

1

.REM *

IDENTIFICATION

PRODUCT CODE: AC-F087C-MC
PRODUCT NAME: CXRPDC0 RP04,5,6 MOD
PRODUCT DATE: JUNE 1979
MAINTAINER: DEC/X11 SUPPORT GROUP

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITALS COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976,1979 DIGITAL EQUIPMENT CORPORATION

1. ABSTRACT

RPD IS AN IOMODX WHICH WILL EXERSIZE UP TO 8 RP04/5/6 DISK DRIVES ON AN RH11/RH70 CONTROLLER, OR DUAL-PORTED BETWEEN TWO SUCH CONTROLLERS. IN SINGLE-PORT MODE, THE ENTIRE SURFACE OF EACH DISK WILL BE TESTABLE (RANDOMLY OR SYSTEMATICALLY).

IN DUAL-PORT MODE, THE "A" CONTROLLER MODULE WILL ACCESS ONLY THE LOW HALF OF THE DISK, AND THE "B" CONTROLLER MODULE WILL ACCESS ONLY THE HIGH HALF OF THE DISK; NO COMMUNICATION WILL BE ATTEMPTED BETWEEN THE TWO PORTS. ONE COPY OF THIS MODULE MUST BE CONFIGURED FOR EACH PORT.

SINGLE/DUAL PORT MODE WILL BE AUTOMATICALLY SELECTED ON THE BASIS OF THE RPDT REGISTER OF EACH SELECTED DRIVE.

DURING TESTING, ANY BAD SECTOR FOUND WILL BE STORED AUTOMATICALLY AND WITHOUT OPERATOR INTERVENTION INTO THE MODULE'S BADSPOT TABLE. A MESSAGE CAN BE PRINTED TO NOTIFY THE OPERATOR OF THE BADSPOT ADDRESS (SEE OPERATOR OPTIONS). SECTORS LISTED IN THE TABLE WILL NOT BE TESTED ON ANY DRIVE.

THE WBUFQ WORD IN THE HEADER MAY BE CHANGED AT WILL TO ALTER THE MODULE'S TRANSFER SIZE (AND OF COURSE A AND B-PORT SIZES NEED NOT MATCH): THIS WILL NOT ALTER THE EFFECTIVENESS OF THE BADSPOT TABLE OR CAUSE THE MODULE TO ATTEMPT ACCESS OF OUT-OF-RANGE DISK ADDRESSES. WITHIN THIS FRAMEWORK, THE MODULES WILL ACCESS AS MUCH OF THE DISK AS POSSIBLE. RANDOM SEEKS WILL BE USED UNLESS THIS OPTION IS DESELECTED IN SR1.

2. REQUIREMENTS

HARDWARE: 1 TO 8 RP04/5/6 DRIVES ON 1 OR 2 RH11/RH70 CONTROLLERS

STORAGE: RPD REQUIRES:

DECIMAL WORDS: 1806
OCTAL WORDS: 3416
OCTAL BYTES: 7034

3. PASS DEFINITION

-
- ONE PASS CONSISTS OF 300 ITERATIONS.
 - AN ITERATION CONSISTS OF THE FOLLOWING STEPS EXECUTED ON EACH SELECTED DRIVE:
 - A. WRITE DATA TO DISK. IN DUAL-PORT MODE, PORT "A" MODULE WILL TEST THE LOW HALF OF THE DISK (CYLINDERS 0 TO 407 FOR RP06, CYLINDERS 0 TO 205 FOR RP04/5), PORT "B" WILL EXERSIZE THE HIGH HALF (CYLINDERS 206 TO 410 FOR RP04/5, CYLINDERS 408 TO 814 FOR RP06). IN SINGLE-PORT MODE, THE MODULE WILL ACCESS THE ENTIRE DISK SURFACE (CYLINDERS 0 TO 410 FOR RP04/5, CYLINDERS 0 TO 814 FOR RP06).
 - B. WRITE-CHECK DATA JUST WRITTEN.
 - C. READ 256 WORDS (1 SECTOR) INTO MODULE READ BUFFER.
 - D. DO IN-CORE COMPARE OF READ BUFFER WITH FIRST 256 WORDS OF WRITE BUFFER.
 - E. RELEASE THE DRIVE FOR OTHER PORT TO TEST.

4. EXECUTION TIME

ONE PASS OF RPD RUNNING ALONE ON A PDP-11/70 TAKES ABOUT A MINUTE.

5. CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:
DVA-176700 VCT-254 BR1-5 DVC-1

REQUIRED PARAMETERS:
SR1 MUST BE SET UP TO INDICATE WHICH PORT (A/B) THIS COPY OF THE
MODULE IS TO TEST, IF ANY DRIVES ARE DUAL-PORTED. ALSO, BIT 7 MUST BE
SET IF THE CONTROLLER IS AN RH70, AND CLEARED IF IT IS NOT AN RH70.

6. DEVICE/OPTION SETUP

MAKE SURE ALL DRIVES ARE POWERED UP, WRITE ENABLED, AND READY.
FOR DUAL-PORTED DRIVES: MAKE SURE CONTROLLER SELECT SWITCH IS IN A/B
POSITION. IF DRIVE IS NOT CYCLED UP WITH SWITCH IN CORRECT POSITION, PLACE
SWITCH CORRECTLY, AND DISABLE THEN RE-ENABLE DRIVE USING DRIVE DISABLE
SWITCH.

7. OPERATOR OPTIONS

SR1

BIT 2
CLEAR(0):
TYPE OUT DATA LATE ERRORS (HARD ERROR) AND COUNT THEM INTERNALLY.
SET(1):
COUNT DATA LATE ERRORS INTERNALLY, BUT DO NOT TYPE OUT.

BIT 4
CLEAR(0):
THIS COPY OF RPD TESTS A-PORT ON ANY DUAL-PORTED DRIVES
SET(1):
THIS COPY OF RPD TESTS B-PORT ON ANY DUAL-PORTED DRIVES

BIT 5
CLEAR(0):
TEST DISKS WITH RANDOM SEEKS
SET(1):
DISABLE RANDOM SEEKS. INCREMENT SECTOR BY ONE EACH CYCLE.

BIT 6
CLEAR(0):
DO NOT TYPE OUT BADSPOTS.
SET(1):

ALWAYS TYPE OUT ANY BADSPOT NOT ALREADY RECORDED ON BADSPOT
TABLE.

BIT 7

CLEAR(0):

CONTROLLER IS RH11

SET(1):

CONTROLLER IS RH70 (HAS RHBAE AND RHCS3)

8. NON-STANDARD PRINTOUTS

A. MOST PRINTOUTS HAVE THE STANDARD FORMATS.

B. ERROR MESSAGE DUMP RH11 REGISTERS IN THE FOLLOWING ORDER:

RHCS1 RHWC RHBA RPDA RHCS2 RPDS RPER1 RPAS
RPLA RHDB RPMR RPDT RPSM RPOF RPDC RPCC
RPER2 RPER3 RPEC1 RPEC2

C. THE BAD SECTOR MESSAGE (WHEN ENABLED) LOOKS LIKE:

'DRIVE X: BADSPOT AT (OCTAL) CYL: XX, TRK: XX, SEC: XX'
WHERE 'X' IS AN OCTAL DIGIT. NO ERROR IS ASSOCIATED WITH THIS
MESSAGE.

3

```
SBTTL MODULE HEADER BLOCK ;DRB001
000000 IOMODX <RPDC >,176700,254,5,0,0,1500,104,RBUF,256,,256.
000000 MODULE 150000,RPDC ,176700,254,5,0,0,1500,104,RBUF,256,,256.
;TITLE RPDC DEC/X11 SYSTEM EXERCISER MODULE
;DDXCOM VERSION 6 23-MAY-78
;LIST BTN
*****
000000 BEGIN;
000000 050122 041504 040 MODNAM: ,ASCII /RPDC / ;MODULE NAME,
000005 000 XFLAG: ,BYTE OPEN ;USED TO KEEP TRACK OF WBUFF USAGE
000006 176700 ADDR: 176700+0 ;1ST DEVICE ADDR,
000010 000254 VECTOR: 254+0 ;1ST DEVICE VECTOR,
000012 240 BR1: ,BYTE PRTY5+0 ;1ST BR LEVEL,
000013 000 BR2: ,BYTE PRTY0+0 ;2ND BR LEVEL,
000014 000001 DVID1: 0+1 ;DEVICE INDICATOR 1,
000016 000000 SR1: OPEN ;SWITCH REGISTER 1
000020 000000 SR2: OPEN ;SWITCH REGISTER 2
000022 000000 SR3: OPEN ;SWITCH REGISTER 3
000024 000000 SR4: OPEN ;SWITCH REGISTER 4
*****
000026 150000 STAT: 150000 ;STATUS WORD,
000030 002044 INIT: START ;MODULE START ADDR,
000032 000252 SPOINT: MODSP ;MODULE STACK POINTER,
000034 000000 PASCNT: 0 ;PASS COUNTER,
000036 001500 ICNT: 1500 ;# OF ITERATIONS PER PASS=1500
000040 000000 ICOUNT: 0 ;LOC TO COUNT ITERATIONS
000042 000000 SOFCNT: 0 ;LOC TO SAVE TOTAL SOFT ERRORS
000044 000000 HRDCNT: 0 ;LOC TO SAVE TOTAL HARD ERRORS
000046 000000 SOFPAS: 0 ;LOC TO SAVE SOFT ERRORS PER PASS
000050 000000 HRDPAS: 0 ;LOC TO SAVE HARD ERRORS PER PASS
000052 000000 SYSCNT: 0 ;# OF SYS ERRORS ACCUMULATED
000054 000000 PANNUM: 0 ;HOLDS RANDOM # WHEN RAND MACHO IS CALLED
000056 000000 CONFIG: ;RESERVED FOR MONITOR USE
000056 000000 RES1: 0 ;RESERVED FOR MONITOR USE
000060 000000 RES2: 0 ;RESERVED FOR MONITOR USE
000062 000000 SVR0: OPEN ;LOC TO SAVE R0,
000064 000000 SVR1: OPEN ;LOC TO SAVE R1,
000066 000000 SVR2: OPEN ;LOC TO SAVE R2,
000070 000000 SVR3: OPEN ;LOC TO SAVE R3,
000072 000000 SVR4: OPEN ;LOC TO SAVE R4,
000074 000000 SVR5: OPEN ;LOC TO SAVE R5,
000076 000000 SVR6: OPEN ;LOC TO SAVE R6,
000100 000000 CSRA: OPEN ;ADDR OF CURRENT CSR,
000102 SBADR: ;ADDR OF GOOD DATA, OR
000102 ACSR: OPEN ;CONTENTS OF CSR,
000104 WASADR: ;ADDR OF BAD DATA, OR
000104 ASAT: OPEN ;STATUS REG CONTENTS,
000106 ERPTYP: ;TYPE OF ERROR
000106 ASB: OPEN ;EXPECTED DATA,
000110 AWAS: OPEN ;ACTUAL DATA,
000112 002170 RSTRT: RSTRT ;RESTART ADDRESS AFTER END OF PASS
000114 WDTO: OPEN ;WORDS TO MEMORY PER ITERATION
000116 WDFR: OPEN ;WORDS FROM MEMORY PER ITERATION
000120 INTR: OPEN ;# OF INTERRUPTS PER ITERATION
000122 000104 IDNUM: 104 ;MODULE IDENTIFICATION NUMBER=104
000124 000716 RBUFVA: RBUF ;READ BUFFER VIRTUAL ADDRESS
```

```
000126 000000 RBUFA: OPEN ;READ BUFFER PHYSICAL ADDRESS
000130 000000 RBUFEA: OPEN ;READ BUFFER EA BITS
000132 000400 RBUSZ: 256 ;SIZE OF THE READ BUFFER
000134 000000 WBUFA: OPEN ;WRITE BUFFER PHYSICAL ADDRESS
000136 000000 WBUFEA: OPEN ;WRITE BUFFER EA BITS
000140 000400 WBUFQ: 256 ;WRITE BUFFER SIZE REQUESTED
000142 000000 WBUSZ: OPEN ;WRITE BUFFER SIZE AVAILABLE
000144 000000 CDRECT: OPEN ;CDATA/DATCK ERROR COUNT
000146 000000 CDWDCT: OPEN ;CDATA/DATCK WORD COUNT
000150 000000 FREE: OPEN ;RESERVED FOR FUTURE USE
;LIST SPSIZ ;MODULE STACK STARTS HERE,
;NLIST
;WORD 0
;LIST
;ENDR
000252 MODSP:
*****
```


