

Micro Fiche Scan

Name of device(s) tested:

RA60/80/81,UDA50,KDA50

Test description:

UDA50/KDA50 DEC/X11 MOD

MAINDEC Number or Package Identifier (after SEP 1977):

CXDUBE0

Fiche Document Part Number:

AH-S915E-MC

Fiche preparation date unknown, using copyright year:

1985

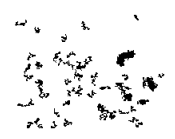
Image resolution:

1-bit black&white, compressed for minimal file size

COPYRIGHT (C) 1981-85 by d|i|g|i|t|a|l



101



101



IDENTIFICATION

PRODUCT CODE: AC S914C MC
 PRODUCT NAME: CXDUBE0 - UDA50A/KDA50Q DEC/X11 MOD
 PRODUCT DATE: 1-OCT-1985
 MAINTAINER: ROGER OAKY
 AUTHOR: MATT TEDONE

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 DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1985 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DEC	DIBOL	RSX
DEC/CMS	EduSystem	UNIBUS
DECnet	IAS	VAX
DECsystem-10	MASSBUS	VMS
DECSYSTEM-20	PDP	VT
DECUS	PDT	Digital Logo
DECwriter	RSTS	

.ENABL LC

.REM 6

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
3.0	START UP
4.0	PASS DEFINITION
5.0	EXECUTION TIME
6.0	CONFIGURATION REQUIREMENTS
7.0	DEVICE/OPTION SETUP
8.0	MODULE OPERATION
9.0	OPERATION OPTIONS
10.0	PRINTOUTS
11.0	DUAL PORT OPERATION
12.0	GLOSSARY
13.0	BIBLIOGRAPHY

1.0 ABSTRACT

The exerciser will be similar to that of other disk subsystem exercisers. Writes will be performed to the disks followed by read and compare of the data read. The controller will do all error retrying. Errors will be reported on the console terminal.

All desired disk drives on the controller will be exercised simultaneously. If disk accessing is not required, then data written will go only as far as the controller's RAM memory.

If the results of the exerciser requires more information, other PDP-11 diagnostic programs are available. They are:

CZUDHA1 - UDA50-A/KDA50-Q Basic Subsystem Diagnostic
CZUDIA0 - UDA50-A/KDA50-Q Disk Drive Exerciser
CZUDJA0 - UDA50-A/KDA50-Q Disk Subsystem Exerciser
CZUDKA0 - UDA50-A/KDA50-Q Disk Formatter.

2.0 REQUIREMENTS

Hardware for all cases:
One DEC/X11 module configures for one UDA50-A or KDA50-Q controller.

Hardware for disk accessing:
One controller with at least one drive is the minimum amount or one controller with four drives is the maximum amount.

Hardware for no disk accessing:
One controller is the only requirement.

Memory: DUBE requires
Decimal words -- 4096 MAX

3.0 START-UP

On the initial start, the program will clear bit1 of 'SR1' and type the following messages.

```
DUBE0 PA:0060162      APC: 000674      PASS #00000
```

```
'IF YOU WISH TO DESTROY CUSTOMER DATA, SET BIT1 (NOT BIT0)  
IN SWITCH REGISTER 1(SR1) OF DUBE? EQUAL TO 1.'
```

```
DUBE0 PA:0060210      APC: 000722      PASS #00000  
'! OPERATING WITH NO DISK ACCESSING !'
```

This will occur regardless of the condition of SR1 (bit1) at configure time.

If the operator wishes to exercise the drive, SR1 (bit1) must be modified at location 16 of CXDUBE0 module (see section 9). This can be accomplished by using the 'MOD' command supplied by the DECX11 run time system. Unless the program is reloaded or the operator modifies the location again, the contents of SR1 will remain the same on all subsequent starts.

On all subsequent starts, the condition of SR1 (bit1) will type to terminal in the following manner.

If bit1 of SR1 is equal to 0 (zero), the following warning will be typed.

```
DUBE0 PA:0060210      APC: 000722      PASS #00000  
'! OPERATING WITH NO DISK ACCESSING !'
```

If bit1 of SR1 is equal to 1 (one), the following warning will be typed.

DUBEO PA:0060210 APC: 000722 PASS #00000
'! CUSTOMER DATA WILL BE DESTROYED !'

<<<< NOTE >>>>

When this DEC/X11 module runs in diskless mode, its data rate exceeds all other devices. This may cause erroneous data lates from other devices.

4.0 PASS DEFINITION

One pass of the DUBE module consists of 512 iterations of the basic test sequence (write, read, data-check). The test sequence writes a user defined number of words (default is 256) words, reads 256 words, and data-compare same.

5.0 EXECUTION TIME

The default execution time of one pass of DUBE running alone on a PDP-11/44 under sequential disk accessing mode will be approximately 20 seconds. Under random accessing mode, the time is 40 seconds. For no disk accessing, the time is five seconds

6.0 CONFIGURATION REQUIREMENTS

Default Parameters:

DEVADR: 172150, VECTOR: 154, BR1: 4, DEVCNT: 1, SR1:
0, SR2: 0

REQUIRED PARAMETERS:

Additional controller module(s) configured must have different bus address(es) and vector(s).

7.0 DEVICE/OPTION SETUP

For disk mode, make certain that all units are powered up, write enabled, connected to a controller via the SDI and ready;

For diskless mode, make certain the controller is powered up

8.0 MODULE OPERATION

TEST SEQUENCE DISK MODE:

- A. Setup device register addresses and module variables.
Set controller characteristics.
- B. Reset all units on-line and drop all that are not.
- C. Get a unit address.
- D. Get a disk address and a fresh block of data.
- E. Do a write -- if errors, report.
- F. Do a read -- if errors, report.
- G. Do a data-check -- if errors, report and continue.
- H. Make unit available.
- I. Wait for available attention message.
- J. If end of pass, report and go to D.
- K. If end of testing unit, go to C; else go to D.

Blocks determined defective won't be replaced by the exerciser.

TEST SEQUENCE DISKLESS MODE:

- A. Get a fresh block of data.
- B. Do a write to controller RAM buffer -- if errors, report.
- C. Do a read from controller RAM buffer -- if errors, report.
- D. Do a data-check -- if errors, report and continue.
- E. If end of pass, report.
- F. Go to A

9.0 OPERATION OPTIONS

One or more software switch registers can be used by the module program general purpose switches. These words are used to define or specify a unique device option or to point to a specific routine in the module. Any option must be specified by the operator before the module is run. Switch Register 1 has the following characteristics.

- SR1 Bit 1 set (1): Allow disk transfers.
 <<<< NOTE >>>> IF SET, CUSTOMER DATA WILL BE DESTROYED!
 reset (0): No disk transfers.
- SR1 Bit 2 set (1): Do not report errors as they occur.
 reset (0): Report errors as they occur.
- SR1 Bit 3 set (1): Do not print error summary at end of pass.
 reset (0): Print error summary at end of pass.
- SR1 Bit 9 set (1): Run Dual port mode (only valid if SR1

reset (0): Bit 1 is set)
 Do not run Dual port mode

SR1 Bit 10 set (1). Select random block addressing.(only
 valid if SR1 Bit 1 is set)
 reset (0): Select sequential block
 addressing.

SR1 Bit 11 set (1): Bypass data compare.
 reset (0): Do data compare

Switch register 2 has the following characteristics.

SR2 Bits 0 to 5: Burst rate.

A burst rate to speed up NPR transfers by the controller can be used. This value is 6 bits maximum and set up in SR2 at configure time.

<<<< NOTE >>>>

The DVID1 mask reflects the number of units chosen for testing and which units on the system are to be tested. Example: If DVID1 contains a 1, only the first unit found on the system will be tested. A unit's order on the system is judged by its unit number. The lowest unit number zero (0). Unit 0 would be the first tested on the system.

If DVID1 contains a 10, the fourth unit on the system will be tested. If the first two units are chosen, DVID1 is 3. Four consecutive units means DVID1 is 17. Six units, DVID1 is 77.

If there is not a unit corresponding to the DVID1 bit setting, the bit set in DVID1 gets cleared. The exerciser will readjust the mask and drop the nonexistent units if more units are chosen than actually are present. The module is dropped if all DVID1 bits are cleared.

If the number of units chosen is less than the actual number of units present, only the desired units will be used during the exercise.

<<<< ANOTHER NOTE >>>>

Make sure all subunited drives are accounted for. Destroying customer data is not desirable.

<<<< ONE MORE NOTE >>>>

If SR1 Bit 3 is reset, a summary status is printed every 15 passes. This status is formatted as follows:

DUBEO PA: 00060470 ACP: 001210 PASS #00000

SOFT ERROR COUNT #00000 *** HARD ERROR COUNT #00000
 CHECK DATA ERROR COUNT #00000

10.0 PRINTOUTS

A. Most printouts have the standard formats described in the DEC/X11 document.

B. Non-standard printouts include error messages which dump the following:

- 1) Summary status
- 2) Flags and endcode
- 3) Unit number
- 4) Byte count
- 5) Hi 16-bit LBN value
- 6) Lo 16-bit LBN value
- 7) Extended address
- 8) Physical address

All values except for PASS, RUNTIME and ERRCNT are printed in octal. PASS, RUNTIME and ERRCNT are printed in decimal.
 Example:

DUBEO PA: 00064116 APC: 004630 PASS: 00000 ERRCNT: 00001
 CSRA 172150 CSRC: 000000 ASTAT: 000006 ERRTP: 000006
 RUNTIME: 000:00:22

DUBEO PA: 00064052 APC: 004564 PASS: 00000

STATUS ENCOD UNITNU BYTECO HI LBN LO LBN EXTADR PHYADR
 000006 000242 000005 000000 000003 116321 000001 062100

STATUS - response of the command sent to the controller.
 This is contained in the last five bits of the word. Here is a list of status codes.

- 0 - success
- 1 - invalid command
- 2 - command aborted
- 3 - unit offline
- 4 - unit available
- 5 - media error
- 6 - write protected
- 7 - compare error
- 10 - data error
- 11 - host buffer access error
- 12 - controller error
- 13 - drive error

ENOCOD - ending code of the command sent. This shows what command was sent to the JDA. Here is a list of all possible endcodes this module uses.

- 100 - AVAILABLE ATTENTION MESSAGE (not a command but a message sent to the host from the UDA)
- 200 - INVALID COMMAND
- 203 - GET UNIT STATUS
- 204 - SET CONTROLLER CHARACTERISTICS
- 210 - AVAILABLE
- 211 - ONLINE
- 230 - MAINTENANCE READ
- 231 - MAINTENANCE WRITE
- 241 - READ
- 242 - WRITE

UNITNU - unit number of the drive that is being accessed. This is not relevant if the user is running diskless mode.

BYTECO - size of the buffer in bytes.

HI LBN - high logical block number (upper 16 bits) which tells the user where on the disk the data is going. This is only valid for disk mode.

LO LBN - low logical block number (lower 16 bits).

EXTADR - extended address of the read/write buffer.

PHYADR - physical address of the read/write buffer.

2. If the controller failed to pass its internal diagnostic, one of the following messages will be printed.

If the diagnostic found a fault:

```
DUBEO PA: 00062052 APC: 002564 PASS: 00000
CONTROLLER INIT ERROR, FOUND BY DIAGNOSTIC
SA REGISTER = xxxxxx IN STEP yyyyy
ADDR = zzzzzz
```

If a step bit was not set as expected during the initialization sequence of the controller:

```
DUBEO PA: 00062152 APC: 002664 PASS: 00000
CONTROLLER INIT ERROR, STEP NOT SET
SA REGISTER = xxxxxx IN STEP yyyyy
ADDR = zzzzzz
```

If data passed back from the controller was not equal to the expected value:

```
DUBEO PA: 00062252 APC: 002764 PASS: 00000
```

CONTROLLER INIT ERROR, EXPECTED DATA WAS INCORRECT
 SA REGISTER = xxxxxx IN STEP yyyy
 ADDR = zzzzzz

Where xxxxxx can have any of the following values and meanings:

104000 - Fatal sequencer error
 104040 - D processor ALU error
 104041 - D proc ROM parity error/ Timeout test error
 105102 - D PROC no board 2 error/D PROC control reg test error/
 D PROC RAM parity error
 105105 - D proc RAM buffer error
 105152 - D proc SDI error
 105153 - D proc write mode wrap SERDES 16 error
 105154 - D PROC read mode, SERDES 16, 10 RSGEN and
 ECC circuitry error
 106040 - U proc ALU error/DFAIL test error/
 Unexpected trap error
 106041 - U proc Control Register error
 106042 - U PROC parity error set erroneously/
 CROM parity test error
 106047 - U proc Constant ROM error with D proc running SDI test
 106055 - Unexpected trap found, aborted diagnostic
 106071 - U PROC Log/Antilog RAM checksum error
 106072 - U PROC ROM parity test error
 106200 - Step 1 data error (MSB not set)
 107103 - U proc k.M parity error
 107107 - U proc RAM buffer error
 107115 - Board #2 test count was wrong
 112300 - Step 2 error
 122240 - NPR error
 122300 - Step 3 error
 142300 - Step 4 error

Where yyyy is the step in which the error was found.

Where zzzzzz is the address of the UDA.

If the maximum number of retries has been exceeded, the following message will be printed.

DUBEO PA: 00061414 APC:002126 PASS #00000

RETRY COUNT EXCEEDED, ABORT

This means the controller did not successfully complete the initialization in four passes. The module is then dropped.

D. If the controller did not successfully clear the ring buffer in the host area, the following message will be printed.

DUBEO PA: 00061414 APC:002126 PASS #00000

RING AREA NOT CLEARED

This is a fatal error. It means that the controller did not access host memory that the controller would use to communicate with the host. The module is then dropped.

E. If the SA register displays a non-zero value after the initialization sequence is done, the following message will be printed.

DUBEO PA: 00064252 APC: 004764 PASS: 00000
 SA REGISTER IS NOT ZERO, = xxxxxx
 CONTROLLER IS GOING THROUGH INITIALIZATION

Where xxxxxx can have the following values and meanings.

- 004400 - controller has been init'd by either a bus init or by writing into the IP register.
- 100001 - bus envelope/packet read error (parity or timeout)
- 100002 - bus envelope/packet write error (parity or timeout)
- 100003 - controller ROM and RAM parity error
- 100004 - controller RAM parity error
- 100005 - controller ROM parity error
- 100006 - Ring read error (parity or timeout)
- 100007 - Ring write error (parity or timeout)
- 100010 - bus interrupt master failure
- 100011 - Host access timeout error
- 100012 - Host exceeded credit limit
- 100013 - Q-bus master error
- 100014 - Diagnostic controller fatal error
- 100015 - Hardware timeout of instruction loop
- 100016 - Invalid virtual circuit identifier
- 100017 - Interrupt write error on bus
- 100020 - Maintenance read/write invalid region identifier
- 100021 - Maintenance write load to non-loadable controller
- 100022 - Controller RAM error (non-parity)
- 100023 - INIT sequence error
- 100024 - High-level protocol incompatibility error
- 100025 - Purge/poll hardware failure
- 100026 - Mapping register read error (parity or timeout)
- 100027 - Mapping option unsupported

E. If a drive is dropped by the exerciser, one of the following messages will be printed.

If the drive had an error it could not handle properly after an iteration, the following message will be printed:

DUBEO PA: 00063012 APC: 003524 PASS #00000

DRIVE 00000 DROPPED.
 DEVICE ID BIT = 000001
 ERRORS CAUSED DRIVE TO BE DROPPED

If the drive was not found by the exerciser, the

following message will be printed:

DUBEO PA: 00063012 APC: 003524 PASS #00000

DRIVE 00000 DROPPED.
 DEVICE ID BIT = 000001
 UNIT WAS NOT FOUND BY THE EXERCISER

If there were more device count bits set than the actual number of drives found, the following message will be printed:

DUBEO PA: 00063012 APC: 003524 PASS #00000

DRIVE 00000 DROPPED.
 DEVICE ID BIT = 000001
 DVID1 BIT SET HIGHER THAN ACTUAL # OF DRIVES FOUND

Solution: try a lesser number of units in DVID1 (loc 14)

11.0 DUAL PORT OPERATION

To run a dual port operation, set bit9 of SR1. The exerciser will check the unit to see if it is offline or available.

The controller will retain control of a unit until the MSCP Available command is entered by the host. During this time, the other controller is not allowed access to the unit through the other port between the write and read. The other controller senses when the unit becomes available and takes it. The MSCP Available command is only executed if SR1 bit 9 and SR1 bit 1 are set. This allows dual porting and disk accessing respectively.

DEC/X11 will only dual port a drive with another DEC/X11 exerciser.

12.0 GLOSSARY

DUBE follows the module name format described in the DEC/X11 Programmer's Guide.

- DU-- Identifies the hardware and thus the module.
- B- Distinguishes between two or more different modules for the same generic device. The sequence A, B, C, ETC. must be used for each additional example.
- E Specifies the module revision.

IOMODX is a type of module in an extended input/output mode. These modules are interrupt driven and are capable of input/output operation. Some added capabilities provided include:

- Ⓢ Use of monitor supplied write buffers.
- Ⓢ Ability to change the size of the write buffers.
- Ⓢ Access to the monitor's check data utility.
- Ⓢ Conversion routines to get 18 and 22 bit addresses from 16 bit addresses.

13.0 BIBLIOGRAPHY

CXQUADO 'DEC/X11 USER'S MANUAL' Sept 1984
CXQUBGO 'DEC/X11 CROSS-REFERENCE MANUAL' Sept 1982
CXQUCAO 'DEC/X11 REFERENCE CARD' January 1979
CXQAFDO 'DEC/X11 PROGRAMMERS'S GUIDE' Sept 1978

&

000120 000000
 000122 000104
 000124 007616
 000126 000000
 000130 000000
 000132 000400
 000134 000000
 000136 000000
 000140 000400
 000142 000000
 000144 000000
 000146 000000
 000150 000000
 000040

INTR: OPEN ;# OF INTERRUPTS PER ITERATION
 IDNUM: 104 ;MODULE IDENTIFICATION NUMBER=104
 RBUFVA: RBUF ;READ BUFFER VIRTUAL ADDRESS
 RBUFPA: OPEN ;READ BUFFER PHYSICAL ADDRESS
 RBUFEA: OPEN ;READ BUFFER EA BITS
 RBUFSZ: 256. ;SIZE OF THE READ BUFFER
 WBUFPA: OPEN ;WRITE BUFFER PHYSICAL ADDRESS
 WBUFEA: OPEN ;WRITE BUFFER EA BITS
 WBUFRQ: 256. ;WRITE BUFFER SIZE REQUESTED
 WBUFSZ: OPEN ;WRITE BUFFER SIZE AVAILABLE
 CDERCT: OPEN ;CDATA/DATCK ERROR COUNT
 CDWDCT: OPEN ;CDATA/DATCK WORD COUNT
 FREE: OPEN ;RESERVED FOR FUTURE USE
 .REPT SPSIZ ;MODULE STACK STARTS HERE.
 .NLIST
 .WORD 0
 .LIST
 .ENDR

000252

MODSP:
 ;*****
 ;*****
 .SBTTL MODULE STORAGE AREA
 ; FOR RELEASE
 ; VERSION 1.0 DON'T TEST STEP 4 COMPLETION.
 ; VERSION 1.1 DON'T WAIT FOR INTERRUPT AFTER SENDING MSCP AVAILABLE
 ; COMMAND.
 ; VERSION 2.0 USE BIT 9 IN SR1 FOR DUAL PORTING. (DON'T SEND MSCP
 ; AVAILABLE COMMAND IF WE WANT JUST SEQUENTIAL OR RANDOM
 ; ACCESS MODE -- IN OTHER WORDS, ONLY SEND ONLINE
 ; COMMAND ONCE DURING PASS UNLESS DUAL PORT MODE).
 ; VERSION 3.0 KDA50-Q SUPPORT ADDED.
 ; VERSION 4.0 JFM - 27-SEP-85
 ; 22 BIT Q-BUS ADDRESSING SUPPORT ADDED.
 ; COMMENTS CLEANED UP AND UNUSED CODE DELETED.
 ; DOCUMENTATION HAS BEEN UPDATED SOMEWHAT.

