

FEPOM

CTAL BRIDGE LINK  
CZFPBA0

COPYRIGHT (c) 1982-84  
AH-T861A-MC  
FICHE 1 OF 1

APR 1984  
digital  
Made In USA

Grid of microfiche frames containing technical data, including tables and diagrams.

BASE 1 1001 1001  
BASE 1 1001 1001  
BASE 1 1001 1001

.REM 6

IDENTIFICATION  
-----

PRODUCT CODE: AC-T860A-MC  
PRODUCT NAME: CZFPBA0 CTRL BRIDGE LINK  
PRODUCT DATE: MAY 1982  
MAINTAINER: CSS GNG DIAGNOSTIC ENGINEERING  
AUTHOR: DALE PROCTOR

COPYRIGHT (C) 1982,1984  
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

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

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL  
DEC  
VAX

PDP  
DECUS  
VMS

UNIBUS  
DECTAPE  
Q-BUS

MASSBUS  
LSI

## MODIFICATION HISTORY:

CZFPB-A.0      DALE PROCTOR      22-DEC-1983  
CHANGED THE NAME TO CZFPB-A0 AND RELEASED TO  
SDC. ALSO UPDATED ALL THE DOCUMENTATION, AND  
REMOVED THE HELP FILE QUESTIONS FROM THE  
INITIALIZATION SECTION.

Z0792-B.0      PAUL HOLSINGER  
MODIFIED TO RUN WITH THE 11/24 BOARD

Z0792-B.1      DALE PROCTOR      1-JUNE-1983  
CORRECTED TEST 18 TO NOT TIMEOUT SO EARLY  
ON THE WATCHDOG TIMER TEST.

Z0792-B.2      DALE PROCTOR      7-JUNE-1983  
INSERTED A NEW TEST 16 AND 17 TO ALLOW  
CHECKING OF THE VAX BUFFERED DATA PATH.

Z0792-C.0      DALE PROCTOR      15-JUNE-1983  
UPDATED REVISION FOR ARCHIVING

## TABLE OF CONTENTS

1.0	GENERAL INFORMATION
1.1	PROGRAM ABSTRACT
1.2	SYSTEM REQUIREMENTS
1.3	RELATED DOCUMENTS AND STANDARDS
1.4	DIAGNOSTIC HIERARCHY PREREQUISITES
1.5	ASSUMPTIONS
2.0	OPERATING INSTRUCTIONS
2.1	COMMANDS
2.2	SWITCHES
2.3	FLAGS
2.4	HARDWARE QUESTIONS
2.5	SOFTWARE QUESTIONS
2.6	EXTENDED P-TABLE DIALOGUE
2.7	QUICK STARTUP PROCEDURE
3.0	ERROR INFORMATION
4.0	PERFORMANCE AND PROGRESS REPORTS
5.0	DEVICE INFORMATION TABLES
6.0	TEST SUMMARIES

## 1.0 GENERAL INFORMATION

## 1.1 PROGRAM ABSTRACT

THIS DIAGNOSTIC CONTAINS LINK MODE TESTS FOR THE FEPCM ON THE PDP SIDE OF THE BRIDGE. THIS PROGRAM MUST BE RUN IN CONJUNCTION WITH THE LINK MODE TESTS FOR THE VAX SIDE OF THE BRIDGE. THE VAX DIAGNOSTIC CONTROLS THE OPERATION OF THIS DIAGNOSTIC. ONCE THE DIAGNOSTIC HAS BEEN STARTED, NOTHING NEEDS TO BE DONE.

BECAUSE OF THE NATURE OF A LINK DIAGNOSTIC, THE STRUCTURE OF THE DIAGNOSTIC IS A LITTLE DIFFERENT FROM NORMAL. BASICLY, THERE IS A MAINLINE TEST CONTROL MODULE, WHICH DOES THE INITIAL AND FINAL HANDSHAKING WITH THE THE VAX, AND CONTROLS THE SELECTION OF THE TEST. THE BASIC LOGIC DESIGN IS AS FOLLOWS:

```

BEGIN MODULE MAINLINE
: REPEAT
: : INPUT QINT REGISTER
: : UNTIL CMDPRS BIT IS SET
: ENOREPEAT
: READ COMMAND REGISTER
: CALCULATE TEST # FROM DATA FIELD IN QCMD REGISTER
: IF TEST # IS TOO LARGE
: : PRINT ERROR MESSAGE
: : ABORT TEST, RETURN TO SUPERVISOR
: ENOREIF
: SET CMDACK TO HANDSHAKE WITH VAX
: GO PERFORM TEST AS SUBROUTINE
: IF RETURN FROM TEST NOT ABORT CODE
: : SET VNTPRS BIT IN QVNT REGISTER
: : REPEAT
: : : INPUT QINT REGISTER
: : : UNTIL VNTACK SET OR TIMEOUT OCCURS
: : ENOREPEAT
: : IF TIMEOUT OCCURRED
: : : PRINT TIMEOUT MESSAGE
: : ENOREIF
: ENOREIF
: RESTART MODULE (NO NORMAL EXIT OUT OF MODULE)
ENDMODULE MAINLINE

```

THIS DIAGNOSTIC HAS BEEN WRITTEN FOR USE WITH THE DIAGNOSTIC RUNTIME SERVICES SOFTWARE (SUPERVISOR). THESE SERVICES PROVIDE THE INTERFACE TO THE OPERATOR AND TO THE SOFTWARE ENVIRONMENT. THIS PROGRAM CAN BE USED WITH XXDP+, ACT, APT, SLIDE AND PAPER TAPE. FOR A COMPLETE DESCRIPTION OF THE RUNTIME SERVICES, REFER TO THE XXDP+ USER'S MANUAL. THERE IS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES IN SECTION 2 OF THIS DOCUMENT.

1.2 SYSTEM REQUIREMENTS

LSI OR PDP CPU WITH AT LEAST 16K OF MEMORY, (PREFREABLY MORE FOR CONVERSATION MODE DMA TEST), CONSOLE DEVICE AND TU-58 OR OTHER LOAD MEDIUM. NORMAL CONFIGURATION FOR THE CONSOLE DEVICE IS THROUGH THE VAX AS A VIRTUAL TERMINAL.

1.3 RELATED DOCUMENTS AND STANDARDS

FEPCH TECHNICAL MANUAL  
 AC-T858A-MC CZFPAAO CTRL BRIDGE INTFC  
 AC-T866A-MC CXFPCA0 CTRL BRIDGE DEC/X11  
 EVDTE COMMAND BRIDGE INTERFACE DIAGNOSTIC. (VAX DIAGNOSTIC)  
 EVDTC COMMAND BRIDGE LINK DIAGNOSTIC. (VAX DIAGNOSTIC)  
 XXDP+ USERS MANUAL - CHQUS??.

1.4 DIAGNOSTIC HIERARCY PREREQUISITES

THE CPU, MEMORY, MEMORY MANAGEMENT, CONSOLE DEVICE AND LOAD MEDIUM ARE ASSUMED TO BE FUNCTIONING PRIOR TO RUNNING THIS TEST. ALSO, THE STAND-ALONE DIAGNOSTICS ON BOTH SIDES ARE ASSUMED TO BE RUNNING SUCCESSFULLY.

1.5 ASSUMPTIONS

2.0 OPERATING INSTRUCTIONS

THIS SECTION CONTAINS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES. FOR DETAILED INFORMATION, REFER TO THE XXDP+ USER'S MANUAL (CHQUS).

2.1 COMMANDS

THERE ARE ELEVEN LEGAL COMMANDS FOR THE DIAGNOSTIC RUNTIME SERVICES (SUPERVISOR). THIS SECTION LISTS THE COMMANDS AND GIVES A VERY BRIEF DESCRIPTION OF THEM. THE XXDP+ USER'S MANUAL HAS MORE DETAILS.

COMMAND	EFFECT
START	START THE DIAGNOSTIC FROM AN INITIAL STATE
RESTART	START THE DIAGNOSTIC WITHOUT INITIALIZING
CONTINUE	CONTINUE AT TEST THAT WAS INTERRUPTED (AFTER ↑C)
PROCEED	CONTINUE FROM AN ERROR HALT
EXIT	RETURN TO XXDP+ MONITOR (XXDP+ OPERATION ONLY!)
ADD	ACTIVATE A UNIT FOR TESTING (ALL UNITS ARE CONSIDERED TO BE ACTIVE AT START TIME)
DROP	DEACTIVATE A UNIT
PRINT	PRINT STATISTICAL INFORMATION (IF IMPLEMENTED BY THE DIAGNOSTIC - SECTION 4.0)
DISPLAY	TYPE A LIST OF ALL DEVICE INFORMATION
FLAGS	TYPE THE STATE OF ALL FLAGS (SEE SECTION 2.3)
ZFLAGS	CLEAR ALL FLAGS (SEE SECTION 2.3)

A COMMAND CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. SO YOU MAY, FOR EXAMPLE, TYPE "STA" INSTEAD OF "START".

2.2 SWITCHES

THERE ARE SEVERAL SWITCHES WHICH ARE USED TO MODIFY SUPERVISOR OPERATION. THESE SWITCHES ARE APPENDED TO THE LEGAL COMMANDS. ALL OF THE LEGAL SWITCHES ARE TABULATED BELOW WITH A BRIEF DESCRIPTION OF EACH. IN THE DESCRIPTIONS BELOW, A DECIMAL NUMBER IS DESIGNATED BY "DDDDD".

\*\* FOR THIS DIAGNOSTIC ONLY!!! - THE SWITCHES MARKED WITH A \*\* SHOULD NOT BE USED.

SWITCH	EFFECT
** /TESTS:LIST	EXECUTE ONLY THOSE TESTS SPECIFIED IN THE LIST. LIST IS A STRING OF TEST NUMBERS, FOR EXAMPLE - /TESTS:1:5:7-10. THIS LIST WILL CAUSE TESTS 1,5,7,8,9,10 TO BE RUN. ALL OTHER TESTS WILL NOT BE RUN.
** /PASS:DDDDD /FLAGS:FLGS	EXECUTE DDDDD PASSES (DDDDD = 1 TO 64000) SET SPECIFIED FLAGS. FLAGS ARE DESCRIBED IN SECTION 2.3.
** /EOP:DDDDD	REPORT END OF PASS MESSAGE AFTER EVERY DDDDD PASSES ONLY. (DDDDD = 1 TO 64000)
** /UNITS:LIST	TEST/ADD/DROP ONLY THOSE UNITS SPECIFIED IN THE LIST. LIST EXAMPLE - /UNITS:0:5:10-12 USE UNITS 0,5,10,11,12 (UNIT NUMBERS = 0-63)

EXAMPLE OF SWITCH USAGE:

START/TESTS:1-5/PASS:1000/EOP:100

THE EFFECT OF THIS COMMAND WILL BE: 1) TESTS 1 THROUGH 5 WILL BE EXECUTED, 2) ALL UNITS WILL TESTED 1000 TIMES AND 3) THE END OF PASS MESSAGES WILL BE PRINTED AFTER EACH 100 PASSES ONLY. A SWITCH CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. YOU MAY, FOR EXAMPLE, TYPE "/TES:1-5" INSTEAD OF "/TESTS:1-5".

BELOW IS A TABLE THAT SPECIFIES WHICH SWITCHES CAN BE USED BY EACH COMMAND.

	TESTS	PASS	FLAGS	EOP	UNITS
START	X	X	X	X	X
RESTART	X	X	X	X	X
CONTINUE		X	X	X	
PROCEED			X		
DROP					X
ADD					X
PRINT					
DISPLAY					X
FLAGS					
ZFLAGS					
EXIT					

### 2.3 FLAGS

FLAGS ARE USED TO SET UP CERTAIN OPERATIONAL PARAMETERS SUCH AS LOOPING ON ERROR. ALL FLAGS ARE CLEARED AT STARTUP AND REMAIN CLEARED UNTIL EXPLICITLY SET USING THE FLAGS SWITCH. FLAGS

ARE ALSO CLEARED AFTER A START COMMAND UNLESS SET USING THE FLAG SWITCH. THE ZFLAGS COMMAND MAY ALSO BE USED TO CLEAR ALL FLAGS. WITH THE EXCEPTION OF THE START AND ZFLAGS COMMANDS, NO COMMANDS AFFECT THE STATE OF THE FLAGS; THEY REMAIN SET OR CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

FLAG	EFFECT
HOE	HALT ON ERROR - CONTROL IS RETURNED TO RUNTIME SERVICES COMMAND MODE
LOE	LOOP ON ERROR
IER*	INHIBIT ALL ERROR REPORTS
IBE*	INHIBIT ALL ERROR REPORTS EXCEPT FIRST LEVEL (FIRST LEVEL CONTAINS ERROR TYPE, NUMBER, PC, TEST AND UNIT)
IXE*	INHIBIT EXTENDED ERROR REPORTS (THOSE CALLED BY PRINTX MACRO'S)
PRI	DIRECT MESSAGES TO LINE PRINTER
PNT	PRINT TEST NUMBER AS TEST EXECUTES
BOE	"BELL" ON ERROR
UAM	UNATTENDED MODE (NO MANUAL INTERVENTION)
ISR	INHIBIT STATISTICAL REPORTS (DOES NOT APPLY TO DIAGNOSTICS WHICH DO NOT SUPPORT STATISTICAL REPORTING)
IDR	INHIBIT PROGRAM DROPPING OF UNITS
ADR	EXECUTE AUTODROP CODE
LOT	LOOP ON TEST
EVL	EXECUTE EVALUATION (ON DIAGNOSTICS WHICH HAVE EVALUATION SUPPORT)

\*ERROR MESSAGES ARE DESCRIBED IN SECTION 3.1

SEE THE XXDP\* USER'S MANUAL FOR MORE DETAILS ON FLAGS. YOU MAY SPECIFY MORE THAN ONE FLAG WITH THE FLAG SWITCH. FOR EXAMPLE, TO CAUSE THE PROGRAM TO LOOP ON ERROR, INHIBIT ERROR REPORTS AND TYPE A "BELL" ON ERROR, YOU MAY USE THE FOLLOWING STRING:

/FLAGS:LOE:IER:BOE

#### 2.4 HARDWARE QUESTIONS

WHEN A DIAGNOSTIC IS STARTED, THE RUNTIME SERVICES WILL PROMPT THE USER FOR HARDWARE INFORMATION BY TYPING "CHANGE HW (L) ?" YOU MUST ANSWER "Y" AFTER A START COMMAND UNLESS THE HARDWARE INFORMATION HAS BEEN "PRELOADED" USING THE SETUP UTILITY (SEE CHAPTER 6 OF THE XXDP\* USER'S MANUAL). WHEN YOU ANSWER THIS QUESTION WITH A "Y", THE RUNTIME SERVICES WILL ASK FOR THE NUMBER OF UNITS (IN DECIMAL). YOU WILL THEN BE ASKED THE FOLLOWING QUESTIONS FOR EACH UNIT.

LSI BRIDGE MODULE BASE ADDRESS?  
ADDRESS OF Q-COMMAND REGISTER

DEFAULT=160500



LSI BRIDGE MODULE VECTOR                    DEFAULT=500  
ADDRESS OF INTERRUPT VECTOR

THE DEFAULTS WILL AUTOMATICALLY BE USED IF YOU ANSWER NO  
TO CHANGE HW?

## 2.5 SOFTWARE QUESTIONS

AFTER YOU HAVE ANSWERED THE HARDWARE QUESTIONS OR AFTER A RESTART  
OR CONTINUE COMMAND, THE RUNTIME SERVICES WILL ASK FOR SOFTWARE  
PARAMETERS. THESE PARAMETERS WILL GOVERN SOME DIAGNOSTIC SPECIFIC  
OPERATION MODES. YOU WILL BE PROMPTED BY "CHANGE SW (L) ?"  
IF YOU WISH TO CHANGE ANY PARAMETERS, ANSWER BY TYPING  
"Y". THE SOFTWARE QUESTIONS AND THE DEFAULT VALUES ARE DESCRIBED  
IN THE NEXT PARAGRAPH(S).

THERE ARE NO SOFTWARE QUESTIONS.

## 2.6 EXTENDED P-TABLE DIALOGUE

WHEN YOU ANSWER THE HARDWARE QUESTIONS, YOU ARE BUILDING ENTRIES  
IN A TABLE THAT DESCRIBES THE DEVICES UNDER TEST. THE SIMPLEST  
WAY TO BUILD THIS TABLE IS TO ANSWER ALL QUESTIONS FOR EACH  
UNIT TO BE TESTED. IF YOU HAVE A MULTIPLEXED DEVICE SUCH AS  
A MASS STORAGE CONTROLLER WITH SEVERAL DRIVES OR A COMMUNICATION  
DEVICE WITH SEVERAL LINES, THIS BECOMES TEDIOUS SINCE MOST OF  
THE ANSWERS ARE REPETITIOUS.

TO ILLUSTRATE A MORE EFFICIENT METHOD, SUPPOSE YOU ARE TESTING  
A FICTIONAL DEVICE, THE XY11. SUPPOSE THIS DEVICE CONSISTS OF  
A CONTROL MODULE WITH EIGHT UNITS (SUB-DEVICES) ATTACHED TO IT.  
THESE UNITS ARE DESCRIBED BY THE OCTAL NUMBERS 0 THROUGH 7. THERE  
IS ONE HARDWARE PARAMETER THAT CAN VARY AMONG UNITS CALLED THE  
Q-FACTOR. THIS Q-FACTOR MAY BE 0 OR 1. BELOW IS A SIMPLE WAY  
TO BUILD A TABLE FOR ONE XY11 WITH EIGHT UNITS.

♦ UNITS (D) ? 8<CR>

UNIT 1  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 0<CR>  
Q-FACTOR (0) 0 ? 1<CR>

UNIT 2  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 1<CR>  
Q-FACTOR (0) 1 ? 0<CR>

UNIT 3  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 2<CR>  
Q-FACTOR (0) 0 ? <CR>

UNIT 4  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 3<CR>  
Q-FACTOR (0) 0 ? <CR>

```
UNIT 5
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 4<CR>
Q-FACTOR (0) 0 ? <CR>
```

```
UNIT 6
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 5<CR>
Q-FACTOR (0) 0 ? <CR>
```

```
UNIT 7
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 6<CR>
Q-FACTOR (0) 0 ? 1<CR>
```

```
UNIT 8
CSR ADDRESS (0) 160000<CR>
SUB-DEVICE # (0) ? 7<CR>
Q-FACTOR (0) 1 ? <CR>
```

NOTICE THAT THE DEFAULT VALUE FOR THE Q-FACTOR CHANGES WHEN A NON-DEFAULT RESPONSE IS GIVEN. BE CAREFUL WHEN SPECIFYING MULTIPLE UNITS!

AS YOU CAN SEE FROM THE ABOVE EXAMPLE, THE HARDWARE PARAMETERS DO NOT VARY SIGNIFICANTLY FROM UNIT TO UNIT. THE PROCEDURE SHOWN IS NOT VERY EFFICIENT.

THE RUNTIME SERVICES CAN TAKE MULTIPLE UNIT SPECIFICATIONS HOWEVER. LET'S BUILD THE SAME TABLE USING THE MULTIPLE SPECIFICATION FEATURE.

```
# UNITS (0) ? 8<CR>
```

```
UNIT 1
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 0,1<CR>
Q-FACTOR (0) 0 ? 1,0<CR>
```

```
UNIT 3
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 2-5<CR>
Q-FACTOR (0) 0 ? 0<CR>
```

```
UNIT 7
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 6,7<CR>
Q-FACTOR (0) 0 ? 1<CR>
```

AS YOU CAN SEE IN THE ABOVE DIALOGUE, THE RUNTIME SERVICES WILL BUILD AS MANY ENTRIES AS IT CAN WITH THE INFORMATION GIVEN IN ANY ONE PASS THROUGH THE QUESTIONS. IN THE FIRST PASS, TWO ENTRIES ARE BUILT SINCE TWO SUB-DEVICES AND Q-FACTORS WERE SPECIFIED. THE SERVICES ASSUME THAT THE CSR ADDRESS IS 160000 FOR BOTH SINCE IT WAS SPECIFIED ONLY ONCE. IN THE SECOND PASS, FOUR ENTRIES WERE BUILT. THIS IS BECAUSE FOUR SUB-DEVICES WERE SPECIFIED. THE

"-" CONSTRUCT TELLS THE RUNTIME SERVICES TO INCREMENT THE DATA FROM THE FIRST NUMBER TO THE SECOND. IN THIS CASE, SUB-DEVICES 2, 3, 4 AND 5 WERE SPECIFIED. (IF THE SUB-DEVICE WERE SPECIFIED BY ADDRESSES, THE INCREMENT WOULD BE BY 2 SINCE ADDRESSES MUST BE ON AN EVEN BOUNDARY.) THE CSR ADDRESSES AND Q-FACTORS FOR THE FOUR ENTRIES ARE ASSUMED TO BE 160000 AND 0 RESPECTIVELY SINCE THEY WERE ONLY SPECIFIED ONCE. THE LAST TWO UNITS ARE SPECIFIED IN THE THIRD PASS.

THE WHOLE PROCESS COULD HAVE BEEN ACCOMPLISHED IN ONE PASS AS SHOWN BELOW.

♦ UNITS (0) ? 8<CR>

UNIT 1

CSR ADDRESS (0) ? 160000<CR>

SUB-DEVICE ♦ (0) ? 0-7<CR>

Q-FACTOR (0) 0 ? 0,1,0,...,1,1<CR>

AS YOU CAN SEE FROM THIS EXAMPLE, NULL REPLIES (COMMAS ENCLOSING A NULL FIELD) TELL THE RUNTIME SERVICES TO REPEAT THE LAST REPLY.

## 2.7 QUICK START-UP PROCEDURE (XXDP+)

TO START-UP THIS PROGRAM:

1. BOOT XXDP+
2. GIVE THE DATE AND ANSWER THE LSI AND 50HZ (IF THERE IS A CLOCK) QUESTIONS
3. TYPE "R NAME", WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM
4. TYPE "START"
5. ANSWER THE "CHANGE HW" QUESTION WITH "Y"
6. ANSWER ALL THE HARDWARE QUESTIONS

WHEN YOU FOLLOW THIS PROCEDURE YOU WILL BE USING ONLY THE DEFAULTS FOR FLAGS AND SOFTWARE PARAMETERS. THESE DEFAULTS ARE DESCRIBED IN SECTIONS 2.3 AND 2.5.