372 000574* 000000 000000 000000
373 000502* 000000 000000 000000
374 000610* 000000 000000 000000
375 000616* 000000 000000 000000
376 000624* 000000

DO NOT CHANGE THE ORDER OF THE NEXT 4 LOCATIONS
NEEDED FOR MAP22 ROUTINE

377
378
379 000626* 000000
380 000630* 000000
381 000632* 000000
382 000634* 000000

PA18: 0
XMEM: 0
PA22: 0
EA22: 0

383
384 000636* 000252*

NXTBBK: BADSPT ; NEXT OPEN BADSPOT IN TABLE

***-5

JDRB001

385
386 000640* 000000

CYLLIM: 0 ; UPPER DISK LIMIT (DEPENDS ON DISK TYPE)

JDRB001

387 000642* 000000

CYLNDR: 0 ; ABSOLUTE CYLINDER ADDRESS OF CURRENT OP, JDRB001

JDRB001

388 000644* 000000

CYL: 0 ; RELATIVE DISK ADDRESS

JDRB001

389 000646* 000

DSKADR: 0 ; FOR REFERENCING TRACK/SECTOR AS WORD

JDRB001

390 000646* 000

SECTOR: ,BYTE 0 ; SECTOR ADDRESS OF CURRENT OPERATION

JDRB001

391 000647* 000

TRACK: ,BYTE 0 ; TRACK ADDRESS OF CURRENT OPERATION

JDRB001

392
393 000650* 000000

BADCYL: 0 ; FROM DRIVE LOCATION REGISTERS, ON FINDING BADSPOT JDRB001

JDRB001

394 000652* 000000

BADSEC: 0 ; FROM RPDA ON BADSPOT OCCURANCE

JDRB001

395 000654* 000000

BADTRK: 0 ; FROM RPDA+1 ON BADSPOT

JDRB001

396
397 000656* 000000

LASCYL: 0 ; LAST CYLINDER WHICH CAN BE WRITTEN

JDRB001

398 000660* 000000

LASTRK: 0 ; LAST TRACK WHICH CAN BE WRITTEN

JDRB001

399 000662* 000000

LASSEC: 0 ; LAST SECTOR WHICH CAN BE WRITTEN

JDRB001

400
401 000664* 000000

SIZTRK: 0 ; NUMBER OF TRACKS IN WBUFSZ

JDRB001

402 000666* 000000

SIZSEC: 0 ; REMAINDER OF WBUFSZ, IN SECTORS

JDRB001

403 000670* 000000

FLAG: 0

404 000672* 000000

DVICE: 0

405 000674* 000000

DRIVE: 0

406 000676* 000000

UNITNO: 0

***-1

JDRB001

407
408 000700* 000000

DLTCNT: 0

409 000702* 000000

FUNC: 0

410 000704* 000000

TIMER: 0

411 000706* 000000

ZERO: 0

412 000710* 000000

FERADR: 0

413 000712* 000000

CLK: 0

414 000714* 000000

TRY: 0

415 000716* 000400

RBUF: ,BLKW 256.

416
417 001716* 000552*

TABLE:

418 001716* 000552*

S

419 001720* 000554*

S+2

420 001722* 000556*

S+4

421 001724* 000560*

S+6

422 001726* 000562*

S+10

423 001730* 000564*

S+12

424 001732* 000566*

S+14

425 001734* 000570*

S+16

426 001736* 000572*

S+20

427 001740* 000574*

S+22

428 001742* 000576*

S+24

429 001744* 000600*

S+26

430 001746* 000602*

S+30

431 001750* 000604*

S+32

432 001752* 000606*

S+34

433 001754* 000610*

S+36

434 001756* 000612*

S+40

435 001760* 000614*

S+42

436 001762* 000616*

S+44

437 001764* 000620*

S+46

438
439 001766* 177777

177777

***-1

JDRB001

440 001770* 000000

RHCS1: 0

441 001772* 000000

RHWC: 0

442 001774* 000000

RHBA: 0

443 001776* 000000

RPDA: 0

444 002000* 000000

RHCS2: 0

445 002002* 000000

RPDS: 0

446 002004* 000000

RPFR: 0

447 002006* 000000

RPAS: 0

448 002010* 000000

RPLA: 0

449 002012* 000000

RHDB: 0

450 002014* 000000

RPMR: 0

451 002016* 000000

RPDT: 0

452 002020* 000000

RPSW: 0

453 002022* 000000

RPOF: 0

454 002024* 000000

RPDC: 0

455 002026* 000000

RPCC: 0

456 002030* 000000

RPFR2: 0

457 002032* 000000

RPFR3: 0

458 002034* 000000

RPEC1: 0

459 002036* 000000

RPEC2: 0

460 002040* 000000

RHBAE: 0

461 002042* 000000

RHCS3: 0

***-4

JDRB001

```

463          ,SBTTL  MODULE CODE                                ;DRB001
464 002044' 012767 000400 176042 START: MOV #256,,WDTO      ;256 WORDS TO MEM/ITERATION
465 002052' 012767 000400 176036 MOV #256,,WDFR      ;256 WORDS FROM MEM/ITERATION
466 002060' 012767 000004 176032 MOV #4,INTR        ;4 INTERRUPTS/ITERATION
467 002066' 016767 175722 176576 MOV DVID1,DVICE    ;GET DRIVES TO TEST
468 002074' 123727 000041 000011 CMPB #41,#11       ;IF RP IS LOAD MEDIUM THEN
469 002102' 001021 BNE 3#              ;BEGIN
470 002104' 113700 000040 MOVB #40,R0          ; GET LOAD-DEVICE NUMBER
471 002110' 012701 000001 MOV #1,R1          ; INITIALIZE DEVICE MASK
472 002114' 105700 18:  TSTB R0              ; WHILE NOT POINTING AT LOAD-DEVICE DO
473 002116' 001403 BEQ 2#              ; BEGIN
474 002120' 006301 ASL R1              ; SHIFT MASK TO NEXT DEVICE
475 002122' 105300 DECB R0              ; KEEP TRACK OF SHIFTING
476 002124' 000773 BR 1#              ; END
477 002126' 130167 176540 20: BITB R1,DVICE    ; IF LOAD DEVICE IS SELECTED THEN
478 002132' 001405 BEQ 3#              ; BEGIN
479 002134' 113767 000040 176534 MOVB #40,UNITNO    ; MOVE LOAD-DEVICE NUMBER TO DRIVE
480 002142' 004767 002252 JSR PC,DROP        ; DROP THE LOAD-DEVICE
481          ; END
482 002146'          30:  CLR CYL              ;END
483 002146' 005067 176472 CLR CYL          ;START AT CYLINDER 0 ;DRB001
484 002152' 012767 000377 176466 MOV #377,DSKADR    ;START AT TRACK 0, SECTOR 0 ;DRB001
485          ;(* INCREMENT BEFORE TEST *) ;DRB001
486 002160' 004767 003652 JSR PC,SETUP
487 002164' 004767 003530 JSR PC,REZET      ;CLEAR THE RH
488 002170'          RESTRT:
489          ;
490 002170' 104415 000000' 000124' GETPA$,BEGIN, RBUFVA ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
491          ;
492 002176' 012767 177777 176472 LOOP1: MOV #1,UNITNO ;PRE-SET UNIT NUMBER
493 002204' 104414 000000' GWFBUF$, BEGIN ;GET WRITE BUFFER INFORMATION
494          ;***1 ;DRB001
495 002210'          LOOP2:
496 002210' 004767 000744 30:  JSR PC,PICKDR ;GO PICK A DRIVE
497 002214' 103407 BCS 1#              ;RETURNS HERE IF ALL DRIVES DONE ;DRB001
498 002216' 004767 001574 JSR PC,WRTLM ;GET DISK LIMITS FOR THIS TYPE OF DISK ;DRB001
499 002222' 005067 176466 CLR TRY          ;ELSE CLR RE TRY COUNT
500 002226' 004767 000022 JSR PC,CYCLE ;GO DO A CYCLE ON THIS DRIVE
501 002232' 000766 BR LOOP2 ;DO IT TO NEXT DRIVE
502          ;
503 002234' 005767 176432 10:  TST DVICE ;ANYBODY LEFT TO CHECK?
504 002240' 001002 BNE 2#              ;BR IF YES
505 002242' 104410 000000' END$,BEGIN ;
506 002246'          20:  ENDIT$,BEGIN ;SIGNAL END OF ITERATION,
507 002246' 104413 000000' BR LOOP1 ;MONITOR SHALL TEST END OF PASS
508          ;BR BACK IF NO
509 002252' 000751
510          ;
511 002254'          CYCLE:
512          ;
513 002254' 004767 000772 JSR PC,PICKK ;SELECT A SECTOR TO TEST ;***4 ;DRB001
514 002260' 004567 000026 JSR R5,WRITE ;GO WRITE A BLOCK
515 002264' 004567 000116 JSR R5,WRTCK ;GO DO WRITE CHECK
516 002270' 004567 000206 JSR R5,READ ;GO READ A BLOCK
517 002274' 104412 000000' 000126' CDATA$,BEGIN,RBUPFA ; REQUEST FOR MONITOR TO CHECK DATA
518 002302' 002304' ,+2 ; IF ERROR, CONTINUE