4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38

000002
 000004
 000010
 001000
 002000
 004000
 000000
 000000
 000000
 000260
 000264
 000000

SR.XFR = BIT01 ;NO DISK TRANSFER 0 = NO DISK TRANSFER, 1 = DO DISK TRANSFER
 SR.REP = BIT02 ;REPORT ERROR AS THEY OCCUR 0 = REPORT, 1 = DON'T REPORT
 SR.SUM = BIT03 ;REPORT ERRORS ON END OF PASS 0 = REPORT, 1 = DON'T REPORT
 SR.DUA = BIT09 ;DUAL PORT 0 = NO DUAL PORT, 1 = DUAL PORT
 SR.SEQ = BIT10 ;DISK ACCESS MODE 0 = SEQUENTIAL, 1 = RANDOM
 SR.CMP = BIT11 ;NO DATA COMPARE 0 = DO DATA COMPARE, 1 = DON'T DO DATA COMPARE
 SAREG: .WORD 0 ; CONTROLLER STATUS REG
 ;**
 ; THE ORDER OF THE NEXT 5 VARIABLES MUST NOT CHANGE
 ;
 CINTR: .WORD 0 ;COMMAND INTERRUPT INDICATOR
 RINTR: .WORD 0 ;RESPONCE INTERRUPT INDICATOR
 RSPONC: .BLKW 2. ;MESSAGE RING
 COMMND: .BLKW 2. ;COMMAND RING
 CMDREF: .WORD 0 ;COMMAND REFERENCE NUMBER
 ;
 ;--

39					
40	000272	000000	RSPPA: .WORD	0	; RESPONSE RING
41	000274	000000	RSPEA: .WORD	0	; PHYSICAL
42	000276	000000	RSPPP: .WORD	0	; ADDRESS
43	000300	000000	RSPEP: .WORD	0	; STORAGE
44					
45	000302	000000	RSPLN: .WORD	0	; RESPONCE PACKET LENGTH
46	000304	000000	RSPVIR: .WORD	0	; RESPONCE PACKET VIRTUAL CIRCUIT
47	000306		RSPACK: .BLKW	24.	; RESPONSE PACKET
48	000366	000000	RPAKPA: .WORD	0	; RESPOINSE PACKET
49	000370	000000	RPAKEA: .WORD	0	; PHYSICAL
50	000372	000000	RPAKPP: .WORD	0	; ADDRESS
51	000374	000000	RPAKEP: .WORD	0	; STORAGE
52					
53	000376	000000	CMPLN: .WORD	0	; COMMAND PACKET LENGTH
54	000400	000000	CMPVIR: .WORD	0	; COMMAND PACKET VIRTUAL CIRCUIT
55	000402		CMPACK: .BLKW	24.	; COMMAND PACKET
56	000462	000000	CPAKPA: .WORD	0	; COMMAND PACKET
57	000464	000000	CPAKEA: .WORD	0	; PHYSICAL
58	000466	000000	CPAKPP: .WORD	0	; ADDRESS
59	000470	000000	CPAKEP: .WORD	0	; STORAGE
60					
61	000472	000000	VA: .WORD	0	; GENERIC VIRTUAL ADDRESS FOR GETPA
62	000474	000000	PA: .WORD	0	; GENERIC PHYSICAL ADDRESS
63	000476	000000	EA: .WORD	0	; GENERIC EXTENDED ADDRESS
64	000500	000000	PA22: .WORD	0	; 22-BIT PHYSICAL ADDRESS
65	000502	000000	EA22: .WORD	0	; EE-BIT EXTENDED ADDRESS
66					
67	000504	000000	RBUFPP: .WORD	0	; READ BUFFER PHYSICAL ADDRESS SAVE AREA
68	000506	000000	RBUFEP: .WORD	0	; READ BUFFER EXTENDED ADDRESS SAVE AREA
69	000510	000000	WBUFPP: .WORD	0	; WRITE BUFFER PHYSICAL ADDRESS SAVE AREA
70	000512	000000	WBUFEP: .WORD	0	; WRITE BUFFER EXTENDED ADDRESS SAVE AREA
71					
72	000514	000000	NUM: .WORD	0	; ADDRESS USED IN OTOA
73	000516	000000	OLGPA: .WORD	0	; THE OLD PHYSICAL ADDRESS
74	000520	000000	OLDEA: .WORD	0	; THE OLD EXTENDED ADDRESS TO CHECK IF
75					; CONTROLLER WILL BE REINITED
76					
77		000017	PRTNUM = 15.		; PRINT MESSAGE EVERY 15TH TIME
78	000522	000017	PRNMSG: .WORD	PRTNUM	; PRINT WORD SAVES THE VALUE TO CHECK FOR THE
79					; NEXT TIME AN END OF PASS MESSAGE IS WRITTEN
80		002260	TIMER = 1200.		; TIME TO WAIT 2-3 SECONDS AFTER DAP COMMAND
81					
82	000524	177777	EXPAV: .WORD	177777	; EXPECTING AN AVAILABLE ATTENTION MESSAGE = 0
83					; ELSE = 177777
84					
85	000526		ADR1: .BLKB	6	
86	000534	000	.BYTE	0	
87	000535		ADR2: .BLKB	6	
88	000543	000	.BYTE	0	
89	000544		ADR3: .BLKB	6	
90	000552	000	.BYTE	0	
91	000553		ADR4: .BLKB	6	
92	000561	000	.BYTE	0	
93	000562		ADR5: .BLKB	6	

94 000570 000
95 000571
96 000577 000
97 000600
98 000606 000
99 000607
100 000615 000
101

ADR6: .BYTE 0
.BLKB 6
.BYTE 0
ADR7: .BLKB 6
.BYTE 0
ADR8: .BLKB 6
.BYTE 0
.EVEN

```

103      .SBTTL MORE MODULE STORAGE
104
105 000616 000000      SECL:  .WORD  0      ;CURRENT SECTOR LO ORDER ADDRESS
106 000620 000000      SECH:  .WORD  0      ;CURRENT SECTOR HI ORDER ADDRESS
107
108 000622 000000      UNSZL: .WORD  0      ;UNIT SIZE LO ORDER LIMIT FROM ONLINE CMND
109 000624 000000      UNSZH: .WORD  0      ;UNIT SIZE HI ORDER LIMIT
110
111 000626 003300      LIMIT: .WORD  3300    ;4K 1200 = MOST WORDS MAITW CAN TAKE
112
113 000630 000001      DVICE. .WORD  1      ;DEVICE TO TEST
114 000632 000000      UNITNO: .WORD  0      ;UNIT NUMBER
115 000634 000000      TRY:  .WORD  0      ;NUMBER OF TRIES
116 000636 000001      PORTID: .WORD  1      ;BIT POSITION SELECTS THE PORT
117 000640 000000      UNITFL: .WORD  0      ;SAVE UNIT FLAGS
118 000642 000000      WORK:  .WORD  0      ;TEMPORARY WORK AREA
119
120      005670      TIMEOUT = 3000.    ;TIME OUT GUADGE
121      C00004      RLIM = 4      ;RETRY LIMIT
122
123 000644 000000 000001  TABLEW: .WORD  0,1      ;TABLE ENTRY UNITNO,PORTID
124 000650 177777 177777      .WORD  -1,-1      ;CURRENT LAST TABLE ENTRY
125 000654      .BLKW  12.      ;REST OF TABLE
126 000704 177777 177777      TEND:  .WORD  -1,-1      ;END MARKER
127

```

```
129          .SBTTL  MODULE PRIVATE DATA
130
131          000001  BIT00 = 1
132          000002  BIT01 = 2
133          000004  BIT02 = 4
134          000010  BIT03 = 10
135          000020  BIT04 = 20
136          000040  BIT05 = 40
137          000100  BIT06 = 100
138          000200  BIT07 = 200
139          000400  BIT08 = 400
140          001000  BIT09 = 1000
141          002000  BIT10 = 2000
142          004000  BIT11 = 4000
143          010000  BIT12 = 10000
144          020000  BIT13 = 20000
145          040000  BIT14 = 40000
146          100000  BIT15 = 100000
147
148          ;
149          ;      ERROR BITS
150          ;
151          000000  ERR.0 = 0      ;NOT DEFINED
152          000001  ERR.1 = 1      ;DATA ERROR
153          000003  ERR.3 = 3      ;CONTROLLER NOT READY
154          000006  ERR.6 = 6      ;DRIVE NOT READY, OFF LINE OR NON EXISTENT
155          000032  ERR.32 = 32    ;NPR ERROR
156
```

```

158          .SBTTL CONTROLLER BIT DEFINITIONS
159          ; SA REGISTER UNIVERSAL READ BITS
160
161          SA.S1= 004000          ;STEP 1 STATUS BIT
162          004000          SA.S2= 010000          ;STEP 2 STATUS BIT
163          010000          SA.S3= 020000          ;STEP 3 STATUS BIT
164          020000          SA.S4= 040000          ;STEP 4 STATUS BIT
165          040000          SA.ERR= 100000         ;ERROR INDICATOR
166          100000
167
168          ; SA REGISTER ERROR STATUS BITS
169
170          SA.ERC= 003777        ;ERROR CODE
171
172          ; SA REGISTER STEP ONE READ BITS
173
174          SA.NSI= 002000        ; NON SETTABLE INTERRUPT
175          001000          SA.Q22= 001000        ; 22 BIT ADDRESS BUS
176          004000          SA.DIA= 000400        ; DIAG BIT IN SA REGISTER
177          000100          SA.MAP= 000100        ; MAPPING BIT
178          000040          SA.SM = 000040        ; SPECIAL MODE BIT FOR KDA50-Q
179
180          ; SA REGISTER STEP ONE WRITE BITS
181
182          SA.VEC= 000177        ; INTERRUPT VECTOR (DIVIDED BY 4)
183          000200          SA.INT= 000200        ; INTERRUPT ENABLE DURING INITIALIZATION
184          003400          SA.RSP= 003400        ; MESSAGE RING LENGTH
185          034000          SA.CMD= 034000        ; COMMAND RING LENGTH
186
187          ; SA REGISTER STEP TWO READ BITS
188
189          SA.VCE= 000177        ; INTERRUPT VECTOR ECHO
190          000200          SA.INE= 000200        ; INTERRUPT ENABLE ECHO
191
192          ; SA REGISTER STEP TWO WRITE BITS
193
194          SA.PRQ= 000001        ; LOW ORDER MESSAGE RING BYTE ADDRESS
195          000001          ; ENABLE UNIBUS ADAPTER PURGE INTERRUPT
196
197          ; SA REGISTER STEP THREE READ BITS
198
199          SA.RSE= 000017        ; RESPONSE RING LENGTH ECHO
200          000360          SA.CME= 000360        ; COMMAND RING LENGTH ECHO
201
202          ; SA REGISTER STEP THREE WRITE BITS
203
204          SA.LFC= 040000        ; HIGH ORDER MESSAGE RING BYTE ADDRESS
205          040000          ; LAST FAILURE CODE REQUEST
206
207          ; SA REGISTER STEP FOUR READ BITS
208
209          SA.MCV= 000377        ; CONTROLLED MICROCODE VERSION
210
211          ; SA REGISTER STEP FOUR WRITE BITS
212

```

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 6-1
CONTROLLER BIT DEFINITIONS

213

000001

SA.GO= BIT0

!GO BIT TO START CONTROLLER FIRMWARE

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 7
 COMMAND/MESSAGE DESCRIPTOR BIT DEFINITIONS

```

215          .SBTTL COMMAND/MESSAGE DESCRIPTOR BIT DEFINITIONS
216          100000          RG.OWN= BIT15          ;SET WHEN CONTROLLER OWNS RING
217          040000          RG.FLG= BIT14          ;FLAG BIT
218
219          ;OFFSETS INTO HOST COMMUNICATIONS AREA WITH ONE DESCRIPTOR TO EACH RING
220
221          000010          HC.SIZ= 8.             ;SIZE OF HOST COMM AREA IN BYTES
222          000060          PKTSIZ= 48.           ;SIZE OF PACKETS IN BYTES
223
224          000000          HC.RES= 0.             ;RESPONCE RING START
225          000002          HC.RCT= 2.             ;RESPONCE RING CONTROL WORD
226          000004          HC.CMD= 4.             ;COMMAND RING START
227          000006          HC.CCT= 6.             ;CONTROL RING CONTROL WORD
228          000306          HC.RPK= RSPACK         ;START OF RESPONCE PACKET BUFFER
229          000366          HC.CPK= HC.RPK+PKTSIZ ;START OF COMMAND PACKET BUFFER
230

```

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 8
 COMMAND PACKET OPCODES

		.SBTTL COMMAND PACKET OPCODES	
232			
233			
234	000001	OP.ABO= 01	; ABORT COMMAND
235	000020	OP.ACC= 20	; ACCESS COMMAND
236	000010	OP.AVL= 10	; AVAILABLE COMMAND
237	000021	OP.CCD= 21	; COMPARE CONTROLLER DATA COMMAND
238	000040	OP.CMP= 40	; COMPARE HOST DATA COMMAND
239	000013	OP.DAP= 13	; DETERMINE ACCESS PATHS COMMAND
240	000022	OP.ERS= 22	; ERASE COMMAND
241	000023	OP.FLU= 23	; FLUSH COMMAND
242	000002	OP.GCS= 02	; GET COMMAND STATUS COMMAND
243	000003	OP.GUS= 03	; GET UNIT STATUS COMMAND
244	000011	OP.ONL= 11	; ONLINE COMMAND
245	000041	OP.RD= 41	; READ COMMAND
246	000024	OP.RPL= 24	; REPLACE COMMAND
247	000004	OP.SCC= 04	; SET CONTROLLER CHARACTERISTICS COMMAND
248	000012	OP.SUC= 12	; SET UNIT CHARACTERISTICS COMMAND
249	000042	OP.WR= 42	; WRITE COMMAND
250	000030	OP.MRD= 30	; MAINTENANCE READ COMMAND
251	000031	OP.MWR= 31	; MAINTENANCE WRITE COMMAND
252	000200	OP.END= 200	; END PACKET FLAG
253	000100	OP.AVA= 100	; AVAILABLE ATTENTION MESSAGE
254	000101	OP.ERL= 101	; ERROR LOG ATTENTION MESSAGE
255	000102	OP.SHC= 102	; SHADOW COPY COMPLETE ATTENTION MESSAGE
256	000102	OP.ACP= 102	; ACCESS PATH ATTENTION MESSAGE
257			
258			
259			
260			

;NOTE: END PACKET OPCODES (ALSO CALLED ENDCODES) ARE FORMED BY ADDING THE END
 ;PACKET FLAG TO THE COMMAND OPCODE. THE UNKNOWN COMMAND END PACKET CONTAINS
 ;JUST THE END PACKET FLAG IN ITS OPCODE FIELD.

.SBTTL COMMAND MODIFIERS			
262			
263			
264	040000	MD.CMP= 040000	;COMPARE
265	100000	MD.EXP= 100000	;EXPRESS REQUEST
266	010000	MD.ERR= 010000	;FORCE ERROR
267	004000	MD.SCH= 004000	;SUPPRESS CACHING (HIGH SPEED)
268	002000	MD.SCL= 002000	;SUPPRESS CACHING (LOW SPEED)
269	001000	MD.SEC= 001000	;SUPPRESS ERROR CORRECTION
270	000400	MD.SER= 000400	;SUPPRESS ERROR RECOVERY
271	000200	MD.SSH= 000200	;SUPPRESS SHADOWING
272	000100	MD.WBN= 000100	;WRITE-BACK (NON-VOLATILE)
273	000040	MD.WBV= 000040	;WRITE BACK (VOLATILE)
274	000001	MD.SPD= 000001	;SPIN-DOWN
275	000001	MD.FEU= 000001	;FLUSH ENTIRE UNIT
276	000002	MD.VOL= 000002	;VOLATILE ONLY
277	000001	MD.NYU= 000001	;NEXT UNIT
278			
279			
.SBTTL END PACKET FLAGS			
280			
281	000200	EF.BBR= 000200	;BAD BLOCK REPORTED
282	000100	EF.BBU= 000100	;BAD BLOCK UNREPORTED
283	000040	EF.LOG= 000040	;ERROR LOG GENERATED
284	000020	EF.SEX= 000020	;SERIOUS EXCEPTION
285			
.SBTTL UNIT FLAGS			
286			
287			
288	000001	UF.CMR= 000001	;COMPARE READS
289	000002	UF.CMW= 000002	;COMPARE WRITES
290	010000	UF.RPL= 010000	;HOST INITIATED BAD BLOCK REPLACEMENT
291	040000	UF.INA= 040000	;INACTIVE SHADOW SET UNIT
292	000200	UF.RMV= 000200	;REMOVEABLE MEDIA
293	004000	UF.SCH= 004000	;SUPPRESS CACHING (HIGH SPEED)
294	002000	UF.SCL= 002000	;SUPPRESS CACHING (LOW SPEED)
295	000040	UF.WBN= 000040	;WRITE-BACK (NON-VOLATILE)
296	020000	UF.WPH= 020000	;WRITE PROTECT(HARDWARE)
297	010000	UF.WPS= 010000	;WRITE PROTECT(SOFTWARE OR VOLUME)
298	000004	UF.576= 000004	;576 BYTE SECTORS

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 10
CONTROLLER FLAGS

```

300      .SBTTL CONTROLLER FLAGS
301      000200      CF.AVL= 000200      ;ENABLE AVAILABLE ATTENTION MESSAGES
302      000100      CF.MSC= 000100      ;ENABLE MISCELLANEOUS ERROR LOG MESSAGES
303      000040      CF.OTH= 000040      ;ENABLE OTHER HOST'S ERROR LOG MESSAGES
304      000020      CF.THS= 000020      ;ENABLE THIS HOST'S ERROR LOG MESSAGES
305      000002      CF.SMD= 000002      ;SHADOWING
306      000001      CF.576= 000001      ;576 BYTE SECTORS
307
308      .SBTTL COMMAND PACKET OFFSETS
309
310      GENERIC COMMAND PACKET OFFSETS:
311      ;
312      000000      P.CRF= 0.      ;COMMAND REFERENCE NUMBER
313      000004      P.UNIT= 4.      ;UNIT NUMBER
314      000010      P.OPCD= 8.      ;OPCODE
315      000012      P.MOD= 10.      ;MODIFIERS
316      000014      P.BCNT= 12.      ;BYTE COUNT
317      000020      P.BUFF= 16.      ;BUFFER DESCRIPTOR
318      000020      P.ADPA= 16.      ;BUFFER'S PHYSICAL ADDRESS (P.BUFF)
319      000022      P.ADEA= 18.      ;BUFFER'S EXTENDED ADDRESS (P.BUFF+2)
320      000034      P.LBN= 28.      ;LOGICAL BLOCK NUMBER
321      000040      P.SFTW= 32.      ;SOFTWARE WORDS
322
323      ;
324      000014      P.OTRF= 12.      ABORT AND GET COMMAND STATUS COMMAND PACKET OFFSETS:
325      ;OUTSTANDING REFERENCE NUMBER
326
327      ;
328      000016      P.UNFL= 14.      ONLINE AND SET UNIT CHARACTERISTICS COMMAND PACKET OFFSETS:
329      000020      P.HSTI= 16.      ;UNIT FLAGS
330      000024      P.UNTI= 20.      ;HOST IDENTIFIER
331      000034      P.ELGF= 28.      ;UNIT IDENTIFIER
332      000040      P.SHUN= 32.      ;ERROR LOG FLAGS
333      000042      P.CPSP= 34.      ;SHADOW UNIT
334      ;
335      000014      P.RBN= 12.      REPLACE COMMAND PACKET OFFSETS:
336      ;REPLACEMENT BLOCK NUMBER
337
338      ;
339      000014      P.VRSN= 12.      SET CONTROLLER CHARACTERISTICS COMMAND PACKET OFFSETS:
340      000016      P.CNTF= 14.      ;MSCP VERSION
341      000020      P.HTMO= 16.      ;CONTROLLER FLAGS
342      000022      P.USEF= 18.      ;HOST TIMEOUT
343      000024      P.TIME= 20.      ;USE FRACTION
344      ;
345      000034      P.RGID= 28.      MAINTENANCE READ AND MAINTENANCE WRITE COMMAND PACKET OFFSETS:
346      000040      P.RGOF= 32.      ;REGION ID
;REGION OFFSET

```

		.SBTTL END PACKET OFFSETS	
348			
349			
350			GENERIC END PACKET OFFSETS:
351	000000	P.CRF= 0.	; COMMAND REFERENCE NUMBER
352	000004	P.UNIT= 4.	; UNIT NUMBER
353	000010	P.OPCD= 8.	; OP CODE (ALSO CALLED ENDCODE)
354	000011	P.FLGS= 9.	; END PACKET FLAGS
355	000012	P.STS= 10.	; MODIFIERS
356	000014	P.BCNT= 12.	; BYTE COUNT
357	000034	P.FBBK= 28.	; FIRST BAD BLOCK
358	000040	P.SFTW= 32.	; SOFTWARE WORDS
359			
360			GET COMMAND STATUS END PACKET OFFSETS:
361	000014	P.OTRF= 12.	; OUTSTANDING REFERENCE NUMBER
362	000020	P.CMST= 16.	; COMMAND STATUS
363			
364			GET UNIT STATUS END PACKET OFFSETS:
365	000014	P.MLUN= 12.	; MULTI-UNIT CODE
366	000016	P.UNFL= 14.	; UNIT FLAGS
367	000020	P.HSTI= 16.	; HOST IDENTIFIER
368	000024	P.UNTI= 20.	; UNIT IDENTIFIER
369	000040	P.SHUN= 32.	; SHADOW UNIT
370	000042	P.SHST= 34.	; SHADOW STATUS
371	000044	P.TRCK= 36.	; TRACK SIZE
372	000046	P.GRP= 38.	; GROUP SIZE
373	000050	P.CYL= 40.	; CYLINDER SIZE
374	000054	P.RCTS= 44.	; RCT TABLE SIZE
375	000056	P.RBNS= 46.	; RBN / TRACK
376	000057	P.RCTC= 47.	; RCT COPIES
377			
378			ONLINE AND SET UNIT CHARACTERISTICS
379	000014	P.MLUN= 12.	; MULTI-UNIT CODE
380	000016	P.UNFL= 14.	; UNIT FLAGS
381	000020	P.HSTI= 16.	; HOST IDENTIFIER
382	000024	P.UNTI= 20.	; UNIT IDENTIFIER
383	000040	P.SHUN= 32.	; SHADOW UNIT
384	000044	P.UNSZ= 36.	; UNIT SIZE
385	000050	P.VSER= 40.	; VOLUME SERIAL NUMBER
386			
387			SET CONTROLLER CHARACTERISTICS END PACKET OFFSETS:
388	000014	P.VRSN= 12.	; MSCP VERSION
389	000016	P.CNTF= 14.	; CONTROLLER FLAGS
390	000020	P.CTMO= 16.	; CONTROLLER TIMEOUT
391	000022	P.CNCL= 18.	; CONTROLLER COMMAND LIMIT
392	000024	P.CNTI= 20.	; CONTROLLER ID
393	000034	P.MEDI= 28.	; MEDIA TYPE
394	000042	P.SHST= 34.	; SHADOW STATUS
395			
396			ERROR LOG ATTENTION MESSAGE PACKET OFFSETS
397			
398	000000	P.CRF= 0.	; COMMAND REFERENCE NUMBER
399	000004	P.UNIT= 4.	; UNIT NUMBER
400	000006	P.CNT= 6.	; COUNT
401	000010	P.OPCD= 8.	; OP CODE
402	000011	P.FLGS= 9.	; ERROR LOG FLAGS

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 11-1
END PACKET OFFSETS

403 000012
404 000014

P.SZOF= 10.
P.LGDT= 12.