## 3.0 ERROR INFORMATION

### 3.1 TYPES OF ERROR MESSAGES

THERE ARE THREE LEVELS OF ERROR MESSAGES THAT MAY BE ISSUED BY A DIAGNOSTIC: GENERAL, BASIC AND EXTENDED. GENERAL ERROR MESSAGES ARE ALWAYS PRINTED UNLESS THE "IER" FLAG IS SET (SECTION 2.3). THE GENERAL ERROR MESSAGE IS OF THE FORM:

```
NAME TYPE NUMBER ON UNIT NUMBER TST NUMBER PC:XXXXXX
ERROR MESSAGE
```

,WHERE; NAME = DIAGNOSTIC NAME

TYPE = ERROR TYPE (SYS FATAL, DEV FATAL, HARD OR SOFT)  
NUMBER = ERROR NUMBER  
UNIT NUMBER = 0 - N (N IS LAST UNIT IN PTABLE)  
TST NUMBER = TEST AND SUBTEST WHERE ERROR OCCURRED  
PC:XXXXXX = ADDRESS OF ERROR MESSAGE CALL

BASIC ERROR MESSAGES ARE MESSAGES THAT CONTAIN SOME ADDITIONAL INFORMATION ABOUT THE ERROR. THESE ARE ALWAYS PRINTED UNLESS THE "IER" OR "IBE" FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL MESSAGE.

EXTENDED ERROR MESSAGES CONTAIN SUPPLEMENTARY ERROR INFORMATION SUCH AS REGISTER CONTENTS OR GOOD/BAD DATA. THESE ARE ALWAYS PRINTED UNLESS THE "IER", "IBE" OR "IXE" FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL ERROR MESSAGE AND ANY ASSOCIATED BASIC ERROR MESSAGES.

### 3.2 SPECIFIC ERROR MESSAGES

MOST OF THE ERROR MESSAGES ARE SELF EXPLANATORY. THE ONLY THING TO WATCH OUT FOR WITH THE ERROR MESSAGES IS IF AN ERROR IS GENERATED ON ONE SIDE, THE OTHER SIDE WILL ALMOST CERTAINLY GENERATE AN ERROR, USUALLY A TIME-OUT. ALSO, THE FIRST LINE OF AN ERROR MESSAGE WILL SAY THAT THE FAULTING TEST IS TEST 1. THIS IS BECAUSE THE SUPERVISOR THINKS IT IS ALWAYS IN TEST 1, AND THE TESTS ARE ACTUALLY SUBROUTINES OF TEST 1. USUALLY THE SECOND OR THIRD LINE PRINTS OUT THE ACTUAL TEST NUMBER.

AN EXAMPLE IS AS FOLLOWS:

```
CZFPBAO HRD ERR 00001 ON UNIT 00 TST 001 SUB 001 PC: 015334
LSI-TEST 04
TIMEOUT WAITING FOR CMDPRS SET
****TEST ABORTED****
```

### 4.0 PERFORMANCE AND PROGRESS REPORTS

THIS DIAGNOSTIC TRICKS THE DIAGNOSTIC SUPERVISOR, BY ALWAYS BEING IN TEST 1 ACCORDING TO THE SUPERVISOR. THE DIFFERENT TESTS ARE ACTUALLY SUBROUTINES OF TEST ONE. AS A RESULT, IT NEVER LEAVES TEST ONE AND WILL NORMALLY NEVER PRINT OUT PERFORMANCE AND PROGRESS REPORTS. IN FACT, IF AN END OF PASS MESSAGE IS PRINTED, SOMETHING WENT WRONG. HOWEVER, THE PROGRAM WILL NORMALLY RESTART.

### 5.0 DEVICE INFORMATION TABLES

P-TABLE DEFINITION-  
WORD 1- CSR ADDRESS (DEFAULT 160500)  
WORD 2- VECTOR ADDRESS (DEFAULT 500)

### 6.0 TEST SUMMARIES

TEST 1: CMD PRS - CMD ACK  
TEST 2: COMMAND REGISTER  
TEST 3: EVENT REGISTER  
TEST 4: VIEW REGISTER READ  
TEST 5: VIEW REGISTER WRITE

TEST 6: VIEW REGISTER COMPLIMENTED READ  
TEST 7: VIEW REGISTER COMPLIMENTED WRITE  
TEST 8: COMMAND INTERRUPTS  
TEST 9: EVENT INTERRUPTS  
TEST 10: VIEW CONTENTION  
TEST 11: PDP-11 NXM INTERRUPT  
TEST 12: DMA 2 WORDS, VAX TO PDP-11  
TEST 13: DMA 2 WORDS, PDP-11 TO VAX  
TEST 14: DMA VAX TO PDP-11  
TEST 15: DMA PDP-11 TO VAX  
TEST 16: DMA VAX (BUFFERED DATA PATH) TO PDP-11  
TEST 17: DMA PDP-11 TO VAX (BUFFERED DATA PATH)  
TEST 18: DMA PDP-11 HI MEMORY TO VAX  
TEST 19: DMA VAX TO PDP-11 (32K BOUNDARY)  
TEST 20: WATCHDOG TIMER  
TEST 21: DMA THROTTLE  
TEST 22: CONVERSATION MODE  
TEST 23: CHECK QHALT (MANUAL)  
TEST 24: CHEK Q POWER FAIL (MANUAL)  
TEST 25: CHECK QBOOT (MANUAL)

ε

```

11          .TITLE CZFPB-A.0 CTRL BRIDGE LINK
12          .SBTTL PROGRAM HEADER
37
39 000000          .ENABL ABS,AMA
40                .DSABL GBL
41                "          2000
43
44 002000          BGNMOD
45
46          ;++
47          ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
48          ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
49          ;--
50
51 002000          POINTER BGNSETUP,BGNRPT
52
69
70 002000          HEADER CZFPB,A,0,0,1
002000          L$NAME::          ;DIAGNOSTIC NAME
002000          103          .ASCII /C/
002001          132          .ASCII /Z/
002002          106          .ASCII /F/
002003          120          .ASCII /P/
002004          102          .ASCII /B/
002005          000          .BYTE 0
002006          000          .BYTE 0
002007          000          .BYTE 0
002010          L$REV::          ;REVISION LEVEL
002010          101          .ASCII /A/
002011          L$DEPO::          ;0
002011          060          .ASCII /O/
002012          L$UNIT::          ;NUMBER OF UNITS
002012          000001          .WORD T$PTHV
002014          L$TIML::          ;LONGEST TEST TIME
002014          000000          .WORD 0
002016          L$HPCP::          ;POINTER TO H.W. QUES.
002016          034256          .WORD L$HARD
002020          L$SPCP::          ;POINTER TO S.W. QUES.
002020          000000          .WORD 0
002022          L$HPTP::          ;PTR. TO DEF. H.W. PTABLE
002022          002130          .WORD L$HW
002024          L$SPTP::          ;PTR. TO S.W. PTABLE
002024          000000          .WORD 0
002026          L$LADP::          ;DIAG. END ADDRESS
002026          034570          .WORD L$LAST
002030          L$STA::          ;RESERVED FOR APT STATS
002030          000000          .WORD 0
002032          L$CO::          .WORD 0
002032          000000          .WORD 0
002034          L$DTYP::          ;DIAGNOSTIC TYPE
002034          000001          .WORD 1
002036          L$APT::          ;APT EXPANSION
002036          000000          .WORD 0
002040          L$DTP::          ;PTR. TO DISPATCH TABLE
002040          002124          .WORD L$DISPATCH
002042          L$PRIO::          ;DIAGNOSTIC RUN PRIORITY
002042          000000          .WORD 0

```

PROGRAM HEADER

002044  
002044 000000  
002046  
002046 000000  
002050  
002050 003  
002051 003  
002052  
002052 000000  
002054 000000  
002056  
002056 000000  
002060  
002060 002534  
002062  
002062 013170  
002064  
002064 000000  
002066  
002066 000000  
002070  
002070 000000  
002072  
002072 000000  
002074  
002074 000000  
002076  
002076 002540  
002100  
002100 104035  
002102  
002102 000000  
002104  
002104 013210  
002106  
002106 013640  
002110  
002110 013634  
002112  
002112 013202  
002114  
002114 000000  
002116  
002116 000000  
002120  
002120 000000

L\$ENVI:: .WORD 0 ;FLAGS DESCRIBE HOW IT WAS SETUP  
L\$EXP1:: .WORD 0 ;EXPANSION WORD  
L\$MREV:: .BYTE C\$REVISION ;SVC REV AND EDIT #  
          .BYTE C\$EDIT  
L\$EF:: .WORD 0 ;DIAG. EVENT FLAGS  
          .WORD 0  
L\$SPC:: .WORD 0  
L\$DEVP:: .WORD L\$DVTYP ; POINTER TO DEVICE TYPE LIST  
L\$REPP:: .WORD L\$RPT ;PTR. TO REPORT CODE  
L\$EXP4:: .WORD 0  
L\$EXP5:: .WORD 0  
L\$AUT:: .WORD 0 ;PTR. TO ADD UNIT CODE  
L\$DUT:: .WORD 0 ;PTR. TO DROP UNIT CODE  
L\$LUN:: .WORD 0 ;LUN FOR EXERCISERS TO FILL  
L\$DESP:: .WORD L\$DESC ;POINTER TO DIAG. DESCRIPTION  
L\$LOAD:: EMT E\$LOAD ;GENERATE SPECIAL AUTOLOAD EMT  
L\$ETP:: .WORD 0 ;POINTER TO ERR TBL  
L\$ICP:: .WORD L\$INIT ;PTR. TO INIT CODE  
L\$CCP:: .WORD L\$CLEAN ;PTR. TO CLEAN-UP CODE  
L\$ACP:: .WORD L\$AUTO ;PTR. TO AUTO CODE  
L\$PRT:: .WORD L\$PROT ;PTR. TO PROTECT TABLE  
L\$TEST:: .WORD 0 ;TEST NUMBER  
L\$DLY:: .WORD 0 ;DELAY COUNT  
L\$HIME:: .WORD 0 ;PTR. TO HIGH MEM  
          .WORD 0

DISPATCH TABLE

83  
84  
85  
86  
87  
88  
89  
90 002122  
002122 000001  
002124  
002124 013740  
91

```
.SBTTL DISPATCH TABLE  
:  
: THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.  
: IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.  
:  
DISPATCH 1  
.WORD 1  
L$DISPATCH:  
.WORD T1
```



DEFAULT HARDWARE P-TABLE

```

99
100
101
102
103
104
105
106
107
108 002126
      002126 000003
      002130
      002130
109
110 002130 160500
111 002132 000500
112 002134 000200
122
123 002136
      002136

```

.SBTTL DEFAULT HARDWARE P-TABLE

```

; **
; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
; THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
; IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES,
; AND IS USED AS A "TEMPLATE" FOR BUILDING THE P-TABLES.
; --

```

```

      BGNHW DFPTBL
      .WORD L10000-L$HW/2
L$HW::
DFPTBL::

```

```

      .WORD QCMDAD ;HARDWARE UNIT 0 DEFAULT BASE ADDRESS
      .WORD VEC ;HARDWARE UNIT 0 DEFAULT VECTOR
      .WORD †80000000010000000 ;DEFAULT BR INTERRUPT LEVEL 4

```

```

      ENDPHW
L10000:

```

SOFTWARE P-TABLE

126  
 127  
 128  
 129  
 130  
 131  
 132  
 133  
 134  
 135 002136  
 002136 000001  
 002140  
 002140  
 136  
 137 002140 000240  
 138  
 146  
 147 002142  
 002142  
 148  
 149 002142

```
.SBTTL SOFTWARE P-TABLE

; **
; THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
; PROGRAM AS OPERATIONAL PARAMETERS. THESE PARAMETERS ARE
; SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
; AT RUN TIME.
; --

          BGNSW  SFPTBL
          .WORD  L10001-L$SW/2

L$SW::
SFPTBL::

          NOP

          ENDSW
L10001:

          ENDMOD
```

## SOFTWARE P-TABLE

```

11          .ENABL  AMA
39
49
50 002142          BGNMOD
51 002142          STARS
                ;*****
52          ;*
53          ;*          MACROS DEFINED ONLY FOR THIS DIAGNOSTIC
54          ;*
55 002142          STARS
                ;*****
56
57          ;
58          ;          MACRO SET
59          ;          USED TO CALL ROUTINE TO PRINT OUT REGISTER BIT NAMES
60          ;
61          .MACRO  SETBITS NAME, POINTER, VALUE
62          .NARG   TE#          ;GET NUMBER OF ARGUMENTS PASSED
63          .IF NE  TE#-3        ;SEE IF RIGHT NUMBER
64          .ERROR  ;SETBITS MACRO REQUIRES 3 PARAMETERS
65          .ENDC
66          .NCHR   $L1 NAME    ;GET NUMBER OF CHARACTERS IN NAME PARAMETER
67          $L1 = $L1+1        ;ALLOW FOR ZERO BYTE IN ASCIZ
68          $TEMP = $L1 & 1    ;CHECK FOR ODD OR EVEN
69          .IF EQ $TEMP        ;IF EVEN LENGTH OF NAME
70          $LEN = 4+$L1
71          .ENDC
72          .IF NE $TEMP        ;IF ODD LENGTH OF NME
73          $LEN = 5+$L1
74          .ENDC
75          MOV     R1, -(SP)    ;SAVE REGISTERS
76          MOV     R5, -(SP)
77          MOV     VALUE, R1    ;GET VALUE REGISTER TO READ
78          JSR     R5, BIT#     ;SUBROUTINE TO OUTPUT VALUES
79          .WORD   $LEN
80          .WORD   POINTER
81          .ASCIZ  /NAME/
82          .EVEN
83          MOV     (SP)+, R5    ;RESTORE REGISTERS
84          MOV     (SP)+, R1
85          .ENDM
86          ;
87          ;          MACRO S#BITS
88          ;          USED TO CALL ROUTINE TO FORMAT PRINT BUFFER OF REGISTER BIT NAMES
89          ;
90          .MACRO  S#BITS  POINTER, VALUE
91          .NARG   TE#          ;GET NUMBER OF ARGUMENTS PASSED
92          .IF NE  TE#-2        ;SEE IF RIGHT NUMBER
93          .ERROR  ;S#BITS MACRO REQUIRES 2 PARAMETERS
94          .ENDC
95          MOV     R1, -(SP)    ;SAVE REGISTERS
96          MOV     R5, -(SP)
97          MOV     VALUE, R1    ;GET VALUE REGISTER TO READ
98          JSR     R5, BIT#     ;SUBROUTINE TO OUTPUT VALUES
99          .WORD   POINTER
100         .EVEN
101         MOV     (SP)+, R5    ;RESTORE REGISTERS

```

## SOFTWARE P-TABLE

```

102          MOV      (SP)+,R1
103      .ENDM
104      .MACRO  HLPMAC  N
105      ;
106      ;MACRO TO INTERFACE WITH THIS SUBROUTINE TO PRINT OUT TEXT FILE
107      ;
108      .NCHR   $SYM,N
109          JSR      R5,$HLP          ;JUMP TO SUBROUTINE
110          .ASCII  /N/              ;NAME OF FILE
111          $TEMP = 10.-$SYM
112          .REPT   $TEMP
113          .BYTE   0
114          .ENDR
115      .ENDM
116      ;
117      ;MACRO TO BUILD THE TEST TABLE
118      ;
119      .MACRO  TESTNM  N
120          .WORD   TEST'N
121      .ENDM
122      .MACRO  NUMTESTS  N
123      .LIST
124      NUM#TST::
125          .WORD   N                ;NUMBER OF TESTS
126      TESTS::
127      .NLIST
128      .LIST  MEB
129      $TNUM = 1
130      .RADIX 10
131      .REPT  N
132          TESTNM \ $TNUM
133      $TNUM = $TNUM + 1
134      .ENDR
135      .RADIX 8
136      .ENDM
137      ;
138      ;MACRO TO NAME TESTS
139      ;
140      $NUM == 0
141      .MACRO  TEST  N
142      .LIST
143      TEST'N::
144          MOV      @N,L$TEST        ;TEST NUMBER FOR ERROR MESSAGES
145      .NLIST
146      .ENDM
147      ;
148      .MACRO  STATST  N$
149      .RADIX 10
150      .NARG  $TEMP
151      .IF  EQ  $TEMP
152      $NUM = $NUM + 1
153      .ENDC
154      .IF  NE  $TEMP
155      $NUM = N$
156      .ENDC
157          TEST \ $NUM
158          CALL   BMSG              ;CHECK FOR PNT FLAG

```

000000

H2

CZFPB-A.O CTRL BRIDGE LINK

MACRO M1200 09-JAN-84 12:25 PAGE 14-2

SEQ 0020

SOFTWARE P-TABLE

159  
160

.RADIX 8  
.ENDM

GLOBAL EQUATES SECTION

162  
163 002142  
  
164  
165  
166  
167 002142  
  
168  
169 002142

.SBTTL GLOBAL EQUATES SECTION

STARS

;;\*\*\*\*\*

;\*  
;\*  
;\*  
;\*  
STARS

GLOBAL EQUATE SECTION

STARS

;;\*\*\*\*\*

EQUALS

;

; BIT DIFINITIONS

;

100000	BIT15==	100000
040000	BIT14==	40000
020000	BIT13==	20000
010000	BIT12==	10000
004000	BIT11==	4000
002000	BIT10==	2000
001000	BIT09==	1000
000400	BIT08==	400
000200	BIT07==	200
000100	BIT06==	100
000040	BIT05==	40
000020	BIT04==	20
000010	BIT03==	10
000004	BIT02==	4
000002	BIT01==	2
000001	BIT00==	1

001000	BIT9==	BIT09
000400	BIT8==	BIT08
000200	BIT7==	BIT07
000100	BIT6==	BIT06
000040	BIT5==	BIT05
000020	BIT4==	BIT04
000010	BIT3==	BIT03
000004	BIT2==	BIT02
000002	BIT1==	BIT01
000001	BIT0==	BIT00

;

; EVENT FLAG DEFINITIONS

; EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

;

000040	EF.START==	32.	; START COMMAND WAS ISSUED
000037	EF.RESTART==	31.	; RESTART COMMAND WAS ISSUED
000036	EF.CONTINUE==	30.	; CONTINUE COMMAND WAS ISSUED
000035	EF.NEW==	29.	; A NEW PASS HAS BEEN STARTED
000034	EF.PWR==	28.	; A POWER-FAIL/POWER-UP OCCURRED

;

; PRIORITY LEVEL DEFINITIONS

;

000340	PRI07==	340
000300	PRI06==	300
000240	PRI05==	240
000200	PRI04==	200

## GLOBAL EQUATES SECTION

```

000140          PRI03== 140
000100          PRI02== 100
000040          PRI01== 40
000000          PRI00== 0
;
; OPERATOR FLAG BITS
;
000004          EVL==      4
000010          LOT==     10
000020          ADR==     20
000040          IDU==     40
000100          ISR==    100
000200          UAM==    200
000400          BOE==    400
001000          PNT==   1000
002000          PRI==   2000
004000          IXE==   4000
010000          IBE==  10000
020000          IER==  20000
040000          LOE==  40000
100000          MOE== 100000
;
; STANDARD REGISTER BASE ADDRESS CONFIGURATION
;
170             QCMDAD == 160500      ; STANDARD BASE ADDRESS FOR UNIT 0
171             VEC    == 500         ; STANDARD VECTOR ADDRESS FOR UNIT 0
172
173             QCMDAD == 160500      ; STANDARD BASE ADDRESS FOR UNIT 0
174             VEC    == 500         ; STANDARD VECTOR ADDRESS FOR UNIT 0
175
176             ;
177             ; BIT DEFINITIONS
178             ;
179             ; QCMD - Q-BUS COMMAND REGISTER
180             ;
181             CMDACK == BIT00        ; NO COMMAND ACKNOWLEDGE
182             PRSIE  == BIT06        ; R/W INTERRUPT ENABLE
183             ;
184             ; QVNT - Q-BUS EVENT REGISTER
185             ;
186             VNTPRS == BIT00        ; NO EVENT PRESENT
187             ACKIE  == BIT06        ; R/W INTERRUPT ENABLE
188             ;
189             ; QINT - Q-BUS INTERRUPT SUMMARY REGISTER
190             ;
191             ERRIE  == BIT06        ; R/W INTERRUPT ENABLE
192             MRDY   == BIT09        ; RO MIRROR OF QDMA-07, READY
193             VNTACK == BIT10        ; RO EVENT ACKNOWLEDGE
194             CMDPRS == BIT11        ; RO COMMAND PRESENT
195             UPMRFL == BIT12        ; RO UNIBUS POWER FAIL
196             QNEX   == BIT13        ; RO Q-BUS NON EXISTENT MEMORY ADDRESS
197             UNEX   == BIT14        ; RO UNIBUS NON EXISTENT MEMORY ADDRESS
198             ERR    == BIT15        ; RO ERROR
199             ;
200             ; QCTL - Q-BUS CONTROL REGISTER
201             ;
202             WDRST  == BIT00        ; NO WATCHDOG TIMER RESET
203             TMRENB == BIT01        ; R/W TIMER ENABLE
204             WDT0   == BIT02        ; NO WATCHDOG TIMER TIME-OUT BIT (1 OF 2)
205             WDT1   == BIT03        ; NO WATCHDOG TIMER TIME-OUT BIT (2 OF 2)

```

## GLOBAL EQUATES SECTION

```

206      000020      QVREN   == BIT04      ; Q-VIEW READ ENABLE
207      000000      TIME.5  == 0          ; 1/2 SEC TIME CODE
208      000004      TIME1    == WDT0      ; 1 SEC TIME CODE
209      000010      TIME2    == WDT1      ; 2 SEC TIME CODE
210      000014      TIME4    == WDT0!WDT1 ; 4 SEC TIME CODE
211      ;
212      ;
213      ;          QDMA - Q-BUS DMA CONTROL AND STATUS REGISTER
214      ;
215      000001      GO        == BIT00      ; WO INITIATES DMA TRANSFER
216      000002      DMAWT    == BIT01      ; R/W DIRECTION FLOW OF DMA
217      000020      UXAD16   == BIT04      ; R/W UNIBUS EXTENDED ADDRESS BIT 16
218      000040      UXAD17   == BIT05      ; R/W UNIBUS EXTENDED ADDRESS BIT 17
219      000100      RDYIE    == BIT06      ; R/W INTERRUPT ENABLE
220      000200      RDY      == BIT07      ; RO READY (CLEARED BY GO)
221      000400      QXAD16   == BIT08      ; R/W Q-BUS EXTENDED ADDRESS BIT 16
222      001000      QXAD17   == BIT09      ; R/W Q-BUS EXTENDED ADDRESS BIT 17
223      002000      QXAD18   == BIT10      ; R/W Q-BUS EXTENDED ADDRESS BIT 18
224      004000      QXAD19   == BIT11      ; R/W Q-BUS EXTENDED ADDRESS BIT 19
225      010000      QXAD20   == BIT12      ; R/W Q-BUS EXTENDED ADDRESS BIT 20
226      020000      QXAD21   == BIT13      ; R/W Q-BUS EXTENDED ADDRESS BIT 21
227      ;
228      ;          QMT1 - Q-BUS MAINTENANCE REGISTER 1
229      ;
230      000002      MVADR    == BIT01      ; R/W MAINTENANCE VIEW ADDRESS
231      000004      FUNC0    == BIT02      ; R/W SELECT MAINTENANCE MODE BIT 02 FOR QMTO
232      000010      FUNC1    == BIT03      ; R/W SELECT MAINTENANCE MODE BIT 03 FOR QMTO
233      000020      FUNC2    == BIT04      ; R/W SELECT MAINTENANCE MODE BIT 04 FOR QMTO
234      000040      MQPFL    == BIT05      ; R/W MAINTENANCE Q-BUS POWER FAIL
235      000100      MQHALT   == BIT06      ; WO MAINTENANCE Q-BUS HALT
236      000200      MIODIS   == BIT07      ; R/W MAINTENANCE I/O DISABLE
237      ;
238      ;          MAINTENANCE FUNCTION, BIT IN QMT1 TO DEFINE USE OF QMTO
239      ;
240      000004      SELQEVEN  == 000004    ; SELECT QEVENT
241      000010      SELQCMD   == 000010    ; SELECT QCMD
242      000014      SELUVADR  == 000014    ; SELECT UVADR
243      000020      SELUBAR   == 000020    ; SELECT UBAR
244      000024      SELDMADAT == 000024    ; SELECT DMADAT
245      000030      SELUVDAT  == 000030    ; SELECT UVDAT
246      000034      SELRDDMA  == 000034    ; SELECT DMA READ
247      ;
248      ;          MEMORY MANAGEMENT MAPPING ADDRESSES
249      ;
250      ;
251      177572      MMRO      == 177572
252      172516      MMR3      == 172516
253      172300      KISDRO    == 172300
254      172340      KISARO    == 172340
255      170370      HIUBM     == 170370
256      172340      APRO      == KISARO
257      172342      APR1      == APRO+2
258      172344      APR2      == APR1+2
259      172346      APR3      == APR2+2
260      172350      APR4      == APR3+2
261      172352      APR5      == APR4+2
262      172354      APR6      == APR5+2

```



GLOBAL EQUATES SECTION

263 172356  
 264  
 265  
 266  
 267  
 268 000010  
 269 000000  
 270 000001  
 271 020040  
 272 026000  
 273 000013  
 274  
 275  
 290

APR7 == APR6+2

;  
 ; MISC. EQUATES  
 ;

MAXUNITS == 8.  
 NO == 0  
 YES == 1  
 WSPACE == 20040  
 TIMEOUT == 26000  
 MILLI == 11.