```

```

519 002304' 004767 000266 JSR PC,RELESE ;RELEASE THE DRIVE ;DRB001
520 002310' 000207 RTS PC ;END CYCLE ;DRB001
521

```

```

522                                     ;***-66
523                                     ;DRB001
524                                     ;
525                                     ;
526                                     ;
527                                     ;
528                                     ;
529                                     ;
530                                     ;
531                                     ;
532                                     ;
533                                     ;
534                                     ;
535                                     ;
536 002312* 012767 000161 176362 WRITE: MOV #161,FUNC ; LOAD WRITE FUNCTION
537 002320* 012767 002312* 176362 MOV #WRITE,FERADR ;SAVE WHERE WE WERE
538 002326* 016746 175610 MOV #WPUFSZ,-(SP) ;GET WRITE SIZE
539 002332* 005416 NEG (SP) ;NEGATE IT
540 002334* 012677 177432 MOV (SP)+,0RHWC ; LOAD WORD COUNT
541 002340* 016777 175570 177426 MOV #WUPFA,0RHBA ; LOAD BUFFER ADDRESS
542 002346* 016777 176274 177422 MOV DSKADR,0RPDA ; LOAD DISK ADDRESS
543 002354* 016777 176262 177442 MOV CYLDR,0RPDC ; LOAD CYLINDER ADDRESS
544 ; LINEUP #WUFEA ; LINE UP EA BITS FOR RHCS1
545 002402* 000167 000302 JMP GO ; CONTINUE
546 002406* 012767 000151 176266 WRITCK: MOV #151,FUNC ; LOAD WRITE-CHECK FUNCTION
547 002414* 012767 002406* 176266 MOV #WRITCK,FERADR ;SAVE WHERE WE WERE
548 002422* 016746 175514 MOV #WPUFSZ,-(SP) ;GET WRITE SIZE
549 002426* 005416 NEG (SP) ;NEGATE IT
550 002430* 012677 177336 MOV (SP)+,0RHWC ; LOAD WORD COUNT
551 002434* 016777 175474 177332 MOV #WUPFA,0RHBA ; LOAD BUFFER ADDRESS
552 002442* 016777 176200 177326 MOV DSKADR,0RPDA ; LOAD DISK ADDRESS
553 002450* 016777 176166 177346 MOV CYLDR,0RPDC ; LOAD CYLINDER ADDRESS
554 ; LINEUP #WUFEA ; LINE UP EA BITS FOR RHCS1
555 002476* 000167 000206 JMP GO ; CONTINUE
556 002502* 012767 000171 176172 READ: MOV #171,FUNC ; LOAD READ FUNCTION
557 002510* 012767 002502* 176172 MOV #READ,FERADR ;SAVE WHERE WE WERE
558 002516* 016746 175410 MOV #RPUFSZ,-(SP) ;GET READ SIZE
559 002522* 005416 NEG (SP) ;NEGATE IT
560 002524* 012677 177242 MOV (SP)+,0RHWC ; LOAD WORD COUNT
561 002530* 016777 175372 177236 MOV #RUPFA,0RHBA ; LOAD BUFFER ADDRESS
562 002536* 016777 176104 177232 MOV DSKADR,0RPDA ; LOAD DISK ADDRESS
563 002544* 016777 176072 177252 MOV CYLDR,0RPDC ; LOAD CYLINDER ADDRESS
564 ; LINEUP #RUFEA ; LINE UP EA BITS FOR RHCS1
565 002572* 000167 000112 JMP GO ; CONTINUE
566 002576* 016777 176074 177174 RELEASE: MOV #UNITNO,0RHCS2 ;SET UP UNIT TO RELEASE
567 002604* 012777 000013 177156 MOV #13,0RHCS1 ;EXECUTE RELEASE COMMAND
568 002612* 104407 000000* BREAK$,BEGIN ;TEMPORARY RETURN TO MONITOR...
569 002616* 104407 000000* BREAK$,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
570 002622* 000207 RTS PC ;IT'S RELEASED
571 ;***-33
572 ;DRB001
573 002624* 016777 176046 177146 CLEAR: MOV #UNITNO,0RHCS2 ; LOAD UNIT ADDRESS
574 002632* 012777 000011 177130 MOV #11,0RHCS1 ; ISSUE A DRIVE CLEAR
575 002640* 000240 NOP ;WAIT
576 002642* 000240 NOP ;FOR DRIVE CLEAR TO FINISH
577 002644* 012777 000023 177116 MOV #23,0RHCS1 ;ISSUE A PACK ACK

```

```

578 002652* 105777 177112 10: TSTB 0RHCS1
579 002656* 100401 BMI 26
580 002660* 000774 BR 18 ; NO, WAIT TILL DONE
581 002662* 017746 177120 20: MOV #0RPAS,-(SP) ;CLEAR AS BIT
582 002666* 012677 177114 MOV (SP)+,0RPAS
583 002672* 012777 040000 177070 MOV #BIT14,0RHCS1 ; CLEAR ANY CONTROLLER ERRORS
584 002700* 012777 010000 177114 MOV #FIT12,0RPOF ; SET BIT FOR 11 FORMAT
585 002706* 000205 RTS R5 ; RETURN
586
587 002710* 016777 175762 177062 GO: MOV #UNITNO,0RHCS2 ; LOAD UNIT SELECT
588 002716* 032767 001000 175132 BIT #ADDR22,RES1 ;22 BIT SUPPORT?
589 002724* 001434 BEQ 18 ;NO
590 002726* 017767 177042 175672 MOV #0RHBA,PA18 ;GET 18 BIT ADDR
591 002734* 006267 175670 ASR #XMEM ;SHIFT EA BITS TO POSITION 4,5
592 002740* 006267 175664 ASR #XMEM
593 002744* 006267 175660 ASR #XMEM
594 002750* 006267 175654 ASR #XMEM
595 002754* 104416 000000* 000626* MAP22$, BEGIN,PA18 ; GET 22-BIT ADDR FROM 18-BIT ADDR
596 002762* 016777 175644 177004 MOV #PA22,0RHBA ;LOAD BA REG
597 002770* 016777 175640 177042 MOV #EA22,0RHBAE ;LOAD BAE REG
598 002776* 042767 000034 175632 BIC #34,EA22 ;CLEAR UNWANTED BITS
599 003004* 000367 175624 SWAB EA22 ;LOAD INTO BITS 8,9
600 003010* 016767 175620 175612 MOV #EA22,XMEM ;LOAD XMEM TO SET INTO FUNCTION CODE
601 003016* 056767 175606 175656 10: BIS #XMEM,FUNC ; LOAD EXTENDED MEMORY BITS
602 003024* 016777 175632 176736 MOV #FUNC,0RHCS1 ; EXECUTE THE FUNCTION
603 003032* 104400 000000* EXIT$,BEGIN ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
604 003036* 010046 NTRUPT: MOV #R0,-(SP) ;SAVE R0
605 003040* 016700 175632 MOV #UNITNO,R0
606 003044* 116000 003242* MOVB #BITTAB(R0),R0
607 003050* 017746 176732 MOV #0RPAS,-(SP)
608 003054* 040016 BIC #R0,(SP)
609 003056* 012677 176724 MOV #R0+,0RPAS
610 003062* 012600 MOV (SP)+,R0
611 ;-----
612 003064* 000004 000000* 003072* PIRQ$,BEGIN,1$ ; QUEUE UP TO CONTINUE AT 1$ AND RTI
613 ;-----
614
615 003072* 004567 001360 10: JSR #R5,ERRORS ; GO CHECK FOR ERRORS
616 003076* 103401 BCS 26 ;ERRORS DETECTED
617 003100* 000205 RTS R5 ;OTHERWISE, RETURN OK
618
619 003102* 012605 20: MOV (SP)+,R5 ;RESTORE R5, SINCE WE WON'T RETURN
620 003104* 005767 175560 TST FLAG ;DON'T RETRY ON HARD ERROR
621 003110* 100410 BMI 48 ;DRB001
622 003112* 005267 175576 INC TRY ;COUNT AN ERROR
623 003116* 026727 175572 000003 CMP TRY,#3 ;TOO MANY FOR THIS CYCLE?
624 003124* 002002 BGE 48 ;DRB001
625 003126* 000177 175556 JMP #FERADR ;RE-EXECUTE THE DRIVER ROUTINE
626 003132* 005067 175532 CLR FLAG ;CLEAR HARD ERROR BIT
627 003136* 104403 000000* 006722* MSGN$,BEGIN,EXCED ;ASCII MESSAGE CALL WITH COMMON HEADER
628 ;***-7
629 003144* 004767 000102 30: JSR #PC,PICKBK ;TRY A DIFFERENT BLOCK
630 003150* 005367 175522 DEC #UNITNO ;WANT TO RE-DO SAME DRIVE WERE ON
631 003154* 000167 177030 JMP #LOOP2 ;GO DO IT
632 ;***-2
633 ;DRB001