;SIZE OR OFFSET
;START OF ERROR LOG DATA

```

406          .SBTTL ERROR LOG FLAGS
407
408          000200      EF.FRS= 000200      ;FIRST PACKET
409          000100      EF.LST= 000100      ;LAST PACKET
410          000001      EF.MIS= 000001      ;MESSAGE MISSING
411
412          ;ERROR LOG MESSAGE OFFSETS
413
414          000000      L.EVNT= 0.          ;EVENT CODE
415          000002      L.SLOT= 2.          ;SLOT NUMBER
416          000004      L.CNTI= 4.          ;CONTROLLER IDENTIFIER
417          000014      L.CNTI= 12.         ;CONTROLLER SOFTWARE REVISION
418          000015      L.CHVR= 13.         ;CONTROLLER HARDWARE REVISION
419          000016      L.UNTI= 14.         ;UNIT IDENTIFIER
420          000026      L.USVR= 22.         ;UNIT SOFTWARE REVISION
421          000027      L.UHVR= 23.         ;UNIT HARDWARE REVISION
422          000030      L.ERLC= 24.         ;ERROR LOCATION
423          000034      L.CYL= 28.          ;CYLINDER
424          000040      L.GRP= 32.          ;GROUP
425          000041      L.TRCK= 33.         ;TRACK
426          000042      L.SCTR= 34.         ;SECTOR
427          000044      L.VSER= 36.         ;VOLUME SERIAL NUMBER
428          000050      L.DATA= 40.         ;EVENT DEPENDENT DATA
429
430          ;STATUS AND EVENT COE DEFINITIONS
431
432          000037      ST.MSK= 37           ;STATUS / EVENT CODE MASK
433          000040      ST.SUB= 40           ;SUB-CODE MULTIPLIER
434          000000      ST.SUC= 0            ;SUCCESS
435          000001      ST.CMD= 1            ;INVALID COMMAND
436          000002      ST.ABO= 2            ;COMMAND ABORTED
437          000003      ST.OFL= 3            ;UNIT-OFFLINE
438          000004      ST.AVL= 4            ;UNIT-AVAILABLE
439          000005      ST.MFE= 5            ;MEDIA ERROR
440          000006      ST.WPR= 6            ;WRITE PROTECTED
441          000007      ST.CMP= 7            ;COMPARE ERROR
442          000010      ST.DAT= 10           ;DATA ERROR
443          000011      ST.HST= 11           ;HOST BUFFER ACCESS ERROR
444          000012      ST.CNT= 12           ;CONTROLLER ERROR
445          000013      ST.DRV= 13           ;DRIVE ERROR
446          000037      ST.DIA= 37           ;MESSAGE FROM AN INTERNAL DIAGNOSTIC
447
448          ;
449          ;          SUBCODES FOR ST.OFL
450          ;
451          000040      SC.NVL = 40           ;NO VOLUME MOUNTED
452          ;          ; OR DRIVE DISAVLED VIA RUN/STOP SWITCH
453          000100      SC.IOP = 100         ;UNIT INOPERATIVE
454          000400      SC.DIS = 400         ;UNIT DISABLED BY FIELD SERVICE
455          ;          ; OR INTERNAL DIAGNOSTIC
456          000200      SC.DUP = 200         ;DUPLICATE UNIT NUMBER
457          ;
458          ;          SUBCODES FOR ST.DRV
459          ;
460          000040      SC.STO = 40           ;SDI RESPONCE TIME OUT
  
```

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 12-1
ERROR LOG FLAGS

461

000100

SC.INV = 100

:INVALID SDI RESPONCE

463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517

```

.SBTTL  MODULE CODE
*****
:
:   INIT VALUES
:   INIT CONTROLLER
:   XFER TO DISK?
:       F FOR J = 1,CYCLE LIMIT
:         MAINTENANCE WRITE
:         MAINTENANCE READ
:         CHECK DATA?
:           T CHECK
:         NEXT J
:       T FOR J = 1,CYCLE LIMIT
:         GET UNIT STATUS
:         IF DRIVE IS NOT AVAILABLE, WAIT UNTIL IT IS
:         DRIVE THERE?
:         F DROP
:           ALL DRIVES DROPPED?
:             T DROP MODULE
:             F ---
:           T ONLINE
:           ONLINE?
:             T PICK BLOCK - IF RANDOM, GET RAND * MOD X
:             ELSE INCREMENT
:             IF LBN > LIMIT THEN LBN <- 0
:
:         WRITE
:         READ
:         CHECK DATA ?
:           T CHECK
:           AVAILABLE DRIVE(I)
:           F TRY TO BRING ONLINE AGAIN
:
:       NEXT J
:
*****

```

```

*****
:
:   START CODE
:
:   IF THE CODE IS RESTARTED, CLEAR THE OLD ADDRESSES SO THE
:   THE CONTROLLER WILL GET REINITED.
:
*****

```

```

START:
      INC      #-1          ;FIRST TIME THRU HERE?
      BNE     1$          ;BR IF NO
      BIC     @SR.XFR,SR1 ;DO NOT ALLOW DISK TRANSFERS
      MSGN$,BEGIN,WARN1  ;ASCII MESSAGE CALL WITH COMMON HEADER
      BIT     @SR.XFR,SR1 ;WILL CUSTOMER DATA BE OVERWRITTEN?
      BEQ     2$          ;BR IF NO
      MSGN$,BEGIN,WARN2  ;ASCII MESSAGE CALL WITH COMMON HEADER
      BR      3$          ;
      1$:
      2$:

```

508	000710			
509	000710	005227	177777	
510	000714	001006		
511	000716	042767	000002	177072
512	000724	104403	000000'	006042'
513	000732	032767	000002	177056
514	000740	001404		
515	000742	104403	000000'	006046'
516	000750	000403		
517	000752			

518	000752	104403	000000'	006052'				
519	000760				3#:			
520	000760	005067	177160			CLR	CDERCT	; CLEAR DATA CHECK ERROR COUNT
521	000764	012767	177777	177532		MOV	#177777,EXPV	; NOT EXPECTING AN INTERRUPT
522	000772	012767	000017	177522		MOV	#PRNUM,PRNMSG	; INITIALIZE PRINT WORD
523	001000	016767	177010	177622		MOV	DVID1,DVICE	; DVICE HAS DESIRED BITS SET
524	001006	005067	177632			CLR	TABLEW	; SET TABLE FOR UNIT 0
525	001012	012767	000001	177626		MOV	#1,TABLEW+2	; SET TABLE FOR PORTID FOR UNIT 0
526	001020	005067	177244			CLR	CMDREF	; COMMAND REF # = 0
527	001024	104417	000000'			RAND#	.BEGIN	
528	001030	016767	177020	177560		MOV	RANUM,SECL	; FOR RESTARTING (INITIAL SECTOR ADDR)
529	001036	005067	177556			CLR	SECL	; STORE IN SA REG
530	001042	016767	176740	177202		MOV	ADP,SAREG	; SA REGISTER HAS PROPER ADDRESS
531	001050	062767	000002	177174		ADD	SAREG	; OLD PHYSICAL ADDRESS CLEARED
532	001056	005067	177434			CLR	OLOPA	; OLD EXTENDED ADDRESS CLEARED
533	001062	005067	177432			CLR	OLDEA	; FOR RESTARTING. THIS WILL FORCE A
534								; CONTROLLER REINIT TO TAKE PLACE
535								


```

537
538
539
540
541
542
543
544
545
546
547
548
549 001066
550 001066 004767 000740
551 001072 026767 177200 177416
552 001100 001004
553 001102 026767 177172 177410
554 001110 001412
555 001112 C16767 177160 177376
556 001120 016767 177154 177372
557 001126 004767 000260
558 001132 005067 177476
559 001136
560 001136 032767 000010 176652
561 001144 001034
562 001146 026767 177350 176660
563 001154 001030
564 001156 062767 000017 177336
565
    001164 104421 000000' 000042'
    001172 000535'
566 001174 105067 177342
567
    001200 104421 000000' 000044'
    001206 000544'
568 001210 105067 177335
569
    001214 104421 000000' 000144'
    001222 000526'
570 001224 105067 177303
571 001230 104403 000000' 006004'
572 001236 012777 005024' 176544 1#:
573
574 001244 005067 177362
575 001250 032767 000002 176540
576 001256 001446
    
```

```

*****
:
: RESTART SEQUENCE
:
: CHECK THE ADDRESS OF THE RINGS TO SEE IF THEY WERE RELOCATED
: IF THEY WERE, REINIT THE CONTROLLER.
:
: GET THE NEW ADDRESSES. IF THE DISKLESS OPERATION IS DESIRED
: THEN DO THE MAINTENENCE WRITE AND READ. ELSE DO THE WRITE
: AND READ WITH A DRIVE.
:
*****
RESTR1: JSR PC,CVTADR ;
CMP RSPPP,OLDPA ;IS THE OLD PHYS ADDR = NEW ONE?
BNE RESTR2 ;IF SO, REINIT
CMP RSPEP,OLDEA ;IS THE OLD EXTN ADDR = NEW ONE?
BEQ RESTR1 ;IF NOT, DON'T REINIT
RESTR2: MOV RSPPP,OLDPA ;ELSE SET THE OLD RING ADDR
MOV RSPEP,OLDEA ;AND THE OLD EXTENDED ADDR
JSR PC,INITUD ;AND INIT THE CONTROLLER
CLR TRY ;CLEAR RETRY COUNT

RESTR1: BIT @SR.SUM,SR1 ;DO WE WANT THE REPORT?
BNE 1# ;IF NOT, SKIP THE REPORT
CMP PRMSG,PASCNT ;DO WE PRINT?
BNE 1# ;IF PASS COUNT IS NOT = PRINT WORD, SKIP
ADD @PRNUM,PRMSG ;PRINT WORD IS INCREMENT
*****
;CONVERT SOFCNT TO ASCII AND
;STORE AT ADR2
BTOD$,BEGIN,SOFCNT,ADR2
*****
CLRB ADR2+5
*****
;CONVERT HRDCNT TO ASCII AND
;STORE AT ADR3
BTOD$,BEGIN,HRDCNT,ADR3
*****
CLRB ADR3+5
*****
;CONVERT CDERCT TO ASCII AND
;STORE AT ADR1
BTOD$,BEGIN,CDERCT,ADR1
*****
CLRB ADR1+5
MSGN$,BEGIN,ERRPAS ;ASCII MESSAGE CALL WITH COMMON HEADER
MOV @NTRUPT,@VECTOR ;GET VECTOR ADDRESS
;SET POINTER
;PRESET UNIT @
CLR UNITNO ;DISK XFER???
BIT @SR.XFR,SR1 ;NO! DO MAINTENENCE (DISKLESS) ROUTINES
BEQ MA10NC
    
```

```

577 ;*****
578 ; DO THE DISK OPERATIONS ;
579 ; CHECK TO SEE WHICH PORTS ARE AVAILABLE ;
580 ;*****
581 JSR PC.SETUP ;FIND DRIVES/SET UP TABLE
582 001260 004767 001524 TST DVICE ;ELSE, TEST FOR ANY MORE DRIVES
583 001264 005767 177340 BNE LOOP1 ;IF TRUE, DO A CYCLE
584 001270 001002
585
586 001272 104410 000000' END$.BEGIN
587
588 001276 004767 000420 LOOP1: JSR PC.GETWB ; ALLOCATE WRITE BUFFER
589 001302 012704 000644' MOV #TABLEW,R4 ;R4 -> TABLE OF UNITNO AND PORTID
590 001306 012703 000001 MOV #1,R3 ;R3 IS AN INDEX TO DVICE
591 001312
592 001312 030367 177312 LOOP2: BIT R3,DVICE ;HAS THE DRIVE BEEN DROPPED
593 001316 001412 BEQ 9$ ;IF SO, SKIP THIS DRIVE
594 001320 016467 000002 177310 MOV 2(R4),PORTID ;SET UP PORTID
595 001326 C11467 177300 MOV (R4),UNITNO ;SET UP UNITNO
596 ; *** DO A DISK CYCLE
597 001332 004767 001756 JSR PC.CYCLED ;DO A CYCLE FOR DISK OPERATION
598 001336 103002 BCC 9$ ;IF SUCCESSFUL, CONTINUE
599 001340 004767 002534 JSR PC.DROP1 ;IF NOT, DROP DRIVE
600 001344
601 001344 062704 000004 9$: ADD #4,R4 ;POINT TO NEXT ENTRY OF THE TABLE
602 001350 006303 ASL R3 ;R3 POINTS TO NEXT BIT
603 001352 022704 000704' CMP #TEND,R4 ; POINT BEYOND LAST ENTRY?
604 001356 001403 BEQ 12$ ; IF NOT, THEN TRY AGAIN.
605 001360 020367 177244 CMP R3,DVICE ;IF R3 > DVICE THEN DONE WITH ITERATION
606 001364 003752 BLE LOOP2 ;IF < OR =, LOOP
607 001366
608 001366 104413 000000' 12$: ENDIT$.BEGIN ;SIGNAL END OF ITERATION.
;MONITOR SHALL TEST END OF PASS
;AND DO AGAIN
609 001372 000741 BR LOOP1
610
611 ;*****
612 ; MAINTENANCE ROUTINE, DO THE DISKLESS CODE ;
613 ;*****
614
615 MA10NC: JSR PC.GETWB ; GET WRITE BUFFER
616 001374 004767 000322 JSR PC.CYCLED
617 001400 004767 002206' ENDIT$.BEGIN ;SIGNAL END OF ITERATION.
618 001404 104413 000000' ;MONITOR SHALL TEST END OF PASS
619 001410 000771 BR MA10NC

```

```

621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642 001412 005004
643 001414 012702 000001
644 001420 005077 176362
645 001424 012701 002260
646 001430 017700 176616
647 001434 032700 100000
648 001440 001007
649 001442 104407 000000'
    001446 104407 000000'
650 001452 005301
651 001454 001365
652 001456 000404
653 001460 012703 004000
654 001464 000167 001150
655 001470 042700 173377
656 001474 022700 004400
657 001500 001402
658 001502 000167 001126
659
660 001506 016705 176276
661 001512 006205
662 001514 006205
663 001516 052705 100200
664
665 001522 010500
666 001524 012703 004000
667 001530 004767 001002
668 001534 042705 100000
669 001540 042700 000200
670 001544 001404
671 001546 052700 010200
672 001552 000167 001056
673 001556 016700 176514
674 001562 004767 000750
  
```

```

*****
INITIALIZE THE CONTROLLER
DO THE 4 STEPS FOR INITIALIZING THE CONTROLLER.
STEP 1 - CHECK FOR ERROR, STEP 1
          SEND VECTOR/4, INTERRUPT ENABLE, RING LEN'S = 0
STEP 2 - CHECK VECTOR ECHO, INTERRUPT ECHO,
          ERROR, STEP 2
          SEND PHYSICAL ADDRESS & PURGE = 0
STEP 3 - CHECK RING LEN = 0, ERROR, STEP 3
          SEND EXTENDED ADDRESS BITS
STEP 4 - CHECK STEP 4
          SEND LFAIL = 0 . GO AND BURST
*****

INITUD: CLR      R4                ;R4 IS USED IF AN ERROR IS DETECTED
        MOV      #1,R2           ;R2 = STEP INDICATOR REG FOR MSG'S
        CLR      @ADDR          ;WRITE TO IP REGISTER TO INIT CONTROLLER
        MOV      @TIMER,R1      ;SET TIME OUT LIMIT
1$:     MOV      @SAREG,R0       ;R0 HAS SA REGISTER DATA
        BIT      @<SA.ERR>,R0   ;CHECK FOR ERROR
        BNE     2$              ;IF FOUND, GET OUT OF LOOP
        BREAK$,BEGIN           ;TEMPORARY RETURN TO MONITOR...
        BREAK$,BEGIN           ;THEN CONTINUE AT NEXT INSTRUCTION.
        DEC     R1              ;TIME OUT?
        BNE     1$              ;IF NOT, LOOP
        BR      4$              ;IF DONE, CONTINUE
2$:     MOV      @SA.S1,R3       ;R3 = STEP 1 BIT
        JMP     ERROR1          ;IF HERE, ERROR
4$:     BIC      @<C<SA.S1>SA.DIA>,R0 ; CLEAR KDA50-Q DEPENDENT BITS
        CMP     @<SA.S1>SA.DIA>,R0 ; DID DATA COMPARE PROPERLY?
        BEQ     5$              ;IF SO, CONTINUE
        JMP     ERROR3          ;REPORT ERROR
        ;
        ; STEP 2
5$:     MOV      VECTOR,R5       ;VECTOR GIVEN
        ASR     R5              ;SET TO APPROPRIATE VALUE
        ASR     R5              ; = VECTOR/4
        BIS     @<SA.INT+BIT15>,R5 ;ACTIVATE INTERRUPTS & SET MSB FOR STEP 1
        ;
        ; LEN'S ARE 0
        MOV     R5,R0           ;STORE R5 IN R0 FOR SUBROUTINE
        MOV     @SA.S1,R3       ;R3 HAS STEP BIT FOR SUBROUTINE
        JSR    PC,SNDSTP        ;SEND STEP DATA
        BIC     @BIT15,R5       ;CLEAR MSB FOR COMPARE DATA
        BIC     @BIT07,R0       ;WAS BIT07 ONLY BIT SET?. SHOULD BE
        BEQ     6$              ;
6$:     BIS     @<SA.S2+BIT07>,R0 ;SET R0 TO REPORT THE ERROR
        JMP     ERROR3          ;REPORT ERROR
        MOV     RSPPP,R0        ;R0 GETS PHYSICAL ADDRESS
        JSR    PC,SNDSTP        ;SEND STEP DATA
  
```

```

675 001566 042705 177400      BIC      #177400,R5      ;HIGH BYTE CLEARED
676 001572 020500              CMP      R5,R0          ;CHECK ECHO DATA
677 001574 001402              BEQ      7#             ;IF OK, SKIP
678 001576 000167 001032      JMP      ERROR3         ;IF NOT, REPORT ERROR
679 001602              ;
680              ; STEP 3
681 001602 016700 176472      MOV      RSPEP,R0       ; SEND THE EXTENDED ADDRESS BITS
682 001606 004767 000724      JSR      PC,SNDSTP     ; SEND STEP DATA
683 001612 012700 000254      MOV      #RSPONC-4,R0  ; RO -> RING ENVELOP
684              ; STEP 4
685 001616 005720              TST      (R0)+          ; IS THE RING ENTRY = 0?
686 001620 001402              BEQ      9#             ;
687 001622 000167 000774      JMP      ERRORS        ; IF NOT, ERROR
688 001626 022700 000270      CMP      #CMDREF,R0   ; IS RO POINT PAST THE RINGS?
689 001632 001371              BNE      8#             ; IF NOT, LOOP
690 001634 016700 176160      MOV      SR2,R0        ; RO = BURST VALUE
691 001640 000241              CLC                      ; CLEAR CARRY
692 001642 006300              ASL      R0             ; ALIGN BURST FOR STEP 4
693 001644 006300              ASL      R0             ;
694 001646 052700 000001      BIS      #SA.GO,R0     ; SET GO BET
695 001652 000077 176374      MOV      R0,#SAREG     ; SEND DATA TO CONTROLLER/INIT DONE
696 001656 016767 176604 176400  MOV      CPAKPP,COMMAND ; STORE ADDRESS IN THE RING
697 001664 016767 176600 176374  MOV      CPAKEP,COMMAND+2 ; MOVE ADJUSTED EA INTO RING
698              ;
699 001672 016767 176474 176360  MOV      RPAKPP,RSPONC  ; STORE ADDRESS IN THE RING
700 001700 016767 176470 176354  MOV      RPAKEP,RSPONC+2 ; MOVE ADJUSTED EA INTO RING
701 001706 012777 005024 176074  MOV      #NTRUPT,#VECTOR ; STORE INTERRUPT ADDRESS IN VECTOR
702 001714 005067 176714              CLR                      ; CLEAR TRY SO DRIVE WILL
703              ; GO BACK ONLINE IF NECESSARY
704 001720 000207              RTS      PC
705              ; **
706              ; GETWB - GET WRITE BUFFER
707              ; --
708 001722              GETWB:
709 001722 104414 000000'          GMBUF#, BEGIN          ;GET WRITE BUFFER INFORMATION
710 001726 032767 000010 176124  BIT      #QMON22,RES2  ; IF NOT USING Q-22 MONITOR,
711 001734 001404              BEQ      11#           ; USE 18 BIT ADDRESSING
712 001736 032767 001000 176112  BIT      #ADDR22,CONFIG ; IF 22-BIT QBUS ADDRESSING,
713 001744 001012              BNE      12#           ; CALCULATE PHYSICAL ADDRESS
714 001746 016767 176162 176534 11# : MOV      WBUFPA,WBUFPP ; CONVERT FROM 18 BIT
715 001754 016700 176156              MOV      WBUFEA,R0     ; PSEUDO ADDRESS
716 001760 004767 000540              JSR      PC,ASR04      ; TO 18 BIT
717 001770 000417              MOV      R0,WBUFEP     ; PHYSICAL ADDRESS
718 001772 016767 176136 176474 12# : BR      20#           ;
719 002000 016767 176132 176470              MOV      WBUFPA,PA     ; SET UP FOR
720 002006 104416 000000' 000474'  MOV      WBUFEA,EA     ; MAP22 CALL
721 002014 016767 176460 176466              MAP22#, BEGIN,PA     ; GET 22-BIT ADDR FROM 18-BIT ADDR
722 002022 016767 176454 176462              MOV      PA22,WBUFPP  ; PHYSICAL ADDRESS
723 002030 000207              MOV      EA22,WBUFEP  ;
724              ; RETURN FROM SUBROUTINE
725              ; **
726              ; CVTADR - CONVERT 16 BIT ADDRESS TO 18 OR 22 BIT ADDRESS
727              ; --
728 002032 012767 000260' 176432  CVTADR: MOV      #RSPONC,VA ; CONVERT RESPONSE RING ADDRESS