; MAXIMUM NUMBER OF UNITS ALLOWED  
 ; LOGICAL VALUE FOR NO  
 ; LOGICAL VALUE FOR YES  
 ; WORD OF ASCII SPACES  
 ; TIME OUT LOOP VALUE FOR 1 SEC  
 ; TIMEOUT VALUE FOR ABOUT 1 MILLISECOND

GLOBAL DATA SECTION

292  
 293  
 294 002142  
  
 295  
 296  
 297  
 298 002142  
  
 299  
 300  
 301  
 302  
 303 002142  
 304 002142 000000  
 305 002144 000000  
 306 002146 000000  
 307 002150 000000  
 308 002152 000000  
 309 002154 000000  
 310 002156 000000  
 311 002160 000000  
 312 002162 000000  
 313 002164 000000  
 314 002166 000000  
 315 002170 000000  
 316  
 317  
 318  
 319  
 320 002172  
 321 002172 000000  
 322 002174 000000  
 323 002176 000000  
 324 002200 000000  
 325 002202 000000  
 326 002204 000000  
 327 002206 000000  
 328 002210 000000  
 329 002212 000000  
 330 002214 000000  
 331 002216 000000  
 332 002220 000000  
 333  
 334 002222  
 335 002222 000000  
 336 002224 000000  
 337 002226 000000  
 338 002230 000000  
 339 002232 000000  
 340 002234 000000  
 341 002236 000000  
 342 002240 000000  
 343 002242 000000  
 344 002244 000000  
 345 002246 000000  
 346 002250 000000

```

.SBTTL GLOBAL DATA SECTION
STARS
;*****
;*
;* GLOBAL DATA SECTION
;*
STARS
;*****

;
; CURRENT UNIT Q-BUS BRIDGE INTERFACE REGISTERS ADDRESSES ARE STORED HERE
;
ADDRESSES::
QCMD:: .WORD 0 ; Q-BUS COMMAND REGISTER
QVNT:: .WORD 0 ; Q-BUS EVENT REGISTER
QINT:: .WORD 0 ; Q-BUS INTERRUPT SUMMARY REGISTER
QCTL:: .WORD 0 ; Q-BUS CONTROL REGISTER
QDMA:: .WORD 0 ; Q-BUS DMA CONTROL & STATUS REGISTER
QUBA:: .WORD 0 ; Q-BUS BUS ADDRESS UNIBUS REGISTER
QBA:: .WORD 0 ; Q-BUS BUS ADDRESS Q-BUS REGISTER
QWC:: .WORD 0 ; Q-BUS WORD COUNT REGISTER
QMT0:: .WORD 0 ; Q-BUS MAINTENANCE REGISTER 0
QMT1:: .WORD 0 ; Q-BUS MAINTENANCE REGISTER 1
QVAD:: .WORD 0 ; Q-BUS VIEW ADDRESS REGISTER
QVDA:: .WORD 0 ; Q-BUS DATA REGISTER

;
; CURRENT UNIT Q-BUS BRIDGE INTERFACE REGISTERS VALUES ARE STORED HERE
;
VALUES::
SQCMD:: .WORD 0 ; COMMAND REGISTER
SQVNT:: .WORD 0 ; EVENT REGISTER
SQINT:: .WORD 0 ; INTERRUPT SUMMARY REGISTER
SQCTL:: .WORD 0 ; CONTROL REGISTER
SQDMA:: .WORD 0 ; DMA CONTROL & STATUS REGISTER
SQUBA:: .WORD 0 ; BUS ADDRESS UNIBUS REGISTER
SQBA:: .WORD 0 ; BUS ADDRESS REGISTER
SQWC:: .WORD 0 ; WORD COUNT REGISTER
SQMT0:: .WORD 0 ; MAINTENANCE REGISTER 0
SQMT1:: .WORD 0 ; MAINTENANCE REGISTER 1
SQVAD:: .WORD 0 ; VIEW ADDRESS REGISTER
SQVDA:: .WORD 0 ; DATA REGISTER

;
; USED FOR TABLE PROCESSING OF EXPECTED RESULTS
;
EXPECTED::
EXQCMD:: .WORD 0 ; COMMAND REGISTER
EXQVNT:: .WORD 0 ; EVENT REGISTER
EXQINT:: .WORD 0 ; INTERRUPT SUMMARY REGISTER
EXQCTL:: .WORD 0 ; CONTROL REGISTER
EXQDMA:: .WORD 0 ; DMA CONTROL & STATUS REGISTER
EXQUBA:: .WORD 0 ; BUS ADDRESS UNIBUS REGISTER
EXQBA:: .WORD 0 ; BUS ADDRESS REGISTER
EXQWC:: .WORD 0 ; WORD COUNT REGISTER
EXQMT0:: .WORD 0 ; MAINTENANCE REGISTER 0
EXQMT1:: .WORD 0 ; MAINTENANCE REGISTER 1
EXQVAD:: .WORD 0 ; VIEW ADDRESS REGISTER
EXQVDA:: .WORD 0 ; DATA REGISTER

```

GLOBAL DATA SECTION

347  
 348  
 349  
 350  
 351 002252  
 002252  
 002252 000031  
 002254  
 002254 014220  
 002256 014240  
 002260 014474  
 002262 014730  
 002264 015274  
 002266 015436  
 002270 016004  
 002272 016146  
 002274 016476  
 002276 017030  
 002300 020132  
 002302 020734  
 002304 021756  
 002306 022536  
 002310 023522  
 002312 024302  
 002314 025266  
 002316 026046  
 002320 026756  
 002322 030116  
 002324 030422  
 002326 031506  
 002330 033440  
 002332 034010  
 002334 034202

; INTERNAL TEST NUMBERS ARE STORED HERE

```

;
NUMTESTS 25. ;VB.2
NUM#TST:: .WORD 25. ;NUMBER OF TESTS
TESTS::
.WORD TEST1
.WORD TEST2
.WORD TEST3
.WORD TEST4
.WORD TEST5
.WORD TEST6
.WORD TEST7
.WORD TEST8
.WORD TEST9
.WORD TEST10
.WORD TEST11
.WORD TEST12
.WORD TEST13
.WORD TEST14
.WORD TEST15
.WORD TEST16
.WORD TEST17
.WORD TEST18
.WORD TEST19
.WORD TEST20
.WORD TEST21
.WORD TEST22
.WORD TEST23
.WORD TEST24
.WORD TEST25

```

352  
 353  
 354  
 355  
 356 002336 121 103 115  
 002341 104 000  
 357 002343 121 126 116  
 002346 124 000  
 358 002350 121 111 116  
 002353 124 000  
 359 002355 121 103 124  
 002360 114 000  
 360 002362 121 104 115  
 002365 101 000  
 361 002367 121 125 102  
 002372 101 000  
 362 002374 121 102 101  
 002377 040 000  
 363 002401 121 127 103  
 002404 040 000  
 364 002406 121 115 124  
 002411 060 000  
 365 002413 121 115 124  
 002416 061 000

; Q-BRIDGE REGISTER NAMES ARE STORED HERE. ACCESS IS INDEXED, EACH TAKING 2 WORDS

```

;
NAMES:: .EVEN
.ASCIZ /QCMD/
.ASCIZ /QVNT/
.ASCIZ /QINT/
.ASCIZ /QCTL/
.ASCIZ /QDMA/
.ASCIZ /QUBA/
.ASCIZ /QBA /
.ASCIZ /QWC /
.ASCIZ /QMT0/
.ASCIZ /QMT1/

```

GLOBAL DATA SECTION

```

366 002420      121      126      101      .ASCIZ /QVAD/
      002423      104      000
367 002425      121      126      104      .ASCIZ /QVDA/
      002430      101      000
368                                     .EVEN
369
370                                     ;
371                                     ;   MISC STORAGE AREAS
372                                     ;
373
374 002432 000000 BRIDGETYPE:: .WORD 0 ;BRIDGE TYPE (0=Q-BRIDGE 1=U-BRIDGE)
375 002434 000000 BRDGFLG:: .WORD 0 ;BRIDGE CONFIGURED FLAG
376 002436 000000 LOGUNIT:: .WORD 0 ;USED TO STORE CURRENT LOGICAL UNIT
377 002440 000000 PLOC:: .WORD 0 ;POINTER TO CURRENT HARDWARE P-TABLE
378 002442 000000 BASEADD:: .WORD 0 ;STORES BASE ADDRESS
379 002444 000000 VECTOR:: .WORD 0 ;STORES VECTOR ADDRESS
380 002446 000000 BRLEVEL:: .WORD 0 ;STORES BR LEVEL
381 002450 000000 TRPFLG:: .WORD 0 ;TRAP FLAG INDICATOR
382 002452 000000 ADD:: .WORD 0 ;STORES ADDRESS OF REGISTER FOR MESSAGES
383 002454 000000 VAL:: .WORD 0 ;STORES VALUE OF REGISTER FOR MESSAGES
384 002456 000000 EXP:: .WORD 0 ;STORES EXPECTED VALUE OF TEST FOR MESSAGES
385 002460 000000 TNUM:: .WORD 0 ;STORES TEST NUMBER
386 002462 000000 ERROR1:: .WORD 0 ;STORES ERROR COUNTS
387 002464 000000 ERROR2:: .WORD 0 ;STORES ERROR COUNTS
388 002466 000000 RAM:: .WORD 0 ;TELLS WHETHER RAM MESSAGE WAS PRINTED OUT
389 002470 000000 VALUE:: .WORD 0 ;STORES CURRENT CPU PRIORITY LEVEL
390 002472 000000 INTFLG:: .WORD 0 ;INTERRUPT FLAG
391 002474 000000 COUNTER:: .WORD 0 ;LOOP COUNTER STORAGE
392 002476 000000 HP1:: .WORD 0 ;TELLS WHETHER DRS HELP MESSAGE ALREADY PRINTED
393 002500 000000 HP2:: .WORD 0 ;TELLS WHETHER DIA HELP MESSAGE ALREADY PRINTED
394 002502 000000 FREMEM:: .WORD 0 ;STORES STARTING ADDRESS OF FREE MEMORY
395 002504 000000 MEMSIZ:: .WORD 0 ;STORE SIZE OF FREE MEMORY IN # OF WORDS
396 002506 000000 DIR:: .WORD 0 ;DIRECTION BIT FOR DMA
397 002510 000000 PATTERN:: .WORD 0 ;STORES CODE FOR DMA PATTERN
398 002512 000000 WORD.COUNT:: .WORD 0 ;STORE WORD COUNT FOR DMA IN 2'S COMPLIMENT FORM
399 002514 000000 LOW16:: .WORD 0 ;LOW 16 BITS OF UNIBUS ADDRESS FOR DMA
400 002516 000000 HI2:: .WORD 0 ;HIGH 2 BITS OF UNIBUS ADDRESS FOR DMA
401 002520 000000 MAPPING:: .WORD NO ;FLAG TO USE MAPPING REGISTERS
402 002522 000000 APR:: .WORD 0 ;STORES CURRENT VALUE OF APR UNDER USE
403 002524 000000 LOOPS:: .WORD 0 ;STORES CURRENT LOOP VALUE
404 002526 000000 DLY1: .WORD 0 ;STORE WRITE DELAY VALUE FOR CONTENTIONS TEST
405 002530 000000 DLY2: .WORD 0 ;STORE READ DELAY VALUE FOR CONTENTIONS TEST
406 002532 000000 DEFAULT:: .WORD 0 ;STORE DEFAULT TIMEOUT VALUE

```

GLOBAL TEXT SECTION

```

408 .SBTTL GLOBAL TEXT SECTION
409
410 ;**
411 ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
412 ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
413 ; MORE THAN ONE TEST.
414 ;--
415
416 ;
417 ; NAMES OF DEVICES SUPPORTED BY PROGRAM
418 ;
419 002534          DEVTYP <FEP>
      002534          L#DVTYP::
      002534      106      105      120      .ASCIZ /FEP/
      002537          .EVEN
420
421 ; TEST DESCRIPTION
422 ;
423 ;
424 002540          DESCRIPT      <COMMAND BRIDGE LINK MODE DIAGNOSTIC>
      002540          L#DESC::
      002540      103      117      115      .ASCIZ /COMMAND BRIDGE LINK MODE DIAGNOSTIC/
      002543      115      101      116
      002546      104      040      102
      002551      122      111      104
      002554      107      105      040
      002557      114      111      116
      002562      113      040      115
      002565      117      104      105
      002570      040      104      111
      002573      101      107      116
      002576      117      123      124
      002601      111      103      000
425          .EVEN
426          .EVEN
427 .NLIST BEX
428 ;
429 ; BIT NAMES IN THE REGISTERS
430 002604      054      054      054      CHDBIT:: .ASCIZ /.....PRsie.....CMDACK/
431 002637      054      054      054      VNTBIT:: .ASCIZ /.....ACKIE.....VNTPRS/
432 002672      105      122      122      INTBIT:: .ASCIZ /ERR,UNEX,QNEX,UPWRFL,CHOPRS,VNTACK,RDY...ERRIE/
433 002751      054      054      054      CTLBIT:: .ASCIZ /.....WDT1,WDT0,THRENB,WDRST/
434 003015      054      054      121      DMABIT:: .ASCII /..QXAD21,QXAD20,QXAD19,QXAD18,QXAD17,QXAD16,RDY,RDYIE/
435 003102      054      125      130      .ASCIZ /,UXAD17,UXAD16...DMAWT,GO/
436 003134      054      054      054      MT1BIT:: .ASCIZ /.....MIODIS,MQHLT,MQPFL,FUNC2,FUNC1,FUNCO,MVADR/
437
438

```

GLOBAL TEXT SECTION

440 003217

STARS

;;\*\*\*\*\*

441

;

442

; FORMAT STATEMENTS USED IN PRINT CALLS

443

;

444

;.IF NDF ONEFILE

445

.NLIST BEX

446

;.ENDC

447

448 003217

045

116

061

CONFIG:: .ASCIZ /#N1#A CONFIGURED FOR #T#A BRIDGE MODULE#N1/

449 003271

121

055

102

QBUS:: .ASCIZ /Q-BUS/

450 003277

125

116

111

UBUS:: .ASCIZ /UNIBUS/

451 003306

045

116

061

FOR0:: .ASCIZ /#N1/ ;LINE FEED CARRIAGE RETURN ONLY

452 003312

045

116

061

FOR1:: .ASCII /#N1#ANAME#S4#ADDRESS#S4#CONTENTS#S4#EXPECTED/

453 003371

040

050

125

.ASCIZ / (UNDETERMINED BITS ARE ZERO)/

454 003427

045

116

061

FOR2:: .ASCIZ /#N1#T#S5#O6#S5#O6#S6#O6/

455 003457

045

101

105

FOR3:: .ASCIZ /#AEVENT SYNC TIMEOUT OCCURED AFTER TEST #Z2#A COMPLETED/

456 003547

045

101

124

FOR4:: .ASCIZ /#ATEST #Z2#A TIMEOUT/

457 003574

045

116

061

FOR5:: .ASCIZ /#N1#S4#O4#S9#O6#S7#O6/

458 003622

045

116

061

FOR6:: .ASCIZ /#N1#ATOTAL RAM CHECK ERRORS #D5/

459 003662

045

101

122

FOR7:: .ASCIZ /#ARAM ADDRESS CONTENTS EXPECTED/

460 003732

045

101

114

FOR8:: .ASCIZ /#ALSI-TEST #Z2#N1/

461 003754

045

101

116

FOR9:: .ASCII /#ANOT ENOUGH MEMORY AVAILABLE FOR DMA TRANSFER;/

462 004033

045

116

061

FOR9.1:: .ASCIZ /#N1#D5#A MAXIMUM WORDS AVAILABLE IN LOW MEMORY/

463 004112

045

116

061

FOR9.2:: .ASCIZ /#N1#D5#A MAXIMUM WORDS AVAILABLE IN HI MEMORY WITH MAPPING/

464 004205

045

116

061

FOR9.3:: .ASCIZ /#N1#D5#A REQUESTED/

465 004230

045

116

061

FOR10:: .ASCIZ /#N1#Z5#S3#O6#S4#O6/

466 004253

045

116

061

FOR11:: .ASCIZ /#N1#A TOTAL DMA ERRORS = #Z5/

467 004310

045

101

124

FOR12:: .ASCIZ /#ATIMEOUT WAITING FOR QINT CHDPRS INTERRUPT/

468 004364

045

101

124

FOR13:: .ASCIZ /#ATIMEOUT WAITING FOR QDMA RDY INTERRUPT/

469 004435

045

101

111

FOR14:: .ASCIZ /#AINVALID DATA AFTER DMA TRANSFER/

470 004477

045

116

062

FOR14.1:: .ASCIZ /#N2#AWORD# VALUE EXPECTED/

471 004536

045

101

124

FOR15:: .ASCIZ /#ATIMEOUT WAITING FOR QINT VNTACK INTERRUPT/

472 004612

045

101

124

FOR16:: .ASCIZ /#ATIMEOUT WAITING FOR QINT CHDPRS SET#N1/

473 004663

045

101

124

FOR17:: .ASCIZ /#ATIMEOUT WAITING FOR QINT VNTACK SET#N1/

474 004734

045

101

127

FOR18:: .ASCIZ /#AWAITING FOR PATTERN # #Z2/

475 004770

045

101

117

FOR19:: .ASCII /#AONLY BITS 4 AND 5 SHOULD BE SET FOR HIGH UNIBUS ADDRESS/

476 005061

045

116

045

.ASCIZ /#N#A CONTENTS = #O6/

477 005104

045

101

105

FOR20:: .ASCIZ /#AERR INTERRUPT OCCURED/

478 005134

045

101

124

FOR21:: .ASCIZ /#ATIMEOUT WAITING FOR QDMA RDY OR QINT ERR INTERRUPT/

479 005221

045

101

121

FOR22:: .ASCIZ /#AQINT QNEX BIT 13 Q-BUS NON EXISTENT MEMORY SHOULD BE SET/

480 005314

045

116

061

FOR23:: .ASCIZ /#N1#ALOOP NUMBER = #Z3/

481 005343

045

116

061

FOR24:: .ASCIZ /#N1#ADMA THROTTLE TEST WAITING FOR LOOP #D1/

482 005417

045

116

061

FOR25:: .ASCIZ /#N1#AERROR IN QBOOT, NO BOOT OCCURED/

483 005464

045

101

124

FOR26:: .ASCIZ /#ATIMEOUT WAITING FOR QINT VNTACK CLEAR/

484 005534

045

116

061

ABO:: .ASCIZ /#N1#A\*\*\*\*TEST ABORTED\*\*\*\*/

485 005566

045

116

061

FORERR:: .ASCIZ /#N1#A/<7>/UNEXPECTED INTERRUPT IN TEST #Z2#A PC=#O6#A/<7>/#N1/

GLOBAL TEXT SECTION

487 005654

STARS

::\*\*\*\*\*

488

:

489

; ASCII MESSAGES USED IN ERROR REPORTING

490

:

491 005654

STARS

::\*\*\*\*\*

492

493

494 005654

120

122

111

HELP1:: .ASCIZ /PRINT DIAGNOSTIC SUPERVISOR HELP FILE?/

495 005723

120

122

111

HELP2:: .ASCIZ /PRINT Z0792 DIAGNOSTIC HELP FILE?/

496 005765

102

122

111

MSG1:: .ASCIZ /BRIDGE TIMEOUT ERROR/

497 006012

102

122

111

MSG2:: .ASCIZ /BRIDGE QCMD ROTATING BITS 1-5 IN ERROR/

498 006061

102

122

111

MSG3:: .ASCIZ /BRIDGE QVNT ROTATING BITS 1-5 IN ERROR/

499 006130

102

122

111

MSG4:: .ASCIZ /BRIDGE QVDA VIEW REGISTER RAM CHECK ERROR/

500

501 006202

102

122

111

MSG6:: .ASCIZ /BRIDGE WRONG INTERRUPT BIT, CHECK QINT FOR WHICH ONE/

502

503

504

.EVEN

## GLOBAL ERROR REPORT SECTION

```

509          .SBTTL GLOBAL ERROR REPORT SECTION
510
511          ;**
512          ; THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
513          ; USED BY MORE THAN TEST TO OUTPUT ADDITIONAL ERROR INFORMATION. PRINTB
514          ; (BASIC) AND PRINTX (EXTENDED) CALLS ARE USED TO CALL PRINT SERVICES.
515          ;--
516
517
518          ;          TIME OUT ERROR
519          ; BASIC FORMAT IS-
520          ;          TIMEOUT OCCURED AFTER TEST ? COMPLETED
521
522          006270          BGNMSG  ERRO
523          006270          ERRO::  PRINTB  #FOR3,TNUM
524          006270 013746 002460      MOV    TNUM,-(SP)
525          006274 012746 003457      MOV    #FOR3,-(SP)
526          006300 012746 000002      MOV    #2,-(SP)
527          006304 010600              MOV    SP,R0
528          006306 104414              TRAP  C#PNTB
529          006310 062706 000006      ADD    #6,SP
530          524 006314          ENDMSG
531          006314 104423          L10002: TRAP  C#MSG
532
533          ;          TIME OUT ERROR
534          ; BASIC FORMAT IS-
535          ;          LSI-TEST XX TIMEOUT
536          ; R1 EXPECTED TO CONTAIN REGISTER NUMBER
537
538          006316          BGNMSG  ERR1
539          006316          ERR1::  RFLAGS  FLAG
540          006316 104421          TRAP  C#RFLA
541          006320 010037 011710      MOV    R0,FLAG
542          533 006324 032737 020000 011710  BIT    #IER,FLAG
543          534 006332 001012          BNE    1#
544          535 006334          PRINTF  #FOR8,L#TEST
545          006334 013746 002114      MOV    L#TEST,-(SP)
546          006340 012746 003732      MOV    #FOR8,-(SP)
547          006344 012746 000002      MOV    #2,-(SP)
548          006350 010600              MOV    SP,R0
549          006352 104417              TRAP  C#PNTF
550          006354 062706 000006      ADD    #6,SP
551          536 006360 022701 000000      1#:  CMP    #0,R1
552          537 006364 001404          BEQ    10#
553          538 006366 022701 000001      CMP    #1,R1
554          539 006372 001412          BEQ    20#
555          540 006374 000421          BR     30#
556          ;COMMAND REGISTER TIMEOUT
557          10#:
558          006376          PRINTX  #FOR16
559          542 006376          MOV    #FOR16,-(SP)
560          006376 012746 004612      MOV    #1,-(SP)
561          006402 012746 000001      MOV    SP,R0
562          006406 010600              TRAP  C#PNTX
563          006410 104415

```

```

;IS INHIBIT FLAGS SET?
;BRANCH OVER IF SET
;PRINT TEST NUMBER

```

```

;SEE WHICH REGISTER
;IF COMMAND REGISTER, BRANCH

```

```

;BRANCH IF QVENT REGISTER
;JUST PRINT ABORT MESSAGE

```



GLOBAL ERROR REPORT SECTION

```

006412 062706 000004
544 006416 000410
545
546 006420
547 006420
006420 012746 004663
006424 012746 000001
006430 010600
006432 104415
006434 062706 000004
548
549 006440
006440 013746 002474
006444 012746 004734
006450 012746 000002
006454 010600
006456 104415
006460 062706 000006
550 006464
006464 012746 005534
006470 012746 000001
006474 010600
006476 104414
006500 062706 000004
551 006504
006504
006504 104423
552
553
554
555
556
557
558
559
560
561
562
563 006506
006506
564 006506
006506 104421
006510 010037 011710
565 006514 032737 020000 011710
566 006522 001012
567 006524
006524 013746 002114
006530 012746 003732
006534 012746 000002
006540 010600
006542 104417
006544 062706 000006
568 006550 022701 000000
569 006554 001404
570 006556 022701 000001
571 006562 001412
572 006564 000421

```

```

ADD #4,SP
BR 30$
;EVENT REGISTER TIMEOUT
20$:
PRINTX #FOR17
MOV #FOR17,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C#PNTX
ADD #4,SP
;COMMON
30$:
PRINTX #FOR18,COUNTER
MOV COUNTER,-(SP)
MOV #FOR18,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C#PNTX
ADD #6,SP
PRINTB #ABO
MOV #ABO,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C#PNTB
ADD #4,SP
ENDMSG
L10003:
TRAP C#MSG
;
; TIME OUT ERROR
; BASIC FORMAT IS-
; LSI-TEST XX TIMEOUT WAITING FOR QINT CMDPRS SET
; *****TEST ABORTED***
;
; INPUTS:
; R1 CONTAINS REGISTER NUMBER
;
ERR2::
BGNMSG ERR2
RFLAGS FLAG
TRAP C#RFLA
MOV RO,FLAG
BIT #IER,FLAG ;IS INHIBIT FLAGS SET?
BNE 1$ ;BRANCH OVER IF SET
PRINTF #FOR8,L#TEST ;PRINT TEST NUMBER
MOV L#TEST,-(SP)
MOV #FOR8,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C#PNTF
ADD #6,SP
1$:
CMP #0,R1 ;SEE WHICH REGISTER
BEQ 10$ ;IF COMMAND REGISTER, BRANCH
CMP #1,R1
BEQ 20$ ;BRANCH IF QVENT REGISTER
BR 30$ ;JUST PRINT ABORT MESSAGE

```

GLOBAL ERROR REPORT SECTION

```

573                                     ;COMMAND REGISTER TIMEOUT
574 006566                             10$:
575 006566 012746 004612                PRINTX  #FOR16
    006572 012746 000001                MOV     #FOR16,-(SP)
    006576 010600                        MOV     #1,-(SP)
    006600 104415                        MOV     SP,R0
    006602 062706 000004                TRAP   C#PNTX
    006606 000410                        ADD     #4,SP
    ;SKIP OVER EVENT REGISTER PRINTING
576 006606 000410
577                                     ;EVENT REGISTER TIMEOUT
578 006610                             20$:
579 006610 012746 004663                PRINTX  #FOR17
    006614 012746 000001                MOV     #FOR17,-(SP)
    006620 010600                        MOV     #1,-(SP)
    006622 104415                        MOV     SP,R0
    006624 062706 000004                TRAP   C#PNTX
    006630 006630 012746 005534          30$: PRINTB  #ABO
    006634 012746 000001                MOV     #ABO,-(SP)
    006640 010600                        MOV     #1,-(SP)
    006642 104414                        MOV     SP,R0
    006644 062706 000004                TRAP   C#PNTB
    006650 006650 000004                ADD     #4,SP
    006650 104423                        ENDMSG
580 006630
581 006650
582 006650
583
584                                     ;
585                                     ; BAD REGISTER VALUE ERROR (SINGLE REGISTER TEST)
586                                     ;REGISTER NUMBER (0-11.) EXPECTED TO BE IN R1
587                                     ;REGISTER ADDRESS EXPECTED TO BE IN LOCATION ADD
588                                     ;REGISTER VALUE EXPECTED TO BE IN LOCATION VAL
589                                     ;REGISTER EXPECTED VALUE EXPECTED TO BE IN LOCATION EXP
590 006652
591 006652
592 006652 010146
593 006652 010246
594 006652 005002
595 006652 060102
596 006652 060102
597 006652 060102
598 006652 060102
599 006652 062702 002336
600 006652 006301
601 006700 016137 002142 002452
602 006706
603 006706 012746 003312
    006712 012746 000001
    006716 010600
    006720 104414
    006722 062706 000004
    006726 013746 002456
    006732 013746 002454
    006736 013746 002452

ERR3::
    BGNMSG  ERR3
    MOV     R1,-(SP)
    MOV     R2,-(SP)
    CLR     R2
    ADD     R1,R2
    ADD     R1,R2
    ADD     R1,R2
    ADD     R1,R2
    ADD     #NAMES,R2
    ASL     R1
    MOV     ADDRESSES(R1),ADD
    PRINTB  #FOR1
    MOV     #FOR1,-(SP)
    MOV     #1,-(SP)
    MOV     SP,R0
    TRAP   C#PNTB
    ADD     #4,SP
    PRINTB  #FOR2,R2,ADD,VAL,EXP
    MOV     EXP,-(SP)
    MOV     VAL,-(SP)
    MOV     ADD,-(SP)

;SAVE REGISTERS USED IN ROUTINES
;CLEAR ACCUMULATOR
;MULTIPLY REGISTER POINTER TIME 5
;GET REGISTER NAME ADDRESS IN R2
;REGISTER POINTER TIMES 2
;MOVE CURRENT ADDRESS TO MESSAGE AREA
;PRINT HEADER LINE
;PRINT FULL ERROR DETAILS
    
```

GLOBAL ERROR REPORT SECTION

```

006742 010246
006744 012746 003427
006750 012746 000005
006754 010600
006756 104414
006760 062706 000014
604 006764
006764 012746 003306
006770 012746 000001
006774 010600
006776 104414
007000 062706 000004
605 007004 010226
606 007006 010126
607 007010
007010
007010 104423
608
609
610
611
612 007012
007012
613 007012
007012 104421
007014 010037 011710
614 007020 032737 020000 011710
615 007026 001012
616 007030
007030 013746 002114
007034 012746 003732
007040 012746 000002
007044 010600
007046 104417
007050 062706 000006
617 007054
007054 012746 003662
007060 012746 000001
007064 010600
007066 104415
007070 062706 000004
618 007074
007074
007074 104423
619
620
621
622 007076
007076
623 007076 013702 002512
624 007102 005402
625 007104 013703 002454
626 007110 005203
627 007112 006303
628 007114 006303
629 007116 006303
630 007120 006303

MOV R2,-(SP)
MOV #FOR2,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C#PNTB
ADD #14,SP
PRINTB #FOR0 ;INSERT LINE FEED CARRIAGE RETURN
MOV #FOR0,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C#PNTB
ADD #4,SP
MOV R2,(SP)+ ;RESTORE REGISTERS
MOV R1,(SP)+
ENDMSG
L10005:
TRAP C#MSG
;BAD VIEW REGISTER RAM ERROR
;PRINTS HEADER ONLY
ERR4::
BGNMSG ERR4
RFLAGS FLAG
TRAP C#RFLA
MOV R0,FLAG
BIT #IER,FLAG ;IS INHIBIT FLAGS SET?
BNE 1$ ;BRANCH OVER IF SET
PRINTF #FOR8,L#TEST ;PRINT TEST NUMBER
MOV L#TEST,-(SP)
MOV #FOR8,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C#PNTF
ADD #6,SP
PRINTX #FOR7 ;PRINT HEADER
MOV #FOR7,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C#PNTX
ADD #4,SP
L10006:
TRAP C#MSG
;NOT ENOUGH MEMORY IN BUFFER TO TRANSFER REQUESTED DMA BYTE COUNT
ERR5::
BGNMSG ERR5
MOV WORD.COUNT,R2
NEG R2 ;CONVERT WORD COUNT FROM 2'S COMPL
MOV VAL,R3
INC R3
ASL R3 ;CONVERT PAR FORMAT TO WORDS
ASL R3
ASL R3
ASL R3

```

GLOBAL ERROR REPORT SECTION

631	007122	006303		ASL	R3	
632	007124			RFLAGS	FLAG	
	007124	104421		TRAP	C#RFLA	
	007126	010037	011710	MOV	RO,FLAG	
633	007132	032737	020000	BIT	#IER,FLAG	;IS INHIBIT FLAGS SET?
634	007140	001012		BNE	1#	;BRANCH OVER IF SET
635	007142			PRINTF	#FOR8,L#TEST	;PRINT TEST NUMBER
	007142	013746	002114	MOV	L#TEST,-(SP)	
	007146	012746	003732	MOV	#FOR8,-(SP)	
	007152	012746	000002	MOV	#2,-(SP)	
	007156	010600		MOV	SP,RO	
	007160	104417		TRAP	C#PNTF	
	007162	062706	000006	ADD	#6,SP	
636	007166			1#:	PRINTB	#FOR9
	007166	012746	003754	MOV	#FOR9,-(SP)	
	007172	012746	000001	MOV	#1,-(SP)	
	007176	010600		MOV	SP,RO	
	007200	104414		TRAP	C#PNTB	
	007202	062706	000004	ADD	#4,SP	
637	007206			PRINTX	#FOR9.1,MEMSIZ	
	007206	013746	002504	MOV	MEMSIZ,-(SP)	
	007212	012746	004033	MOV	#FOR9.1,-(SP)	
	007216	012746	000002	MOV	#2,-(SP)	
	007222	010600		MOV	SP,RO	
	007224	104415		TRAP	C#PNTX	
	007226	062706	000006	ADD	#6,SP	
638	007232			PRINTX	#FOR9.2,R3	
	007232	010346		MOV	R3,-(SP)	
	007234	012746	004112	MOV	#FOR9.2,-(SP)	
	007240	012746	000002	MOV	#2,-(SP)	
	007244	010600		MOV	SP,RO	
	007246	104415		TRAP	C#PNTX	
	007250	062706	000006	ADD	#6,SP	
639	007254			PRINTX	#FOR9.3,R2	
	007254	010246		MOV	R2,-(SP)	
	007256	012746	004205	MOV	#FOR9.3,-(SP)	
	007262	012746	000002	MOV	#2,-(SP)	
	007266	010600		MOV	SP,RO	
	007270	104415		TRAP	C#PNTX	
	007272	062706	000006	ADD	#6,SP	
640	007276			PRINTB	#AB0	
	007276	012746	005534	MOV	#AB0,-(SP)	
	007302	012746	000001	MOV	#1,-(SP)	
	007306	010600		MOV	SP,RO	
	007310	104414		TRAP	C#PNTB	
	007312	062706	000004	ADD	#4,SP	
641	007316			ENDMSG		
	007316			L10007:	TRAP	C#MSG
	007316	104423				
642						
643						
644						
645	007320					
	007320					
646	007320			ERR6::	BGNMSG	ERR6
	007320	104421		RFLAGS	FLAG	
	007322	010037	011710	TRAP	C#RFLA	
				MOV	RO,FLAG	

GLOBAL ERROR REPORT SECTION

```

647 007326 032737 020000 011710      BIT      @IER,FLAG      ;IS INHIBIT FLAGS SET?
648 007334 001012                      BNE      1$           ;BRANCH OVER IF SET
649 007336                      PRINTF   @FOR8,L$TEST    ;PRINT TEST NUMBER
      007336 013746 002114      MOV      L$TEST,-(SP)
      007342 012746 003732      MOV      @FOR8,-(SP)
      007346 012746 000002      MOV      @2,-(SP)
      007352 010600      MOV      SP,RO
      007354 104417      TRAP    C$PNTF
      007356 062706 000006      ADD     @6,SP
650 007362                      1$: PRINTX   @FOR12
      007362 012746 004310      MOV      @FOR12,-(SP)
      007366 012746 000001      MOV      @1,-(SP)
      007372 010600      MOV      SP,RO
      007374 104415      TRAP    C$PNTX
      007376 062706 000004      ADD     @4,SP
651 007402                      PRINTB   @ABO
      007402 012746 005534      MOV      @ABO,-(SP)
      007406 012746 000001      MOV      @1,-(SP)
      007412 010600      MOV      SP,RO
      007414 104414      TRAP    C$PNTB
      007416 062706 000004      ADD     @4,SP
652 007422                      ENDMSG
      007422                      L10010: TRAP    C$MSG
      007422 104423
653
654
655                      ;NO QDMA RDY INTERRUPT ERROR
656 007424                      BGNMSG   ERR7
      007424
657 007424                      ERR7:: RFLAGS  FLAG
      007424 104421      TRAP    C$RFLA
      007426 010037 011710      MOV      RO,FLAG
658 007432 032737 020000 011710      BIT      @IER,FLAG      ;IS INHIBIT FLAGS SET?
659 007440 001012                      BNE      1$           ;BRANCH OVER IF SET
660 007442                      PRINTF   @FOR8,L$TEST    ;PRINT TEST NUMBER
      007442 013746 002114      MOV      L$TEST,-(SP)
      007446 012746 003732      MOV      @FOR8,-(SP)
      007452 012746 000002      MOV      @2,-(SP)
      007456 010600      MOV      SP,RO
      007460 104417      TRAP    C$PNTF
      007462 062706 000006      ADD     @6,SP
661 007466                      1$: PRINTX   @FOR13
      007466 012746 004364      MOV      @FOR13,-(SP)
      007472 012746 000001      MOV      @1,-(SP)
      007476 010600      MOV      SP,RO
      007500 104415      TRAP    C$PNTX
      007502 062706 000004      ADD     @4,SP
662 007506                      PRINTB   @ABO
      007506 012746 005534      MOV      @ABO,-(SP)
      007512 012746 000001      MOV      @1,-(SP)
      007516 010600      MOV      SP,RO
      007520 104414      TRAP    C$PNTB
      007522 062706 000004      ADD     @4,SP
663 007526                      ENDMSG
      007526                      L10011: TRAP    C$MSG
      007526 104423
664

```

GLOBAL ERROR REPORT SECTION

```

665
666
667 007530
    007530
668 007530
    007530 104421
    007532 010037 011710
669 007536 032737 020000 011710
670 007544 001012
671 007546
    007546 013746 002114
    007552 012746 003732
    007556 012746 000002
    007562 010600
    007564 104417
    007566 062706 000006
672 007572
    007572 012746 004435
    007576 012746 000001
    007602 010600
    007604 104414
    007606 062706 000004
673 007612
    007612 012746 004477
    007616 012746 000001
    007622 010600
    007624 104415
    007626 062706 000004
674 007632
    007632
    007632 104423
675
676
677
678 007634
    007634
679 007634
    007634 104421
    007636 010037 011710
680 007642 032737 020000 011710
681 007650 001012
682 007652
    007652 013746 002114
    007656 012746 003732
    007662 012746 000002
    007666 010600
    007670 104417
    007672 062706 000006
683 007676
    007676 012746 004536
    007702 012746 000001
    007706 010600
    007710 104415
    007712 062706 000004
684 007716
    007716 012746 005534
    007722 012746 000001

```

;UNEXPECTED DATA AFTER DMA TRANSFER ERROR

BGNMSG ERR8

ERR8::

```

RFLAGS FLAG
TRAP C#RFLA
MOV RO,FLAG
BIT #IER,FLAG
BNE 1$
PRINTF #FOR8,L#TEST
MOV L#TEST,-(SP)
MOV #FOR8,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C#PNTF
ADD #6,SP
1$: PRINTB #FOR14
MOV #FOR14,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C#PNTB
ADD #4,SP
PRINTX #FOR14.1
MOV #FOR14.1,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C#PNTX
ADD #4,SP
ENDMSG

```

```

;IS INHIBIT FLAGS SET?
;BRANCH OVER IF SET
;PRINT TEST NUMBER

```

L10012:

TRAP C#MSG

;NO QINT VNTACK INTERRUPT ERROR

BGNMSG ERR9

ERR9::

```

RFLAGS FLAG
TRAP C#RFLA
MOV RO,FLAG
BIT #IER,FLAG
BNE 1$
PRINTF #FOR8,L#TEST
MOV L#TEST,-(SP)
MOV #FOR8,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C#PNTF
ADD #6,SP
1$: PRINTX #FOR15
MOV #FOR15,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C#PNTX
ADD #4,SP
PRINTB #AB0
MOV #AB0,-(SP)
MOV #1,-(SP)

```

```

;IS INHIBIT FLAGS SET?
;BRANCH OVER IF SET
;PRINT TEST NUMBER

```

GLOBAL ERROR REPORT SECTION

007726	010600			MOV	SP,RO
007730	104414			TRAP	C#PNTB
007732	062706	000004		ADD	#4,SP
685 007736				ENDMSG	
007736				L10013:	
007736	104423			TRAP	C#MSG
686					
687					
688					
689 007740					
007740					
690 007740					
007740	104421			BGNMSG	ERR10
007742	010037	011710		ERR10::	
691 007746	032737	020000	011710	RFLAGS	FLAG
692 007754	001012			TRAP	C#RFLA
693 007756				MOV	RO,FLAG
007756	013746	002114		BIT	#IER,FLAG
007762	012746	003732		BNE	1#
007766	012746	000002		PRINTF	#FOR8,L#TEST
007772	010600			MOV	L#TEST,-(SP)
007774	104417			MOV	#FOR8,-(SP)
007776	062706	000006		MOV	#2,-(SP)
694 010002				MOV	SP,RO
010002	013746	002516		TRAP	C#PNTF
010006	012746	004770		ADD	#6,SP
010012	012746	000002		1#:	
010016	010600			PRINTX	#FOR19,HI2
010020	104415			MOV	HI2,-(SP)
010022	062706	000006		MOV	#FOR19,-(SP)
695 010026				MOV	#2,-(SP)
010026				MOV	SP,RO
010026	104423			TRAP	C#PNTX
696				ADD	#6,SP
697				ENDMSG	
698				L10014:	
699 010030				TRAP	C#MSG
010030					
700 010030					
010030	104421				
010032	010037	011710			
701 010036	032737	020000	011710		
702 010044	001012				
703 010046					
010046	013746	002114			
010052	012746	003732			
010056	012746	000002			
010062	010600				
010064	104417				
010066	062706	000006			
704 010072					
010072	012746	005104			
010076	012746	000001			
010102	010600				
010104	104415				
010106	062706	000004			
705 010112					

```

L10013:
TRAP C#MSG
;MORE THAN BITS 4 AND 5 SET FOR HIGH UNIBUS ADDRESS

```

```

ERR10::
BGNMSG ERR10
RFLAGS FLAG
TRAP C#RFLA
MOV RO,FLAG
BIT #IER,FLAG
BNE 1#
PRINTF #FOR8,L#TEST
MOV L#TEST,-(SP)
MOV #FOR8,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C#PNTF
ADD #6,SP
1#:
PRINTX #FOR19,HI2
MOV HI2,-(SP)
MOV #FOR19,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C#PNTX
ADD #6,SP
ENDMSG
L10014:
TRAP C#MSG
;ERR INTERRUPT OCCURRED ERROR