```

```
634 ; -----  
635 ;  
636 ;  
637 ;  
638 003160* PICKDR: ;  
639 003160* 005267 175512 18: INC UNITNO ;POINT TO NEXT DRIVE  
640 003164* 026727 175506 000010 CMP UNITNC,#8. ;DONE LOOKING?  
641 003172* 001002 BNE 20 ;BR IF NO, ELSE  
642 003174* 000261 SEC ;SET CARRY FOR NO-MORE-DRIVES JDRB001  
643 003176* 000420 BR 58 ;EXIT ROUTINE JDRB001  
644 003200* 016700 175472 28: MOV UNITNO,R0 ;USE AS AN INDEX  
645 003204* 136067 003242* 175460 BITB BITTAB(R0),DVICE ;TEST THIS DEVICE?  
646 003212* 001001 BNE 38 ;BR IF YES, ELSE  
647 003214* 000761 BR 16  
648 003216*  
649 ;  
650 003216* 004567 002322 JSR R5,READY ;SEE IF DRIVE IS READY JDRB001  
651 003222* 103402 BCS 48 ;BR IF IT WAS READY JDRB001  
652 003224* 004767 002120 JSR PC,NOTRDY ;ELSE GO CLEAR IT AND CHECK AGAIN  
653 003230* 004767 000702 48: JSR PC,GETDVT ;IS THIS LEGAL DRIVE TYPE? JDRB001  
654 003234* 103751 BCS 18 ;ILLEGAL TYPE: DROPPED, TRY NEXT JDRB001  
655 003236* 000241 CLC ;RETURN, NEXT DRIVE AVAILABLE JDRB001  
656 003240* 000297 58: RTS PC ;GO HOME JDRB001  
657 003242* 001001 BITTAB: 1001  
658 003244* 004004 4004  
659 003246* 020020 20020  
660 003250* 100100 100100  
661 ;  
662 ;  
663 003252* PICKRR: ;  
664 ; (* FUNCTION: SELECT NEXT DISK ADDRESS TO TEST, *) JDRB001  
665 ; (* AVOID WRITING OVER KNOWN BAD SPOTS OR DISK'S BAD *) JDRB001  
666 ; (* SPOT FILE (IF OPERATOR TELLS US IT'S THERE). CYL *) JDRB001  
667 ; (* IS "RELATIVE ADDRESS", MAX RANGE 0-410 (RP04/5) OR *) JDRB001  
668 ; (* 0-407 (RP06); THIS IS MOVED TO CYLNR AND INCR- *) JDRB001  
669 ; (* MENTED TO SECOND HALF IF THIS COPY OF RPD IS IN *) JDRB001  
670 ; (* PORT A MODE, *) JDRB001  
671 ;  
672 003252* 032767 000040 174536 BIT #NORAND,SR1 ;IF SEQUENTIAL SEEKS THEN JDRB001  
673 003260* 001403 BEQ 18 ;BEGIN JDRB001  
674 003262* 105267 175360 INCB SECTOR ; SECTOR := SECTOR + 1 JDRB001  
675 003266* 000432 BR 20 ;END JDRB001  
676 003270* 18: ;ELSE (* RANDOM SEEKS *) JDRB001  
677 ;BEGIN (* FUDGE 3 UN-RELATED JDRB001  
678 ; (* NUMBERS FROM 1 RAND CALL *) JDRB001  
679 003270* 104417 000000* RAND$,BEGIN  
680 003274* 016700 174554 MOV RANUM,R0 ; R0 := RANUM JDRB001  
681 003300* 042700 176000 BIC #176000,R0 ; R0 := R0 MOD 1024, JDRB001  
682 003304* 010067 175334 MOV R0,CYL ; CYL := R0 JDRB001  
683 003310* 016700 174540 MOV RANUM,R0 ; "RECHARGE" R0 JDRB001  
684 003314* 000300 SWAB R0 ; R0<010 := R0<810> JDRB001  
685 003316* 042700 177740 BIC #177740,R0 ; R0 := R0 MOD 32, JDRB001  
686 003322* 110067 175320 MOVFB F0,SECTOR ; SECTOR := R0 JDRB001  
687 003326* 016700 174522 MOV RANUM,R0 ; AND AGAIN JDRB001  
688 003332* 000241 CLC ; CARRY := 0 JDRB001  
689 003334* 006100 ROL R0 ; CARRY := R0<15:1> JDRB001
```

```
690 003336* 006100 ROL R0 ; JDRB001  
691 003340* 006100 ROL R0 ; JDRB001  
692 003342* 006100 ROL R0 ; JDRB001  
693 003344* 042700 177740 BIC #177740,R0 ; R0 := (R0<0:13>*8.) + R0<13:3> JDRB001  
694 ; (* NOTE: THE FORMULAE USED ARE *) JDRB001  
695 ; (* ABSOLUTELY IRRELEVANT *) JDRB001  
696 003350* 110067 175273 MOVFB R0,TRACK ; TRACK := R0 JDRB001  
697 003354* 28: ;END JDRB001  
698 003354* 126727 175266 000025 CMPB SECTOR,#21. ;IF SECTOR GT 21, THEN  
699 003362* 003412 BLE 38 ;BEGIN JDRB001  
700 003364* 105267 175257 INCB TRACK ; TRACK := TRACK + 1 JDRB001  
701 003370* 016700 175252 MOV SECTOR,R0 ; R0 := SECTOR JDRB001  
702 003374* 012701 000026 MOV #22,R1 ; R1 := # OF SECTORS IN TRACK JDRB001  
703 003400* 004767 000762 JSR PC,MODLUS ; R0 := R0 MOD R1 JDRB001  
704 003404* 110067 175236 MOVFB R0,SECTOR ; SECTOR := R0 JDRB001  
705 003410* 38: ;END JDRB001  
706 ; (* NOW, CUT VALUES DOWN TO LEGAL RANGES FOR CURRENT DRIVE TYPE *) JDRB001  
707 003410* 126727 175233 000022 CMPB TRACK,#18. ;IF TRACK GT 18, THEN  
708 003416* 003412 BLE 48 ;BEGIN JDRB001  
709 003420* 005267 175220 INC CYL ; CYL := CYL + 1 JDRB001  
710 003424* 116700 175217 MOVFB TRACK,R0 ; R0 := TRACK JDRB001  
711 003430* 012701 000023 MOV #19,R1 ; R1 := 19. JDRB001  
712 003434* 004767 000726 JSR PC,MODLUS ; R0 := R0 MOD R1 JDRB001  
713 003440* 110067 175203 MOVFB R0,TRACK ; TRACK := R0 JDRB001  
714 003444* 48: ;END JDRB001  
715 003444* 026767 175174 175166 CMP CYL,CYLLIM ;IF CYL GT # OF CYLINDERS IN RANGE THEN  
716 003452* 002421 BLT 58 ;BEGIN JDRB001  
717 003454* 032767 000040 174334 BIT #NORAND,SR1 ; IF SEQUENTIAL SEEKS THEN  
718 003462* 001405 BEQ 68 ; BEGIN JDRB001  
719 003464* 005067 175154 CLR CYL ; RESET CYLINDER JDRB001  
720 003470* 005067 175152 CLR DSKADR ; TRACK := 0, SECTOR := 0 JDRB001  
721 003474* 000410 BR 78 ; END JDRB001  
722 003476* 68: ; ELSE JDRB001  
723 ; BEGIN JDRB001  
724 003476* 016700 175142 MOV CYL,R0 ; R0 := CYL JDRB001  
725 003502* 016701 175132 MOV CYLLIM,R1 ; R1 := CYLLIM JDRB001  
726 003506* 004767 000654 JSR PC,MODLUS ; R0 := R0 MOD R1 JDRB001  
727 003512* 010067 175126 MOV R0,CYL ; CYL := R0 JDRB001  
728 003516* 78: ; END JDRB001  
729 003516* 58: ;END JDRB001  
730 003516* 016767 175122 175116 MOV CYL,CYLNDR ;CYLNDR := CYL JDRB001  
731 003524* 032767 000020 174264 BIT #BPORT,SR1 ;IF BPORT AND DUAL PORT THEN  
732 003532* 001407 BEQ 88 ;BEGIN JDRB001  
733 003534* 032767 000001 175126 BIT #BIT0,FLAG ;  
734 003542* 001403 BEQ 88 ;  
735 003544* 006767 175070 175070 ADD CYLLIM,CYLNDR ; INCREMENT TO SECOND HALF OF DISK  
736 003552* 88: ;END JDRB001  
737 003552* 026767 175064 175076 CMP CYLNDR,LASCYL ;IF CYLNDR GT LASCYL THEN  
738 003560* 003431 BLE 98 ;BEGIN JDRB001  
739 003562* 016767 175070 175052 MOV LASCYL,CYLNDR ; CYLNDR := LASCYL (* TRUNCATE *)  
740 003570* 032767 000040 174220 BIT #NORAND,SR1 ; IF SEQUENTIAL THEN  
741 003576* 001404 BEQ 108 ; BEGIN JDRB001  
742 003600* 005067 175040 CLR CYL ; RESET CYLINDER JDRB001  
743 003604* 005067 175036 CLR DSKADR ; TRACK := 0, SECTOR := 0 JDRB001  
744 003610* 108: ; END JDRB001  
745 003610* 126767 175033 175042 CMPB TRACK,LASTRCK ; IF TRACK GT LASTRCK THEN JDRB001
```

```

RPDC DEC/X11 SYSTEM EXERCISER MODULE MACY11 30A(1052) 13-JUN-79 13:05 PAGE 18
XRPDC0,P11 13-JUN-79 13:01 MODULE CODE ;DRB001 SEQ 0017

746 003616* 003412 BLE 118 ; BEGIN ;DRB001
747 003620* 116767 175034 175021 MOVB LASTRK,TRACK ; TRACK I= LASTRK ;DRB001
748 003626* 126767 175014 175026 CMPB SECTOR,LASSEC ; IF SECTOR GT LASSEC THEN ;DRB001
749 003634* 003403 BLE 128 ; BEGIN ;DRB001
750 003636* 116767 175020 175002 MOVB LASSEC,SECTOR ; SECTOR I= LASSEC ;DRB001
751 003644* 1281 ; END ;DRB001
752 003644* 1181 ; END ;DRB001
753 003644* 981 ; END ;DRB001
754 ;DRB001
755 ;(* NOW, CHECK BADSPOT TABLE *) ;DRB001
756 003644* 012700 000252* MOV #BADSPOT,R0 ;R0 I= ADDRESS OF TABLE ;DRB001
757 003650* 016701 174762 MOV NEXTBBK,R1 ;R1 I= END OF TABLE ;DRB001
758 003654* 116702 174766 MOVB SECTOR,R2 ;R2 I= SECTOR ;DRB001
759 003660* 116703 174763 MOVB TRACK,R3 ;R3 I= TRACK ;DRB001
760 003664* 016704 174752 MOV CYLNDR,R4 ;R4 I= CYLNDR ;DRB001
761 ;(* CALCULATE UPPER LIMIT OF THIS WRITE *) ;DRB001
762 003670* 066702 174772 ADD SIZSEC,R2 ;R2 I= R2 * # OF SECTORS IN WRITE ;DRB001
763 003674* 020227 000025 CMP R2,#21. ;IF R2 GT 21. THEN ;DRB001
764 003700* 003403 BLE 138 ;BEGIN ;DRB001
765 003702* 005203 INC R3 ; R3 I= R3 + 1 (INCR, UPPER TRACK) ;DRB001
766 003704* 162702 000026 SUB #22,,R2 ; TRUNCATE R2 ;DRB001
767 003710* 1381 ; END ;DRB001
768 003710* 066703 174750 ADD SIZTRK,R3 ;R3 I= R3 + TRKSIZ ;DRB001
769 003714* 020327 000022 CMP R3,#18. ;IF R3 GT 18. THEN ;DRB001
770 003720* 003403 BLE 148 ;BEGIN ;DRB001
771 003722* 005204 INC R4 ; R4 I= R4 + 1 (INCR, UPPER CYLINDER) ;DRB001
772 003724* 162703 000023 SUB #19,,R3 ; TRUNCATE R3 ;DRB001
773 003730* 1481 ; END ;DRB001
774 003730* 020001 1581 CMP R0,R1 ;WHILE R0 LT R1 DO ;DRB001
775 003732* 002030 BGE 168 ;BEGIN ;DRB001
776 003734* 026710 174702 CMP CYLNDR,(R0) ; IF CYLNDR LE (R0) AND R4 GE (R0) ;DRB001
777 003740* 003022 BGT 178 ; AND TRACK LE 3(R0) AND R3 GE 3(R0) ;DRB001
778 003742* 020410 CMP R4,(R0) ; AND SECTOR LE 2(R0) AND R2 GE 2(R0) THEN ;DRB001
779 003744* 002420 BLT 178 ; ;DRB001
780 003746* 126760 174675 000003 CMPB TRACK,3(R0) ; ;DRB001
781 003754* 003014 RGT 178 ; ;DRB001
782 003756* 120360 000003 CMPB R3,3(R0) ; ;DRB001
783 003762* 002411 BLT 178 ; ;DRB001
784 003764* 126760 174656 000002 CMPB SECTOR,2(R0) ; ;DRB001
785 003772* 003005 BGT 178 ; ;DRB001
786 003774* 120260 000002 CMPB R2,2(R0) ; ;DRB001
787 004000* 002402 BLT 178 ; ;DRB001
788 004002* 000167 177244 JMP PICKBK ; TRY ANOTHER SECTOR ;DRB001
789 004006* 1781 ; END ;DRB001
790 004006* 062700 000004 ADD #4,R0 ; INCREMENT TO NEXT BAD SPOT ;DRB001
791 004012* 000746 BR 158 ; END ;DRB001
792 004014* 1681 ; ;DRB001
793 004014* 000207 RTS PC ;CPU GO HOME!!!! ;DRB001
794 ;DRB001
795 ;DRB001
796 004016* WRTLIM: ;DRB001
797 ;(* WE ALLOW USER TO CHANGE WBUF SZ "AT WILL"--THEREFORE *) ;DRB001
798 ;(* WE DO NOT AUTOMATICALLY KNOW THE TRANSFER SIZE, *) ;DRB001
799 ;(* COMPUTATION IS MADE EASIER, FURTHERMORE, IF WE HAVE *) ;DRB001
800 ;(* THE SIZE BROKEN INTO TRACKS AND SECTORS (SINCE *) ;DRB001
801 ;(* WBUF SZ IS ONE WORD, THE TRANSFER CAN'T BE GREATER *) ;DRB001