```

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 15 2
MODULE CODE

729	002040	104415	000000'	000472'		GETPA#,BEGIN, VA	;GET PHYSICAL ADDRESS FROM 16 BIT VA
730	002046	016767	176424	176220		MOV EA,RSPEA	; SAVE EA BITS
731	002054	016767	176414	176210		MOV PA,RSPPA	; SAVE PA BITS
732	002062	032767	000010	175770		BIT #QMON22,RES2	; IF NOT USING Q-22 MONITOR,
733	002070	001404				BEQ 11#	USE 18 BIT ADDRESSING
734	002072	032767	001000	175756		BIT #ADDR22.CONFIG	; IF 22-BIT QBUS ADDRESSING,
735	002100	001012				BNE 12#	CALCULATE PHYSICAL ADDRESS
736	002102	016767	176366	176166	11#:	MOV PA,RSPPP	; CONVERT FROM 18 BIT
737	002110	016700	176362			MOV EA,RO	PSEUDO ADDRESS
738	002114	004767	000404			JSR PC,ASR04	; TO 18 BIT
739	002120	010067	176154			MOV RO,RSPEP	PHYSICAL ADDRESS
740	002124	000411				BR 20#	
741	002126				12#:	MAP22#, BEGIN,PA	; GET 22-BIT ADDR FROM 18-BIT ADDR
	002126	104416	000000'	000474'		MOV PA22,RSPPP	PHYSICAL ADDRESS
742	002134	016767	176340	176134		MOV EA22,RSPEP	
743	002142	016767	176340	176130			
744							
745	002150	016767	175750	176314	20#:	MOV RBUFVA,VA	; CONVERT READ BUFFER ADDRESS
746	002156	104415	000000'	000472'		GETPA#,BEGIN, VA	;GET PHYSICAL ADDRESS FROM 16-BIT VA
747	002164	016767	176306	175736		MOV EA,RBUFEA	; SAVE EA BITS
748	002172	016767	176276	175726		MOV PA,RBUFPA	; SAVE PA BITS
749	002200	032767	000010	175652		BIT #QMON22,RES2	; IF NOT USING Q-22 MONITOR,
750	002206	001404				BEQ 21#	USE 18 BIT ADDRESSING
751	002210	032767	001000	175640		BIT #ADDR22.CONFIG	; IF 22-BIT QBUS ADDRESSING,
752	002216	001012				BNE 22#	CALCULATE PHYSICAL ADDRESS
753	002220	016767	176250	176256	21#:	MOV PA,RBUFPP	; CONVERT FROM 18 BIT
754	002226	016700	176244			MOV EA,RO	PSEUDO ADDRESS
755	002232	004767	000266			JSR PC,ASR04	; TO 18 BIT
756	002236	010067	176244			MOV RO,RBUFEP	PHYSICAL ADDRESS
757	002242	000411				BR 30#	
758	002244				22#:	MAP22#, BEGIN,PA	; GET 22-BIT ADDR FROM 18-BIT ADDR
	002244	104416	000000'	000474'		MOV PA22,RBUFPP	PHYSICAL ADDRESS
759	002252	016767	176222	176224		MOV EA22,RBUFEP	
760	002260	016767	176216	176220			
761							
762	002266	012767	000402'	176176	30#:	MOV #CMPACK,VA	; CONVERT COMMAND PACKET ADDRESS
763	002274	104415	000000'	000472'		GETPA#,BEGIN, VA	;GET PHYSICAL ADDRESS FROM 16 BIT VA
764	002302	016767	176170	176154		MOV EA,CPAKEA	; SAVE EA BITS
765	002310	016767	176160	176144		MOV PA,CPAKPA	; SAVE PA BITS
766	002316	032767	000010	175534		BIT #QMON22,RES2	; IF NOT USING Q-22 MONITOR,
767	002324	001404				BEQ 31#	USE 18 BIT ADDRESSING
768	002326	032767	001000	175522		BIT #ADDR22.CONFIG	; IF 22-BIT QBUS ADDRESSING,
769	002334	001012				BNE 32#	CALCULATE PHYSICAL ADDRESS
770	002336	016767	176132	176122	31#:	MOV PA,CPAKPP	; CONVERT FROM 18 BIT
771	002344	016700	176126			MOV EA,RO	PSEUDO ADDRESS
772	002350	004767	000150			JSR PC,ASR04	; TO 18 BIT
773	002354	010067	176110			MOV RO,CPAKEP	PHYSICAL ADDRESS
774	002360	000411				BR 40#	
775	002362				32#:	MAP22#, BEGIN,PA	; GET 22-BIT ADDR FROM 18-BIT ADDR
	002362	104416	000000'	000474'		MOV PA22,CPAKPP	PHYSICAL ADDRESS
776	002370	016767	176104	176070		MOV EA22,CPAKEP	
777	002376	016767	176100	176064			
778							
779	002404	012767	000306'	176060	40#:	MOV #RSPACK,VA	; CONVERT RESPONSE PACKET ADDRESS
780	002412	104415	000000'	000472'		GETPA#,BEGIN, VA	;GET PHYSICAL ADDRESS FROM 16 BIT VA

```

781 002420 016767 176052 175742      MOV     EA,RPAKEA      ; SAVE EA BITS
782 002426 016767 176042 175732      MOV     PA,RPAKPA     ; SAVE PA BITS
783 002434 032767 000010 175416      BIT     @QMON22,RES2  ; IF NOT USING Q-22 MONITOR,
784 002442 001404                BEQ     41$           ; USE 18 BIT ADDRESSING
785 002444 032767 001000 175404      BIT     @ADDR22,CONFIG ; IF 22-BIT QBUS ADDRESSING,
786 002452 001012                BNE     42$           ; CALCULATE PHYSICAL ADDRESS
787 002454 016767 176014 175710 41$: MOV     PA,RPAKPP     ; CONVERT FROM 18 BIT
788 002462 016700 176010                MOV     EA,R0         ; PSEUDO ADDRESS
789 002466 004767 000032                JSR     PC,ASR04      ; TO 18 BIT
790 002472 010067 175676                MOV     RO,RPAKEP    ; PHYSICAL ADDRESS
791 002476 000411                BR      50$          ;
792 002500                42$: MAP22$, BEGIN,PA      ; GET 22-BIT ADDR FROM 18-BIT ADDR
793 002506 104416 000000' 000474'    MOV     PA22,RPAKPP   ; PHYSICAL ADDRESS
794 002514 016767 175762 175652      MOV     EA22,RPAKEP  ;
795                                50$: RTS     PC          ; RETURN FROM SUBROUTINE
796 002522 000207
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811 002524
812 002524 006200
813 002526 006200
814 002530 006200
815 002532 006200
816 002534 000207
817
818
819
820
821
822
823
824
825
826
827
828
829 002536 016701 175246
830 002542 012721 002562'
831 002546 116711 175240
832 002552 010077 175474
833
834 002556 104400 000000'

```

```

;*****
;
; ASR04
; ARITHMETIC SHIFT RIGHT REG 0 FOUR TIMES
;
; EXTENDED ADDRESS BITS (16 & 17) ARE SET IN BIT POSITION 4 & 5
; RESPECTIVELY. SHIFT RIGHT FOUR TIMES TO REPOSITION THE VALUE
;
; INPUT RO = UNADJUSTED EXTENDED ADDRESS BITS
;
; OUTPUT RO = ADJUSTED EXTENDED ADDRESS BITS
;*****
ASR04:
    LSR     RO          ;SHIFT 10
    LSR     RO          ;SHIFT 4
    LSR     RO          ;SHIFT 2
    LSR     RO          ;SHIFT 1
    RTS     PC          ;RETURN
;*****
;
; SEND STEP DATA
;
; INPUT: RO HAS DATA TO BE SENT TO CONTROLLER FOR STEP
;        R3 HAS PREVIOUS STEP FLAG SET
;
; OUTPUT: RO HAS DATA SENT FROM CONTROLLER TO HOST FOR ECHO AND NEXT STEP
;         R3 HAS CURRENT STEP FLAG SET
;*****
SNDSTP: MOV     VECTOR,R1      ;SET UP INTERRUPT HANDLER ADDRESS
        MOV     @INTA,(R1)    ;SET PRIORITY LEVEL
        MOVB   BR1,(R1)      ;SEND STEP1 WRITE FORMATED DATA
        MOV     RO,@SAREG
        EXIT$,BEGIN          ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

```

```

835
836 002562          INTA:
837 002562 000004 000000' 002570'  ;-----;
                                ; PIRQ$,BEGIN,3$ ; QUEUE UP TO CONTINUE AT 3$ AND RTI
                                ;-----;
838 002570          3$:
839 002570 017700 175456      MOV     @SAREG,RO      ;GET STEP N FORMATTED DATA
840 002574 032700 100000      BIT     @SA.ERR,RO   ;TEST FOR ERROR
841 002600 001017              BNE     ERROR1       ;IF NOT OK, REPORT
842 002602 005202              INC     R2             ;SET STEP REGISTER
843 002604 006303              ASL     R3             ;R3 HAS STEP BIT PROPERLY SET
844 002606 030300              BIT     R3,RO        ;WAS STEP N SET?
845 002610 001002              BNE     4$           ;IF SO, CONTINUE
846 002612 000167 000020      JMP     ERROR2       ;IF NOT CORRECT STEP, ERROR
847 002616 040300              BIC     R3,RO        ;CLEAR THE STEP BIT, FOR COMPARE
848 002620 000207              RTS     PC             ;RETURN
  
```

850
 851
 852
 853
 854
 855
 856
 857
 858
 859
 860
 861
 862
 863
 864
 865
 866
 867
 868
 869
 870
 871
 872
 873
 874
 875
 876
 877
 878
 879
 880
 881
 882
 383
 884
 885
 886
 887
 888
 889
 890
 891
 892
 893

002622 104403 000000' 006062'
 002622 104410 000000'
 002634 005204
 002636 005204
 002640 005204
 002642 010267 175646
 002646 104420 000000' 000514'
 002654 000535'
 002656 017767 175370 175630
 002664 104420 000000' 000514'
 002672 000526'
 002674 005304
 002676 001003
 002700 104403 000000' 005672'
 002706 005304
 002710 001003
 002712 104403 000000' 005716'
 002720
 002720 005304

```

*****
ERROR 1
PRINT AN ERROR REPORTED BY THE CONTROLLER DIAGNOSTICS

ERROR2
PRINT THE VALUE OF THE SA REGISTER WHEN THE STEP BIT WAS NOT SET

ERROR3
PRINT A THE VALUE OF THE SA REGISTER WHEN THE ECHO WAS NOT SET
CORRECTLY

INPUT  R0 -> SA REGISTER
        R2 = STEP COUNT

OUTPUT THE RETRY COUNT IS INCREMENTED
        IF THE RETRY COUNT > RETRY LIMIT, END MODULE

ERRORS
RING WASN'T ALL ZERO -> ERROR
DROP UDBAO
*****
ERRORS:
MSGN$,BEGIN,ZERO          ;ASCII MESSAGE CALL WITH COMMON HEADER
END$,BEGIN                ;
;
ERROR3: INC      R4          ;R4 = 3 FOR ERROR3
ERROR2: INC      R4          ;R4 = 2 FOR ERROR2
ERROR1: INC      R4          ;R4 = 1 FOR ERROR1
MOV     R2,NUM            ;STORE STEP REG IN A NUMBER FOR CONVRT
;*****
;CONVERT NUM TO ASCII AND
;STORE AT ADR2
OTOA$,BEGIN,NUM,ADR2
;*****
MOV     BSAREG,NUM        ;STORE VALUE IN A NUMBER
;*****
;CONVERT NUM TO ASCII AND
;STORE AT ADR1
OTOA$,BEGIN,NUM,ADR1
;*****
DEC     R4                ;ERROR 1?
BNE     1$                ;IF NOT, CHECK IF IT IS THE NEXT ERROR
MSGN$,BEGIN,INITE1       ;ASCII MESSAGE CALL WITH COMMON HEADER
;
1$: DEC     R4                ;ERROR 2?
BNE     2$                ;IF NOT, CHECK IF IT IS THE NEXT ERROR
MSGN$,BEGIN,INITE2       ;ASCII MESSAGE CALL WITH COMMON HEADER
;
2$: DEC     R4                ;ERROR 3?

```


908
 909
 910
 911
 912
 913
 914
 915
 916
 917
 918
 919
 920
 921
 922
 923 003010
 924
 925 003010 004767 001650
 926 003014 C05367 175250
 927 003020 001110
 928 003022 012703 000001
 929 003026 012704 000644
 930 003032 011467 175574
 931 003036 016714 175570
 932 003042 010367 175570
 933 003046 010364 000002
 934 003052 012764 177777 000004
 935 003060 016464 000004 000006
 936 003066 012767 002400 175546
 937 003074 004767 001526
 938 003100 103006
 939 003102 005367 175534
 940 003106 001372
 941 003110 004767 000774
 942 003114 000437
 943 003116 016767 175170 175506
 944
 945
 946 003124 012702 000644
 947 003130 012705 000001
 948 003134 020227 000704
 949 003140 001420
 950 003142 020305
 951 003144 001416
 952 003146 026712 175460
 953 003152 001404
 954 003154 062702 000004
 955 003160 006305
 956 003162 000764
 957 003164 011467 175442
 958 003170 010367 175442
 959 003174 004767 000720
 960 003200 000405
 961 003202
 962

```

*****
:
: SFT UP
:
: GO FIND OUT WHAT DRIVES ARE OUT THERE.
: A TABLE IS FILLED WITH UNIT NUMBERS(MAX IS 16)
:
: THIS SHOULD ONLY BE DONE AT THE VERY BEGINNING OF RUNNING
: THIS DECX MODULE; THEN NOT RUN AGAIN.
:
: INPUT: DVICE HAS APPROPRIATE BITS SET. THE # OF BITS =
: # OF DRIVES WANTED TO TEST.
: POSITION OF BITS = WHICH DRIVE IN THE SYSTEM IS DESIRED.
*****
SETUP:
: *** SET CONTRL CHAR AND WAIT FOR THE ATTENTION MESSAGES
: JSR PC,SCC ;SET CONTROLLER CHARACTERISTICS
: DEC CMDREF ;ONLY SET UP AT BEGINNING OF MODULE
: BNE 19# ; (USE DRIVES FOUND AT BEGINNING)
: MOV #1,R3 ;INITIAL PORTID VALUE
: MOV #TABLEW,R4 ;R4 -> TABLEW
: MOV (R4),UNITNO ;INITIAL UNITNO IN TABLEW
: MOV UNITNO,(R4) ;UNIT NO SET IN TABLEW;READY TO TEST
1#: MOV R3,PORTID ;PORT ID SET
: MOV R3,2(R4) ;PORTID SET IN TABLEW
: MOV #17777,4(R4) ;INSERT NEW -1,-1 FOR LAST ENTRY
: MOV 4(R4),6(R4) ; OF THE TABLEW
: MOV #2400,WORK ;WORK = RETRY LIMIT
3#: JSR PC,GTSTAT ;GET STATUS, GET NEXT UNIT NUMBER
: BCC 7# ;OK, CONTINUE
: DEC WORK ;ELSE IF OFFLINE, DECR COUNT
: BNE 3# ;IF COUNT > 0, TRY AGAIN.
5#: JSR PC,DROP2 ;DROP THE DRIVE
: BR 17# ;TRY NEXT UNIT
7#: MOV P,UNIT+RSPACK,UNITNO ;UNIT NUMBER FROM RESPONCE PACKET IN UNITNO
: *** CHECK FOR CASE WHERE THE MORE UNITS THEN DRIVES HAVE BEEN SPECIFIED.
: *** NEXT UNIT MODIFIER WILL GIVE A DUPLICATE UNIT NUMBER.
: MOV #TABLEW,R2 ;R2 -> TABLE TO FIND DUPLICATE
: MOV #1,R5 ;R5 IS TEMP PORTID
9#: CMP R2,#TEND ;REACHED THE BOTTOM?
: BEQ 15# ;IF SO, EXIT
: CMP R3,R5 ;REACHED THE LATEST ENTRY?
: BEQ 15# ;IF SO, EXIT
: CMP UNITNO,(R2) ;DO WE HAVE A DUPLICATE UNIT NUMBER?
: BEQ 13# ;IF SO, ERROR
11#: ADD #4,R2 ;IF NOT, POINT TO NEXT POINTER
: ASL R5
: BR 9# ;AND CONTINUE
13#: MOV (R4),UNITNO ;DROP DRIVE FROM TABLE
: MOV R3,PORTID
: JSR PC,DROP3 ;AND DROP IT
: BR 17#
15#:
: ***

```

```

963 003202 026714 175424      CMP      UNITNO,(R4)      ;IS THE UNITNO CORRECT?
964 003206 001402              BEQ      17$              ;IF SO, CHECK FOR NEXT UNIT
965 003210 016714 175416      MOV      UNITNO,(R4)      ;ELSE, CORRECT THE UNIT NUMBER IN TABLE
966 003214              17$: ASL      R3                ;NEXT PCRTID SET
967 003214 006303              CMP      DVICE,R3        ;DONE?
968 003216 026703 175406      BMI     19$              ;IF R3 > DVICE, ALL DESIRED DRIVES ARE FOUND.
969 003222 100407              INC     UNITNO           ;NEXT UNITNO SET
970 003224 005267 175402      ADD     #4,R4            ;POINT TO NEXT ENTRY TO TEST DRIVE
971 003230 062704 000004      CMP     #TEND,R4        ;POINT TO END? IF SO, TABLE FULL
972 003234 022704 000704      BHI     1$               ;IF R4 NOT REACHED END, GO TEST
973 003240 101276              19$: RTS      PC
974 003242
975 003242 000207
976
977
978
979
980
981
982
983 003244 022700 000003      TSTOFL: CMP     #ST.OFL,R0 ; WAS THE DRIVE FOUND OFFLINE?
984 003250 001403              BEQ     10$              ; CHECK WHAT KIND OF OFFLINE
985 003252 022700 000013      CMP     #ST.DRV,R0      ; WAS IT A DRIVE ERROR? -> SDI?
986 003256 001012              BNE     13$              ; IF IT WAS NOT, ERROR (DROP DRIVE)
987 003260 032767 000740 175032 10$: BIT     #<SC.NVL+SC.DIS+SC.DUP+SC.IOP>,P.STS+RSPACK ; WERE ANY OF THESE BITS SET?
988
989
990 003266 001004              BNE     12$              ; = NO VOLUME MOUNTED, UNIT DISABLED BY FIELD SREVICE
991 003270 032767 177000 175022  BIT     #<SC.NVL+SC.DIS+SC.DUP+SC.IOP+ST.MSK>,P.STS+RSPACK ; OR DUPLICATE UNIT NUMBER OR UNIT INOPERATIVE
992 003276 001002              BNE     13$              ; IF SO, EXIT
993 003300 000241              12$: CLC                ; IF SO, DROP
994 003302 000207              RTS     PC                ; CLEAR CARRY
995 003304 000261              13$: SEC                ; RETURN
996
997 003306 004767 002124      JSR     PC,ERRORH        ; SET CARRY, DRIVE WAS FOUND TO BE OFFLINE
998 003312 000207              RTS     PC                ; OR ANOTHER ERROR
                                ; REPORT ERROR
                                ; RETURN
  