```

```

;IS INHIBIT FLAGS SET?
;BRANCH OVER IF SET
;PRINT TEST NUMBER

```

```

;IS INHIBIT FLAGS SET?
;BRANCH OVER IF SET
;PRINT TEST NUMBER

```

```

ERR11::
BGNMSG ERR11
RFLAGS FLAG
TRAP C#RFLA
MOV RO,FLAG
BIT #IER,FLAG
BNE 1#
PRINTF #FOR8,L#TEST
MOV L#TEST,-(SP)
MOV #FOR8,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C#PNTF
ADD #6,SP
1#:
PRINTX #FOR20
MOV #FOR20,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C#PNTX
ADD #4,SP
SETBITS QINT,INTBIT,@QINT

```

GLOBAL ERROR REPORT SECTION

```

010112 010146          MOV      R1,-(SP)          ;SAVE REGISTERS
010114 010546          MOV      R5,-(SP)
010116 017701 172024   MOV      @QINT,R1          ;GET @QINT REGISTER TO READ
010122 004537 012130   JSR      R5,BITS$         ;SUBROUTINE TO OUTPUT VALUES
010126 000012          .WORD   $LEN
010130 002672          .WORD   INTBIT
010132      121      111      116   .ASCIZ  /QINT/
010140 012605          MOV      (SP)+,R5          ;RESTORE REGISTERS
010142 012601          MOV      (SP)+,R1
706 010144          ENDMSG
010144          L10015:
010144 104423          TRAP    C#MSG
707
708          ;NO QDMA RDY OR ERR INTERRUPT ERROR
709
710 010146          BGNMSG  ERR12
010146          ERR12::
711 010146          RFLAGS  FLAG
010146 104421          TRAP    C#RFLA
010150 010037 011710   MOV      R0,FLAG
712 010154 032737 020000 011710   BIT      @IER,FLAG          ;IS INHIBIT FLAGS SET?
713 010162 001012          BNE     1$                 ;BRANCH OVER IF SET
714 010164          PRINTF  #FOR8,L$TEST      ;PRINT TEST NUMBER
010164 013746 002114   MOV      L$TEST,-(SP)
010170 012746 003732   MOV      #FOR8,-(SP)
010174 012746 000002   MOV      #2,-(SP)
010200 010600          MOV      SP,R0
010202 104417          TRAP    C#PNTF
010204 062706 000006   ADD     #6,SP
715 010210          1$: PRINTX  #FOR21
010210 012746 005134   MOV      #FOR21,-(SP)
010214 012746 000001   MOV      #1,-(SP)
010220 010600          MOV      SP,R0
010222 104415          TRAP    C#PNTX
010224 062706 000004   ADD     #4,SP
716 010230          PRINTB  #ABO
010230 012746 005534   MOV      #ABO,-(SP)
010234 012746 000001   MOV      #1,-(SP)
010240 010600          MOV      SP,R0
010242 104414          TRAP    C#PNTB
010244 062706 000004   ADD     #4,SP
717 010250          ENDMSG
010250          L10016:
010250 104423          TRAP    C#MSG
718
719          ;QNX NOT SET ERROR
720
721 010252          BGNMSG  ERR13
010252          ERR13::
722 010252          RFLAGS  FLAG
010252 104421          TRAP    C#RFLA
010254 010037 011710   MOV      R0,FLAG
723 010260 032737 020000 011710   BIT      @IER,FLAG          ;IS INHIBIT FLAGS SET?
724 010266 001012          BNE     1$                 ;BRANCH OVER IF SET
725 010270          PRINTF  #FOR8,L$TEST      ;PRINT TEST NUMBER
010270 013746 002114   MOV      L$TEST,-(SP)
010274 012746 003732   MOV      #FOR8,-(SP)

```



GLOBAL ERROR REPORT SECTION

```

010300 012746 000002
010304 010600
010306 104417
726 010310 062706 000006
010314
010314 012746 005221
010320 012746 000001
010324 010600
010326 104415
727 010330 062706 000004
010334
010334
010334 104423
728
729
730
731 010336
010336
732 010336
010336 104421
010340 010037 011710
733 010344 032737 020000 011710
734 010352 001012
735 010354
010354 013746 002114
010360 012746 003732
010364 012746 000002
010370 010600
010372 104417
010374 062706 000006
736 010400
010400 012746 004435
010404 012746 000001
010410 010600
010412 104414
010414 062706 000004
737 010420
010420 013746 002524
010424 012746 005343
010430 012746 000002
010434 010600
010436 104415
010440 062706 000006
738 010444
010444 012746 004477
010450 012746 000001
010454 010600
010456 104415
010460 062706 000004
739 010464
010464
010464 104423
740
741
742
743 010466
010466

```

```

MOV #2,-(SP)
MOV SP,RO
TRAP C:PNTF
ADD #6,SP
18: PRINTX #FOR22
MOV #FOR22,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C:PNTX
ADD #4,SP
ENDMSG
L10017: TRAP C:MSG

```

;UNEXPECTED DATA AFTER DMA TRANSFER ERROR IN THROTTLE TEST

```

BGNMSG ERR14
ERR14:: RFLAGS FLAG
TRAP C:RFLA
MOV RO,FLAG
BIT #IER,FLAG
BNE 18
PRINTF #FOR8,L:TEST
MOV L:TEST,-(SP)
MOV #FOR8,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C:PNTF
ADD #6,SP
18: PRINTB #FOR14
MOV #FOR14,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C:PNTB
ADD #4,SP
PRINTX #FOR24,LOOPS
MOV LOOPS,-(SP)
MOV #FOR24,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C:PNTX
ADD #6,SP
PRINTX #FOR14.1
MOV #FOR14.1,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C:PNTX
ADD #4,SP
ENDMSG
L10020: TRAP C:MSG

```

```

;IS INHIBIT FLAGS SET?
;BRANCH OVER IF SET
;PRINT TEST NUMBER

```

;NO QBOOT ERROR

```

BGNMSG ERR15
ERR15::

```

GLOBAL ERROR REPORT SECTION

```

744 010466          RFLAGS FLAG
      010466 104421 TRAP  C0RFLA
      010470 010037 011710 MOV  RO,FLAG
745 010474 032737 020000 011710 BIT  @IER,FLAG          ;IS INHIBIT FLAGS SET?
746 010502 001012    BNE  1@          ;BRANCH OVER IF SET
747 010504          PRINTF @FOR8,L@TEST          ;PRINT TEST NUMBER
      010504 013746 002114 MOV  L@TEST,-(SP)
      010510 012746 003732 MOV  @FOR8,-(SP)
      010514 012746 000002 MOV  @2,-(SP)
      010520 010600    MOV  SP,RO
      010522 104417    TRAP  C@PNTF
      010524 062706 000006 ADD  @6,SP
748 010530          1@: PRINTX @FOR25
      010530 012746 005417 MOV  @FOR25,-(SP)
      010534 012746 000001 MOV  @1,-(SP)
      010540 010600    MOV  SP,RO
      010542 104415    TRAP  C@PNTX
      010544 062706 000004 ADD  @4,SP
749 010550          L10021: ENDMSG
      010550          TRAP  C@MSG
      010550 104423
750
751          ; TIME OUT ERROR
752          ; BASIC FORMAT IS-
753          ; LSI-TEST XX TIMEOUT WAITING FOR QINT VNTACK CLEAR
754          ;
755
756 010552          BGNMSG ERR16
      010552          ERR16::
757 010552          RFLAGS FLAG
      010552 104421 TRAP  C0RFLA
      010554 010037 011710 MOV  RO,FLAG
758 010560 032737 020000 011710 BIT  @IER,FLAG          ;IS INHIBIT FLAGS SET?
759 010566 001012    BNE  1@          ;BRANCH OVER IF SET
760 010570          PRINTF @FOR8,L@TEST          ;PRINT TEST NUMBER
      010570 013746 002114 MOV  L@TEST,-(SP)
      010574 012746 003732 MOV  @FOR8,-(SP)
      010600 012746 000002 MOV  @2,-(SP)
      010604 010600    MOV  SP,RO
      010606 104417    TRAP  C@PNTF
      010610 062706 000006 ADD  @6,SP
761 010614          1@: PRINTX @FOR26
      010614 012746 005464 MOV  @FOR26,-(SP)
      010620 012746 000001 MOV  @1,-(SP)
      010624 010600    MOV  SP,RO
      010626 104415    TRAP  C@PNTX
      010630 062706 000004 ADD  @4,SP
762 010634          PRINTB @ABO
      010634 012746 005534 MOV  @ABO,-(SP)
      010640 012746 000001 MOV  @1,-(SP)
      010644 010600    MOV  SP,RO
      010646 104414    TRAP  C@PNTB
      010650 062706 000004 ADD  @4,SP
763 010654          L10022: ENDMSG
      010654          TRAP  C@MSG
      010654 104423
764

```

GLOBAL SUBROUTINES SECTION

```

766
767
768
769
770
771
772
773 010656

774
775
776
777
778
779
780
781
782
783
784
785
786 010656

787
788 010656
789
790 010656
    010656 104421
    010660 010037 011710
791 010664 032737 004000 011710
792 010672 001135
793 010674 010146
794 010676 010246
795 010700 010346
796 010702 010446
797 010704 010546
798
799 010706 012701 000014
800 010712 013702 002142
801 010716 012703 002172
802 010722 022701 000007
803 010726 001002
804 010730 000137 011410
805 010734 022701 000002
806 010740 001002
807 010742 000137 011476
808 010746 012223
809 010750 005301
810 010752 001363
811
812 010754
    010754 012746 011570
    010760 012746 000001
    010764 010600
    010766 104415
    010770 062706 000004
813 010774 012704 002331
    
```

```

.SBTTL GLOBAL SUBROUTINES SECTION

; **
; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
; THAT ARE USED IN MORE THAN ONE TEST.
; **

STARS
; *****
; **
; SUBROUTINE DUMP-
; FUNCTIONAL DESCRIPTION:
; SUBROUTINE TO PRINT THE CONTENTS OF ALL THE REGISTERS
; IT USES THE PRINTX PRINT EXTENDED FORMAT AND CAN BE DISABLED
; BY THE OPERATOR BY /FLAG:IXE, INHIBIT EXTENDED ERROR REPORTS
; INPUTS:
; NONE
; OUTPUT:
; DEVICE REGISTER DUMP PRINTED IF EXTENDED ERROR REPORTS ENABLED
; CALLING SEQUENCE:
; JSR PC,DUMP
STARS
; *****

DUMP::
; FIRST CHECK FOR INHIBIT EXTENDED ERROR REPORTS FLAG SET
RFLAGS FLAG
TRAP C#RFLA
MOV R0,FLAG
BIT #BIT11,FLAG ;IS INHIBIT FLAGS SET?
BNE EDUMP ;EXIT ROUTINE IF SET
;SAVE CPU REGISTERS
MOV R1,-(SP)
MOV R2,-(SP)
MOV R3,-(SP)
MOV R4,-(SP)
MOV R5,-(SP)

; FIRST READ THE REGISTERS
MOV #12,R1 ;NUMBER OF REGISTERS TO READ
MOV QCMD,R2 ;BASE REGISTER ADDRESS TO R2
MOV #VALUES,R3 ;RECEIVING STORAGE
E10: CMP #7,R1 ;IS IT QUBA REGISTER?
BNE 11:
JMP P1 ;BRANCH IF QUBA
11: CMP #2,R1 ;IS IT QVAD REGISTER?
BNE 12:
JMP P2 ;BRANCH IF QVAD
12: MOV (R2)+,(R3)+ ;STORE REGISTER VALUE
E5: DEC R1 ;DECREMENT LOOP COUNTER
BNE E10 ;IF NOT DONE, LOOP

; END READ, NOW PRINT THE RESULTS
PRINTX #RP1 ;PRINT FIRST LINE OF MESSAGE
MOV #RP1,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C#PNTX
ADD #4,SP
MOV #NAMES-5,R4 ;POINT R4 TO REGISTER NAMES TABLE
    
```

## GLOBAL SUBROUTINES SECTION

```

814 011000 005002          CLR      R2          ;CLEAR POINTER REGISTER
815 011002 012703 000014  MOV      #12.,R3     ;LOAD R3 AS LOOP COUNTER
816 011006 062704 000005  E64:    ADD      #5,R4       ;POINT R4 TO NEXT REGISTER NAME
817 011012 016237 002142 002452  MOV      ADDRESSES(R2),ADD ;GET ADDRESS OF REGISTER READY TO PRINT
818 011020 016237 002172 002454  MOV      VALUES(R2),VAL  ;GET VALUE OF REGISTER READY TO PRINT
819 011026          PRINTX  #R4,ADD,VAL ;PRINT REGISTER NAME, ADDRESS, VALUE
      011026 013746 002454  MOV      VAL,-(SP)
      011032 013746 002452  MOV      ADD,-(SP)
      011036 010446          MOV      R4,-(SP)
      011040 012746 011660  MOV      #R2,-(SP)
      011044 012746 000004  MOV      #4,-(SP)
      011050          MOV      SP,R0
      011052 104415          TRAP     C:PNTX
      011054 062706 000012  ADD      #12,SP
820          ;CHECK TO SEE IF ONE OF THE BIT DEFINED REGISTERS, IF YES, BRANCH
821 011060 022703 000014  CMP      #12.,R3     ;IS IT CMD REG
822 011064 001441          BEQ      CMDB
823 011066 022703 000013  CMP      #11.,R3     ;IS IT VNT REG
824 011072 001450          BEQ      VNTB
825 011074 022703 000012  CMP      #10.,R3     ;IS IT INT REG
826 011100 001457          BEQ      INTB
827 011102 022703 000011  CMP      #9.,R3      ;IS IT CTL REG
828 011106 001466          BEQ      CTLB
829 011110 022703 000010  CMP      #8.,R3      ;IS IT DMA REG
830 011114 001475          BEQ      DMAB
831 011116 022703 000003  CMP      #3.,R3      ;IS IT MT1 REG
832 011122 001504          BEQ      MT1B
833 011124 062702 000002  X4:    ADD      #2,R2       ;POINT R2 TO NEXT REGISTER DATA
834 011130 005303          DEC      R3          ;DECREMENT LOOP COUNTER
835 011132 001325          BNE     E64         ;GO DO NEXT REGISTER IF NOT DONE
836 011134          PRINTX  #RPO ;INSERT LINE FEED CARRIAGE RETURN
      011134 012746 011564  MOV      #RPO,-(SP)
      011140 012746 000001  MOV      #1,-(SP)
      011144 010600          MOV      SP,R0
      011146 104415          TRAP     C:PNTX
      011150 062706 000004  ADD      #4,SP
837 011154 012605          MOV      (SP)+,R5
838 011156 012604          MOV      (SP)+,R4
839 011160 012603          MOV      (SP)+,R3
840 011162 012602          MOV      (SP)+,R2
841 011164 012601          MOV      (SP)+,R1
842 011166 000207  EDUMP:  RTS      PC          ;RETURN
843
844          ;PROCESS PRINTING SET BIT DEFINITIONS
845 011170  CMDB:  S:BITS  CMDBIT,VAL
      011170 010146          MOV      R1,-(SP)     ;SAVE REGISTERS
      011172 010546          MOV      R5,-(SP)
      011174 013701 002454  MOV      VAL,R1      ;GET VAL REGISTER TO READ
      011200 004537 012110  JSR      R5,BIT#    ;SUBROUTINE TO OUTPUT VALUES
      011204 002604          .WORD  CMDBIT
      011206 012605          MOV      (SP)+,R5
      011210 012601          MOV      (SP)+,R1
846 011212 000461          BR      PRINTIT
847
848 011214  VNTB:  S:BITS  VNTBIT,VAL
      011214 010146          MOV      R1,-(SP)     ;SAVE REGISTERS
      011216 010546          MOV      R5,-(SP)

```

GLOBAL SUBROUTINES SECTION

```

011220 013701 002454      MOV      VAL,R1 ;GET VAL REGISTER TO READ
011224 004537 012110      JSR      R5,BIT# ;SUBROUTINE TO OUTPUT VALUES
011230 002637              .WORD     VNTBIT
011232 012605              MOV      (SP)+,R5 ;RESTORE REGISTERS
011234 012601              MOV      (SP)+,R1
849 011236 000447          BR       PRINTIT
850
851 INTB: S#BITS INTBIT,VAL
011240 010146              MOV      R1,-(SP) ;SAVE REGISTERS
011242 010546              MOV      R5,-(SP)
011244 013701 002454      MOV      VAL,R1 ;GET VAL REGISTER TO READ
011250 004537 012110      JSR      R5,BIT# ;SUBROUTINE TO OUTPUT VALUES
011254 002672              .WORD     INTBIT
011256 012605              MOV      (SP)+,R5 ;RESTORE REGISTERS
011260 012601              MOV      (SP)+,R1
852 011262 000435          BR       PRINTIT
853
854 CTLB: S#BITS CTLBIT,VAL
011264 010146              MOV      R1,-(SP) ;SAVE REGISTERS
011266 010546              MOV      R5,-(SP)
011270 013701 002454      MOV      VAL,R1 ;GET VAL REGISTER TO READ
011274 004537 012110      JSR      R5,BIT# ;SUBROUTINE TO OUTPUT VALUES
011300 002751              .WORD     CTLBIT
011302 012605              MOV      (SP)+,R5 ;RESTORE REGISTERS
011304 012601              MOV      (SP)+,R1
855 011306 000423          BR       PRINTIT
856
857 DMAB: S#BITS DMABIT,VAL
011310 010146              MOV      R1,-(SP) ;SAVE REGISTERS
011312 010546              MOV      R5,-(SP)
011314 013701 002454      MOV      VAL,R1 ;GET VAL REGISTER TO READ
011320 004537 012110      JSR      R5,BIT# ;SUBROUTINE TO OUTPUT VALUES
011324 003015              .WORD     DMABIT
011326 012605              MOV      (SP)+,R5 ;RESTORE REGISTERS
011330 012601              MOV      (SP)+,R1
858 011332 000411          BR       PRINTIT
859
860 MT1B: S#BITS MT1BIT,VAL
011334 010146              MOV      R1,-(SP) ;SAVE REGISTERS
011336 010546              MOV      R5,-(SP)
011340 013701 002454      MOV      VAL,R1 ;GET VAL REGISTER TO READ
011344 004537 012110      JSR      R5,BIT# ;SUBROUTINE TO OUTPUT VALUES
011350 003134              .WORD     MT1BIT
011352 012605              MOV      (SP)+,R5 ;RESTORE REGISTERS
011354 012601              MOV      (SP)+,R1
861
862 ;PRINT BIT DEFINITION LINE
863 PRINTIT:
864 MOV      R0,R5 ;MOVE ADDRESS TO R5
865 TSTB     (R5) ;IS IT A BLANK
866 BEQ      X# ;BRANCH IF NOTHING TO PRINT
867 PRINTB  @RP4,R5 ;PRINT LINE
011364 010546              MOV      R5,-(SP)
011366 012746 011705      MOV      @RP4,-(SP)
011372 012746 000002      MOV      @2,-(SP)
011376 010600              MOV      SP,R0
011400 104414              TRAP    C#PNTB

```

GLOBAL SUBROUTINES SECTION

```

011402 062706 000006          ADD    #6,SP
868 011406 000646          BR     X#          ;RETURN TO MAINLINE
869
870
871          ;PROCESS QUBA REGISTER (NOT NORMAL REGISTER READ
P1:        MOV    @QMT1,-(SP)      ;SAVE CURRENT QMT1
          MOV    @QINT,-(SP)      ;SAVE QINT
          CLR    @QINT            ;CLEAR QINT REGISTER
          BIS    #MIODIS,@QMT1    ;ENABLE INTERNAL LOOPBACK
          BIS    #BIT4,@QMT1     ;SELECT UBAR FUNCTION
          MOV    @QMT0,(R3)+      ;READ QUBA
          TST    (R2)+           ;CORRECT R2
          BIC    #MIODIS,@QMT1    ;DISABLE INTERNAL LOOPBACK
          BIC    #BIT4,@QMT1     ;DESELECT UBAR FUNCTION
          MOV    (SP)+,@QINT      ;RESTORE QINT
          MOV    (SP)+,@QMT1     ;RESTORE QMT1
          JMP    E5#            ;BRANCH BACK
872 011410 017746 170550
873 011414 017746 170526
874 011420 005077 170522
875 011424 052777 000200 170532
876 011432 052777 000020 170524
877 011440 017723 170516
878 011444 005722
879 011446 042777 000200 170510
880 011454 042777 000020 170502
881 011462 012677 170460
882 011466 012677 170472
883 011472 000137 010750
884
885          ;PROCESS QVAD REGISTER READ (NOT NORMAL REGISTER READ)
P2:        MOV    @QMT1,-(SP)      ;SAVE CURRENT QMT1
          MOV    @QINT,-(SP)      ;SAVE QINT
          CLR    @QINT            ;CLEAR QINT REGISTER
          BIS    #MIODIS,@QMT1    ;ENABLE INTERNAL LOOPBACK
          BIS    #MVADR,@QMT1     ;ENABLE ADDRESS REGISTER READ
          MOV    @QVAD,(R3)+      ;READ ADDRESS REGISTER
          TST    (R2)+           ;CORRECT R2
          BIC    #MIODIS,@QMT1    ;DISABLE INTERNAL LOOPBACK
          BIC    #MVADR,@QMT1     ;DISABLE ADDRESS REGISTER READ
          MOV    (SP)+,@QINT      ;RESTORE QINT
          MOV    (SP)+,@QMT1     ;RESTORE QMT1
          JMP    E5#            ;BRANCH BACK
886 011476 017746 170462
887 011502 017746 170440
888 011506 005077 170434
889 011512 052777 000200 170444
890 011520 052777 000002 170436
891 011526 017723 170434
892 011532 005722
893 011534 042777 000200 170422
894 011542 042777 000002 170414
895 011550 012677 170372
896 011554 012677 170404
897 011560 000137 010750
898
899          ;FORMATS USED IN REGISTER DUMP PRINTING
900
901 011564 045 116 061 RP0:: .ASCIZ /#N1/          ;LINE FEED CARRIAGE RETURN ONLY
902 011570 045 116 061 RP1:: .ASCIZ /#N1#S7#AREGISTER DUMP#N1#ANAME#S4#ADDRESS#S4#ACONTENTS/
903          .EVEN
904 011660 045 116 061 RP2:: .ASCIZ /#N1#T#S5#06#S5#06#S3/
905 011705 045 124 000 RP4:: .ASCIZ /#T/
906          .EVEN
907 011710 000000          FLAG: .WORD 0          ;STORES RETURNED FLAG SETTINGS
908