```

```

RPDC DEC/X11 SYSTEM EXERCISER MODULE MACY11 30A(1052) 13-JUN-79 13:05 PAGE 19
XRPDC0,P11 13-JUN-79 13:01 MODULE CODE ;DRB001 SEQ 0018

802 ;(* THAN 12 TRACKS), THIS ROUTINE ALSO CALCULATES THE *) ;DRB001
803 ;(* UPPER LIMITS ON DISK WRITES SO AS NOT TO OVERFLOW *) ;DRB001
804 ;(* HIGH CYLINDER, *) ;DRB001
805 ;DRB001
806 004016* 016700 174120 MOV WBUF SZ,R0 ;R0 I= ALLOCATED WRITE BUFFER ;DRB001
807 004022* 000300 SWAB R0 ;R0 I= R0 / 256. ;DRB001
808 004024* 042700 177400 BIC #177400,R0 ; (* R0 15 WBUF SZ IN SECTORS *) ;DRB001
809 004030* 005067 174630 CLR SIZTRK ;SIZTRK I= 0 ;DRB001
810 004034* 020027 000026 CMP R0,#22. ;WHILE R0 GE 22. DO ;DRB001
811 004040* 002405 BLT 28 ;BEGIN ;DRB001
812 004042* 105267 174616 INCB SIZTRK ; SIZTRK I= SIZTRK + 1 ;DRB001
813 004046* 162700 000026 SUB #22,,R0 ; DECREMENT SECTORS BY 1 TRACK ;DRB001
814 004052* 000770 BR 18 ;END ;DRB001
815 004054* 010067 174606 MOV R0,SIZSEC ;SIZSEC I= REMAINDER ;DRB001
816 004060* 012701 000022 MOV #18,,R1 ;R1 I= HIGH TRACK ;DRB001
817 004064* 012702 000025 MOV #21,,R2 ;R2 I= HIGH SECTOR ;DRB001
818 004070* 160002 SUB R0,R2 ;R2 I= R2 - SIZSEC ;DRB001
819 004072* 005702 TST R2 ;IF R2 LT 0 THEN ;DRB001
820 004074* 002003 BGE 38 ;BEGIN ;DRB001
821 004076* 062702 000026 ADD #22,,R2 ; INCREMENT BACK UP ;DRB001
822 004102* 005301 DEC R1 ; CAN'T FIT THAT LAST TRACK... ;DRB001
823 004104* 166701 174554 ;END ;DRB001
824 004104* 005701 SUB SIZTRK,R1 ;R1 I= R1 - SIZTRK ;DRB001
825 004110* 002004 TST R1 ;IF R1 LT 0 THEN ;DRB001
826 004112* 002004 BGE 48 ;BEGIN ;DRB001
827 004114* 062701 000023 ADD #19,,R1 ; INCREMENT R1 BACK UP ;DRB001
828 004120* 005367 174532 DEC LASCYL ; LASCYL I= LASCYL - 1 ;DRB001
829 004124* 481 ; END ;DRB001
830 004124* 010167 174530 MOV R1,LASTRK ;LASTRK I= HIGHEST TRACK WE CAN WRITE ;DRB001
831 004130* 010267 174526 MOV R2,LASSEC ;LASSEC I= HIGHEST SECTOR WE CAN WRITE ;DRB001
832 004134* 000207 RTS PC ;END OF ROUTINE ;DRB001
833 004136* GETDVT: ;DRB001
834 ;(* FIND WHICH TYPE OF DRIVE THIS IS, IF RP04/5, SET CYLLIM *) ;DRB001
835 ;(* TO 206. IF RP06, SET CYLLIM TO 408. DROP THE DRIVE IF *) ;DRB001
836 ;(* IT IS NOT A RP04/5/6 *) ;DRB001
837 ;DRB001
838 004136* 016777 174534 175634 MOV UNITNO,0RHCS2 ;SELECT THE DRIVE ;DRB001
839 004144* 027727 175646 024022 CMP 0RPDT,#24022 ;IF IT IS AN RP06 DUAL PORT THEN ;DRB001
840 004152* 001012 BNE 18 ;BEGIN ;DRB001
841 004154* 052767 000001 174506 BIS #BIT0,FLAG ; DUAL PORT ;DRB001
842 004162* 012767 000630 174450 MOV #408,,CYLLIM ; CYLLIM I= 408. ;DRB001
843 004170* 012767 001456 174460 MOV #814,,LASCYL ; LASCYL I= DISK HIGH LIM ;DRB001
844 004176* 000471 BR 28 ;END ;DRB001
845 004200* 181 ;ELSE ;DRB001
846 004200* 027727 175612 024021 CMP 0RPDT,#24021 ;IF IT IS RP04/5 DUAL PORT THEN ;DRB001
847 004206* 001404 REQ 38 ; ;DRB001
848 004210* 027727 175602 024020 CMP 0RPDT,#24020 ; ;DRB001
849 004216* 001012 BNE 48 ;BEGIN ;DRB001
850 004220* 381 ; ;DRB001
851 004220* 052767 000001 174442 BIS #BIT0,FLAG ; DUAL PORT ;DRB001
852 004226* 012767 000316 174404 MOV #206,,CYLLIM ; CYLLIM I= 206. ;DRB001
853 004234* 012767 000632 174414 MOV #410,,LASCYL ; LASCYL I= DISK HIGH LIM ;DRB001
854 004242* 000447 BR 58 ;END ;DRB001
855 004244* 481 ;ELSE ;DRB001
856 004244* 027727 175546 020022 CMP 0RPDT,#20022 ;IF RP06 SINGLE PORT THEN ;DRB001
857 004252* 001012 BNE 68 ;BEGIN ;DRB001