```

```

1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017 003314
1018 003314 C32767 001000 174474
1019 003322 001004
1020
1021 003324 005767 175304
1022 003330 100443
1023 003332 000422
1024
1025
1026
1027
1028 003334 012701 000010
1029 003340 004767 001262
1030 003344 103013
1031 003346 004767 177672
1032 003352 103507
1033
1034 003354 005067 175144
1035 003360 052767 140000 174674
1036 003366 004767 001426
1037
1038 003372 000402
1039 003374 005301
1040 003376 001360
1041 003400 004767 001316
1042 003404 103753
1043 003406 016767 174742 175210
1044 003414 016767 174732 175200
1045 003422 001006
1046 003424 005767 175174
1047 003430 001731
1048
1049 003432 012767 100000 175174
1050
1051
1052
1053
1054
    
```

```

*****
CYCLE DISK
DO THE DISK CYCLE
DO GET STATUS COMMANDS TO ASSURE THAT THE DRIVE
IS AVAILABLE (FOR DUAL PORTING)
CHECK DRIVE TO BE ONLINE
IF TRUE
    PICK THE BLOCK
    WRITE
    READ
    DATA CHECK
    MAKE THE DRIVE AVAILABLE
ELSE DROP DRIVE
*****
CYCLED:
    BIT    #SR.DUA,SR1          ;DUAL PORT?
    BNE    2$                  ;IF NOT, CONTINUE
; *** CHECK IF WE DO ONLINE FOR THE FIRST TIME.
    TST    TRY                  ;IF TRY HAS SET MSB, DON'T DO ONLINE
    BMI    16$                 ; DON'T DO ONLINE
    BR     10$                 ; ELSE DO ONLINE (1ST TIME THROUGH IN THIS PASS)
; ***
; *** DO GET STATUS COMMANDS TO ASSURE THE DRIVE IS AVAILABLE TO THE CONTROLLER
; *** FOR DUAL PORTING.
; ***
2$:    MOV    #10,R1            ;R1 = # OF GET STATUS TO DO
4$:    JSR    PC.GTSTAT         ;IS THE DRIVE OFFLINE?
        BCC    6$              ;IF ALL OK, DO THE CYCLE
        JSR    PC.TSTOFL       ;ELSE, CHECK IF OFFLINE
        BCS    24$             ;IF IT ERRED, DROP THE DRIVE
; *** HANDLE OFF LINE DRIVE. WAIT FOR AVAILABLE ATTENTION MESSAGE
        CLR    EXPV            ;EXPECT AN AVAILABLE ATTENTION MESSAGE
        BIS    #<RG.OWN+RG.FLG>,RSPONC*2 ;SET RING FOR ATTN MESSAGE
        JSR    PC.INTERP       ;WAIT FOR MESSAGE
        ; 2ND ATTENTION MESSAGE
        BR     10$
6$:    DEC    R1                ;DONE?
        BNE    4$              ;IF NOT DCNE, TRY AGAIN
10$:   JSR    PC.ONLINE        ;DO AND ONLINE COMMAND
        BCS    2$              ;IF CARRY WAS SET, TRY AGAIN
14$:   MOV    P.UNSZ*2,RSPACK,UNSZH ;IS THE UNIT SIZE HI ADDRESS
        MOV    P.UNSZ,RSPACK,UNSZL ;GET UNIT SIZE/IS IT = 0?
        BNE    16$             ;IF NOT ZERO, CONTINUE WITH ITERATION
        TST    UNSZH           ;IS UNSZH ALSO 0?
        BEQ    CYCLED         ;IF 0, TRY TO BRING ONLINE AGAIN
; *** SET MSB OF TRY TO SHOW THAT INITIAL ONLINE IS DONE
        MOV    #100000,TRY
*****
;
; THE FOLLOWING SEGMENT SETS THE LIMIT FOR THE UNIT SIZE.
; THE VALUE (UNIT SIZE - (WRITE BUFFER SIZE/NORMAL BLOCK SIZE))
;
    
```

```

1055      ;           IS THE LAST SECTOR POSSIBLE TO RIGHT TO.           ;
1056      ; .....
1057      ; .....
1058 003440      016700 174476      16: MOV      WBUF SZ,RO      ;WBUF SZ IN RO AS A LIMIT
1059 003440      005001      CLR      R1           ;R1 = # OF BLOCKS
1060 003444      005201      18: INC      R1           ;INCREMENT THE # OF BLOCKS
1061 003446      005201      SUB      #400,RO      ;DECREMENT A BLOCK
1062 003450      162700 000400      BPL     18:          ;BR IF > 0
1063 003454      100374      SUB      R1,UNSZL     ;ADJUST THE UNIT SIZE
1064 003456      160167 175140      ; *** NOW PICK WHICH BLOCK TO WRITE TO
1065      JSR     PC,PICKBK      ;ELSE SELECT A SECTOR TO TEST
1066 003462      004767 000156      JSR     PC,GTSTAT     ;DID WE NOT GET THE DRIVE ONLINE?
1067 003466      004767 001134      BCS     2:           ;IF WE DID NOT, GO BACK TO TOP AND TRY AGAIN
1068 003472      103720      CMP      #ST.AVL,RO   ;IS IT AVAILABLE?
1069 003474      022700 000004      BEQ     2:           ;IF SO, GO BACK TO TOP AND TRY AGAIN
1070 003500      001715      ; *** WRITE TO THE BLOCK SELFCTED
1071      JSR     PC,WRITE     ;WRITE THE DATA FOR USER DEFINED # OF WORDS
1072 003502      004767 000720      BCC     19:          ;IF OK, CONTINUE
1073 003506      103007      BIT     #SR.DUA,SR1   ;ARE WE DOING DUAL PORT?
1074 003510      032767 001000 174300 BNE     2:           ;IF YES, RETRY
1075 003516      001306      JSR     PC,ERRORH     ;ELSE, HARD ERROR
1076 003520      004767 001712      BR      22:          ;AND EXIT; BCS 22: ;IF ERROR, EXIT
1077 003524      000421      ; *** READ IT BACK
1078      JSR     PC,READ     ;READ A BLOCK
1079 003526      004767 000730      BCS     22:          ;IF ERROR, EXIT
1080 003532      103416      BIT     #SR.CMP,SR1   ;DO A DATA COMPARE?
1081 003534      032767 004000 174254 BNE     20:          ;IF NOT, SKIP THE COMPARE
1082 003542      001004      ; *** COMPARE DATA
1083      CDATA$,BEGIN,RBUFA ; REQUEST FOR MONITOR TO CHECK DATA
1084 003544      104412 000000' 000126' .+2 ; IF ERROR, CONTINUE
1085 003552      003554'      BIT     #SR.DUA,SR1   ;DO WE DO AN AVAILABLE?
1086 003554      032767 001000 174234 BEQ     22:          ;IF NOT(BIT NOT SET) SKIP AVAILABLE
1087 003562      001402      ; *** MAKE THE DRIVE AVAILABLE
1088      JSR     PC,AVAILB   ;RELEASE THE DRIVE
1089 003564      004767 001014      CLC     ;EVERY THING WAS OK
1090 003570      000241      ;WASTE A LITTLE TIME SO OTHER
1091      RTS     PC         ;CONTROLLER CAN GRAB DRIVE
1092 003572      000207      ;RETURN
1093      ; *** SUBROUTINE TO WAIT FOR AN INTERRUPT
1094      ; *** RETURNS AFTER THE INTERRUPT OCCURS
1095      DOINTR:
1096 003574      005067 174724      CLR     EXPV          ;EXPECT AN AVAILABLE ATTENTION MESSAGE
1097 003574      005067 174724      BIS     #<RG.OWN+RG.FLG>,RSPONC*2 ; SET OWN AND FLAG FOR RESPONCE RING
1098 003600      052767 140000 174454 JMP     INTERP        ;WAIT FOR ATTENTION MESSAGE & RETURN
1099 003606      000167 001206
1100      ; .....
1101      ; .....
1102      ; .....
1103      ; .....
1104      ; .....
1105      ; .....
1106      ; .....
1107      ; .....
1108      ; .....
    
```

```
1109 003612      CYCLEL: JSR      PC.MAITW      ;DO A MAINTENENCE WRITE
1110 003612      JSR      PC.MAITR      ;DO A MAINTENENCE READ
1111 003616      004767 000430      BIT      @SR.CMP.SR1    ;DO A DATA COMPARE?
1112 003622      032767 004000 174166 BNE      21#           ;IF NOT, SKIP THE COMPARE
1113 003630      001004          CDATA#,BEGIN,RBUFPA ; REQUEST FOR MONITOR TO CHECK DATA
1114 003632      104412 000000' 000126' .+2          ; IF ERROR, CONTINUE
1115 003642      003642'          RTS      PC
1116 003642      000207          ;
```

```

1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128 003644
1129 003644 032767 002000 174144
1130 003652 001467
1131 003654
1132 003654 104417 000000'
1133 003660 016746 174170
1134 003664 104417 000000'
1135 003670 016746 174160
1136
1137
1138
1139 003674 000241
1140 003676 042716 100000
1141 003702 012667 174712
1142 003706 005767 174712
1143 003712 001430
1144
1145 003714 016700 174704
1146 003720 005100
1147 003722 012701 100000
1148 003726 030100
1149 003730 001403
1150 003732 000241
1151 003734 006001
1152 003736 000773
1153 003740 040100
1154 003742 000241
1155 003744 006001
1156 003746 001374
1157 003750 040067 174644
1158 003754 026767 174640 174642
1159 003762 002420
1160 003764 001405
1161 003756 006267 174626
1162 003772 000414
1163
1164
1165
1166 003774 005067 174620
1167 004000 005767 174616
1168 004004 001406
1169 004006 166716 174610
1170 004012 103375
1171 004014 066716 174602
1172 004020 000401

*****
:
: PICK A BLOCK TO WRITE TO.
:
: EITHER PICK THE NEXT SEQUENTIAL BLOCK (DEFAULT) OR TAKE ONE AT
: RANDOM.
:
: OUTPUT: FILL SECH & SECL (CURRENT SECTOR ADDR)
:
*****
PICKBK: BIT #SR,SEQ,SR1 ;CHECK SR1 FOR RANDOM ACCESS MODE
        BEQ SEQACC ;BR IF SEQUENTIAL ACCESS

RANACC: RAND#,BEGIN ;GENERATE THE SECTOR ADDRESS
        MOV RANNUM,-(SP)
        RAND#,BEGIN ;GENERATE THE SECTOR ADDRESS
        MOV RANNUM,-(SP)

: ADJUST HI ADDRESS FIRST
:
        CLC ;CLEAR CARRY FOR ROTATE
        BIC #100000,(SP) ;CLEAR UPPER BIT MAKES SURE VALUE'S
        MOV (SP),SECH ;STORE IN SECTOR HI ADDRESS
        TST UNSZH ;IS THE MAX SIZE 0?
        BEQ 3# ;IF 0, GET LOW SECTOR ADDRESS
: *** UNSZH > 0 IF CODE FALLS THROUGH HERE
        MOV UNSZH,R0 ;R0 = MAX VALUE
        COM R0 ;R0 COMPLEMENT, NOW FIND MS ZERO
        MOV #100000,R1 ;R1 IS INDEX INTO MAX VALUE
1#: BIT R1,R0 ;HAVE 0 YET?
        BEQ 2# ;IF 1ST 0 REACHED, CLEAR REST OF THE BITS
        CLC ;CLEAR CARRY FOR ROR
        ROR R1 ;POINT TO NEXT BIT
        BR 1# ;BRANCH TO TEST AGAIN
2#: BIC R1,R0 ;CLEAR REST OF THE BITS
        CLC ;CLEAR CARRY FOR ROR
        ROR R1 ;IF R1 ROTATES INTO CARRY, R1 = 0
        BNE 2# ;IF R1 NOT 0, MORE BITS TO CLEAR
        BIC R0,SECH ;CLEAR UPPER BITS OF HIGH SECTOR VALUE
        CMP SECH,UNSH ;IF THE HIGH SECTOR VALUE > MAX VALUE?
        BLT 7# ;IF <, EXIT
        BEQ 4# ;IF =, TEST LOW ORDER VALUE
        ASR SECH ;SECH = SECH/2 - CAN'T BE > MAX NOW
        BR 7# ;EXIT

: GET LOW SECTOR ADDRESS
:
3#: CLR SECH ;CLEAR HI SECTOR SIZE
4#: TST UNSZL ;IS THE HIGHEST POSSIBLE = 0?
        BEQ 6# ;IF TRUE, DON'T DO LOOP
5#: SUB UNSZL,(SP) ;ELSE, SECL = SECL - UNSZL (ADJUST)
        BCC 5# ;IF UNSZL > SECL, LOOP
        ADD UNSZL,(SP) ;ELSE SUBTRACTED ONCE TOO OFTEN
        BR 7# ; AND EXIT
    
```

```

1173 004022 005016          6#: CLR (SP) ;CLEAR LO SECTOR ADDRESS (IF HIGHEST POSSIBLE 0)
1174 004024 012667 174566 7#: MOV (SP)+,SECL ;SAVE LO SECTOR ADDRESS
1175 004030 000207          RTS PC ;RETURN
1176
1177
1178 ;GENERATE DISK ADDRESS BY SEQUENTIAL ADDRESSING
1179
1180 004032          SEQACC:
1181 004032 005267 174560      INC SECL ;INCREMENT THE SECTOR ADDRESS
1182 004036 001405          BEQ 16# ;BR IF ZERO
1183 004040 026767 174552 174554  CMP SECL,UNSZL ;OVER LIMIT?
1184 004046 103413          BLO 18# ;BR IF LOWER
1185 004050 000402          BR 17# ;SKIP THE INCREMENT
1186 004052          16#:
1187 004052 005267 174542      INC SECH ;INCREMENT SECTOR HIGH ADDRESS
1188 004056          17#:
1189 004056 026767 174536 174540  CMP SECH,UNSZH ;OVER LIMIT?
1190 004064 103404          BLO 18# ;BR IF LOWER
1191 004066 C0C67 174524          CLR SECL ;RESET THE STARTING SECTOR ADDRESS
1192 004072 005067 174522          CLR SECH ;
1193
1194 004076          18#:
1195 004076 000207          RTS PC
  
```


1197
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1210
 1211
 1212 004100
 1213 004100 012767 000001 174406
 1214 004106 000407
 1215 004110
 1216 004110 012767 000002 174376
 1217 004116 000403
 1218 004120
 1219 004120 012767 000003 174366
 1220 004126 036767 174504 174474
 1221 004134 001445
 1222 004136 022767 177777 174472
 1223
 1224 004144 001441
 1225
 1226 004146 046767 174464 174454
 1227

 004154 104421 000000' 000632'
 004162 000535'
 1228 004164 105067 174352
 1229

 004170 104420 000000' 000636'
 004176 000526'
 1230 004200 012764 177777 000002
 1231 004206 005367 174302
 1232 004212 001004
 1233 004214 10' 303 000000' 005732'
 1234 004222 00412
 1235 004224 005367 174264 1#:
 1236 004230 001004
 1237 004232 104403 000000' 005750'
 1238 004240 000403
 1239 004242 104403 000000' 005766' 2#:
 1240

```

*****
:
: DROP A DRIVE
:
: A DRIVE WOULDN'T RESPOND, DROP IT. SET THIS UP IN DVICE.
:
: INPUT UNITNO = UNIT NUMBER OF DRIVE TO DROP
:        PORTID = BIT SET TO DROP DRIVE
:
: OUTPUT DVICE HAS A BIT CLEARED. THE BIT POSITION
:         REPRESENTS THE DRIVE
:
*****
DROP1:  MOV    #1,NUM
        BR     DROP4
DROP2:  MOV    #2,NUM
        BR     DROP4
DROP3:  MOV    #3,NUM
DROP4:  BIT    PORTID,DVICE      ;HAS THE DRIVE BEEN DROPPED, DON'T DROP AGAIN
        BEQ    10$              ;
        CMP    #177777,PORTID  ;IF DRIVE HAS BEEN DROPPED, DON'T DROP AGAIN
                                ;(WILL ZERO DVICE PREMATURE)
        BEQ    10$              ;IF =, DRIVE HAS BEEN DROPPED -> EXIT ROUTINE
                                ;
        BIC    PORTID,DVICE      ;DROP THE DRIVE
;*****
;CONVERT UNITNO TO ASCII AND
;STORE AT ADR2
BTOD$,BEGIN,UNITNO,ADR2
;*****
CLRB   ADR2+5
;*****
;CONVERT PORTID TO ASCII AND
;STORE AT ADR1
OTOA$,BEGIN,PORTID,ADR1
;*****
MOV    #177777,2(R4)          ;DESELECT DRIVE SO IT WON'T BE USED AGAIN.
DEC    NUM                    ;DROPPED FOR WHICH ERROR?
BNE    1$                      ;IF NOT FOR ERRORS, CONTINUE
MSGN$,BEGIN,DRP1             ;ASCII MESSAGE CALL WITH COMMON HEADER
BR     10$
DEC    NUM                    ;WAS UNIT NOT FOUND?(NON EXISTENT UNIT)
BNE    2$                      ;IF NOT, CONTINUE
MSGN$,BEGIN,DRP2             ;ASCII MESSAGE CALL WITH COMMON HEADER
BR     10$
MSGN$,BEGIN,DRP3             ;ASCII MESSAGE CALL WITH COMMON HEADER
; ACTUAL UNITS FOUND

```

1241 004250 000207
1242
1243

101: RTS PC

1245
 1246
 1247
 1248
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257 004252 004767 001066
 1258 004256 012767 000030 174126
 1259 004264 016767 174216 174132
 1260 004272 016767 174206 174122
 1261 004300 016700 173626
 1262 004304 000424
 1263
 1264
 1265
 1266
 1267
 1268
 1269
 1270
 1271
 1272
 1273
 1274
 1275
 1276
 1277 004306 004767 001032
 1278 004312 012767 000031 174072
 1279 004320 016767 174166 174076
 1280 004326 016767 174156 174066
 1281 004334 026767 173602 174264
 1282 004342 100403
 1283 004344 016700 174256
 1284 004350 000402
 1285 004352 016700 173564
 1286 004356 006300
 1287 004360 010067 174032
 1288 004364 012767 000020 173710
 1289 004372 012767 000044 173776
 1290 004400 012767 000001 174030
 1291 004406 012767 177777 173764
 1292 004414 012767 177777 173662
 1293 004422 000167 000322
 1294

```

*****
:
:      MAITENENCE READ
:
:      SET UP A PACKET WITH:
:          OPCODE & MODIFIER
:          REGION ID & REGION OFFSET
:          READ BUFFER DESCRIPTOR
:          BYTE COUNT
:      THEN SEND THE PACKET
:
*****
  
```

```

*****
:
:      MAITR: JSR      PC,CLRPAK          ;CLEAR THE PACKETS
:            MOV      #OP.MRD,P.OPCD+CMACK ;SET THE OPCODE
:            MOV      RBUFEP,P.ADEA+CMACK  ;SET THE BUFFER DESCRIPTOR
:            MOV      RBUFPP,P.ADPA+CMACK  ;
:            MOV      RBUFSZ,RO            ;STORE THE BUFFER SIZE IN WORDS
:            BR       MAITP               ;SET UP THE REST OF THE PACKET
:
*****
  
```

```

*****
:
:      MAITENENCE WRITE
:
:      SET UP A PACKET WITH:
:          OPCODE & MODIFIER
:          REGION ID & REGION OFFSET
:          WRITE BUFFER DESCRIPTOR
:          BYTE COUNT (EITHER WBUFSZ OR LIMIT IF WBUFSZ > LIMIT)
:      THEN SEND THE PACKET
:
*****
  
```

```

*****
:
:      MAITW: JSR      PC,CLRPAK          ;CLEAR THE PACKETS
:            MOV      #OP.MWR,P.OPCD+CMACK ;SET THE OPCODE
:            MOV      WBUFEP,P.ADEA+CMACK  ;SET THE BUFFER DESCRIPTOR
:            MOV      WBUFPP,P.ADPA+CMACK  ;
:            CMP      WBUFSZ,LIMIT        ;IS THE BUFFER SIZE > LIMIT?
:            BMI      1$                  ;IF NOT, WBUFSZ IS OK
:            MOV      LIMIT,RO            ;STORE THE BUFFER SIZE IN WORDS
:            BR       MAITP               ;AND SKIP
:            1$: MOV      WBUFSZ,RO        ;STORE THE BUFFER SIZE IN WORDS
:            MAITP: ASL      RO            ;MAKE IT NUMBER OF BYTES
:            MOV      RO,P.BCNT+CMACK     ;SET WRITE BUFFER SIZE
:            MOV      #16.,RSPLN          ;SET RESPONSE PACKET LENGTH
:            MOV      #36.,CMPLN          ;SET COMMAND PACKET LENGTH
:            MOV      #1,P.RGIL+CMACK     ;SET REGION ID = 1
:            MOV      #177777.CMPVIR      ;SET COMMAND VIRTUAL CIRCUIT (-1 FOR DM)
:            MOV      #177777.RSPVIR      ;SET COMMAND VIRTUAL CIRCUIT
:            JMP      SEND                ;SEND THE PACKET
:
*****
  