```

GLOBAL SUBROUTINES SECTION

```

910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928 011712 010537 012106
929 011716
    011716 010500
    011720 104434
930
931 011722 012702 000015
932 011726 004537 012010
933 011732 012702 000012
934 011736 004537 012010
935 011742 005001
936 011744
    011744 104426
    011746 110002
937 011750
    011750 103011
938 011752 022702 000000
939 011756 001406
940 011760 022702 000011
941 011764 001437
942 011766 004537 012010
943 011772 000764
944
945 011774
    011774 104435
946 011776 013705 012106
947 012002 062705 000012
948 012006 000205
949
950 012010 010237 012104
951 012014
    012014 012746 012102
    012020 012746 000001
    012024 010600
    012026 104414
    012030 062706 000004
952 012034 005201
953 012036 022701 000010
954 012042 001406
955 012044 022702 000012

```

```

*****
**
: FUNCTIONAL DESCRIPTION:
: SUBROUTINE TO PRINT OUT A TEXT OR HELP FILE
: INPUTS:
: NONE
: OUTPUTS:
: CONTENTS OF EITHER DRS.HLP OR DIAGNOSTIC DEVICE HELP FILE
: PRINTED ON CONSOLE.
: CONTENTS OF R1 AND R2 ARE DESTROYED.
: CALLING SEQUENCE:
: USES MACRO HLPMAC; FORMAT IS HLPMAC 'FILENAME.EXT'
: MACRO IS DEFINED AT END OF SUBROUTINE.
: MACRO USES FORMAT JSR R5,HLP.
: --
*****

:HLP:: MOV R5,#STORE ;SAVE RETURN ADDRESS
      OPEN R5 ;OPEN FILE NAME
      MOV R5,R0
      TRAP C#OPEN
:FIRST PRINT CR/LF
      MOV #15,R2 ;CARRIAGE RETURN TO R2
      JSR R5,60# ;GO PRINT IT
      MOV #12,R2 ;LINE FEED TO R2
      JSR R5,60# ;GO PRINT IT
      CLR R1 ;CLEAR POSITION COUNTER
10#: GETBYTE R2 ;GET BYTE COUNT OF RECORD
     TRAP C#GETB
     MOVB R0,R2
     BNCOMPLETE 50# ;BRANCH IF END OF FILE
     BCC 50#
     CMP #0,R2 ;END OF RECORDS?
     BEQ 50# ;BRANCH IF END
30#: CMP #11,R2 ;IS IT TAB CHARACTER?
     BEQ 80# ;IF YES HANDLE TAB
40#: JSR R5,60# ;GO PRINT IT
     BR 10# ;GO PROCESS NEXT CHARACTER
:END OF FILE OR LAST RECORD
50#: CLOSE ;CLOSE FILE
     TRAP C#CLOS
     MOV #STORE,R5 ;RESTORE RETURN ADDRESS
     ADD #10.,R5 ;SKIP PAST FILE NAME
     RTS R5
:PRINT AND COUNT ROUTINE
60#: MOV R2,#MESLNE ;GET CHARACTER TO PRINT IN R2
     PRINTB #HLPFOR ;GO PRINT LINE
     MOV #HLPFOR,-(SP)
     MOV #1,-(SP)
     MOV SP,R0
     TRAP C#PNTB
     ADD #4,SP
     INC R1 ;INCREMENT POSITION COUNTER
     CMP #8.,R1 ;IS IT MAXIMUM?
     BEQ 65# ;IF MAXIMUM, BRANCH
     CMP #12,R2 ;IS IT A LINE FEED?

```

## GLOBAL SUBROUTINES SECTION

956	012050	001403			BEQ	65\$				
957	012052	022702	000015		CMP	#15,R2				; IF LINE FEED, BRANCH
958	012056	001001			BNE	70\$				; IS IT A CARRIAGE RETURN?
959	012060	005001			65\$: CLR	R1				; IF NOT, BRANCH
960	012062	000205			70\$: RTS	R5				; CLEAR POSITION COUNTER
961					;TAB ROUTINE					; FROM PRINT AND COUNT ROUTINE
962	012064	012702	000040		80\$: MOV	#40,R2				; MOVE SPACE CHARACTERS
963	012070	004537	012010		JSR	R5,60\$				; GO PRINT SPACE
964	012074	005701			TST	R1				; DONE YET?
965	012076	001372			BNE	80\$				; IF NOT DONE, AGAIN
966	012100	000721			BR	10\$				; RETURN FROM TAB ROUTINE
967										
968										
969										
970	012102									
971	012102	045	101							
972	012104									
973	012104	000								
974	012105	000								
975										
976	012106	000000								

; STORAGE, FORMAT AND MESSAGE LINE  
; \$MLPFOR: :  
. ASCII /#A/  
; \$MESLNE: :  
. BYTE 0 ; RESERVED FOR MESSAGE LINE  
. BYTE 0  
. EVEN  
; \$STORE: .WORD 0 ; STORES MAIN SUBROUTINE RETURN ADDRESS



GLOBAL SUBROUTINES SECTION

978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034

012110  
012110 010246  
012112 010346  
012114 010446  
012116 012503  
012120 012737 177777 012642  
012126 000411  
012130  
012130 005037 012642  
012134 010246  
012136 010346  
012140 010446  
012142 010502  
012144 061505  
012146 005722  
012150 012203

```

*****
;
;          BITS.MAC
; PROVIDES CODE FOR A SUBROUTINE TO INTERPRET THE BITS SET
; IN A WORD (OR REGISTER), AND OPTIONALLY FOR EVERY BIT SET PRINT OUT
; THE NAME OF THAT BIT IF DEFINED IN AN ASCIZ STATEMENT, AND ALWAYS
; PUT THE NAMES IN A BUFFER.
;
; AUTHOR: DALE PROCTOR
; DATE: 12/23/81
; VERSION: 1.00
;
; ASCIZ STATEMENT FORMAT -
; LABEL: .ASCIZ /BIT15,BIT14,...,BIT10...BIT7/
; MAXIMUM NUMBER OF DEFINITIONS IS 16
;
; TO USE THE FULL ROUTINE, CALL BY USING THE MACRO SETBITS.
; SETBITS IS IN THE FORMAT-
; SETBITS NAME, POINTER, VALUE
; WHERE NAME IS AN ASCII NAME THAT WILL BE PRINTED IN THE
; MESSAGE LINE, POINTER IS AN ADDRESS THAT POINTS TO THE
; ASCIZ FORMAT STATEMENT THAT CONTAINS THE BIT NAMES, AND
; VALUE IS THE WORD TO INTERPRET AND CONVERT TO BIT NAMES.
;
; TO CREATE ONLY A PRINT BUFFER WITH THE SET BIT NAMES CALL
; THIS ROUTINE WITH THE MACRO S$BITS.
; S$BITS IS IN THE FORMAT-
; S$BITS POINTER, VALUE
; ON RETURN FROM S$BITS ROUTINE R0 CONTAINS THE ADDRESS
; OF THE BUFFER.
;
; --
;*****
;CALL BY S$BITS MACRO, NO PRINTING
BIT$::
;SAVE SOME REGISTERS
MOV R2,-(SP)
MOV R3,-(SP)
MOV R4,-(SP)
MOV (R5),R3 ;GET POINTER OF FORMAT IN R3
MOV #-1,STATU$ ;TELL ROUTINE THAT THE S$BITS MACRO CALLED
BR BUILD$ ;BRANCH OVER SETBITS STUFF
;CALL BY SETBITS MACRO, PRINT LINE IF SOMETHING TO PRINT
BITS$::
CLR STATU$ ;TELL ROUTINE THAT THE SETBITS MACRO CALLED
;SAVE SOME REGISTERS
MOV R2,-(SP)
MOV R3,-(SP)
MOV R4,-(SP)
MOV R5,R2 ;GET ADDRESS IN R2
ADD (R5),R5 ;ADD LENGTH OF DATA TO RETURN PC
TST (R2)+ ;SKIP PAST LENGTH
MOV (R2)+,R3 ;GET POINTER OF FORMAT IN R3
;REGISTER USAGE
; R0 = LOOP COUNTER
; R1 = CONTAINS REGISTER CONTENTS
; R2 = POINTS TOWARD REGISTER NAME

```

## GLOBAL SUBROUTINES SECTION

```

1035      ;      R3 = POINTS TO DEFINITION FORMAT
1036      ;      R4 = POINTS TO BUFFER
1037      ;      R5 = CONTAINS RETURN ADDRESS
1038      ;NOW BUILD BIT NAME BUFFER
1039 012152 BUILD$:  MOV      #MES2$,R4      ;POINT R4 TO BUFFER
1040 012152 012704 012431 ;ZERO OUT BUFFER
1041      MOV      #132.,R0
1042 012156 012700 000204 1$:  CLRB      (R4)+      ;CLEAR BYTE
1043 012162 105024      DEC      R0
1044 012164 005300      BNE      1$
1045 012166 001375      MOV      #MES2$,R4      ;POINT R4 BACK TO BUFFER
1046 012170 012704 012431      MOV      #16.,R0      ;NUMBER OF LOOPS TO LOOP COUNTER R1
1047 012174 012700 000020      MOV      R1,STORE$    ;SAVE ORIGINAL BIT PATTERN
1048 012200 010137 012636      CLR      NO$CHAR      ;CLEAR CHARACTER MOVED FLAG
1049 012204 005037 012640 10$: CLRB      NO$CHAR+1    ;CLEAR BIT DEFINED AND MOVED FLAG
1050 012210 105037 012641      ROL      R1           ;CHECK BIT
1051 012214 006101      BCS      20$          ;IF BITS SET, GO MOVE BIT NAME
1052 012216 103410      ;BIT WAS NOT SET, FIND NEXT COMMA
1053      5$:  CMPB      #0,(R3)      ;END OF DEFINITION?
1054 012220 122713 000000      BEQ      40$          ;IF END, GO PRINT MESSAGE
1055 012224 001433      CMPB      #'',(R3)     ;IS IT A COMMA?
1056 012226 122713 000054      BEQ      35$          ;IF COMMA, CONTINUE TO PROCESS
1057 012232 001425      INC      R3           ;POINT TO NEXT CHARACTER IN DEFINITION
1058 012234 005203      BR      5$           ;CHECK NEXT CHARACTER IF NOT A COMMA
1059 012236 000770      ;BIT WAS SET, MOVE BIT NAME IF THERE
1060      20$:  CMPB      #0,(R3)      ;IS IT END OF FORMAT?
1061 012240      BEQ      45$          ;IF END, GO PRINT MESSAGE
1062 012240 122713 000000      CMPB      #'',(R3)     ;IS IT A COMMA?
1063 012244 001425      BEQ      30$          ;IF IT WAS A COMMA, GO
1064 012246 122713 000054      ;SHOULD BE A VALID CHARACTER, MOVE TO PRINT BUFFER
1065 012252 001406      MOVB     (R3)+,(R4)+    ;MOVE CHARACTER
1066      INCB     NO$CHAR      ;INCREMENT CHARACTER FLAG
1067 012254 112324      INCB     NO$CHAR+1    ;INCREMENT BIT DEFINED AND MOVED FLAG
1068 012256 105237 012640      BR      20$          ;GO PROCESS NEXT CHAR
1069 012262 105237 012641      ;END OF CHARACTER MOVE, ADVANCE POINTER TO PUT COMMA AFTER NAME
1070 012266 000764      30$:  TSTB     NO$CHAR+1    ;WAS THE LAST BIT DEFINED
1071      BEQ      35$          ;IF LAST BIT NOT DEFINED, BRANCH
1072 012270      MOVB     #'',(R4)+    ;PUT COMMA AFTER BIT NAME
1073 012270 105737 012641      MOVB     #40,(R4)+    ;PUT SPACE AFTER COMMA
1074 012274 001404      INC      R3           ;POINT TO NEXT CHARACTER IN DEFINITION
1075 012276 112724 000054      DEC      R0           ;DECREMENT LOOP COUNTER
1076 012302 112724 000040      BNE      10$         ;IF NOT DONE, GO PROCESS NEXT BIT
1077 012306 005203      ;NOW SET UP TO PRINT THE MESSAGE
1078 012310 005300      40$:  DEC      R4           ;POINT BACK TO LAST COMMA AND SPACE
1079 012312 001336      DEC      R4
1080      MOVB     #0,(R4)      ;PUT ZERO AS TERMINATOR IN BUFFER
1081 012314      CLR      R0           ;DEFAULT RETURN CODE = NOTHING PRINTED
1082 012314 005304      TST      STATU$      ;SEE WHICH MACRO CALLED ROUTINE
1083 012316 005304      BNE      50$          ;BRANCH IF S$BITS MACRO CALLED
1084 012320 112714 000000      TST      NO$CHAR      ;SEE IF NOTHING PUT IN PRINT BUFFER
1085 012324 005000      BEQ      50$          ;IF NOTHING THERE, DON'T PRINT IT
1086 012326 005737 012642      PRINTB  #MES1$,R2,STORE$,#MES2$ ;PRINT LINE
1087 012332 001015      MOV      #MES2$,-(SP)
1088
1089
1090 012334      ;
      012334 012746 012431

```

GLOBAL SUBROUTINES SECTION

```

012340 013746 012636      MOV      STORE$, -(SP)
012344 010246              MOV      R2, -(SP)
012346 012746 012402      MOV      @MES1$, -(SP)
012352 012746 000004      MOV      @4, -(SP)
012356 010600              MOV      SP, R0
012360 104414              TRAP     C$PNTB
012362 062706 000012      ADD      @12, SP
1091                               ;RESTORE AND GO BACK
1092 012366 50$:
1093 012366 012700 012431      MOV      @MES2$, R0      ;RETURN WITH R0 POINTING TO BUFFER
1094 012372 012604              MOV      (SP)+, R4
1095 012374 012603              MOV      (SP)+, R3
1096 012376 012602              MOV      (SP)+, R2
1097 012400 000205              RTS      R5
1098
1099 012402      045      116      061 MES1$:: .ASCIZ /#N1#T#A = #06#A; #T#N1/
1100 012431      MES2$:: .BLKB 132.      ;SAVE ALL KINDS OF SPACE FOR MESSAGE LINE
1101              .EVEN
1102 012636 000000      STORE$: .WORD 0      ;STORES ORIGINAL BIT PATTERN
1103 012640 000000      NO$CHAR: .WORD 0      ;FLAG IF CHARACTERS WERE MOVED TO BUFFER
1104 012642 000000      STATU$: .WORD 0      ;STORE WHICH MACRO CALLED

```

GLOBAL SUBROUTINES SECTION

```

1106 012644
1107
1108
1109
1110 012644
1111
1112
1113
1114
1115 012644
      012644
1116 012644 005237 002450
1117 012650
      012650
      012650 000002
1118
1119
1120
1121 012652
      012652
1122 012652 011637 002454
1123 012656
      012656 013746 002454
      012662 013746 002114
      012666 012746 005566
      012672 012746 000003
      012676 010600
      012700 104417
      012702 062706 000010
1124 012706
      012706 010146
      012710 010546
      012712 017701 167230
      012716 004537 012130
      012722 000012
      012724 002672
      012726      121      111      116
      012734 012605
      012736 012601
1125 012740 005077 167202
1126 012744 052777 000100 167174
1127 012752
      012752
      012752 000002
1128

```

```

STARS
;*****
;**
;          INTERRUPT SERVICE ROUTINE
;:-
STARS
;*****
;
;GENERAL PURPOSE INTERRUPT ROUTINE TO INCREMENT A FLAG
;
      BGNSRV TRAP4
TRAP4::
      INC TRPFLG          ;SET TRAP FLAG
      ENDSRV
L10023:
      RTI
;
;UNKNOWN ERR INTERRUPT OCCURRED, TELL OPERATOR
;
      BGNSRV ERRINT
ERRINT::
      MOV (SP),VAL          ;SAVE RETURN ADDRESS FOR MESS.
      PRINTF #FORERR,L$TEST,VAL
      MOV VAL,-(SP)
      MOV L$TEST,-(SP)
      MOV #FORERR,-(SP)
      MOV #3,-(SP)
      MOV SP,R0
      TRAP C$PNTF
      ADD #10,SP
      SETBITS QINT,INTBIT,@QINT
      MOV R1,-(SP)          ;SAVE REGISTERS
      MOV R5,-(SP)
      MOV @QINT,R1          ;GET @QINT REGISTER TO READ
      JSR R5,BITS$          ;SUBROUTINE TO OUTPUT VALUES
      .WORD $LEN
      .WORD INTBIT
      .ASCIZ /QINT/
      MOV (SP)+,R5          ;RESTORE REGISTERS
      MOV (SP)+,R1
      CLR @QINT              ;CLEAR ENTIRE REGISTER
      BIS @ERRIE,@QINT      ;INABLE ERR INTERRUPT
      ENDSRV
L10024:
      RTI

```

GLOBAL SUBROUTINES SECTION

1130 012754

```

STARS
:*****
:
: MMINIT
: FUNCTIONAL DESCRIPTION-
: SUBROUTINE TO INITIALIZE MEMORY MANAGEMENT REGISTERS
: INPUTS-
: NONE
: OUTPUT-
: NONE
: CALLING SEQUENCE-
: JSR PC,MMINIT
:
: TO ENABLE MEMORY MAPPING. MOV #BITO,MMRO
:--
STARS
:*****

```

1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143

1144 012754

1145  
1146

1147 012754  
1148 012754 010146  
1149 012756 010246  
1150 012760 005001  
1151 012762 012702 172300  
1152 012766 012703 000007  
1153 012772 012712 077406  
1154 012776 010162 000040  
1155 013002 005722  
1156 013004 062701 000200  
1157 013010 005303  
1158 013012 001367  
1159 013014 012712 077406  
1160 013020 012762 177600 000040  
1161 013026 012602  
1162 013030 012601  
1163 013032 000207

```

MMINIT::
MOV R1,-(SP) ;SAVE REGISTERS
MOV R2,-(SP)
CLR R1
MOV #KISDR0,R2 ;GET DESCRIPTOR REGISTER ADDRESS
MOV #7,R3 ;DO 7 REGISTERS
1$: MOV #77406,(R2) ;SET 4K, R/W UP
MOV R1,40(R2) ;SET ADDRESS REGISTER
TST (R2)+ ;SET TO NEXT REGISTER ADDRESS
ADD #200,R1
DEC R3 ;DONE?
BNE 1$ ;IF NOT DONE
MOV #77406,(R2) ;SET I/O PAGE
MOV #177600,40(R2)
MOV (SP)+,R2 ;RESTORE REGISTERS
MOV (SP)+,R1
RTS PC

```

GLOBAL SUBROUTINES SECTION

1165 013034

1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177

1178 013034

1179

1180 013034

1181 013034

013034 104421  
013036 010005  
1182 013040 032705 001000  
1183 013044 001412  
1184 013046  
013046 013746 002114  
013052 012746 013074  
013056 012746 000002  
013062 010600  
013064 104417  
013066 062706 000006  
1185 013072 000207

1186

1187 013074

1188

045 116

STARS

::.....

```

: **
: BMSG
: FUNCTIONAL DESCRIPTION-
: SUBROUTINE TO SIMULATE THE PNT FLAG IF SET
: INPUTS-
: NONE
: OUTPUT-
: NONE, REGISTER R5 ALTERED
: CALLING SEQUENCE-
: JSR PC,BMSG
:
: ..

```

STARS

::.....