```

```
858 004254* 042767 000001 174406 BIC #BIT0,FLAG ; SINGLE PORT JDRB001
859 004262* 012767 001457 174350 MOV #815,,CYLLIM ; CAN USE ENTIRE DISK JDRB001
860 004270* 012767 001456 174360 MOV #814,,LASCYL ; LASCYL != DISK HIGH LIMIT JDRB001
861 004276* 000431 BR 28 ;END JDRB001
862 004300* 68: ;ELSE JDRB001
863 004300* 027727 175512 020021 CMP @RPDT,#20021 ;IF RP04/5 SINGLE PORT THEN JDRB001
864 004306* 001404 BEQ 88 ; JDRB001
865 004310* 027727 175502 020020 CMP @RPDT,#20020 ; JDRB001
866 004316* 001012 BNE 98 ; JDRB001
867 004320* 88: ;BEGIN JDRB001
868 004320* 042767 000001 174342 BIC #BIT0,FLAG ; SINGLE PORT JDRB001
869 004326* 012767 000633 174304 MOV #411,,CYLLIM ; USE ENTIRE DISK JDRB001
870 004334* 012767 000632 174314 MOV #410,,LASCYL ; LASCYL != DISK HIGH LIMIT JDRB001
871 004342* 000407 BR 28 ;END JDRB001
872 004344* 98: ;ELSE JDRB001
873 ;BEGIN JDRB001
874 004344* 004767 000050 JSR PC,DROP ; DROP THE DRIVE JDRB001
875 004350* 104403 000000* 007012* MSGN#,BEGIN,BADTYP ;ASCII MESSAGE CALL WITH COMMON HEADER JDRB001
876 004356* 000261 SEC ; ERROR RETURN JDRB001
877 004360* 000401 BR 78 ; JDRB001
878 004362* 58: ;END JDRB001
879 004362* 28: ; JDRB001
880 004362* 000241 CLC ;NO ERROR JDRB001
881 004364* 78: ; JDRB001
882 004364* 000207 RTS PC ;RETURN JDRB001
883 004366* MODLUS: JDRB001
884 ;(* THIS LITTLE ROUTINE COMPUTES R0 MODULUS R1 *) JDRB001
885 ;(* I.E., THE REMAINDER OF R0/R1, AND RETURNS THE *) JDRB001
886 ;(* ANSWER IN R0. *) JDRB001
887 004366* 020001 18: CMP R0,R1 ;WHILE R0 GE R1 DO JDRB001
888 004370* 002402 BLT 28 ;BEGIN JDRB001
889 004372* 160100 SUB R1,R0 ; R0 != R0-R1 JDRB001
890 004374* 000774 BF 18 ;END JDRB001
891 004376* 28: ; JDRB001
892 004376* 000207 RTS PC ;THAT'S ALL JDRB001
893
894 004400* 012700 000716* CLRBR: MOV #RBUF,R0 ;CLEAR RBUF BUFFER JDRB001
895 004404* 016701 173522 MOV RRUF$Z,R1 ;GET ITS ADDR AND SIZE JDRB001
896 004410* 005020 CLRCLR: CLR (R0)+ ;CLEAR ANOTHER JDRB001
897 004412* 005301 DEC R1 ;COUNT ANOTHER JDRB001
898 004414* 001375 BNE CLRCLR ;BR BACK TILL DONE JDRB001
899 004416* 000207 RTS PC
900
901 004420* 012701 000001 DROP: MOV #1,R1 ; INITIALIZE DROP PICKER JDRB001
902 004424* 016700 174246 MOV UNITNO,R0 ; GET THE DRIVE NUMBER JDRB001
903 004430* 001403 BEQ 28 ; IF DRIVE 0 GO DROP IT JDRB001
904 004432* 006301 18: AGL R1 ; POINT TO NEXT DRIVE JDRB001
905 004434* 005300 DEC R0 ; IS THIS THE ONE ? JDRB001
906 004436* 001375 BNE 18 ; NO, LOOK AGAIN JDRB001
907 004440* 040167 174226 BIC R1,DVICE ; DROP THE DRIVE JDRB001
908 ;***** JDRB001
909 ;CONVERT UNITNO TO ASCII AND JDRB001
910 ;STORE AT ADRI JDRB001
911 004444* 104420 000000* 000676* OTOA$,BEGIN,UNITNO,ADR1
912 004452* 007024* ;*****
913 ;*****
```

```
914 004454* 000207 RTS PC ; RETURN
915 ;
916
917 004456* ERRORS: JDRB001
918 004456* 005777 175306 TST @RHCS1 ;IF NO ERROR THEN JDRB001
919 004462* 100403 BMI 18 ;BEGIN JDRB001
920 004464* 000241 CLC ;SET NO ERROR JDRB001
921 004466* 000167 000654 JMP 28 ; RETURN NO ERROR JDRB001
922 ;END JDRB001
923 004472* 18: ;ELSE JDRB001
924 ;BEGIN JDRB001
925 004472* 005000 CLR R0 ; R0 != 0 JDRB001
926 004474* 016701 175270 MOV RHCS1,R1 ; BEGIN OF REGISTER POINTERS JDRB001
927 004500* 38: ; REPEAT JDRB001
928 ; BEGIN JDRB001
929 004500* 012160 000552* MOV (R1)+,S(R0) ; MOV REGISTER TO TEMPORARY JDRB001
930 004504* 005720 TST (R0)+ ; ADVANCE POINTER JDRB001
931 004506* 020027 000046 CME R0,#46 ; END JDRB001
932 004512* 002772 BLT 38 ; UNTIL R0 GE 46 JDRB001
933 004514* 004767 001162 JSR PC,ERSUB1 ; GET SOME TRIVIAL INFO JDRB001
934 004520* 005767 174036 TST S+10 ; IF DATA LATE ERROR THEN JDRB001
935 004524* 100015 BPL 48 ; BEGIN JDRB001
936 004526* 005267 174146 INC DLTCNT ; COUNT IT UNCONDITIONALLY JDRB001
937 004532* 032767 000004 173256 BIT @DLATE,SRI ; IF OPERATOR WANTS MESSAGE THEN JDRB001
938 004540* 001005 BNE 58 ; BEGIN JDRB001
939 004542* 104403 000000* 006726* MSGN$,BEGIN,DLTERR ;ASCII MESSAGE CALL WITH COMMON HEADER JDRB001
940 004550* 000167 000440 JMP 68 ; HARD ERROR PROCESS JDRB001
941 ;END JDRB001
942 004554* 58: ; ELSE JDRB001
943 004554* 000167 000502 JMP 78 ; NO MESSAGE "HARD ERROR" RETURN JDRB001
944 ;END JDRB001
945 004560* 032767 000100 174000 48: BIT #BIT6,S+14 ; IF ER1<6> OR ( ER1<4> AND NOT JDRB001
946 004566* 001020 BNE 88 ; ER1<15> AND NOT ER1<8> ) THEN JDRB001
947 004570* 032767 000020 173770 BIT #BIT4,S+14 ; JDRB001
948 004576* 001471 BEQ 98 ; JDRB001
949 004600* 032767 100000 173760 BIT #BIT5,S+14 ; JDRB001
950 004606* 001402 BEQ 308 ; JDRB001
951 004610* 000167 000400 JMP 68 ; (* HARD ERROR *) JDRB001
952 004614* 032767 000400 173744 308: BIT #BIT8,S+14 ; JDRB001
953 004622* 001402 BEQ 88 ; JDRB001
954 004624* 000167 000442 JMP 108 ; (* SOFT ERROR *) JDRB001
955 004630* 88: ; BEGIN (* BADSPOT *) JDRB001
956 004630* 032767 000100 173160 BIT #BDSPT,SRI ; IF OPERATOR WANTS TYPEOUT THEN JDRB001
957 004636* 001434 BEQ 128 ; JDRB001
958 004640* 118: ; BEGIN JDRB001
959 004640* 116767 173714 174004 MOV#B S+6,BADSEC ; SET CURRENT SECTOR JDRB001
960 004646* 116767 173707 174000 MOV#B S+7,BADTRK ; SET CURRENT TRACK JDRB001
961 004654* 016767 173730 173766 MOV#B S+36,BADCYL ; SET CURRENT CYLINDER JDRB001
962 ;***** JDRB001
963 ;CONVERT UNITNO TO ASCII AND JDRB001
964 ;STORE AT ADRI JDRB001
965 004662* 104420 000000* 000676* OTOA$,BEGIN,UNITNO,ADR1
966 004670* 007024* ;*****
967 ;*****
968 ;CONVERT BADCYL TO ASCII AND
969
```

```
970 ;STORE AT CYLNO
971 004672 104420 000000 000650 OTOAS,BEGIN,BADCYL,CYLNO
972 004700 006614
973 ;*****
974 ;*****
975 ;CONVERT BADTRK TO ASCII AND
976 ;STORE AT TRKNO
977 004702 104420 000000 000654 OTOAS,BEGIN,BADTRK,TRKNO
978 004710 006632
979 ;*****
980 ;*****
981 ;CONVERT BADSEC TO ASCII AND
982 ;STORE AT SECNO
983 004712 104420 000000 000652 OTOAS,BEGIN,BADSEC,SECNO
984 004720 006650
985 ;*****
986 ;*****
987 MSGNS,BEGIN,BADMES ;ASCII MESSAGE CALL WITH COMMON HEADER
988 ; END ;DRB001
989 ; IF MORE SPACE IN TABLE THEN ;DRB001
990 ; BEGIN ;DRB001
991 ; R0 POINTS TO NEXT FREE SLOT IN TABLE ;DRB001
992 ; STUFF CYLINDER NUMBER ;DRB001
993 ; STUFF REST OF DISK ADDRESS ;DRB001
994 ; UPDATE POINTER ;DRB001
995 ; END ;DRB001
996 ; NO MESSAGE "HARD ERROR" SO NO RE-TRY ;DRB001
997 ; END (* BAD SPOT *) ;DRB001
998 ; IF MASSBUSS CTRL PARITY ERROR THEN ;DRB001
999 ; BEGIN ;DRB001
1000 ;ASCII MESSAGE CALL WITH COMMON HEADER ;DRB001
1001 ; HARD ERROR ;DRB001
1002 ; END ;DRB001
1003 ; IF MASSBUSS DATA PARITY ERROR THEN ;DRB001
1004 ; BEGIN ;DRB001
1005 ;ASCII MESSAGE CALL WITH COMMON HEADER ;DRB001
1006 ; HARD ERROR ;DRB001
1007 ; END ;DRB001
1008 ; IF WRITE-CHECK ERROR THEN ;DRB001
1009 ; BEGIN ;DRB001
1010 ;ASCII MESSAGE CALL WITH COMMON HEADER ;DRB001
1011 ; TRERR ;DRB001
1012 ; SOFT ERROR ;DRB001
1013 ; END ;DRB001
1014 ; IF ANY OTHER TRANSFER ERROR THEN ;DRB001
1015 ; BEGIN ;DRB001
1016 ;ASCII MESSAGE CALL WITH COMMON HEADER ;DRB001
1017 ; HARD ERROR ;DRB001
1018 ; END ;DRB001
1019 ; IF OTHER HARD ERROR THEN ;DRB001
1020 ; BEGIN ;DRB001
1021 ;ASCII MESSAGE CALL WITH COMMON HEADER ;DRB001
1022 ; TRERR ;DRB001
1023 ; SOFT ERROR ;DRB001
1024 ; END ;DRB001
1025 ; (* SOFT--SPECIAL CASE *) ;DRB001
```

```
1026 005124 005767 173462 TST S+40 ;DRB001
1027 005130 001014 BNE 200 ;DRB001
1028 005132 032767 000156 173454 BIT #156,S+42 ;DRB001
1029 005140 001010 BNE 200 ;DRB001
1030 005142 032767 000200 172646 BIT #RH70,SR1 ;DRB001
1031 005150 001405 BEQ 220 ;DRB001
1032 005152 032777 140000 174662 BIT #140000,APHCS3 ;DRB001
1033 005160 001401 BEQ 220 ;DRB001
1034 005162 ;DRB001
1035 005162 000414 BR 60 ;DRB001
1036 005164 ;DRB001
1037 005164 032767 120400 173374 BIT #120400,S+14 ;DRB001
1038 005172 001004 BNE 230 ;DRB001
1039 005174 032767 140000 173412 BIT #140000,S+42 ;DRB001
1040 005202 001401 BEQ 240 ;DRB001
1041 005204 ;DRB001
1042 005204 000432 BR 100 ;DRB001
1043 005206 ;DRB001
1044 ;DRB001
1045 005206 104403 000000 006754 ;(* IF WE FALL THRU TO HERE, WE HAVE AN UNKNOWN ERROR *) ;DRB001
1046 005214 ;ASCII MESSAGE CALL WITH COMMON HEADER ;DRB001
1047 005214 016700 174572 ; (* PROCESS HARD ERROR *) ;DRB001
1048 005220 105767 173336 MOV PHDR,R0 ;SAVE RHDB ADDRESS ;DRB001
1049 005224 100407 TSTB S+10 ;IF OR NOT SET THEN ;DRB001
1050 005226 012767 000706 174556 BMI 250 ;BEGIN ;DRB001
1051 005234 012767 000001 172644 MOV #ZERO,RHDB ;FAKE ADDRESS TO PREVENT DLT ON ACCESS ;DRB001
1052 005242 000402 MOV #1,ERRTYP ;DATA PROBLEM ;DRB001
1053 005244 ;DRB001
1054 005244 005067 172636 BR 260 ;END ;DRB001
1055 005250 ;DRB001
1056 ;DRB001
1057 005250 104405 000000 001716 ;***** ;DRB001
1058 HRDRS,BEGIN,TABLE ;DRB001
1059 ;***** ;DRB001
1060 005256 010067 174530 MOV R0,RHDB ;RESTORE DATA BUFFER ADDRESS ;DRB001
1061 005262 052767 000200 173400 BIS #BIT7,FLAG ;HARD ERROR FLAG I NO RETRY ;DRB001
1062 005270 000423 BR 270 ;DO COMMON ERROR EXIT ;DRB001
1063 ;DRB001
1064 ; (* PROCESS SOFT ERROR *) ;DRB001
1065 005272 016700 174514 MOV RHDB,R0 ;SAVE RHDB ADDRESS ;DRB001
1066 005276 105767 173260 TSTB S+10 ;IF OR NOT SET THEN ;DRB001
1067 005302 100407 BMI 280 ;BEGIN ;DRB001
1068 005304 012767 000706 174500 MOV #ZERO,RHDB ;DON'T TRY TO ACCESS REGISTER ;DRB001
1069 005312 012767 000001 172566 MOV #1,ERRTYP ;DATA ERROR ;DRB001
1070 005320 000402 BR 290 ;END ;DRB001
1071 005322 005067 172560 CLR ERRTYP ;ELSE ;DRB001
1072 005326 ;DRB001
1073 ;DRB001
1074 005326 104406 000000 001716 ;***** ;DRB001
1075 SOFRS,BEGIN,TABLE ;DRB001
1076 005334 005067 173330 CLR FLAG ;NO HARD ERROR ;DRB001
1077 ;DRB001
1078 005340 004767 000004 2700 JSR PC,NOTRDY ;CLEAR OUT ERRORS, MAKE SURE DRIVE'S OK ;DRB001
1079 005344 000261 SEC ;SET ERROR RETURN ;DRB001
1080 005346 000205 RTS R5 ;RETURN HOMEWARD, ;DRB001
1081 ;DRB001
```

```

1082
1083 005350 012767 077777 173334 NOTRDY: MOV #77777,CLK ; SET THE TIMER
1084 005356 012767 077777 173334 48:
1085 005356 032777 004000 174404 BIT #BIT11,0RHCS1 ;DO I HAVE THE DRIVE DVA?
1086 005364 001406 BEQ 68 ;NO
1087 005366 010046 MOV R0,=(SP)
1088 005370 012600 MOV (SP)+,R0 ;JUST WASTE A LITTLE TIME
1089 005372 032777 004000 174370 BIT #BIT11,0RHCS1 ;STILL GOT IT?
1090 005400 001017 BNE 28
1091 005402 000000 68:
1092 005402 104407 000000 BREAK0,BEGIN ;TEMPORARY RETURN TO MONITOR....
1093 005406 104407 000000 BREAK0,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
1094 005412 104407 000000 BREAK0,BEGIN ;TEMPORARY RETURN TO MONITOR....
1095 005416 104407 000000 BREAK0,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
1096 005422 005367 173264 DEC CLK ;COUNT # OF TRIES
1097 005426 001353 BNE 48 ;NOT DONE YET
1098 005430 104403 000000 MSGN0,BEGIN,NOT ;ASCII MESSAGE CALL WITH COMMON HEADER
1099 005436 000426 BR 78 ;COULD NOT GET DRIVE
1100 005440 004567 175160 JSR R5,CLEAR ;SET THE CONTROLLER AND DRIVE
1101 005444 004567 000074 JSR P5,READY ; IS DRIVE READY ?
1102 005450 000434 BR 18 ; YES, CONTINUE
1103 005452 004767 000224 JSR PC,ERSUR1 ; LOAD ERROR INFORMATION
1104 005456 005000 CLR R0 ;MOVE DRIVE REG INTO TABLE
1105 005460 016701 174304 MOV RHCS1,R1
1106 005464 012160 000552 MOV (R1)+,S(R0)
1107 005470 005720 TST (R0)+
1108 005472 022700 CMP #46,R0
1109 005476 001372 BNE 228
1110 005500 012767 000000 172400 MOV #6,ERRTYP ;DRIVE NOT READY
1111 *****
1112 005506 104405 000000 001716 HRDR0,BEGIN,TABLE ; DRIVE NOT READY
1113 *****
1114 ***** ;**=1
1115 005514 004767 175056 78: JSR PC,RELESE ;RELEASE THE DRIVE
1116 005520 004767 176674 JSR PC,DROP ; NO, DROP THE DRIVE
1117 005524 104403 000000 007000 MSGN0,BEGIN,DRP ;ASCII MESSAGE CALL WITH COMMON HEADER
1118 005532 016706 172274 MOV SPOINT,R6
1119 005536 000167 174446 JMP LOOP2
1120 005542 000207 18: RTS PC ;RETURN
1121 ;
1122 -----
1123
1124 005544 016777 173126 174226 READY: MOV UNITNO,0RHCS2 ; LOAD UNIT ADDRESS
1125 005552 017700 174212 MOV 0RHCS1,R0
1126 005556 032700 004000 BIT #BIT11,R0
1127 005562 001433 BEQ 18
1128 005564 017700 174212 MOV 0RPDS,R0 ; SAVE STATUS IN R0
1129 005570 105700 TSTB R0 ; DRIVE READY ?
1130 005572 100027 BPL 18 ; NO
1131 005574 032700 000100 BIT #BIT6,R0 ; VOLUME VALID ?
1132 005600 001424 BEQ 18 ; NO
1133 005602 032700 000400 BIT #BIT8,R0 ; DRIVE PRESENT ?
1134 005606 001421 BEQ 18 ; NO
1135 005610 032700 004000 BIT #BIT11,R0 ; WRITE LOCKED ?
1136 005614 001016 BNE 18 ; YES
1137 005616 032700 010000 BIT #BIT12,R0 ; MEDIUM ON LINE ?