```

```

1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307 004426 004767 000712
1308 004432 012767 000042 173752
1309 004440 016700 173476
1310 004444 016767 174040 173750
1311 004452 016767 174034 173744
1312 004460 000415
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325 004462 004767 000656
1326 004466 012767 000041 173716
1327 004474 016700 173432
1328 004500 016767 174002 173716
1329 004506 016767 173772 173706
1330 004514 012767 000040 173560
1331 004522 012767 000040 173646
1332 004530 006300
1333 004532 010067 173660
1334 004536 016767 174054 173672
1335 004544 016767 174050 173666
1336 004552 000476
    
```

```

*****
WRITE
SET UP OP CODE, MODIFIERS, BUFFER SIZE (BYTE COUNT),
BUFFER DESCRIPTOR (PHYSICAL AND EXTENDED ADDRESS)
LET READ SET SIMILAR DATA IN THE PACKET:
DISK ADDRESS AND CYLINDER ID (LOGICAL BLOCK NUMBER).
THEN SEND THE PACKET.
*****
    
```

```

WRITE: JSR PC, CLRPAK ; CLEAR PACKETS
        MOV #OP.WR, P.OPCD+CMACK ; SET THE OPCODE
WRITEA: MOV WBLF SZ, RO ; STORE THE BUFFER SIZE IN WORDS
        MOV WBUFPP, P.ADPA+CMACK ; SET THE BUFFER DESCRIPTOR(PA)
        MOV WBUFEP, P.ADEA+CMACK ; SET THE BUFFER DESCRIPTOR(EA)
        BR READA ;
    
```

```

*****
READ
SET UP OP CODE, MODIFIERS, BUFFER SIZE (BYTE COUNT),
BUFFER DESCRIPTOR (PHYSICAL AND EXTENDED ADDRESS),
DISK ADDRESS AND CYLINDER ID (LOGICAL BLOCK NUMBER).
THEN SEND THE PACKET.
*****
    
```

```

READ: JSR PC, CLRPAK ; CLEAR PACKETS
        MOV #OP.RD, P.OPCD+CMACK ; SET THE OPCODE
        MOV RBUFSZ, RO ; STORE THE BUFFER SIZE IN WORDS
        MOV RBUFEP, P.ADEA+CMACK ; SET THE BUFFER DESCRIPTOR
        MOV RBUFPP, P.ADPA+CMACK ;
READA: MOV #32., RSPLN ; SET RESPONSE PACKET LENGTH
        MOV #32., CMPLN ; SET COMMAND PACKET LENGTH
        ASL RO ; MAKE IT NUMBER OF BYTES
        MOV RO, P.BCNT+CMACK ; SET READ BUFFER SIZE
        MOV SECL, P.LBN+CMACK ; SET LOGICAL BLOCK NUMBER
        MOV SECH, P.LBN*2+CMACK ;
        BR SEND ; SEND THE PACKET
    
```

```

1338
1339
1340
1341
1342
1343
1344
1345 004554 004767 000564
1346 004560 012767 000013 173624
1347 004566 012767 000074 173506
1348 004574 012767 000074 173574
1349 004602 000462
1350
1351
1352
1353
1354
1355
1356
1357
1358 004604 004767 000534
1359 004610 012767 000010 173574
1360 004616 012767 000014 173456
1361 004624 000413
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371 004626 004767 000512
1372 004632 012767 000003 173552
1373 004640 012767 000001 173546
1374 004646 012767 000060 173426
1375 004654 012767 000014 173514
1376 004662 000432
  
```

```

.....
:
:   DETERMINE ACCESS PATHS
:
:   SET UP CODE, GO SEND PACKET
:
:.....
DAP:  JSR      PC,CLRPAK      ;CLEAR PACKETS
      MOV      #OP.DAP,P.OPCD+CHPACK ;SET OPCODE
      MOV      #60.,RSPLN     ;SET LENGTHS
      MOV      #60.,CHPLEN
      BR       SEND          ;SEND THE PACKET
:.....
:
:   AVAILABLE PACKET
:
:   SET OP CODE AND MODIFIERS THEN SEND THE PACKET
:
:.....
AVAILB: JSR      PC,CLRPAK      ;CLEAR PACKETS
        MOV      #OP.AVL,P.OPCD+CHPACK ;SET THE OPCODE
        MOV      #12.,RSPLN     ;SET RESPONSE PACKET LENGTH
        BR       GTSTAA        ;SEND THE PACKET
:.....
:
:   GET UNIT STATUS
:
:   SET OPCODE AND MODIFIER (FOR THEN NEXT UNIT
:   THEN SEND THE PACKET
:
:.....
GTSTA~: JSR      PC,CLRPAK      ;CLEAR PACKETS
        MOV      #OP.GUS,P.OPCD+CHPACK ;SET THE OPCODE
        MOV      #48.,RSPLN     ;CLEAR MODIFIERS
        MOV      #48.,RSPLN     ;SET RESPONSE PACKET LENGTH
GTSTAA: MOV      #12.,CHPLEN    ;SET COMMAND PACKET LENGTH
        BR       SEND          ;SEND THE PACKET
:.....
  
```

1378
 1379
 1380
 1381
 1382
 1383
 1384
 1385
 1386
 1387
 1388 004664
 1389 004664 004767 000454
 1390 004670 012767 000040 173500
 1391 004676 012767 000034 173376
 1392 004704 012767 000004 173500
 1393 004712 012767 000200 173500
 1394
 1395 004720 000413
 1396
 1397
 1398
 1399
 1400
 1401
 1402
 1403
 1404
 1405
 1406 004722 004767 000416
 1407 004726 012767 000040 173346
 1408 004734 012767 000044 173434
 1409 004742 012767 000011 173442

```

*****
:
:   SET CONTROLLER CHARACTERISTICS
:
:   SET OP CODE AND CONTROLLER FLAG (ENABLE ATTENTION MSGS)
:   CLEAR MSCP VERSION, HOST TIMEOUT, USE FRACTION,
:   AND ALL OF QUAD WORD TIME AND DATE.
:   THEN SEND PACKET
:
*****
  
```

```

SCC:
      JSR      PC,CLRPAK          ;GO CLEAR THE COMMAND PACKET
      MOV      #32.,CMPLN        ;SET UP COMMAND PACKET LENGTH
      MOV      #28.,RSPLN        ;SET UP RESPONSE PACKET LENGTH
      MOV      #OP.SCC,P.OPCD+CMPACK ;SET THE OPCODE
      MOV      #CF.AVL,P.CNTF+CMPACK ;SET THE CONTROLLER FLAGS
:                                     ; TO ENABLE ATTENTION MSGS
      BR       SEND              ;SEND THE PACKET
  
```

```

*****
:
:   ONLINE
:
:   SET OPCODE, MODIFIERS, UNIT ID, HOST ID
:   SHADOW UNIT, ERROR FLAGS
:   THEN SEND PACKET
:
*****
  
```

```

ONLINE: JSR      PC,CLRPAK          ;CLEAR PACKETS
      MOV      #32.,RSPLN        ;SET RESPONSE PACKET LENGTH
      MOV      #36.,CMPLN        ;SET COMMAND PACKET LENGTH
      MOV      #OP.ONL,P.OPCD+CMPACK ;SET THE OPCODE
  
```

```

1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430 004750 005267 173314
1431 004754 001775
1432 004756 016767 173306 173416
1433 004764 016767 173642 173414
1434 004772 042767 040000 173266
1435 005000 052767 100000 173260
1436 005006 052767 140000 173246
1437 005014 005777 172766
1438 005020
1439 005020 104400 000000'
1440
1441 005024
1442 005024 000004 000000' 005032'

1443
1444 005032
1445 005032 005067 173220
1446 005036 022767 000100 173252
1447 005044 001524
1448 005046 022767 000102 173242
1449 005054 001527
1450
1451 005056
1452 005056 016700 173236
1453 005062 001513
1454 005064 042700 177740
1455 005070 001510
1456 005072 005067 173010
1457 005076 122700 000013
1458 005102 001015
1459 005104 032767 001000 172704
1460 005112 001472
1461 005114 022767 000053 173176
1462 005122 001464
1463 005124 022767 000113 173166
    
```

```

.....
SEND - SEND A PACKET
INTERP - WAIT FOR AN INTERRUPT

SET UP THE COMMAND REFERENCE NUMBER AND UNITNO IN THE PACKET
SET OWN, CLEAR FLAG IN THE COMMAND RING (FOR CONTROLLER)
SET OWN & FLAG IN MESSAGE RING (FOR INTERRUPTS BY CONTROLLER)
AFTER INTERRUPT, MAKE SURE THE PACKET WAS PROCESSED (NO HARD
OR SOFT ERRORS) THEN RETURN TO CYCLED.

INPUT:  CMPACK IS FILLED EXCEPT FOR CMDREF & UNITNO
        INTERRUPT VECTOR AND BR LEVEL ARE ESTABLISHED

OUTPUT: MSPACK IS FILLED
        CLEAR CARRY IF COMMAND PACKET WAS OK
        ELSE GO DO A HARD/SOFT ERROR.
.....
SEND:  INC      CMDREF      ;NEW COMMAND REFERENCE NUMBER
        BEQ      SEND        ;COMMAND REF # CANNOT = 0
        MOV      CMDREF,P.CRF+CMPPACK ;SET COMMAND REF NUMBER
        MOV      UNITNO,P.UNIT+CMPPACK ;SET UNIT NUMBER
        BIC      @RG.FLG,COMMAND*2    ;CLEAR FLAG
        BIS      @RG.OWN,COMMAND*2    ;SET OWN FOR COMMAND RING
        BIS      @<RG.OWN+RG.FLG>,RSPONC*2 ;SET OWN AND FLAG FOR MESSAGE RING
        TST      @ADDR              ;FORCE POLLING TO PACKET

INTERP: EXIT$,BEGIN                ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

INTERRUPT:
-----
PIRQ$,BEGIN,1$                ; QUEUE UP TO CONTINUE AT 1$ AND RTI
-----

1$:
CLR      RINTR                ;CLEAR INTERRUPT FLAG
CMP      @OP.AVA,P.OPCD+RSPACK' ;WAS AN AVAILABLE ATTENTION RECIEVED?
BEQ      15$                  ;IF IT WAS, EXIT
CMP      @OP.ACP,P.OPCD+RSPACK' ;WAS THE ACCESS PATH ATTENTION RECIEVED?
BEQ      16$                  ;IF IT WAS, GO PROCESS
                                ; ELSE CHECK SUCCESS

2$:
MOV      P.STS+RSPACK,RO      ; SUCCESS?
BEQ      14$                  ;IF YES, EXIT
BIC      @177740,RO           ;CLEAR UPPER 11 BITS OF SUB-STATUS
BEQ      14$                  ;IF SUCCESS = 0, EXIT OK
CLR      ERRYP                ;IF GOT HERE, ERROR
CMPB     @ST.DRV,RO           ; DRIVE ERROR?
BNE      3$                  ;IF NOT NEXT TEST
BIT      @SR.DUA,SR1         ;ARE WE DUAL PORTING?
BEQ      12$                  ;IF NOT, GO REPORT ERROR/ELSE EXPECTED
CMP      @<ST.DRV+SC.STO>,P.STS+RSPACK ;IS IT AN SDI RESPONCE TIMEOUT?
BEQ      10$                  ;IF TRUE, DRIVE IS NOT ONLINE, EXIT
CMP      @<ST.DRV+SC.INV>,P.STS+RSPACK ;IS IT THE INVALID SDI RESPONCE?
    
```

```

1464 005132 001460      BEQ      10$      ; IF TRUE, DRIVE IS NOT ONLINE. EXIT?
1465 005134 000461      BR       12$      ; ELSE HARD ERROR
1466 005136          3$:      CMPB     @ST.CNT,RO  ; CONTROLLER ERROR?
1467 005136 122700 000012      BNE     4$      ; IF NOT NEXT TEST
1468 005142 001004          MOV     @ERR.3,ERRTYP ; ELSE, SET ERROR TYPE
1469 005144 012767 000003 172734      BR      ERRORH    ; AND HARD ERROR
1470 005152 000531          4$:      CMPB     @ST.HST,RO  ; HOST BUFFER ACCESS ERROR?
1471 005154          BNE     5$      ; IF NOT NEXT TEST
1472 005154 122700 000011      MOV     @ERR.32,ERRTYP ; ELSE, SET ERROR TYPE
1473 005160 001004          BR      ERRORH    ; AND HARD ERROR
1474 005162 012767 000032 172716          5$:      CMPB     @ST.DAT,RO  ; DATA ERROR?
1475 005170 000522      BNE     6$      ; IF NOT NEXT TEST
1476 005172 122700 000010      MOV     @ERR.1,ERRTYP ; ELSE, SET ERROR TYPE
1477 005172 001004          BR      ERRORS    ; AND SOFT ERROR
1478 005176 001004          6$:      CMPB     @ST.WPR,RO  ; WRITE PROTECTED?
1479 005200 012767 000001 172700      BEQ     12$      ; ELSE HARD ERROR
1480 005206 000533          8$:      CMPB     @ST.AVL,RO  ; STILL AVAILABLE?
1481 005210 122700 000006      BNE     9$      ; IF NOT NEXT TEST
1482 005214 001431          CMP     @OP.GUS,P.OPCD+CMPACK ; ELSE, IF COMMAND WAS
1483 005216 122700 000004          ; GET UNIT STATUS
1484 005216 001005          BEQ     14$      ; THEN EXPECTED & LEAVE ROUTINE
1485 005222 001005          BR      12$      ; ELSE HARD ERROR
1486 005224 022767 000003 173160          9$:      CMPB     @ST.OFL,RO  ; UNIT OFFLINE?
1487 005224          BNE     13$      ; IF NOT NEXT TEST
1488          ; *** OFFLINE WHEN TRIED ONLINE OR GET UNIT STATUS
1489 005232 001427          CMP     @OP.ONL,P.OPCD+CMPACK ; WAS IT AN ONLINE COMMAND?
1490 005234 000421          BEQ     10$      ; IF SO, SET CARRY/EXIT
1491 005236 122700 000003          CMP     @OP.GUS,P.OPCD+CMPACK ; IS IT GET UNIT STATUS COMMAND?
1492 005236 001022          BEQ     10$      ; IF SO, SET CARRY/EXIT
1493 005242          CMP     @OP.WR,P.OPCD+CMPACK ; IS IT WRITE COMMAND?
1494          BNE     12$      ; IF NOT, REPORT HARD ERROR
1495 005244 022767 000011 173140          SEC     PC      ; ELSE, SET CARRY TO
1496 005252 001410          RTS     PC      ; AND RETURN TO DROP DRIVE/AWAIT AVAILABLE DRIVE
1497 005254 022767 000003 173130          ; *** HARD ERROR EXIT WITH ERROR TYPE = 6
1498 005262 001404          12$:      MOV     @ERR.6,ERRTYP ; ELSE, SET ERROR TYPE
1499 005264 022767 000042 173120      BR      ERRORH    ; AND HARD ERROR
1500 005272 001002          ; *** SOFT ERROR EXIT WITH ERROR TYPE = 0
1501 005274 000261          13$:      BR      ERRORS    ; ERROR WITH ERRTYP = 0 & IS A SOFT ERROR
1502 005276 000207          ; ST.CMP,ST.MFE,..ST.ABO,ST.CMD
1503          ; *** SUCESSFUL EXIT
1504 005300          14$:      CLC     PC      ; CLEAR CARRY 'CAUSE PACKET IS OK
1505 005300 012767 000006 172600      RTS     PC      ; ELSE, OK, SO FAR.
1506 005306 000453          15$:      TST     EXPV    ; *** WAIT FOR ATTENTION INTERRUPT
1507          ; *** DID WE GET AN AVAILABLE ATTENTION MESSAGE THAT WE EXPECTED?
1508 005310          ;
1509 005310 000472          ;
1510          ;
1511          ;
1512 005312          ;
1513 005312 000241          ;
1514 005314 000207          ;
1515 005316          ;
1516          ;
1517          ;
1518 005316 005767 173202          ;

```


1519	005322	001004			
1520	005324	012767	177777	173172	
1521	005332	000767			
1522	005334				164.
1523	005334	052767	140000	172720	
1524	005342	000626			
1525					

```

BNE 164
MOV @177777,EXPV
BR 144
BIS @<RG.OWN+RG.FLG>,RSPONC+2
BR INTERP

```

```

;IF EXPV IS NOT 0, WE GOT ONE WE DIDN'T EXPECT
;CLEAR EXPECTED AVAILABLE ATTENTION MESSAGE WORD
; AND RETURN
;
;
;
;
;WAIT FOR RESPONSE OF LAST PACKET SENT

```

```

1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538 005344
1539 005344 017767 172702 173142
1540 005352 001421
1541
    005354 104420 000000 000514'
    005362 C00526'
1542 005364 104403 000000' 006024'
1543 005372 010346
1544 005374 010446
1545 005376 004767 174010
1546 005402 012603
1547 005404 012604
1548 005406 004767 177252
1549 005412 005267 172426
1550
1551 005416 012702 000064
1552 005422 012705 000302'
1553 005426 005025
1554 005430 005302
1555 005432 001375
1556 005434 000207
1557
1558
1559
1560
1561
1562
1563 005436
1564 005436 032767 000004 172352
1565 005444 001403
1566 005446 005267 172372
1567
1568 005452 000407
1569 005454 004767 000056
1570
    005460 104405 000000' 000000
1571 005466 004767 000070
1572 005472 000261
1573 005474 000207
1574
    
```

```

:*****:
:      CLEAR PACKETS      :
:      ASSUMPTION: 1) RESPONSE BUFFER PRECEDES THE COMMAND BUFFER :
:                  2) TWO WORDS BEFORE EACH BUFFER IS FOR LENGTH :
:                  OF PACKET AND VIRTUAL CIRCUIT :
:      OUTPUT: R2 = 0 WHEN DONE :
:              R5 = END OF COMMAND PACKET WHEN DONE :
:*****:
CLRPAK:  MOV     @SAREG,NUM      ;IF SA REG NOT ZERO, STORE IN NUM
        BEQ     5$           ;IF SA REG IS ZERO, CLEAR PACKETS
:*****:
:                  ;CONVERT NUM TO ASCII AND
:                  ;STORE AT ADR1
        OTOA$,BEGIN,NUM,ADR1
:*****:
MSGN$,BEGIN,SANOTO ;ASCII MESSAGE CALL WITH COMMON HEADER
MOV     R3,-(SP)      ;SAVE R3
MOV     R4,-(SP)      ;SAVE R4
JSR     PC,INITUD     ;RE INIT SA REGISTER
MOV     (SP)+,R3      ;RESTORE R3
MOV     (SP)+,R4      ;RESTORE R4
JSR     PC,SCC        ;SET CONTROLLER CHARS AGAIN
INC     HRDCNT        ;INCREMENT HARD ERROR COUNT
:DOING THIS WILL CAUSE ANOTHER CALL TO CLRPAK
5$:     MOV     #52,R2    ;R2 = # OF WORDS TO CLEAR
        MOV     @RSPLN,R5 ;R5 -> RSPLN, 1ST WORD TO CLEAR
6$:     CLR     (R5)+     ;CLEAR WORD
        DEC     R2        ;R2 = ZERO? (DONE CONDITION)
        BNE     6$        ;IF NOT ZERO, LOOP
        RTS     PC        ;RETURN
:*****:
:      HARD ERROR      CARRY WILL BE SET :
:*****:
ERRORH:  BIT     @SR.REP,SR1 ;DO WE REPORT THE ERROR?
        BEQ     7$        ;IF SO, REPORT
        INC     HRDCNT    ;ELSE, INCREMENT THE HARD ERROR
:                  ;COUNT IF NOT REPORTED
        BR     8$        ;SKIP REPORT
7$:     JSR     PC,SETTAB ;SET UP TABLE
:*****:
        HRDR$,BEGIN,NULL ;
:*****:
8$:     JSR     PC,PRINTE
        SEC
        RTS     PC        ;RETURN TO CYCLED
    
```

```

1575
1576
1577
1578
1579
1580 005476
1581 005476 032767 000004 172312
1582 005504 001403
1583 005506 005267 172330
1584
1585 005512 000407
1586 005514 004767 000016
1587
      00520 104406 000000' 000000

1588 005526 004767 000030
1589 005532 000261
1590 005534 000207
1591
1592
1593
1594
1595
1596
1597
1598
1599 005536
1600 005536 016767 172244 172334
1601 005544 016767 172550 172332
1602 005552 017767 172474 172322
1603 005560 000207
1604
1605
1606
1607
1608
1609
1610
1611 005562
1612

      005562 104420 000000' 000320'
      005570 000526'

1613

      005572 104420 000000' 000316'
      005600 000535'

1614

      005602 104420 000000' 000312'
  