BMSG::

```

RFLAGS R5
TRAP C:RFLA
MOV RO,R5
BIT @PNT,R5 ;IS PNT FLAG SET
BEQ 100 ;BRANCH IF NOT SET
PRINTF @PNTMSG,L:TEST ;PRINT MESSAGE
MOV L:TEST,-(SP)
MOV @PNTMSG,-(SP)
MOV @2,-(SP)
MOV SP,RO
TRAP C:PNTF
ADD @6,SP
100: RETURN ;FROM SUBROUTINE

```

061 PNTMSG:: .ASCIZ /#N1#ATEST #Z2#A/<26> ;PNT FLAG MESSAGE  
.EVEN

1190 013116

1191  
 1192  
 1193  
 1194  
 1195  
 1196  
 1197  
 1198  
 1199  
 1200  
 1201  
 1202

1203 013116

1204

1205 013116 010446  
 1206 013116 010446  
 1207 013120 104421  
       013120 104421  
       013122 010004  
 1208 013124 032704 020000  
 1209 013130 001010  
 1210 013132 012746 013156  
       013136 012746 000001  
       013142 010600  
       013144 104417  
       013146 062706 000004  
 1211 013152 012604  
 1212 013154 000207  
 1213  
 1214 013156 045 116 061  
 1215  
 1216 013166  
 1217

```

STARS
;*****
;
;   ENDERR
;   FUNCTIONAL DESCRIPTION-
;   SUBROUTINE TO PRINT OUT CONTROL CODE 26 IF MESSAGE PRINTED
;   INPUTS-
;   NONE
;   OUTPUT-
;   NONE
;   CALLING SEQUENCE-
;   JSR   PC,ENDERR
;
;--
STARS
;*****
    
```

```

ENDERR::
    MOV     R4, -(SP)           ;SAVE R4
    RFLAGS R4                 ;READ FLAGS
    TRAP   C:RFLA
    MOV    RO,R4
    BIT    @IER,R4            ;INHIBIT ERROR REPORTING FLAG SET?
    BNE    101                ;BRANCH IF YES
    PRINTF @FERR              ;PRINT TERMINATING CHARACTER
    MOV    @FERR, -(SP)
    MOV    @1, -(SP)
    MOV    SP,RO
    TRAP   C:PNTF
    ADD    @4,SP
    101:  MOV    (SP)+,R4      ;RESTORE R4
    RETURN                                ;FROM SUBROUTINE

061 FERR:: .ASCIZ /#N1#A/<26> ;TERMINATING FORMAT
    .EVEN
    ENDMOD
    
```

15  
16  
45  
46 013170  
47  
48  
49  
50  
51  
52  
53 013170  
013170  
54 013170 004737 010656  
66  
67 013174  
013174 000167  
013176 000000  
79  
80  
81  
82 013200  
013200  
013200 104425

.ENABL AMA  
.SBTTL REPORT CODING SECTION

BGNMOD

\*\*\*  
; THE REPORT CODING SECTION CONTAINS THE  
; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.  
;--

BGNRPT  
L#RPT::

JSR PC.DUMP

;PRINT OUT REGISTER DUMP

EXIT RPT  
.WORD J#JMP  
.WORD L10025-2-

.EVEN

ENDRPT  
L10025:

TRAP C#RPT



PROTECTION TABLE

84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98

013202  
013202  
013202 177777  
013204 177777  
013206 177777

```

.SBTTL PROTECTION TABLE
; **
; THIS TABLE IS USED BY THE RUNTIME SERVICES
; TO PROTECT THE LOAD MEDIA.
; **

      BGNPROT
L$PROT::
      -1          ;OFFSET INTO P-TABLE FOR CSR ADDRESS
      -1          ;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
      -1          ;OFFSET INTO P-TABLE FOR DRIVE NUMBER

      ENDPROT

```

INITIALIZE SECTION

```

113
114
115
116
117
118
119
120 013210
    013210
121 013210
    013210 104433
122 013212 005077 166746
123 013216
    013216 012700 000000
    013222 104441
124 013224
    013224 012700 000036
    013230 104447
125 013232
    013232 103002
126 013234 000137 013546
127 013240
    013240 012700 000035
    013244 104447
128 013246
    013246 103035
129 013250 005737 002434
130 013254 001026
131 013256 005237 002434
132 013262 005037 002432
133 013266 012702 003271
134 013272
    013272 104407
135 013274
    013274 103405
136 013276 012737 177777 002432
137 013304 012702 003277
138 013310
    013310 010246
    013312 012746 003217
    013316 012746 000002
    013322 010600
    013324 104414
    013326 062706 000006
139
    
```

```

.SBTTL INITIALIZE SECTION

; **
; THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
; AT THE BEGINNING OF EACH PASS.
; --

L$INIT::
INITO: BGNINIT
      BRESET
      TRAP C$RESET ;RESET BUS
      CLR QMT1 ;MAKE SURE QMT1 IS REALLY CLEAR
      SETPRI @PRI00 ;INITIALIZE WITH CPI ZERO PRIORITY
      MOV @PRI00,R0
      TRAP C$SPRI
      REDEF @EF.CONTINUE
      MOV @EF.CONTINUE,R0 ; WAS THERE A CONTINUE COMMAND
      TRAP C$REFG
      BNCOMPLETE INIT1 ; IF NO, CONTINUE
      BCC INIT1
      JMP ENDI ;YES, SKIP INITIALIZATION
INIT1: REDEF @EF.NEW ;NEW PASS OR SUB PASS?
      MOV @EF.NEW,R0
      TRAP C$REFG
      BNCOMPLETE NEXT ; IF NOT NEW PASS, SKIP SETUP
      BCC NEXT
INIT2: TST BRDGLG ;HAS BRIDGE TYPE BEEN DETERMINED?
      BNE INIT3 ;IF NE YES
      INC BRDGLG ;SET FLAG
      CLR BRIDGETYPE ;CLEAR BRIDGE TYPE FLAG (Q-BRIDGE)
      MOV @QBUS,R2 ;LOAD BRIDGE TYPE PTR (Q-BRIDGE)
      READBUS ;DETERMINE BRIDGE TYPE
      TRAP C$RDBU
      BCOMPLETE 10$ ; IF COMPLETE THEN Q-BRIDGE
      BCS 10$
      MOV @-1,BRIDGETYPE ;SET BRIDGE TYPE FLAG (U-BRIDGE)
      MOV @UBUS,R2 ;LOAD BRIDGE TYPE PTR (U-BRIDGE)
10$: PRINTB @CONFIG,R2 ;IDENTIFY BRIDGE TYPE
      MOV R2,-(SP)
      MOV @CONFIG,-(SP)
      MOV @2,-(SP)
      MOV SP,R0
      TRAP C$PNTB
      ADD @6,SP
    
```

## INITIALIZE SECTION

```

141 013332 000240      INIT3:  NOP
158 013334 012737 177777 002436  SETUP:  MOV      #-1,LOGUNIT      ;INITIALIZE LOGICAL UNIT COUNTER
159 013342 005237 002436      NEXT:  INC      LOGUNIT      ;POINT TO NEXT LOGICAL UNIT
160 013346 023737 002436 002012  CMP      LOGUNIT,L#UNIT  ;HAVE WE PASSED MAXIMUM
161 013354 001525      BEQ      ABORT          ;GO IF MAXIMUM PASSED
162 013356      GPHARD LOGUNIT,PLOC    ;GET P-TABLE ADDRESS
      013356 013700 002436      MOV      LOGUNIT,RO
      013362 104442      TRAP     C#GPHRD
      013364 010037 002440      MOV      RO,PLOC
163 013370      BNCOMPLETE NEXT      ;IF NOT AVAILABLE, GET NEXT UNIT
      013370 103364      BCC      NEXT
164 013372 013737 002436 002074      MOV      LOGUNIT,L#LUN  ;STORE CURRENT LOGICAL UNIT IN HEADER
165      ;GET ALL HARDWARE TABLE ENTRIES IN STORAGE
166 013400 013700 002440      MOV      PLOC,RO        ;PLACE HEADER POINTER IN RO
167 013404 016037 000000 002442      MOV      0(RO),BASEADD  ;BASE ADDRESS OF BOARD
168 013412 016037 000002 002444      MOV      2(RO),VECTOR   ;VECTOR ADDRESS OF BOARD
169 013420 016037 000004 002446      MOV      4(RO),BRLEVEL  ;BR INTERRUPT LEVEL ON THE BOARD
170      ;FILL IN BRIDGE INTERFACE REGISTER TABLE
171 013426 013701 002442      MOV      BASEADD,R1    ;GET BASE ADDRESS IN R1
172 013432 012702 002142      MOV      #QCMD,R2      ;POINT TO REGISTER TABLE ADDRESS
173 013436 012703 000014      MOV      #12.,R3       ;# OF REGISTERS IN TABLE
174 013442 010122      10#:  MOV      R1,(R2)+      ;PLACE REGISTER ADDRESS IN TABLE
175 013444 062701 000002      ADD      #2,R1         ;ADJUST R1 TO NEXT REGISTER ADDRESS
176 013450 005303      DEC      R3            ;ARE WE DONE?
177 013452 001373      BNE     10#           ;IF NOT DONE, GO AGAIN
178      ;GET FREE MEMORY ADDRESS AND SIZE
179 013454      MEMORY FREMEM        ;GET MEMORY ADDRESS
      013454 104431      TRAP     C#MEM
      013456 010037 002502      MOV      RO,FREMEM
180 013462 017737 167014 002504      MOV      #FREMEM,MEMSIZ ;GET FREE MEMORY SIZE
181      ;CHECK MEMORY SIZE
182 013470 022737 007577 002120      CMP      #7577,L#HIME  ;COMPARE 128K TO MEMORY SIZE
183 013476 100003      BPL     25#           ;OK IF SIZE <= 128K
184 013500 012737 007577 002120      MOV      #7577,L#HIME  ;RESET MEMORY SIZE TO 128K
185      ;SET UP FOR UNEXPECTED INTERRUPTS
186 013506      25#:  SETVEC VECTOR,#ERRINT,#PRI07 ;SET UP INTERRUPT VECTOR
      013506 012746 000340      MOV      #PRI07,-(SP)
      013512 012746 012652      MOV      #ERRINT,-(SP)
      013516 013746 002444      MOV      VECTOR,-(SP)
      013522 012746 000003      MOV      #3,-(SP)
      013526 104437      TRAP     C#SVEC
      013530 062706 000010      ADD      #10,SP
187 013534 052777 000100 166404      BIS      #ERRIE,#QINT  ;ENABLE ERROR INTERRUPT
188      ;SET UP AND ENABLE MEMORY MANAGEMENT
189 013542 004737 012754      JSR     PC,#MINIT     ;SET UP MM REGISTERS
190      ;SET UP FOR UNEXPECTED INTERRUPTS
191 013546      ENDI: SETVEC VECTOR,#ERRINT,#PRI07 ;SET UP INTERRUPT VECTOR
      013546 012746 000340      MOV      #PRI07,-(SP)
      013552 012746 012652      MOV      #ERRINT,-(SP)
      013556 013746 002444      MOV      VECTOR,-(SP)
      013562 012746 000003      MOV      #3,-(SP)
      013566 104437      TRAP     C#SVEC
      013570 062706 000010      ADD      #10,SP
192 013574 052777 000100 166344      BIS      #ERRIE,#QINT  ;ENABLE ERROR INTERRUPT
193      ;SET UP DEFAULT TIMEOUT VALUE
194 013602 012737 000003 002532      MOV      #3,DEFAULT
195 013610 005737 002432      TST     BRIDGETYPE    ;3 SECOND DEFAULT TIMEOUT FOR 11/23
                          ;TEST FOR Q-BRIDGE

```

INITIALIZE SECTION

```

196 013614 001403      BEQ      308
197 013616 012737 000005 002532      MOV      #5,DEFAULT
198 013624      104432      308:    EXIT    INIT
      013626 000004      TRAP    C#EXIT
199 013630      104444      ABORT:  .WORD  L10027-.
      013630      TRAP    C#DCLN
200
224
225
237
238
239
240 013632      .EVEN
      013632      ENDINIT
      013632 104411      L10027: TRAP    C#INIT

```

```

;IF EQ THEN YES: CONTINUE
;5 SEC DEFAULT TIMEOUT FOR 11/24
;THAT'S ALL TO INITIALIZE PASS

```

```

;DO CLEANUP AND ABORT THE PASS

```

AUTODROP SECTION

242  
 243  
 244  
 245  
 246  
 247  
 248  
 249  
 250  
 251 013634  
       013634  
 252  
 253 013634 000240  
 254  
 261  
 262 013636  
       013636  
       013636 104461

```
.SBTTL AUTODROP SECTION
; **
; THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
; THE "ADR" FLAG WAS SET. THE UNIT(S) UNDER TEST ARE CHECKED TO
; SEE IF THEY WILL RESPOND. THOSE THAT DON'T ARE IMMEDIATELY
; DROPPED FROM TESTING.
; --
                BGNAUTO
L$AUTO::
                NOP                ; NO ON-LINE SIGNAL PROVIDED TO USER FOR AUTODROP

                ENDAUTO
L10030:        TRAP    C$AUTO
```

CLEANUP CODING SECTION

```

264
265
266
267
268
269
270
271 013640
    013640
272
273
274 013640 013701 002142
275 013644 012702 002172
276 013650 012703 000014
277 013654 012122
278 013656 077302
279
280 013660 042777 000100 166254
281 013666 042777 000100 166250
282 013674 042777 000100 166244
283 013702 042777 000100 166242
284 013710
    013710 013700 002444
    013714 104436
285
305
306
307
308 013716
    013716
    013716 104412
    
```

```

.SBTTL CLEANUP CODING SECTION
; **
; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
; AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
; --

        BGNCLN
L$CLEAN::

; START BY READING AND STORING THE REGISTER VALUES FOR DEBUG
        MOV     QCMD,R1           ;POINT R1 TO Q COMMAND REGISTER
        MOV     @VALUES,R2       ;POINT R2 TO STORAGE AREA
        MOV     @12.,R3          ;# OF REGISTER TO READ TO R3
10$:    MOV     (R1),.(R2),      ;READ AND SAVE REGISTER
        SOB     R3,10$          ;SEE IF DONE, BRANCH IF NOT
; CLEAR INTERRUPT ENABLES IN ALL THE REGISTERS
        BIC     @PRSIE,@QCMD
        BIC     @ACKIE,@QVNT
        BIC     @ERRIE,@QINT
        BIC     @RDYIE,@QDMA
        CLRVEC VECTOR           ;CLEAR INTERRUPT VECTOR
        MOV     VECTOR,R0
        TRAP   C$CVEC

        .EVEN
        ENDCLN
L10031: TRAP   C$CLEAN
    
```

DROP UNIT SECTION

```

310 .SBTTL DROP UNIT SECTION
311
312 ;**
313 ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
314 ; TO NO LONGER BE TESTED.
315 ;--
316
317 013720 BGNDU
013720 L#DU::
318
319 013720 000240 NOP ;NO SPECIAL DROP UNIT CODING NEEDED
320
329
330 013722 EXIT DU
013722 000167 .WORD J$JMP
013724 000000 .WORD L10032-2-.
331
343
344 .EVEN
345
346 013726 ENDDU
013726 L10032:
013726 104453 TRAP C#DU

```

ADD UNIT SECTION

348  
 349  
 350  
 351  
 352  
 353  
 354  
 355  
 356 013730  
 013730  
 357  
 358 013730 000240  
 359  
 368  
 369 013732  
 013732 000167  
 013734 000000  
 370  
 382  
 383  
 384  
 385 013736  
 013736  
 013736 104452  
 386  
 387 013740  
 388

```

.SBTTL  ADD UNIT SECTION
; **
; THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
; TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
; TO THE TEST CYCLE.
; --
L$AU::  BGNAU
NOP          ;NO SPECIAL ADD UNIT CODING NEEDED

EXIT  AU
.WORD J$JMP
.WORD L10033-2-.

.EVEN
L10033: ENDAU
TRAP  C$AU
ENDMOD

```



M5

CZFPB-A.O CTRL BRIDGE LINK

MACRO M1200 09-JAN-94 12:25 PAGE 38

SEQ 0064

MAINLINE TEST CONTROL

1  
2  
14

.SBTTL MAINLINE TEST CONTROL  
.ENABL AMA  
.DSABL GBL

MAINLINE TEST CONTROL

```

52 013740          BGNMOD
53                ;*****
54                ;**
55                ;
56                ;   MAINLINE TEST CONTROL MODULE
57                ;
58                ;--
59                ;*****
60 013740          BGNTST
   013740          T1::
61
62 013740          START:
63                ;
64                ; FIRST TEST QINT CMDPRS AND LOOP UNTIL SET
65                ;
66 013740          2$:   BREAK                ; ALLOW CONTROL C
   013740 104422   TRAP                C#BRK
67 013742 032777   BIT                #CMDPRS,@QINT ; IS CMDPRS SET
68 013750 001773   BEQ                2$         ; IF NOT, TRY AGAIN
69                ;
70                ; GET FUNCTION NUMBER AND SET READY TO SEND BACK
71                ;
72 013752 017701   MOV                @QCMD,R1    ; READ QCMD
73 013756 042701   BIC                #177701,R1 ; GET RID OF UNWANTED BITS
74 013762 006201   ASR                R1        ; PUT IN PROPER PLACE
75 013764 010137   MOV                R1,TNUM   ; STORE ORIGINAL VALUE
76 013770 000301   SWAB               R1        ; TEST NUMBER TO ECHO BACK
77 013772 010177   MOV                R1,@QCMD  ; PUT PATTERN BACK IN QCMD
78                ;
79                ; CALCULATE TEST NUMBER AND GO DO IT AS A SUBROUTINE
80                ;
81 013776 013701   MOV                TNUM,R1   ; GET TEST NUMBER BACK IN R1
82 014002 005237   INC                TNUM     ; INCREMENT TNUM TO REAL TEST NUMBER
83 014006 023737   CMP                TNUM,NUM#TST ; TEST # TOO LARGE
84 014014 003051   BGT                BAD          ; BRANCH IF TOO LARGE
85 014016 006301   ASL                R1        ; MULTIPLY BY 2 TO GET WORD VALUE
86 014020 022771   CMP                #CONVERSATION,@TESTS(R1) ; IS IT CONVERSATION TEST
87 014026 001403   BEQ                5$         ; IF CONVERSION TEST, SKIP CMDACK SET
88 014030 052777   BIS                #CMDACK,@QCMD ; TELL UBRIDGE WHAT WAS SENT
89 014036 004771   JSR                PC,@TESTS(R1) ; GO EXECUTE TEST AS SUBROUTINE
90                ;
91                ; PROGRAM RETURNS HERE AFTER TEST IS COMPLETE
92                ; R1 RETURNS A CODE AS FOLLOWS:
93                ; IF R1 = ABORT, SKIP EVENT HANDSHAKE
94                ; R1 = ANYTHING ELSE, NUMBER OF SECONDS FOR THE TIME OUT
95                ;
96 014042 012737   MOV                #1,L#TEST   ; RESTORE REAL DRS TEST NUMBER
97 014050 020127   CMP                R1,#ABORT        ; ABORT RETURNED FROM TEST?
98 014054 001731   BEQ                START          ; IF ABORT, SKIP EVENT HANDSHAKE
99 014056 052777   BIS                #VNTPRS,@QVNT    ; SET VNTPRS, EVENT COMPLETE
100 014064 005701  TST                R1            ; CHECK TO SEE IF ZERO (DEFAULT)
101 014066 001002  BNE                10$           ; IF NOT ZERO, BRANCH
102 014070 013701  MOV                DEFAULT,R1     ; DEFAULT VALUE FOR TIMEOUT
103 014074 012702  MOV                #TIMOUT,R2     ; SET UP FOR TIME OUT
104 014100 032777  BIT                #VNTACK,@QINT  ; IS EVENT ACKNOWLEDGE SET
105 014106 001314  BNE                START          ; IF SET, GO BACK TO START
106 014110          BREAK           ; ALLOW CONTROL C

```

MAINLINE TEST CONTROL

107	014110	104422				TRAP	C:BRK		
	014112	005302				DEC	R2		: DECREMENT TIME OUT COUNTER
108	014114	001371				BNE	308		: IF NOT TIME OUT, TRY AGAIN
109	014116	005301				DEC	R1		: DECREMENT SECOND COUNT
110	014120	001365				BNE	108		: GO WAIT ANOTHER SECOND
111									
112									
113									
114	014122								
	014122	104456				ERRMRD	1, ERRO		: TELL OPERATOR VNTACT TIME OUT
	014124	000001				TRAP	C:ERMRD		
	014126	000000				.WORD	1		
	014130	006270				.WORD	0		
115	014132	004737	013116			.WORD	ERRO		
116	014136	000700				CALL	ENDERR		: GO PRINT OUT ENDING TERMINATOR
117	014140					BR	START		: GO TO START ANYWAY
	014140	012746	014170			BAD: PRINTB	@BADMSG		: TELL OPERATOR TOO LARGE TEST NUMBER
	014144	012746	000001			MOV	@BADMSG, -(SP)		
	014150	010600				MOV	@1, -(SP)		
	014152	104414				MOV	SP, R0		
	014154	062706	000004			TRAP	C:PNTB		
118	014160	004737	013116			ADD	@4, SP		
119	014164					CALL	ENDERR		: GO PRINT OUT ENDING TERMINATOR
	014164	104432				EXIT	TST		
	014166	000030				TRAP	C:EXIT		
120	014170	045	116	061	BADMSG:	.WORD	L10034		
121						.ASCIZ	/'@N1@ATEST @ TOO LARGE/'		
122	014216					.EVEN			
	014216					ENDTST			
	014216	104401			L10034:	TRAP	C:ETST		

TEST 1: CMD PRS - CMD ACK

124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137

```
.SBTTL TEST 1: CMD PRS - CMD ACK
;*****
;
;TEST DESCRIPTION-
; CHECKS CMD PRS - CMD ACK
;TEST STEPS-
; U CMD QIO
; Q SET CMD CMD ACK
;
; THIS TEST IS DONE IN THE INITIAL HANDSHAKE BEFORE THE TESTS ARE EXECUTED.
; THIS ROUTINE IS JUST A DUMMY.
;--
;*****
```

138 014220  
014220  
014220 012737 000001 002114  
014226 004737 013034  
139 014232 012701 013630  
140 014236 000207

```
STATST
TEST1::
MOV #1,L1TEST ;TEST NUMBER FOR ERROR MESSAGES
CALL BMSG ;CHECK FOR PNT FLAG
MOV #ABORT,R1 ; FORCE ABORT
RTS PC ; RETURN TO MAINLINE
```

Df.