```

```

1138 005622 001413 BEQ 18 ; NO
1139 005624 032700 040000 BIT #BIT14,R0 ; ANY ERRORS ?
1140 005630 001010 BNE 18 ; YES
1141 005632 005700 TST R0 ; ATTENTION SET ?
1142 005634 100406 BMI 18 ; YES
1143 005636 032777 004000 174124 BIT #BIT11,0RHCS1 ;DVA SET?
1144 005644 001402 BEQ 18 ;BR IF NOT
1145 ***** ;**=3
1146 005646 000261 SEC ; READY
1147 005650 000401 BR 28 ; RETURN
1148 005652 000241 18: CLC ; NOT READY
1149 005654 000205 28: RTS R5 ;RETURN
1150 ;
1151 -----
1152
1153 005656 014167 172224 ERSUB2: MOV =(R1),ASH ; LOAD THE DATA
1154 005662 010167 172214 MOV F1,SBADR ; LOAD ADDRESS OF DATA WRITTEN
1155 005666 014267 172216 MOV =(R2),AWAS ; LOAD THE DATA
1156 005672 010267 172206 MOV R2,WASADR ; LOAD ADDRESS OF DATA READ
1157 005676 005721 TST (R1)+ ; RESET REG. 1
1158 005700 005722 TST (R2)+ ; RESET REG. 2
1159
1160 005702 016767 174062 172170 ERSUB1: MOV RHCS1,CSRA ; LOAD ADR OF CURRENT CSR
1161 005710 017767 174054 172164 MOV 0RHCS1,ACSR ; LOAD CONTENTS OF CURRENT CSR
1162 005716 000207 RTS PC ; RETURN
1163
1164
1165 005720 017700 174044 REZET: MOV 0RHCS1,R0
1166 005724 032700 004000 BIT #BIT11,R0
1167 005730 001005 BNE 38
1168 005732 104407 000000 BREAK0,BEGIN ;TEMPORARY RETURN TO MONITOR....
1169 005736 104407 000000 BREAK0,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
1170 005742 000766 ER REZET
1171 005744 012777 000040 174026 38: MOV #BITS,0RHCS2 ; ISSUE AN RH11 INIT
1172 005752 012767 077777 172732 MOV #77777,CLK ; SET THE TIMER
1173 005760 105777 174004 18: TSTB 0PHCS1 ; CONTROLLER READY ?
1174 005764 100417 BMI 28 ; YES, CONTINUE
1175 005766 104407 000000 BREAK0,BEGIN ;TEMPORARY RETURN TO MONITOR....
1176 005772 104407 000000 BREAK0,BEGIN ;THEN CONTINUE AT NEXT INSTRUCTION.
1177 005776 005367 172710 DEC CLK ; WAIT SOME MORE ?
1178 006002 001366 BNE 18 ; YES
1179 006004 012767 000003 172074 MOV #3,ERRTYP ;CONTROLLER NOT READY
1180 *****
1181 006012 104405 000000 000000 HRDR0,BEGIN,NULL ; CONTROLLER NOT READY
1182 *****
1183 006020 104410 000000 END0,BEGIN ;
1184 006024 017746 173756 28: MOV 0RPAS,=(SP)
1185 006030 012677 173752 MOV (SP)+,0RPAS
1186 006034 000207 RTS PC ; RETURN
1187 ;
1188 -----
1189
1190 006036 016700 171744 SETUP: MOV ADDR,R0 ; GET DEVICE ADDRESS
1191 006042 010067 173722 MOV R0,RHCS1 ; GENERATE REGISTER ADDRESSES
1192 006046 062700 000002 ADD #2,R0
1193 006052 010067 173714 MOV R0,RHWC