```

```

;*****
;
;      SOFT ERROR      CARRY WILL BE SET
;
;*****
ERRORS:
      BIT      #SR.REP,SR1      ;DO WE REPORT THE ERROR?
      BEQ      9#              ;IF SO, REPORT
      INC      SOFCNT          ;ELSE, INCREMENT THE HARD ERROR
                              ;COUNT IF NOT REPORTED
      BR       10#            ;SKIP REPORT
9#:   JSR      PC,SETTAB      ;SET UP TABLE
;*****
      SOFER$,BEGIN,NULL      ;
;*****
10#:  JSR      PC,PRINTE
      SEC
      RTS      PC              ;SET CARRY
                              ;RETURN TO CYCLED
;*****
;
;      SETTAB
;
;      SET UP A TABLE OF VALUES FOR A SOFT OR HARD ERROR
;
;*****
SETTAB:
      MOV      ADDR,CSRA      ;SET UP CONTROL STATUS REG REPORT
      MOV      P.STS+RSPACK,ASTAT ;SET UP STATUS
      MOV      BSAREG,ACSR    ;REPORT WHAT IS STATUS REG
      RTS      PC
;*****
;
;      PRINT EXTENDED ERROR MESSAGE
;
;      PRINT STATUS, OPCODE, UNIT NUMBER, BYTE COUNT, LBN AND ADDRESS
;
;*****
PRINTE:
;*****
;CONVERT P.STS+RSPACK TO ASCII AND
;STORE AT ADR1
      OTOA$,BEGIN,P.STS+RSPACK,ADR1
;*****
;CONVERT P.OPCD+RSPACK TO ASCII AND
;STORE AT ADR2
      OTOA$,BEGIN,P.OPCD+RSPACK,ADR2
;*****
;CONVERT P.UNIT+RSPACK TO ASCII AND
;STORE AT ADR3
      OTOA$,BEGIN,P.UNIT+RSPACK,ADR3
  
```

005610 000544'
 1615
 005612 104420 000000' 000322'
 005620 000553'
 1616
 005622 104420 000000' 000440'
 005630 000562'
 1617
 005632 104420 000000' 000436'
 005640 000571'
 1618
 005642 104420 000000' 000424'
 005650 000600'
 1619
 005652 104420 000000' 000422'
 005660 000607'
 1620 005662 104403 000000' 006066'
 1621 005670 000207
 1622

```

;*****
;*****
;CONVERT P.BCNT+RSPACK TO ASCII AND
;STORE AT ADR4
OTOA$,BEGIN,P.BCNT+RSPACK,ADR4
;*****
;*****
;CONVERT P.LBN+2+CMPPACK TO ASCII AND
;STORE AT ADR5
OTOA$,BEGIN,P.LBN+2+CMPPACK,ADR5
;*****
;*****
;CONVERT P.LBN+CMPPACK TO ASCII AND
;STORE AT ADR6
OTOA$,BEGIN,P.LBN+CMPPACK,ADR6
;*****
;*****
;CONVERT P.ADEA+CMPPACK TO ASCII AND
;STORE AT ADR7
OTOA$,BEGIN,P.ADEA+CMPPACK,ADR7
;*****
;*****
;CONVERT P.ADPA+CMPPACK TO ASCII AND
;STORE AT ADR8
OTOA$,BEGIN,P.ADPA+CMPPACK,ADR8
;*****
;*****
MSGN$,BEGIN,BANNER ;ASCII MESSAGE CALL WITH COMMON HEADER
RTS PC
  
```

	.SBTTL	MODULE MESSAGES
1624		
1625	005672	006134'
1626	005674	006205'
1627	005676	177777
1628		
1629	005700	006165'
1630	005702	000526'
1631	005704	006365'
1632	005706	000535'
1633	005710	006513'
1634	005712	000544'
1635	005714	177777
1636		
1637	005716	006134'
1638	005720	006232'
1639	005722	177777
1640		
1641	005724	006134'
1642	005726	006250'
1643	005730	177777
1644		
1645	005732	006342'
1646	005734	000535'
1647	005736	006352'
1648	005740	006735'
1649	005742	000526'
1650	005744	007422'
1651	005746	177777
1652		
1653	005750	006342'
1654	005752	000535'
1655	005754	006352'
1656	005756	006735'
1657	005760	000526'
1658	005762	007466'
1659	005764	177777
1660		
1661	005766	006342'
1662	005770	000535'
1663	005772	006352'
1664	005774	006735'
1665	005776	000526'
1666	006000	007530'
1667	006002	177777
1668		
1669	006004	006377'
1670	006006	000535'
1671	006010	006423'
1672	006012	000544'
1673	006014	006461'
1674	006016	000526'
1675	006020	006132'
1676	006022	177777
1677		
1678	006024	006575'

INITE1:	MSG2
	MSG4
	177777
INITER:	MSG3
	ADR1
	MSG10
	ADR2
	MSG14
	ADR3
	177777
INITE2:	MSG2
	MSG5
	177777
INITE3:	MSG2
	MSG6
	177777
DRP1:	MSG8
	ADR2
	MSG9
	MSG20
	ADR1
	MSGD1
	177777
DRP2:	MSG8
	ADR2
	MSG9
	MSG20
	ADR1
	MSGD2
	177777
DRP3:	MSG8
	ADR2
	MSG9
	MSG20
	ADR1
	MSGD3
	177777
ERRPAS:	MSG11
	ADR2
	MSG12
	ADR3
	MSG13
	ADR1
	MSG1
	177777
SANOTO:	MSG17

1679	006026	000526'	ADR1
1680	006030	006632'	MSG18
1681	006032	177777	177777
1682			
1683	006034	006524'	UNIOFF: MSG16
1684	006036	000526'	ADR1
1685	006040	177777	177777
1686			
1687	006042	007246'	WARN1: MSG40
1688	006044	177777	177777
1689			
1690	006046	007123'	WARN2: MSG37
1691	006050	177777	177777
1692			
1693	006052	007054'	WARN3: MSG36
1694	006054	177777	177777
1695			
1696	006056	006305'	ABOR: MSG7
1697	006060	177777	177777
1698			
1699	006062	006705'	ZERO: MSG19
1700	006064	177777	177777
1701			
1702	006066	006760'	BANNER: MSG21
1703	006070	000526'	ADR1
1704	006072	007052'	MSG23
1705	006074	000535'	ADR2
1706	006076	007052'	MSG23
1707	006100	000544'	ADR3
1708	006102	007052'	MSG23
1709	006104	000553'	ADR4
1710	006106	007052'	MSG23
1711	006110	000562'	ADR5
1712	006112	007052'	MSG23
1713	006114	000571'	ADR6
1714	006116	007052'	MSG23
1715	006120	000600'	ADR7
1716	006122	007052'	MSG23
1717	006124	000607'	ADR8
1718	006126	006132'	MSG1
1719	006130	177777	177777

```

1721
1722
1723
1724 006132      045      000
1725 006134      045      103
1726 006165      045      123
1727 006205      106      117
1728 006232      123      124
1729 006250      105      130
1730 006305      045      122
1731 006342      045      104
1732 006352      040      104
1733 006365      040      111
1734 006377      045      123
1735 006423      040      040
1736 006461      045      103
1737 006513      045      101
1738 006524      045      125
1739 006575      045      123
1740 006632      045      103
1741 006705      045      122
1742 006735      045      104
1743 006760      045      123
1744 007052      040      000
1745 007054      040      041
1746 007123      007      007
1747 007174      040      055
1748 007246      040      111
1749 007341      040      111
1750 007422      045      105
1751 007466      045      125
1752 007530      045      104
1753
1754 007616      000001
1755

.SBTTL MORE MODULE MESSAGES
.NLIST BEX
MSG1: .ASCIZ 'CONTROLLER INIT ERROR.'
MSG2: .ASCIZ 'SA REGISTER = '
MSG3: .ASCIZ 'FOUND BY DIAGNOSTIC '
MSG4: .ASCIZ 'STEP NOT SET.'
MSG5: .ASCIZ 'EXPECTED DATA WAS INCORRECT '
MSG6: .ASCIZ 'RETRY COUNT EXCEEDED, ABORT'
MSG7: .ASCIZ 'DRIVE '
MSG8: .ASCIZ 'DROPPED.'
MSG9: .ASCIZ 'IN STEP '
MSG10: .ASCIZ 'SOFT ERROR COUNT #'
MSG11: .ASCIZ '*** HARD ERROR COUNT #'
MSG12: .ASCIZ 'CHECK DATA ERROR COUNT #'
MSG13: .ASCIZ 'ADDR = '
MSG14: .ASCIZ 'UNIT WAS FOUND OFFLINE. UNIT NUMBER = '
MSG15: .ASCIZ 'SA REGISTER IS NOT ZERO. = '
MSG16: .ASCIZ 'CONTROLLER IS GOING THROUGH INITIALIZATION'
MSG17: .ASCIZ 'RING AREA NOT CLEARED'
MSG18: .ASCIZ 'DEVICE ID BIT = '
MSG19: .ASCIZ 'STATUS ENCOD UNITMU BYTECO HI LBN LO LBN EXTADR PHYADR'
MSG20: .ASCIZ '
MSG21: .ASCIZ '
MSG22: .ASCIZ '
MSG23: .ASCIZ '
MSG24: .ASCIZ '
MSG25: .ASCIZ '
MSG26: .ASCIZ '
MSG27: .ASCIZ '
MSG28: .ASCIZ '
MSG29: .ASCIZ '
MSG30: .ASCIZ '
MSG31: .ASCIZ '
MSG32: .ASCIZ '
MSG33: .ASCIZ '
MSG34: .ASCIZ '
MSG35: .ASCIZ '
MSG36: .ASCIZ '
MSG37: .ASCIZ '
MSG38: .ASCIZ '
MSG39: .ASCIZ '
MSG40: .ASCIZ '
MSGD1: .ASCIZ '
MSGD2: .ASCIZ '
MSGD3: .ASCIZ '
RBUF: .BLKW 256.
.END
;THE READ BUFFER
    
```

Symbol table

ABORT	006056R	CF.AVL	000200	ERROR5	002622R	L.UHVR	000027	NCPUOP	000020
ACSR	000102R	CF.MSC	000100	ERRPAS	006004R	L.UNTI	000016	NOAPTY	000002
ADDR	000006R	CF.OTH	000040	ERRTYP	000106R	L.USVR	000026	NTRUPT	005024R
ADDR22	001000	CF.SMD	000002	ERR.0	000000	L.VSER	000044	NULL	000000
ADR1	000526R	CF.THS	000020	ERR.1	000001	MAITP	004356R	NUM	000514R
ADR2	000535R	CF.576	000001	ERR.3	000003	MAITR	004252R	OLDEA	000520R
ADR3	000544R	CINTR	000254R	ERR.32	000032	MAITW	004306R	OLDPA	000516R
ADR4	000553R	CKMNG#	000001	ERR.6	000006	MAP22#	104416	ONLINE	004722R
ADR5	000562R	CLKPRE	000001	_XIT#	104400	MA10NC	001374R	OPEN	000000
ADR6	000571R	CLKSP#	104422	EXPAV	000524R	MD.CMP	040000	OP.ABO	000001
ADR7	000600R	CLRPAK	005344R	FREE	000150R	MD.ERR	010000	OP.ACC	000020
ADR8	000607R	CHOREF	000270R	GETPA#	104415	MD.EXP	100000	OP.ACP	000102
APTPRE	000200	CMPACK	000402R	GETWB	001722R	MD.FEU	000001	OP.AVA	000100
ASB	000106R	CMPLEN	000376R	GTSTAA	004654R	MD.NXU	000001	OP.AVL	000010
ASR04	002524R	CMPVIR	000400R	GTSTAT	004626R	MD.SCH	004000	OP.CCD	000021
ASTAT	000104R	COMPND	000264R	GMBUF#	104414	MD.SCL	002000	OP.CMP	000040
AUTO	000010	CONFIG	000056R	HC.CCT	000006	MD.SEC	001000	OP.DAP	000013
AVAILB	004604R	CPAKEA	000464R	HC.CMD	000004	MD.SER	000400	OP.END	000200
AWAS	000110R	CPAKEP	000470R	HC.CPK	000366R	MD.SPD	000001	OP.ERL	000101
BANNER	006066R	CPAKPA	000462R	HC.RCT	000002	MD.SSH	000200	OP.ERS	000022
BEGIN	000000R	CPAKPP	000466R	HC.RES	000000	MD.VOL	000002	OP.FLU	000023
BIT0	000001	CSRA	000100R	HC.RPK	000306R	MD.WBN	000100	OP.GCS	000002
BIT00	000001	CVTADR	002032R	HC.SIZ	000010	MD.WBV	000040	OP.GJS	000003
BIT01	000002	CYCLED	003314R	HRDCNT	000044R	MODNAM	000000R	OP.HRD	000030
BIT02	000004	CYCLEL	003612R	HRDER#	104403	MODSP	000252R	OP.HWR	000031
BIT03	000010	DAP	004554R	HRDPAS	000050R	MSG01	007422R	OP.ONL	000011
BIT04	000020	DATCK#	104411	ICONT	000036R	MSG02	007466R	OP.RD	000041
BIT05	000040	DATER#	104404	ICOUNT	000040R	MSG03	007530R	OP.RPL	000024
BIT06	000100	DOINTR	003574R	IDNUM	000122R	MSG#	104403	OP.SCC	000004
BIT07	000200	DROP1	004100R	IMODx	000000	MSG#	104402	OP.SHC	000102
BIT08	000400	DROP2	004110R	INDPAR	000040	MSG#	104401	OP.SUC	000012
BIT09	001000	DROP3	004120R	INIT	000030R	MSG1	006132R	OP.WR	000042
BIT1	000002	DRIP4	004126R	INITER	005700R	MSG10	006365R	OTOA#	104420
BIT11	004000	DRF1	005732R	INITE1	005672R	MSG11	006377R	PA	000474R
BIT12	010000	DRP2	005750R	INITE2	005716R	MSG12	006423R	PARPRE	002000
BIT13	020000	DRP3	005766R	INITE3	005724R	MSG13	006461R	PASCNT	000034R
BIT14	040000	DVICE	000630R	INITUD	001412R	MSG14	006513R	PA22	000500R
BIT15	100000	DVID1	000014R	INTA	002562R	MSG16	006524R	PDPF11	000002
BIT2	000004	EA	000476R	INTERP	005020R	MSG17	006575R	PDPLSI	020000
BIT3	000010	EA22	000502R	INTR	000120R	MSG18	006632R	PDP44	100000
BIT4	000020	ECCMEM	000100	KTPRES	000400	MSG19	006705R	PDP60	004000
BIT5	000040	EF.BBR	000200	KTXTND	040000	MSG2	006134R	PDP70	010000
BIT6	000100	EF.BBU	000100	LIMIT	000626R	MSG20	006735R	PICKBK	003644R
BIT7	000200	EF.FRS	000200	LOOP1	001276R	MSG21	006760R	PIRQ#	000004
BIT8	000400	EF.LOG	000040	LOOP2	001312R	MSG23	007052R	PKTSIZ	000060
BIT9	001000	EF.LST	000100	L.CHVR	000015	MSG3	006165R	POPSP	005726
BREAK#	104407	EF.MIS	000001	L.CNTI	000014	MSG36	007054R	POPSP2	022626
BR1	000012R	EF.SEX	000020	L.CYL	000034	MSG37	007123R	PORTID	000636R
BR2	000013R	ENDIT#	104413	L.DATA	000050	MSG4	006205R	PRMS#	000002
BTOD#	104421	END#	104410	L.ERLC	000030	MSG40	007246R	PRINTE	005562R
CAPRES	000004	ERRORH	005436R	L.EVNT	000000	MSG5	006232R	PRMSG	000522R
CDATA#	104412	ERROR5	005476R	L.GRP	000040	MSG6	006250R	PRTNUM	000017
CDERCT	000144R	ERROR1	002640R	L.SCTR	000042	MSG7	006305R	PRTY	000000
CDWCT	000146R	ERROR2	002636R	L.SLOT	000002	MSG8	006342R	PRTY0	000000
		ERROR3	002634R	L.TRCK	000041	MSG9	006352R	PRTY1	000040

PRTY2 = 000100	P. SMST = 000042	RSPONC 000260R	SETTAB 005536R	TABLEW 000644R
PRTY3 = 000140	P. SHUN = 000040	RSPPA 000272R	SETUP 003010R	TEND 000704R
PRTY4 = 000200	P. STS = 000012	RSPPP 000276R	SNDSTP 002536R	TIMER = 002260
PRTY5 = 000240	P. SZOF = 000012	RSPVIR 000304R	SOF CNT 000042R	TIMOUT = 005670
PRTY6 = 000300	P. TIME = 000024	RSTRT 000112R	SOFER1 = 104406	TRPDFD = 000023
PRTY7 = 000340	P. TRCK = 000044	R6 = *000006	SOFPAS 000046R	TRY 000634R
PS = 177776	P. UNFL = 000016	R7 = *000007	SPOINT 000032R	TSTOFL 003244R
PSW = 177776	P. UNIT = 000004	SANOTO 006024R	SPSIZ = 000040	UF.CMR = 000001
PUSH = 005746	P. UNSZ = 000044	SAREG 000252R	SR.CMP = 004000	UF.CMW = 000002
PUSH2 = 024646	P. UNTI = 000024	SA.CMD = 034000	SR.DUA = 001000	UF.INA = 040000
PWRFLG = 000002	P. USEF = 000022	SA.CME = 000360	SR.REP = 000004	UF.RMV = 000200
P.ADEA = 000022	P. VRSN = 000014	SA.DIA = 000400	SR.SEG = 002000	UF.RPL = 010000
P.ADPA = 000020	P. VSER = 000050	SA.ERC = 003777	SR.SUM = 000010	UF.SCH = 004000
P.BCNT = 000014	QMON22 = 000010	SA.ERR = 100000	SR.XFR = 000002	UF.SCL = 002000
P.BUFF = 000020	RANACC 003654R	SA.GO = 000001	SR1 000016R	UF.MBN = 000040
P.CMST = 000020	RAND1 = 104417	SA.INE = 000200	SR2 000020R	UF.MPH = 020000
P.CNCL = 000022	RANRUM 000054R	SA.INT = 000200	SR3 000022R	UF.MPS = 010000
P.CNT = 000006	RANRUF 007616R	SA.LFC = 040000	SR4 000024R	UF.576 = 000004
P.CNTF = 000016	RB# 000130R	SA.MAP = 000100	START 000710R	UNIOFF 006034R
P.CNTI = 000024	RB.#EP 000506R	SA.MCV = 000377	STAT 000026R	UNITFL 000640R
P.CPSP = 000042	RBUFPA 000126R	SA.NSI = 002000	ST.ABO = 000002	UNITMO 000632R
P.CRF = 000000	RBUFPP 000504R	SA.PRG = 000001	ST.AVL = 000004	UNSZH 000624R
P.CTMO = 000020	RBUFSZ 000132R	SA.Q22 = 001000	ST.CMD = 000001	UNSZL 000622R
P.CYL = 000050	RBUFVA 000124R	SA.RSE = 000017	ST.CMP = 000007	USTACK = 000001
P.ELGF = 000034	READ 004462R	SA.RSP = 003400	ST.CNT = 000012	VA 000472R
P.FBKX = 000034	READA 004514R	SA.SM = 000040	ST.DAT = 000010	VECTOR 000010R
P.FLGS = 000011	RESTR 001066R	SA.S1 = 004000	ST.DIA = 000037	WARN1 006042R
P.GRP = 000046	RESTR1 001136R	SA.S2 = 010000	ST.DRV = 000013	WARN2 006046R
P.HSTI = 000020	RESTR2 001112R	SA.S3 = 020000	ST.HST = 000011	WARN3 006052R
P.HTMO = 000020	RES1 000056R	SA.S4 = 040000	ST.HFE = 000005	MASADR 000104R
P.LBN = 000034	RES2 000060R	SA.VCE = 000177	ST.HSK = 000037	MBUFEA 000136R
P.LGDT = 000014	RG.FLG = 040000	SA.VEC = 000177	ST.CFL = 000003	MBUFEP 000512R
P.MEDI = 000034	RG.OWN = 100000	SBADR 000102R	ST.SUB = 000040	MBUFPA 000134R
P.MLUN = 000014	RH70 = 001000	SCC 004664R	ST.SUC = 000000	MBUFPP 000510R
P.MOD = 000012	RINTR 000256R	SC.DIS = 000400	ST.WPR = 000006	MBUFRQ 000140R
P.OPCD = 000010	RLIM = 000004	SC.DUP = 000200	SVR0 000062R	MBUFSZ 000142R
P.OTRF = 000014	RPAKEA 000370R	SC.INV = 000100	SVR1 000064R	MDFR 000116R
P.RBN = 000014	RPAKEP 000374R	SC.IOP = 000100	SVR2 000066R	MDTO 000114R
P.RBNS = 000056	RPAKPA 000366R	SC.NVL = 000040	SVR3 000070R	WORK 000642R
P.RCTC = 000057	RPAKPP 000372R	SC.STO = 000040	SVR4 000072R	WRITE 004426R
P.RCTS = 000054	RSPACK 000306R	SECH 000620R	SVR5 000074R	WRITEA 004440R
P.RGID = 000034	RSPEA 000274R	SECL 000616R	SVR6 000076R	XFLAG 000005R
P.RGOF = 000040	RSPEP 000300R	SEND 004750R	SYSCNT 000052R	ZERO 006062R
P.SFTW = 000040	RSPLEN 000302R	SEQACC 004032R		

. ABS. 000000 000 (R,I,GBL,ABS,OVR)
 010616 001 (R,I,LCL,RFL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 13663 Words (54 Pages)

DUBE DEC/X11 SYSTEM EXERCISER M MACRO V05.03 Friday 27-Sep-85 16:23 Page 28-3
Symbol table

Size of core pool: 19372 Words (74 Pages)
Operating system: RSX-11M/PLUS (Under VAX/VMS)

