3-000001C9	CHAR	IOS_WRITERET

RKDISK_QIO

F 12
16-Sep-1984 00:13:19
5-Sep-1984 14:21:27

VAX-11 FORTRAN V3.4-56
DISK\$VMSMASTER:[ERF.SRC]RKDISK.FOR;1 Page 18

3-0000017E CHAR IOS_WRITETRACKD
3-00000448 CHAR IOS_WRITEWITHBUF
AP-00000004a L*1 LUN

3-00000326 CHAR IOS_WRITEVBLK
3-00000257 CHAR IOS_WRTTMKR
3-000004A1 CHAR QIO_STRING

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	I*4	QIOCODE	512	(0:1, 0:63)

LABELS

Address	Label
**	10

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name
	IRP\$W_FUNC	I*4	LIB\$EXTZV

COMMAND QUALIFIERS

FORTRAN /LIS=LISS:RKDISK/OBJ=OBJ\$:RKDISK MSRC\$:RKDISK

/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:	11.39 seconds
Elapsed Time:	27.02 seconds
Page Faults:	371
Dynamic Memory:	245 pages

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