```

XRPDCC,P11	13-JUN-79	13:01	MODULE	CODE	OPERATION	ADDRESS	DATA	MESSAGE	STATUS
1194	006056*	062700	000002		ADD	#2,R0			
1195	006062*	010067	173706		MOV	R0,RHBA			
1196	006066*	062700	000002		ADD	#2,R0			
1197	006072*	010067	173700		MOV	R0,RPDA			
1198	006076*	062700	000002		ADD	#2,R0			
1199	006102*	010067	173672		MOV	R0,RHCS2			
1200	006106*	062700	000002		ADD	#2,R0			
1201	006112*	010067	173664		MOV	R0,RPDS			
1202	006116*	062700	000002		ADD	#2,R0			
1203	006122*	010067	173656		MOV	R0,RPER1			
1204	006126*	062700	000002		ADD	#2,R0			
1205	006132*	010067	173650		MOV	R0,RPAS			
1206	006136*	062700	000002		ADD	#2,R0			
1207	006142*	010067	173642		MOV	R0,RPLA			
1208	006146*	062700	000002		ADD	#2,R0			
1209	006152*	010067	173634		MOV	R0,RHDB			
1210	006156*	062700	000002		ADD	#2,R0			
1211	006162*	010067	173626		MOV	R0,RPFR			
1212	006166*	062700	000002		ADD	#2,R0			
1213	006172*	010067	173620		MOV	R0,RPDT			
1214	006176*	062700	000002		ADD	#2,R0			
1215	006202*	010067	173612		MOV	R0,RPFRN			
1216	006206*	062700	000002		ADD	#2,R0			
1217	006212*	010067	173604		MOV	R0,RPOF			
1218	006216*	062700	000002		ADD	#2,R0			
1219	006222*	010067	173576		MOV	R0,RPDC			
1220	006226*	062700	000002		ADD	#2,R0			
1221	006232*	010067	173570		MOV	R0,RPCC			
1222	006236*	062700	000002		ADD	#2,R0			
1223	006242*	010067	173562		MOV	R0,RPER2			
1224	006246*	062700	000002		ADD	#2,R0			
1225	006252*	010067	173554		MOV	R0,RPFR3			
1226	006256*	062700	000002		ADD	#2,R0			
1227	006262*	010067	173546		MOV	R0,RPPEC1			
1228	006266*	062700	000002		ADD	#2,R0			
1229	006272*	010067	173540		MOV	R0,RPPEC2			
1230									
1231	006276*	032767	001000	171552	BIT	#ADDR22,RES1		***7	JDRB001
1232	006304*	001415			BEQ	10		;DO WE HAVE 22-BIT ADDR SUPPORT?	JDRB001
1233	006306*	032767	000200	171502	BIT	#RH70,SR1		;NO	JDRB001
1234	006314*	001411			BEQ	10		;IS THIS AN RH70	JDRB001
1235	006316*	062700	000002		ADD	#2,R0			JDRB001
1236	006322*	010067	173512		MOV	R0,RHBAE			
1237	006326*	062700	000002		ADD	#2,R0			
1238	006332*	010067	173504		MOV	R0,RHCS3			
1239	006336*	000403			BR	20			JDRB001
1240	006340*	042767	001000	171510	BIC	#ADDR22,RES1		; CAN'T USE 22 BIT ADDRESSING	JDRB001
1241	006346*	016700	171436	281	MOV	VECTOR,R0		; GET VECTOR ADDRESS	
1242	006352*	012720	003036*		MOV	#NTRUPT,(R0)+		; SET POINTER JUST IN CASE	
1243	006356*	116710	171430		MOVB	BR1,(R0)		; SET PRIORITY	
1244	006362*	000207			PTS	PC		; RETURN	
1245									

XRPDCC,P11	13-JUN-79	13:01	MODULE	MESSAGES	OPERATION	ADDRESS	DATA	MESSAGE	STATUS
1246					.SBTTL	MODULE MESSAGES			JDRB001
1247								***40	JDRB001
1248	006364*	042440	051122	051117	MES1:	.ASCIZ	' ERROR%		IDFB001
1249	006372*	000045							
1250	006374*	050040	051101	052111	MES2:	.ASCIZ	' PARITY'		IDFB001
1251	006402*	000131							
1252	006404*	042040	044522	042526	MES3:	.ASCIZ	' DRIVE '		JDRB001
1253	006412*	000040							
1254	006414*	046440	051501	041123	MES4:	.ASCIZ	' MASSBUS'		JDRB001
1255	006422*	051525	000						
1256	006425*	040	040504	040524	MES5:	.ASCIZ	' DATA'		JDRB001
1257	006432*	000							
1258	006433*	040	042522	051124	MES6:	.ASCIZ	' RETRY EXCEEDED%'		JDRB001
1259	006440*	020131	054105	042503					
1260	006446*	042105	042105	000045					
1261	006454*	046040	052101	000105	MES7:	.ASCIZ	' LATE'		JDRB001
1262	006462*	047516	020124	042522	MES8:	.ASCIZ	' NOT READY%		JDRB001
1263	006470*	042101	022531	000					
1264	006475*	040	047503	046125	MES9:	.ASCIZ	' COULD NOT GET'		JDRB001
1265	006502*	020104	047516	020124					
1266	006510*	042507	000124						
1267	006514*	042040	047522	050120	MES10:	.ASCIZ	' DROPPED'		JDRB001
1268	006522*	042105	000						
1269	006525*	072	047040	052117	MES11:	.ASCIZ	' ! NOT RP04/5/6%'		JDRB001
1270	006532*	051040	030120	027464					
1271	006540*	027465	022466	000					
1272	006545*	040	051124	047101	MES12:	.ASCIZ	' TRANSFER'		JDRB001
1273	006552*	043123	051105	000					
1274	006557*	045	000		MES13:	.ASCIZ	' %'		JDRB001
1275	006561*	072	041040	042101	MES15:	.ASCIZ	' ! BADSPOT AT (OCTAL) CYL: '		JDRB001
1276	006566*	050123	052117	040440					
1277	006574*	020124	047450	052103					
1278	006602*	046101	020051	054503					
1279	006610*	035114	000040						
1280	006614*	000002			CYLNO:	.BLKB	2		JDRB001
1281	006616*	020040	020040	020054	MES16:	.ASCIZ	' , TRK1 '		JDRB001
1282	006624*	051124	035113	000040					
1283	006632*	000004			TRKNO:	.BLKB	4		JDRB001
1284	006636*	020040	020054	042523	MES17:	.ASCIZ	' , SEC1 '		JDRB001
1285	006644*	035103	000040						
1286	006650*	000004			SECNO:	.BLKB	4		JDRB001
1287	006654*	020040	000045		MES18:	.ASCIZ	' %'		JDRB001
1288	006660*	052440	045516	047516	MES19:	.ASCIZ	' UNKNOWN'		JDRB001
1289	006666*	047127	000						
1290		006672*			.EVEN				JDRB001
1291									JDRB001
1292	006672*	006545*			TRERR:	MES12			JDRB001
1293	006674*	006364*			MES1				JDRB001
1294	006676*	177777			177777				JDRB001
1295									JDRB001
1296	006700*	006414*			MCPERR:	MES4			JDRB001
1297	006702*	006374*			MES2				JDRB001
1298	006704*	006364*			MES1				JDRB001
1299	006706*	177777			177777				JDRB001
1300									JDRB001
1301	006710*	006414*			MDPERR:	MES4			JDRB001

RES2	000060R	221#																			
REXET	005720R	487	1165#	1170																	
RHBA	001774R	442#	541*	551*	561*	590	596*	1195*													
RHBAE	002040R	460#	597*	1236*																	
RHCS1	001770R	440#	567*	574*	577*	578	583*	602*	918	926	1085	1089	1105	1125							
RHCS2	002000R	1143	1160	1161	1165	1173	1191*														
RHCS3	002042R	444#	566*	573*	587*	838*	1124*	1171*	1199*												
RHDB	002012R	461#	1032	1238*																	
RHWC	001772R	449#	1047	1050*	1059*	1064	1067*	1209*													
RH70	000200R	441#	540*	550*	560*	1193*															
RPAS	002006R	367#	1030	1233																	
RPAS	002006R	447#	581	582*	607	609*	1184	1185*	1205*												
RPCC	002026R	455#	1221*																		
RPDA	001776R	443#	542*	552*	562*	1197*															
RPDC	002024R	454#	543*	553*	563*	1219*															
RPDS	002002R	445#	1128	1201*																	
RPDT	002016R	451#	839	846	848	856	863	865	1213*												
RPEC1	002034R	458#	1227*																		
RPEC2	002036R	459#	1229*																		
RPER1	002004R	446#	1203*																		
RPER2	002000R	456#	1233*																		
RPER3	002032R	457#	1225*																		
RPLA	002010R	448#	1207*																		
RPMB	002014R	450#	1211*																		
RPOF	002022R	453#	584*	1217*																	
RPSN	002000R	452#	1215*																		
RETRT	000112R	237#																			
S	000552R	369#	418	419	420	421	422	423	424	425	426	427	428	429							
		430	431	432	433	434	435	436	437	929*	934	945	947	949							
		952	959	960	961	992	997	1002	1007	1012	1017	1019	1021	1024							
		1026	1028	1037	1039	1048	1065	1106*													
SBADR	000102R	230#	1154*																		
SECNO	006650R	983	1206#																		
SECTOR	000646R	390#	674*	686*	698	701	704*	748	750*	758	784										
SETUP	000636R	486	1190#																		
SIZSEC	000666R	402#	762	815*																	
SIZTRK	000664R	401#	760	809*	812*	824															
SOFCNT	000042R	213#																			
SOFER#	104406	260#	1074																		
SOPPAS	000046R	215#																			
SPOINT	000032R	209#	1110																		
SPSIZ	000040	1#	253																		
SR1	000016R	202#	672	717	731	740	937	956	1030	1233											
SR2	000020R	203#																			
SR3	000022R	204#																			
SR4	000024R	205#																			
START	002044R	208	464#																		
STAT	000026R	207#																			
SVR0	000062R	222#																			
SVR1	000064R	223#																			
SVR2	000066R	224#																			
SVR3	000070R	225#																			
SVR4	000072R	226#																			
SVR5	000074R	227#																			
SVR6	000076P	228#																			
SYSCNT	000052R	217#																			

TABLE	001716R	417#	1057	1074	1112																
TIMER	000704R	410#																			
TOUT	006770R	1331#																			
TRACK	000647R	391#	696*	700*	707	710	713*	745	747*	759	780										
TRERR	006672R	1009	1014	1292#																	
TRKNO	006632R	977	1283#																		
TRPDFM	000022	260#																			
TRY	000714R	414#	499*	622*	623																
UNITNO	000676R	406#	479*	492*	566	573	587	605	630*	639*	640	644	838	902							
		911	965	1124																	
UNKNWN	006754R	1045	1323#																		
VECTOR	000010R	198#	1241																		
WASADR	000104R	232#	1156*																		
WBUFEA	000136R	247#	545	555																	
WBUFFA	000134R	246#	541	551																	
WBUFRQ	000140R	248#																			
WBUFSZ	000142R	249#	538	540	806																
WDFR	000116R	239#	465*																		
WDTO	000114R	238#	464*																		
WRITCK	002406R	515	546*	547																	
WRITE	002312R	514	536*	537																	
WRTLIM	004016R	498	796#																		
XFLAG	000005R	196#																			
XMEM	000630R	380#	545*	555*	565*	591*	592*	593*	594*	600*	601										
ZERO	000706P	411#	1050	1067																	
	007034R	415#	518	1280#	1283*	1286*	1290#	1348#	1350#												

. ABS. 000000 000
007034 001

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

XRPDC0,XRPDC0/SOL/CRF;SYN=DDXCOM,XRPDC0
RUN-TIME: 7 12 1 SECONDS
RUN-TIME RATIO: 40/20=1.9
CORE USED: 7K (13 PAGES)