Elapsed time: 00:00:49.34
XDUBEO, XDUBEO/CR/ ?=XDUBEO.DOC, DDXCJM.MAC, XDUBEO.MAC

SYMBOL	VALUE	REFERENCES	CREF
ABORT	006056 R	16 904 #27-1696	
ACSR	000102 R	#3-3 *26-1602	
ADDR	000006 R	#3-3 13-530 15-644 16-897 25-1437 26-1600	
ADDR22	001000	#3-3 15-711 15-734 15-751 15-768 15-785	
ADR1	000526 R	#3-85 14-569 *14-570 16-883 20-1229 26-1541 26-1612 27-1630 27-1649	
		27-1657 27-1665 27-1674 27 1679 27-1684 27-1703	
ADR2	000535 R	#3-87 14-565 *14-566 16-881 20-1227 *20-1228 26-1613 27-1632 27-1646	
		27-1654 27-1662 27-1670 27-1705 16-897 26-1614 27-1634 27-1672 27-1707	
ADR3	000544 R	#3-89 14-567 *14-568	
ADR4	000553 R	#3-91 26-1615 27-1709	
ADR5	000562 R	#3-93 26-1616 27-1711	
ADR6	000571 R	#3-95 26-1617 27-1713	
ADR7	000600 R	#3-97 26-1618 27-1715	
ADR8	000607 R	#3-99 26-1619 27-1717	
APTPRE	000200	#3-3	
ASB	000106 R	#3-3	
ASR04	002574 R	15-715 15-738 15-755 15-772 15-789 #15-811	
ASTAT	000104 R	#3-3 *26-1601	
AUTO	000010	#3-3	
AVAILB	004604 R	18-1088 #23-1358	
AWAS	000110 R	#3-3	
BANNER	006066 R	26-1620 #27-1702	
BEGIN	000000 R	#3-3 13-512 13-515 13-518 13-527 14-565 14-567 14-569 14-571	
		14-586 14-608 14-618 15-649 15-708 15-720 15-729 15-741	
		15-746 15-758 15-763 15-775 15-780 15-792 15-834 15-837 16-874	
		16-875 16-881 16-883 16-887 16-891 16-895 16-897 16-898 16-899	
		16-904 16-905 18-1084 18-1114 19-1132 19-1134 20-1227 20-1229 20-1233	
		20-1237 20-1239 25-1439 25-1442 26-1541 26-1542 26-1570 26-1587 26 1612	
		26-1613 26-1614 26-1615 26-1616 26-1617 26-1618 26-1619 26-1620	
BIT0	000001	#3-3 6-213	
BIT00	000001	#5-131	
BIT01	000002	3-20 #5-132	
BIT02	000004	3-21 #5-133	
BIT03	000010	3-22 #5-134	
BIT04	000020	#5-135	
BIT05	000040	#5-136	
BIT06	000100	#5-137	
BIT07	000200	#5-138 15-669 15-671	
BIT08	000400	#5-139	
BIT09	001000	3-23 #5-140	
BIT1	000002	#3-3	
BIT10	002000	#3-3 3-24 #5-141	
BIT11	004000	#3-3 3-25 #5-142	
BIT12	010000	#3-3 #5-143	
BIT13	020000	#3-3 #5-144	
BIT14	040000	#3-3 #5-145 7-218	
BIT15	100000	#3-3 #5-146 7-217 15-663 15-668	
BIT2	000004	#3-3	
BIT3	000010	#3-3	
BIT4	000020	#3-3	

SYMBOL	VALUE	REFERENCES	CREF
BITS	= 000040	#3-3	
BIT6	= 000100	#3-3	
BIT7	= 000200	#3-3	
BIT8	= 000400	#3-3	
BIT9	= 0J1000	#3-3	
BREAK#	= 104407	#3-3	15-649 15-649
BR1	000012 R	#3-3	15-831
BR2	000013 R	#3-3	
BTOD#	= 104421	#3-3	14-565 14-567 14-569 20-1227
CAPRES	= 000004	#3-3	
CDATA#	= 104412	#3-3	18-1084 18-1114
CDERCT	000144 R	#3-3	*13-520 14-569
CDMDCT	000146 R	#3-3	
CF.AVL	= 000200	#10-302	24-1393
CF.MSC	= 000100	#10-303	
CF.OTH	= 000040	#10-304	
CF.SMD	= 000002	#10-306	
CF.TMS	= 000020	#10-305	
CF.S76	= 000001	#10-307	
CINTR	000254 R	#3-31	
CKING#	= 000001	#3-3	
CLKPTE	= 000001	#3-3	
CLKSP#	= 104422	#3-3	
CLRPAK	005344 R	21-1257 21-1277 22-1307 22-1325 23-1345 23-1358 23-1371 24-1389 24-1406	
		#26-1538	
CMDFEF	000270 R	#3-36 *13-526 15-688 *17-926 *25-1430 25-1432	
CMPPACK	000402 R	#3-55 15-762 *21-1258 *21-1259 *21-1260 *21-1278 *21-1279 *21-1280 *21-1287	
		*21-1290 *22-1308 *22-1310 *22-1311 *22-1326 *22-1328 *22-1329 *22-1333 *22-1334	
		*22-1335 *23-1346 *23-1359 *23-1372 *23-1373 *24-1392 *24-1393 *24-1409 *25-1432	
		*25-1433 25-1487 25-1495 25-1497 25-1499 26-1616 26-1617 26-1618 26-1619	
		*21-1289 *22-1331 *23-1348 *23-1375 *24-1390	
CMPLN	000376 R	#3-53	
CMVJR	000400 R	#3-54	
COMMNO	000264 R	#3-34	*15-696 *15-697 *25-1434 *25-1435
CONFIG	000056 R	#3-3	15-711 15-734 15-751 15-768 15-785
CPAKEA	000464 R	#3-57	*15-764
CPAKEP	000470 R	#3-59	15-697 *15-773 *15-777
CPAKPA	000462 R	#3-56	*15-765
CPAKPP	000466 R	#3-58	15-696 *15-770 *15-776
CSRA	000100 R	#3-3	*26-1600
CVTADR	002032 R	14-550	#15-728
CYCLED	003312 ?	14-597	#18-1017 18-1047
CYCLEL	003412 ?	14-617	#18-1109
DAP	004512 R	#23-1345	
DATCK#	= 104411	#3-3	
DATER#	= 104404	#3-3	
DOINTR	003574 R	#18-1096	
DROP1	004100 R	14-599	#20-1212
DROP2	004110 R	17-941	#20-1215
DROP3	004120 R	17-959	#20-1218
DROP4	004126 R	20-1214	20-1217 #20-1220

XDU8E0		CREATED BY M 70 ON 27-SEP-85 AT 16:24		PAGE 4	
SYMBOL		CROSS REFERENCE		CREF 04.00	
SYMBOL	VALUE	REFERENCES			
ICOUNT	000040 R	#3-3			
IDNUM	000122 R	#3-3			
IMODX.	- 000000	#3-3	15-708		
INDPAR	- 000040	#3-3			
INIT	000030 R	#3-3			
INITER	005700 R	16-899	#27-1629		
INITE1	005672 R	16-887	#27-1625		
INITE2	005716 R	16-891	#27-1637		
INITE3	005724 R	16-895	#27-1641		
INITUD	001412 R	14-557	#15-642	26-1545	
INTA	002562 R	15-830	#15-836		
INTERP	005020 R	18-1036	18-1099	#25-1438	25-1524
INTR	000120 R	#3-3			
KTPRES	- 000400	#3-3			
KTXTND	- 040000	#3-3			
LIMIT	000626 R	#4-111	21-1281	21-1283	
LOOP1	001276 R	14-584	#14-588	14-609	
LOOP2	001312 R	#14-591	14-606		
L.CHVR	- 000015	#12-418			
L.CNTI	- 000014	#12-416	#12-417		
L.CYL	- 000034	#12-423			
L.DATA	- 000050	#12-428			
L.ERLC	- 000030	#12-422			
L.EVNT	- 000000	#12-414			
L.GRP	- 000040	#12-424			
L.SCTR	- 000042	#12-426			
L.SLOT	- 000002	#12-415			
L.TRCK	- 000041	#12-425			
L.UHVR	- 000027	#12-421			
L.UNTI	- 000016	#12-419			
L.USVR	- 000026	#12-420			
L.VSER	- 000044	#12-427			
MAITP	004356 R	21-1262	21-1284	#21-1286	
MAITR	004252 R	18-1111	#21-1257		
MAITH	004306 R	18-1110	#21-1277		
MAP22	- 104416	#3-3	15-720	15-741	15-758
MA10NC	001774 R	14-576	#14-616	14-619	15-775
MD.CMP	- 040000	#9-264			
MD.ERR	- 010000	#9-266			
MD.EXP	- 100000	#9-265			
MD.FEU	- 000001	#9-275			
MD.NXU	- 000001	#9-277	23-1373		
MD.SCH	- 004000	#9-267			
MD.SCL	- 002000	#9-268			
MD.SEC	- 001000	#9-269			
MD.SER	- 000400	#9-270			
MD.SPD	- 000001	#9-274			
MD.SSH	- 000200	#9-271			
MD.VOL	- 000002	#9-276			
MD.WBN	- 000100	#9-272			

XDUCEO		CREATED BY	MACRO	ON	27-SEP-85	AT	16:24	PAGE	5	CREF	04.00
SYMBOL	CROSS REFERENCE	VALUE	REFERENCES								
MO.MBV	= 000040		#9-273								
MOONAM	000000 R		#3-3								
MOOSP	000252 R		3-3	#3-3							
MSGD1	007422 R		27-1650	#28-1750							
MSGD2	007466 R		27-1658	#28-1751							
MSGD3	007530 R		27-1666	#28-1752							
MSGN1	= 104403		#3-3	13-512	13-515	13-518	14-571	16-874	16-887	16-891	16-895
			16-899	16-904	20-1233	20-1237	20-1239	26-1542	26-1620		
MSG51	= 104402		#3-3								
MSG1	= 104401		#3-3								
MSG1	006132 R		27-1675	27-1718	#28-1724						
MSG10	006365 R		27-1631	#28-1733							
MSG11	006377 R		27-1669	#28-1734							
MSG12	006423 R		27-1671	#28-1735							
MSG13	006461 R		27-1673	#28-1736							
MSG14	006513 R		27-1633	#28-1737							
MSG16	006524 R		27-1683	#28-1738							
MSG17	006575 R		27-1678	#28-1739							
MSG18	006632 R		27-1680	#28-1740							
MSG19	006705 R		27-1699	#28-1741							
MSG2	006134 R		27-1625	27-1637	27-1641	#28-1725					
MSG20	006735 R		27-1648	27-1656	27-1664	#28-1742					
MSG21	006760 R		27-1702	#28-1743							
MSG23	007052 R		27-1704	27-1706	27-1708	27-1710	27-1712	27-1714	27-1716	#28-1744	
MSG3	006165 R		27-1629	#28-1726							
MSG36	007054 R		27-1693	#28-1745							
MSG37	007123 R		27-1690	#28-1746							
MSG4	006205 R		27-1626	#28-1727							
MSG40	007246 R		27-1687	#28-1748							
MSG5	006232 R		27-1638	#28-1728							
MSG6	006250 R		27-1642	#28-1729							
MSG7	006305 R		27-1696	#28-1730							
MSG8	006342 R		27-1645	27-1657	27-1661	#28-1751					
MSG9	006352 R		27-1647	27-1655	27-1663	#28-1752					
NCPUDP	= 000020		#3-3								
NOAPTY	= 000002		#3-3								
NTRUPT	005024 R		14-572	15-701	#25-1441						
NULL	= 000000		#3-3	16-898	26-1570	26-1587					
NUM	000514 R		#3-72	#16-880	16-881	#16-882	16-883	#20-1213	#20-1216	#20-1219	#20-1231
			#20-1235	#26-1539	26-1541						
OLDEA	000520 R		#3-74	#13-533	14-553	#14-556					
OLDPA	000516 R		#3-73	#13-532	14-551	#14-555					
ONLINE	004722 R		18-1041	#24-1406							
OPEN	= 000000		3-3	3-3	3-3	3-3	3-3	3-3	3-3	3-3	3-3
			3-3	3-3	3-3	3-3	3-3	3-3	3-3	3-3	3-3
			3-3	3-3	3-3	3-3	3-3	3-3	3-3	3-3	3-3
			3-3	#3-3							
OP.ABC	= 000001		#8-234								
OP.ACC	= 000020		#8-235								
OP.ACP	= 000102		#8-236	25-1448							

SYMBOL	VALUE	REFERENCES	CREF
PRTY7	= 000340	03-3	
PS	= 177776	03-3	
PSW	= 177776	03-3	
PUSH	= 005746	03-3	
PUSH2	= 024646	03-3	
PWRFLG	= 000002	03-3	
P.ADEA	= 000022	010-319 *21-1259 *21-1279 *22-1311 *22-1328 26-1618	
P.ADPA	= 000020	010-318 *21-1260 *21-1280 *22-1310 *22-1329 26-1619	
P.BCNT	= 000014	010-316 *11-356 *21-1287 *22-1333 26-1615	
P.BUFF	= 000020	010-317	
P.CHST	= 000020	011-362	
P.CNCL	= 000022	011-391	
P.CNT	= 000006	011-400	
P.CNTF	= 000016	010-339 *11 389 *24-1393	
P.CNTI	= 000024	011-392	
P.CPSP	= 000042	010-332	
P.CRF	= 000000	010-312 *11-351 *11-398 *25-1432	
P.CTMO	= 000020	011-390	
P.CYL	= 000050	011-373	
P.ELGF	= 000034	010-330	
P.FBBK	= 000034	011-357	
P.FLGS	= 000011	011-354 *11-402	
P.GRP	= 000046	011-372	
P.HSTI	= 000020	010-328 *11-367 *11-381	
P.HTMO	= 000020	010-340	
P.LBN	= 000034	010-320 *22-1334 *22-1335 26-1616 26-1617	
P.LGDT	= 000014	011-404	
P.MEDI	= 000034	011-393	
P.MLUN	= 000014	011-365 *11-379	
P.MOD	= 000012	010 315 *23-1373	
P.OPCD	= 000010	010-314 *11-353 *11-401 *21-1258 *21-1278 *22-1308 *22-1326 *23-1346 *23 1359	
		*23-1372 *24-1392 *24-1409 25-1446 25-1448 25 1487 25-1495 25-1497 25 1499	
		26-1613	
P.OTRF	= 000014	010-324 *11-361	
P.RBN	= 000014	010-335	
P.RBNS	= 000056	011-375	
P.RCTC	= 000057	011-376	
P.RCTS	= 000054	011-374	
P.RGID	= 000034	010-345 *21-1290	
P.RGOF	= 000040	010-346	
P.SFTW	= 000040	010-321 *11-356	
P.SHST	= 000042	70 *11-394	
P.SHUN	= 000040	531 *11-369 *11-383 25-1452 25 1461 25 1463 26-1601 26-1612	
P.STS	= 000012	*11-355 17-987 17-991	
P.SZOF	= 000012	011-403	
P.TIME	= 000024	010-342	
P.TRCK	= 000044	011-371	
P.UNFL	= 000016	010-327 *11-366 *11-380	
P.UNIT	= 000004	010-313 *11-352 *11-399 17-943 *25-1433 26 1614	
P.UNSZ	= 000044	011-384 18-1043 18-1044	

XDUBEO SYMBOL	CROSS REFERENCE	REFERENCES	MACRO	ON 27 SEP 85 AT 16:24	PAGE 8	CREF 04.00
P. UN14	= 000024	010-329		011-368	011-382	
P. USEF	= 000022	010-341				
P. VRSN	= 000014	010-338		011-388		
P. VSER	= 000050	011-385				
QMON22	= 000010	03-3		15-709	15-732	15-749 15-766 15-783
RANACC	003654 R	019-1131				
RAND8	= 104417	03-3		13-527	19-1132	19-1134
RANNUM	000054 R	03-3		13-528	19-1133	19-1135
RBUF	007616 R	3-3		028-1754		
RBUFEA	000130 R	03-3		015-747		
RBUFEF	000506 R	03-68		015-756	015-760	21-1259 22-1328
RBUFPA	000126 R	03-3		015-748	18-1084	18-1114
RBUFPP	000504 R	03-67		015-753	015-759	21-1260 22-1329
RBUFSZ	000132 R	03-3		21-1261	22-1327	
RBUFVA	000124 R	03-3		15-745		
READ	004462 R	18-1079		022-1325		
READA	004514 R	22-1312		022-1330		
RESTR	001066 R	3-3		014-549		
RESTR1	001136 R	14-554		014-559		
RESTR2	001112 R	14-552		014-555		
RES1	000056 R	03-3				
RES2	000060 R	03-3		15-709	15-732	15-749 15-766 15-783
RG.FLG	= 040000	07-218		18-1035	18-1098	25-1434 25-1436 25-1523
RG.OVN	= 100000	07-217		18-1035	18-1098	25-1435 25-1436 25-1523
RH70	= 001000	03-3				
RINTR	000256 R	03-32		025-1445		
RLIM	= 000000	04-121		16-901		
RPAKEA	000004 R	03-49		015-781		
RPAKEP	000366 R	03-51		15-700	015-790	015-794
RPAKPA	000372 R	03-48		015-782		
RPAKPP	000306 R	03-50		15-699	015-787	015-793
RSPACK	000306 R	03-47		7-229	15-77	17-987 17-991 18-1043 19-1044 25-1446
		25-1448		25-1452	15-77	17-943 17-943 26-1601 26-1612 26-1613 26-1614 26-1615
RSPEA	000274 R	03-41		015-730		
RSPEP	000300 R	03-43		14-553	14-556	15-681 015-739 015-743
RSPLEN	000302 R	03-45		021-1288	022-1330	023-1347 023-1360 023-1374 024-1391 024-1407 26-1552
RSPONC	000260 R	03-33		15-683	015-699	015-700 15-728 018-1035 018-1098 025-1436 025-1523
RSPPA	000272 R	03-40		015-731		
RSPPP	000276 R	03-42		14-551	14-555	15-673 015-736 015-742
RSPVIR	000304 R	03-46		021-1292		
RSTR	000112 R	03-3				
R6	= 0000006	03-3				
R7	= 0000007	03-3				
SANOTO	006024 R	26-1542		027-1678		
SAREG	000252 R	03-27		013-530	013-531	15-646 15-695 15-832 15-839 16-882 26-1539
SA.CMD	= 034000	06-185				
SA.CME	= 000360	06-200				
SA.DIA	= 000400	06-176		15-655	15-656	
SA.ERC	= 003777	06-170				

XDU8EO		CREATED BY	MACRO	ON 27-SEP-85	AT 16:24	PAGE 9	CREF 04.00						
SYMBOL	CROSS REFERENCE	VALUE	REFERENCES										
SA.ERR		100000	06-166	15-647	15-840								
SA.GO		000001	06-213	15-694									
SA.INE		000200	06-190										
SA.INT		000200	06-183	15-663									
SA.LFC		040000	06-205										
SA.MAP		000100	06-177										
SA.MCV		000377	06-209										
SA.NSI		002000	06-174										
SA.PRG		000001	06-195										
SA.Q22		001000	06-175										
SA.RSE		000017	06-199										
SA.RSP		003400	06-184										
SA.SM		000040	06-178										
SA.S1		004000	06-162	15-653	15-655	15-656	15-666						
SA.S2		010000	06-163	15-671									
SA.S3		020000	06-164										
SA.S4		040000	06-165										
SA.VCE		000177	06-189										
SA.VEC		000177	06-182										
SBADR		000102 R	03-3										
SCC		004664 R	17-925	024-1388	26-1548								
SC.DIS		000400	012-454	17-987	17-991								
SC.DUP		000200	012-456	17-987	17-991								
SC.INV		000100	012-461	25-1463									
SC.IOP		000100	012-453	17-987	17-991								
SC.NVL		000040	012-451	17-987	17-991								
SC.STO		000040	012-460	25-1461									
SECH		000620 R	04-106	013-529	019-1141	019-1157	19-1158	019-1161	019-1166	019-1187	19-1189		
			019-1192	22-1335									
SECL		000616 R	04-105	013-528	019-1174	019-1181	19-1183	019-1191	22-1334				
SEND		004750 R	21-1293	22-1336	23-1349	23-1376	24-1395	025-1430	25-1431				
SEQACC		004032 R	19-1130	019-1180									
SETTAB		005536 R	26-1569	26-1586	026-1599								
SETUP		003010 R	14-582	017-923									
SNDSTP		002536 R	15-667	15-674	15-682	015-829							
SOF CNT		000042 R	03-3	14-565	026-1583								
SOFER		104406	03-3	26-1587									
SOFPAS		000046 R	03-3										
SPOINT		000032 R	03-3										
SPSIZ		000040	02-28	3-3									
SR.CMP		004000	03-25	18-1081	18-1112								
SR.DUA		001000	03-23	18-1018	18-1074	18-1085	25-1459						
SR.REP		000004	03-21	26-1564	26-1581								
SR.SEG		002000	03-24	19-1129									
SR.SUM		000010	03-22	14-560									
SR.XFR		000002	03-20	13-511	13-513	14-575							
SR1		000016 R	03-3	013-511	13-513	14-560	14-575	18-1018	18-1074	18-1081	18-1085		
			18-1112	19-1129	25-1459	26-1581							
SR2		000020 R	03-3	15-690									
SR3		000022 R	03-3										