TEST 2: COMMAND REGISTER

```

142
143
144
145
146
147
148
149
150
151
152
153
154
155
156 014240
    014240
    014240 012737 000002 002114
    014246 004737 013034
157 014252 012737 000002 002456
158 014260 005037 002474
159 014264 013701 002532
160 014270 005237 002474
161 014274 012702 026000
162 014300
    014300 104422
163 014302 032777 004000 165636
164 014310 001017
165 014312 005302
166 014314 001371
167 014316 005301
168 014320 001365
169
170
171
172 014322 012701 000000
173 014326
    014326 104456
    014330 000002
    014332 000000
    014334 006316
174 014336 012701 013630
175 014342 004737 013116
176 014346 000451
177
178
179
180
181
182 014350
183 014350 017737 165566 002454
184 014356 042737 177701 002454
185 014364 023737 002454 002456
186 014372 001410
187
188
189
190 014374 012701 000000

```

```

.SBTTL TEST 2: COMMAND REGISTER
;*****
;..
;TEST DESCRIPTION-
; CHECKS CMDPRS - CMDACK
; CHECKS ALL REGISTER DATA BITS
;TEST STEPS-
; U   CMD QIO WITH MOVING BIT IN CMD REGISTER
; Q   CHECK BITS AND RETURN THEM IN BITS 8-15
; Q   SET CMDACK
; U   REPEAT FOR ALL BITS
;--
;*****
;
;          STATST
TEST2::
;
;          MOV     #2,L0TEST          ;TEST NUMBER FOR ERROR MESSAGES
;          CALL   BMSG                ;CHECK FOR PNT FLAG
;          MOV     #2,EXP              ; EXPECTED INITIAL PATTERN TO EXP
;          CLR    COUNTER              ; CLEAR LOOP COUNTER
;          MOV     DEFAULT,R1          ; NUMBER OF SECONDS TO TIMEOUT
;          INC    COUNTER              ; INCREMENT LOOP COUNTER
;          MOV     #100,R2             ; TIME OUT VALUE TO R2
;          BREAK  120                  ; ALLOW CONTROL C
;          TRAP   C0BRK                ;
;          BIT    @CMDPRS,BQINT        ; COMMAND PRESENT?
;          BNE    200                  ; IF COMMAND PRESENT, BRANCH
;          DEC    R2                    ; DECREMENT TIMEOUT REGISTER
;          BNE    120                  ; IF NOT IN TIMEOUT, TRY AGAIN
;          DEC    R1                    ; DECREMENT SECOND TIMER
;          BNE    100                  ; IF NOT TIMEOUT, TRY AGAIN
;
;          ; TIMEOUT ERROR CONDITION
;
;          MOV     #0,R1                ; REGISTER NUMBER TO R1
;          ERRHRD 2,,ERR1              ; TELL OPERATOR TIMEOUT
;          TRAP   C0ERRHRD
;          .WORD  2
;          .WORD  0
;          .WORD  ERR1
;          MOV     @ABORT,R1           ; ABORT CODE TO R1
;          CALL   ENDERR               ; GO PRINT OUT ENDING TERMINATOR
;          BR     1000                 ; ABORT TEST
;
;          ; END TIMEOUT ERROR CONDITION
;          ; READ QCMD BITS 1-5 FOR BIT PATTERN AND CHECK
;
;          200:
;          MOV     @QCMD,VAL           ; READ COMMAND REGISTER
;          BIC    @177701,VAL          ; CLEAR OUT UNWANTED BITS
;          CMP    VAL,EXP               ; COMPARE EXPECTED WITH VALUE
;          BEQ    300                  ; BRANCH IF OK
;
;          ; ERROR CONDITION, DATA NOT WHAT EXPECTED
;
;          MOV     #0,R1                ; REGISTER NUMBER TO R1

```

## TEST 2: COMMAND REGISTER

```

191 014400          ERRMRD 3.,MSG2,ERR3          ; TELL OPERATOR DATA ERROR
    014400 104456   TRAP    C#ERRMRD
    014402 000003   .WORD   3
    014404 006012   .WORD   MSG2
    014406 006652   .WORD   ERR3
192 014410 004737 013116   CALL    ENDERR          ; GO PRINT OUT ENDING TERMINATOR
193
194
195
196
197
198 014414          ;
199 014414 013702 002456   300:   MOV    EXP,R2          ; SET UP TO OUTPUT EXPECTED VALUE
200 014420 006202          ASR    R2              ; MOVE OVER 1 BIT
201 014422 000302          SWAB   R2              ; GET INTO HIGH BYTE
202 014424 010204          MOV    R2,R4          ; STORE THIS VALUE
203 014426 012703 000005   MOV    #5,R3         ; LOOP 5 TIMES TO EXERCISE HI BITS
204 014432 006302          400:   ASL    R2              ; MOVE BIT LEFT
205 014434 005303          DEC    R3            ; DECREMENT COUNTER
206 014436 001375          BNE   400            ; BRANCH BACK IF NOT DONE
207 014440 050402          BIS   R4,R2         ; SET LOW BIT BACK ON
208 014442 010277 165474   MOV    R2,@QCMD     ; SEND TO QCMD
209 014446 052777 000001 165466   BIS   @CMDACK,@QCMD ; TELL MASTER WHAT WAS
210
211
212
213 014454          ;
214 014454 006337 002456   500:   ASL    EXP          ; SHIFT EXPECTED PATTERN LEFT
215 014460 032737 000100 002456   BIT   @BIT06,EXP   ; ALREADY IN LAST POSITION?
216 014466 001676          BEQ   500            ; GO TRY WITH NEXT PATTERN
217
218
219
220 014470 005001          ;
221 014472          ;
222 014472 000207          900:   CLR    R1          ; TEST END NORMALLY FLAG
                                1000:  RTS    PC          ; EXIT TEST TO MAINLINE

```

TEST 3: EVENT REGISTER

```

224 .SBTTL TEST 3: EVENT REGISTER
225 ; .....
226 ; ..
227 ; TEST DESCRIPTION-
228 ; CHECKS VNT PRS - VNT ACK
229 ; CHECKS ALL REGISTER DATA BITS
230 ;
231 ; TEST STEPS-
232 ; U VNT QIO WITH MOVING BIT IN VNT REGISTER
233 ; Q SET BITS 8-15 AND VNT PRS
234 ; Q CHECK BITS
235 ; U CHECK BITS 8-15 IN IOSB
236 ; REPEAT FOR ALL BITS
237 ; --
238 ; .....
239 ;
240 014474 STATST
014474 TEST3::
014474 012737 000003 002114 MOV #3,L#TEST ; TEST NUMBER FOR ERROR MESSAGES
014502 004737 013034 CALL BMSG ; CHECK FOR PNT FLAG
241 014506 005037 002474 CLR COUNTER ; CLEAR LOOP COUNTER
242 014512 012737 000002 002456 MOV #2,EXP ; INITIAL PATTERN TO EXP
243 ;
244 ; FIRST SEND PATTERN TO MASTER
245 ;
246 014520 100:
247 014520 013702 002456 MOV EXP,R2 ; SET UP TO OUTPUT RECEIVED VALUE
248 014524 006202 ASR R2 ; MOVE OVER 1 BIT
249 014526 000302 SWAB R2 ; GET INTO HIGH BYTE
250 014530 010204 MOV R2,R4 ; STORE THIS VALUE
251 014532 012703 000005 MOV #5,R3 ; LOOP 5 TIMES TO EXERCISE HI BITS
252 014536 006302 200: ASL R2 ; MOVE BIT LEFT
253 014540 005303 DEC R3 ; DECREMENT COUNTER
254 014542 001375 BNE 200 ; BRANCH BACK IF NOT DONE
255 014544 050402 BIS R4,R2 ; SET LOW BIT BACK ON
256 014546 010277 165372 MOV R2,@QVNT ; SEND TO QVNT
257 014552 052777 000001 165364 BIS @VNTPRS,@QVNT ; TELL MASTER WHAT WAS
258 ;
259 ; NOW WAIT FOR EVENT ACKNOWLEDGED
260 ;
261 014560 005237 002474 INC COUNTER ; INCREMENT LOOP COUNTER
262 014564 013701 002532 MOV DEFAULT,R1 ; NUMBER OF SECONDS TO TIME OUT
263 014570 012702 026000 300: MOV #TIMOUT,R2 ; TIME OUT VALUE TO R2
264 014574 400: BREAK ; ALLOW CONTROL C
014574 104422 TRAP C#BRK
265 014576 032777 002000 165342 BIT @VNTACK,@QINT ; EVENT PRESENT?
266 014604 001017 BNE 500 ; IF EVENT PRESENT, BRANCH
267 014606 005302 DEC R2 ; DECREMENT TIMEOUT REGISTER
268 014610 001371 BNE 400 ; IF NOT IN TIMEOUT, TRY AGAIN
269 014612 005301 DEC R1 ; DECREMENT SECOND TIMER
270 014614 001365 BNE 300 ; IF NOT TIMEOUT, AGAIN
271 ;
272 ; TIMEOUT ERROR CONDITION
273 ;
274 014616 012701 000001 MOV #1,R1 ; REGISTER NUMBER TO R1
275 014622 ERRHRD 4,,ERR1 ; TELL OPERATOR TIMEOUT
014622 104456 TRAP C#ERRHD
    
```

## TEST 3: EVENT REGISTER

```

014624 000004          .WORD 4
014626 000000          .WORD 0
014630 006316          .WORD ERR1
276 014632 012701 013630  MOV #ABORT,R1          ; ABORT CODE TO R1
277 014636 004737 013116  CALL ENDERR           ; GO PRINT OUT ENDING TERMINATOR
278 014642 000431          BR 100#              ; ABORT TEST
279
280          ; END TIMEOUT ERROR CONDITION
281
282          ; READ QVNT BITS 1-5 FOR BIT PATTERN AND CHECK
283
284 014644          50#:
285 014644 017737 165274 002454  MOV #QVNT,VAL          ; READ QVNT
286 014652 042737 177701 002454  BIC #177701,VAL       ; CLEAR OUT UNWANTED BITS
287 014660 023737 002454 002456  CMP VAL,EXP           ; COMPARE EXPECTED WITH VALUE
288 014666 001410          BEQ 60#              ; BRANCH IF OK
289
290          ; ERROR CONDITION, DATA NOT WHAT EXPECTED
291
292 014670 012701 000001          MOV #1,R1              ; REGISTER NUMBER TO R1
293 014674          ERRHRD 5.,MSG3,ERR3          ; TELL OPERATOR DATA ERROR
      014674 104456          TRAP C#ERRHRD
      014676 000005          .WORD 5
      014700 006061          .WORD MSG3
      014702 006652          .WORD ERR3
294 014704 004737 013116          CALL ENDERR           ; GO PRINT OUT ENDING TERMINATOR
295
296          ; END ERROR CONDITION
297
298          ; CHECK FOR END OF TEST, IF NOT END, NEW EXPECTED PATTERN
299
300 014710          60#:
301 014710 006337 002456          ASL EXP                ; SHIFT EXPECTED PATTERN LEFT
302 014714 032737 000100 002456  BIT #BIT06,EXP        ; ALREADY IN LAST POSITION?
303 014722 001676          BEQ 10#              ; GO TRY WITH NEXT PATTERN
304
305          ; EXIT TEST ROUTINE
306
307 014724 005001          90#: CLR R1              ; TEST END NORMALLY FLAG
308 014726          100#:
309 014726 000207          RTS PC                ; EXIT TEST TO MAINLINE

```



TEST 4: VIEW REGISTER READ

```

311 .SBTTL TEST 4: VIEW REGISTER READ
312 ;*****
313 ;**
314 ;TEST DESCRIPTION-
315 ; WRITES DATA TO VIEW REGISTER
316 ; READS DATA FROM VIEW REGISTER
317 ;
318 ;TEST STEPS-
319 ; U LOAD ADDRESS VALUES IN VIEW BUFFER
320 ; U CMD QIO USING VIEW REGISTER BUFFER
321 ; Q READ ALL VIEW REGISTER LOCATIONS AND CHECK
322 ; Q LOAD ADDRESS VALUES IN VIEW REGISTER
323 ; U VNT QIO USING VIEW REGISTER BUFFER
324 ; Q SET UNT PRS
325 ; U CHECK VIEW REGISTER BUFFER
326 ;--
327 ;*****
328
329 014730 STATST
014730 TEST4::
014730 012737 000004 002114 MOV #4,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
014736 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
330 014742 013701 002532 MOV DEFAULT,R1 ; SECONDS OF TIME OUT
331 014746 012702 026000 8#: MOV #TIMOUT,R2 ; TIME OUT VALUE TO R2
332 014752 104422 10#: BREAK
014752 104422 TRAP C#BRK
333 014754 032777 004000 165164 BIT #CMDPRS,#QINT ; COMMAND PRESENT?
334 014762 001017 BNE 20# ; IF COMMAND PRESENT, BRANCH
335 014764 005302 DEC R2 ; DECREMENT TIMEOUT REGISTER
336 014766 001371 BNE 10# ; IF NOT IN TIMEOUT, TRY AGAIN
337 014770 005301 DEC R1 ; DECREMENT SECOND TIMER
338 014772 001365 BNE 8# ; IF NOT TIMEOUT, AGAIN
339 ;
340 ; TIMEOUT ERROR CONDITION
341 ;
342 014774 012701 000000 MOV #0,R1 ; REGISTER # TO R1
343 015000 ERRHRD 6,,ERR2 ; TELL OPERATOR TIMEOUT#
015000 104456 TRAP C#ERRRD
015002 000006 .WORD 6
015004 000000 .WORD 0
015006 006506 .WORD ERR2
344 015010 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
345 015014 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
346 015020 000524 BR 100# ; ABORT TEST
347 ;
348 ; END TIMEOUT ERROR CONDITION
349 ;
350 015022 20#:
351 ;
352 ; SET UP TO READ VIEW RAM
353 ;
354 015022 052777 000020 165120 BIS #QVREN,#QCTL ; READ ENABLE
355 015030 012777 000000 165130 MOV #0,#QVAD ; SET READ RAM INITIAL ADD. TO ZERO
356 015036 013701 002502 MOV FREMEM,R1 ; START OF BUFFER TO R1
357 015042 012702 002000 MOV #2000,R2 ; LOOP COUNTER TO R2
358 ;
359 ; READ 1K VIEW RAM, PLACE CONTENTS IN BUFFER

```

## TEST 4: VIEW REGISTER READ

```

360
361 015046
362 015046 017721 165116
363 015052 005302
364 015054 001374
365
366
367
368 015056 005037 002462
369 015062 012737 000000 002456
370 015070 013701 002502
371 015074 012737 000000 002456
372
373
374
375 015102
376 015102 012137 002454 002454
377 015106 023737 002456 002454
378 015114 001434
379
380
381
382 015116 005237 002462
383 015122 022737 000001 002462
384 015130 001004
385 015132
   015132 104456
   015134 000007
   015136 006130
   015140 007012
386 015142 022737 000011 002462 50:
387 015150 100416
388 015152
   015152 013746 002456
   015156 013746 002454
   015162 013746 002456
   015166 012746 003574
   015172 012746 000004
   015176 010600
   015200 104415
   015202 062706 000012
389
390
391
392 015206
393 015206 005237 002456 002000
394 015212 023727 002456 002000
395 015220 001330
396
397
398
399 015222 005737 002462
400 015226 001414
401
402
403
404 015230

```

```

;
30:
MOV    @QVDA,(R1)+      ; READ AND PLACE IN BUFFER
DEC    R2                ; DECREMENT LOOP COUNTER
BNE    30:               ; IF NOT DONE, AGAIN

;
; DONE READING RAM, NOW SET UP TO CHECK IT
;
CLR    ERROR1           ; INITIAL ERROR COUNT
MOV    #0,EXP           ; INITIAL EXPECTED DATA VALUE
MOV    FREMEM,R1        ; START OF BUFFER TO R1
MOV    #0,EXP           ; LOOP COUNTER AND ADDRESS VALUE

;
; NOW CHECK THE VALUES
;
40:
MOV    (R1)+,VAL        ; MOVE READ VALUE TO VAL
CMP    EXP,VAL          ; COMPARE EXPECTED WITH VALUE
BEQ    60:              ; BRANCH IF OK

;
; ERROR CONDITION, BAD RAM VALUE
;
INC    ERROR1           ; INCREMENT ERROR COUNTER
CMP    #1,ERROR1        ; IS IT THE FIRST ERROR?
BNE    50:              ; IF NOT FIRST ERROR, BRANCH PRINT HEADER
ERRHRD 7.,MSG4,ERR4     ; PRINT ERROR HEADER ONLY
TRAP   C#ERRHD
.WORD  7
.WORD  MSG4
.WORD  ERR4

50:
CMP    #9.,ERROR1      ; SEE IF TOO MANY ERRORS TO PRINT
BMI    60:              ; BRANCH IF TOO MANY ERRORS TO PRINT
PRINTX #FOR5,EXP,VAL,EXP ; PRINT DATA MESSAGE
MOV    EXP,-(SP)
MOV    VAL,-(SP)
MOV    EXP,-(SP)
MOV    #FOR5,-(SP)
MOV    #4,-(SP)
MOV    SP,R0
TRAP   C#PNTX
ADD    #12,SP

;
; END ERROR CONDITION
;
60:
INC    EXP              ; INCREMENT ADDRESS VALUE AND LOOP COUNTER
CMP    EXP,#2000        ; IS IT DONE YET
BNE    40:              ; IF NOT DONE, GO CHECK NEXT VALUE

;
; TEST DONE, PRINT FINAL ERROR TOTALS IF ANY
;
TST    ERROR1           ; ANY ERRORS
BEQ    70:              ; BRANCH IF NO ERRORS

;
; PRINT FINAL ERROR TOTALS
;
PRINTB #FOR6,ERROR1    ; PRINT # OF ERRORS

```

TEST 4: VIEW REGISTER READ

```

015230 013746 002462      MOV     ERROR1,-(SP)
015234 012746 003622      MOV     #FOR6,-(SP)
015240 012746 000002      MOV     #2,-(SP)
015244 010600              MOV     SP,R0
015246 104414              TRAP   C#PNTB
015250 062706 000006      ADD    #6,SP
405 015254 004737 013116      CALL   ENDERR                ; GO PRINT OUT ENDING TERMINATOR
406
407                          ; SET QCMD CMDACK TO COMPLETE HANDSHAKE
408                          ;
409 015260 052777 000001 164654 70#:  BIS    #CMDACK,#QCMD        ; SET CMDACK
410 015266 012701 000005              MOV     #5.,R1              ; TEST END NORMALLY WITH 5 SEC DELAY
411 015272
412 015272 000207              RTS     PC                  ; EXIT TEST TO MAINLINE

```

TEST 5: VIEW REGISTER WRITE

```

414 .SBTTL TEST 5: VIEW REGISTER WRITE
415 ;*****
416 ;**
417 ;TEST DESCRIPTION-
418 ; WRITES DATA TO VIEW REGISTER
419 ;
420 ;TEST STEPS-
421 ; U CMD QIO USING VIEW REGISTER BUFFER
422 ; Q LOAD ADDRESS VALUES IN VIEW REGISTER
423 ; U VNT QIO USING VIEW REGISTER BUFFER
424 ; Q SET VNT PRS
425 ; U CHECK VIEW REGISTER BUFFER
426 ;--
427 ;*****
428
429 015274 STATST
015274 TEST5::
015274 012737 000005 002114 MOV #5,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
015302 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
430 ;
431 ; SET UP TO WRITE TO RAM ON VAX SIDE
432 ;
433 015306 042777 000020 164634 BIC #QVREN,#QCTL ; CLEAR READ ENABLE
434 015314 012777 000000 164644 MOV #0,#QVAD ; STARTING ADDRESS
435 015322 012701 000000 MOV #0,R1 ; STARTING VALUE TO WRITE
436 ;
437 ; WRITE TO U-RAM
438 ;
439 015326 010177 164636 10#: MOV R1,#QVDA ; WRITE TO RAM
440 015332 005201 INC R1 ; INCREMENT VALUE
441 015334 022701 002000 CMP #2000,R1 ; IS IT DONE YET?
442 015340 001372 BNE 10# ; IF NOT DONE, BRANCH
443 ;
444 ; TELL VAX RAM DATA WAS WRITTEN
445 ;
446 015342 052777 000001 164574 BIS #VNTPRS,#QVNT ; SET EVENT PRESENT
447 015350 013701 002532 MOV DEFAULT,R1 ; NUMBER OF SECONDS TO TIMEOUT
448 015354 012702 026000 15#: MOV #TIMEOUT,R2 ; TIME OUT VALUE TO R2
449 015360 032777 002000 20#: BIT #VNTACK,#QINT ; EVENT ACKNOWLEDGE?
450 015366 001020 BNE 30# ; IF COMMAND PRESENT, BRANCH
451 015370 BREAK
015370 104422 TRAP C#BRK
452 015372 005302 DEC R2 ; DECREMENT TIMEOUT REGISTER
453 015374 001371 BNE 20# ; IF NOT IN TIMEOUT, TRY AGAIN
454 015376 005301 DEC R1 ; DECREMENT SECOND COUNTER
455 015400 001365 BNE 15# ; IF NOT DONE, AGAIN
456 ;
457 ; TIMEOUT ERROR CONDITION
458 ;
459 015402 012701 000001 MOV #1,R1 ; REGISTER # TO R1
460 015406 ERRHRD #...ERR2 ; TELL OPERATOR TIMEOUT
015406 104456 TRAP C#ERHRD
015410 000010 .WORD #
015412 000000 .WORD 0
015414 006506 .WORD ERR2
461 015416 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
462 015422 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR

```

TEST 5: VIEW REGISTER WRITE

```
463 015426 000402          BR      100$          ; ABORT TEST
464                          ;
465                          ; END TIMEOUT ERROR CONDITION
466                          ;
467                          ;
468 015430 012701 000005   30$:  MOV     #5.,R1          ; TEST END NORMALLY FLAG FOR 5 SECS
469 015434                100$:  RTS      PC              ; EXIT TEST TO MAINLINE
470 015434 000207
```

TEST 6: VIEW REGISTER COMPLIMENTED READ

```

472 .SBTTL TEST 6: VIEW REGISTER COMPLIMENTED READ
473 ;*****
474 ;**
475 ;TEST DESCRIPTION-
476 ; READS DATA FROM VIEW REGISTER
477 ;
478 ;TEST STEPS-
479 ; U LOAD COMPLIMENTED ADDRESS VALUES IN VIEW BUFFER
480 ; U CMD QIO USING VIEW REGISTER BUFFER
481 ; Q READ ALL VIEW REGISTER LOCATIONS AND CHECK
482 ; Q SET VNT PRS
483 ; U CHECK VIEW REGISTER BUFFER
484 ;--
485 ;*****
486
487 015436 STATST
015436 TEST6::
015436 012737 000006 002114 MOV #6,L$TEST ;TEST NUMBER FOR ERROR MESSAGES
015444 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
488 015450 013701 002532 MOV DEFAULT,R1 ; NUMBER OF SECONDS TO TIME OUT
489 015454 012702 026000 5$: MOV #TIMEOUT,R2 ; TIME OUT VALUE TO R2
490 015460 032777 004000 164460 10$: BIT #CMDPRS,@QINT ; COMMAND PRESENT?
491 015466 001020 BNE 20$ ; IF COMMAND PRESENT, BRANCH
492 015470 BREAK
015470 104422 TRAP C$BRK
493 015472 005302 DEC R2 ; DECREMENT TIMEOUT REGISTER
494 015474 001371 BNE 10$ ; IF NOT IN TIMEOUT, TRY AGAIN
495 015476 005301 DEC R1 ; DECREMENT SECOND TIMER
496 015500 001365 BNE 5$ ; IF NOT DONE, AGAIN
497 ;
498 ; TIMEOUT ERROR CONDITION
499 ;
500 015502 012701 000000 MOV #0,R1 ; REGISTER # TO R1
501 015506 ERRHRD 9,,ERR2 ; TELL OPERATOR TIMEOUT
015506 104456 TRAP C$ERRRD
015510 000011 .WORD 9
015512 000000 .WORD 0
015514 006506 .WORD ERR2
502 015516 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
503 015522 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
504 015526 000525 BR 100$ ; ABORT TEST
505 ;
506 ; END TIMEOUT ERROR CONDITION
507 ;
508 ; SET UP TO READ VIEW RAM
509 ;
510 015530 20$:
511 015530 052777 000020 164412 BIS #QVREN,@QCTL ; READ ENABLE
512 015536 012777 000000 164422 MOV #0,@QVAD ; SET READ RAM INITIAL ADD. TO ZERO
513 015544 013701 002502 MOV FREMEM,R1 ; START OF BUFFER TO R1
514 015550 012702 002000 MOV #2000,R2 ; LOOP COUNTER TO R2
515 ;
516 ; READ 1K VIEW RAM, PLACE CONTENTS IN BUFFER
517 ;
518 015554 30$:
519 015554 017721 164410 MOV @QVDA,(R1)+ ; READ AND PLACE IN BUFFER
520 015560 005302 DEC R2 ; DECREMENT LOOP COUNTER

```

## TEST 6: VIEW REGISTER COMPLIMENTED READ

```

521 015562 001374          BNE      30$          ; IF NOT DONE, AGAIN
522                          ;
523                          ; DONE READING RAM, NOW SET UP TO CHECK IT
524                          ;
525 015564 005037 002462    CLR      ERROR1          ; INITIAL ERROR COUNT
526 015570 013701 002502    MOV      FREMEM,R1       ; START OF BUFFER TO R1
527 015574 012702 000000    MOV      #0,R2          ; LOOP COUNTER
528 015600 010237 002456    MOV      R2,EXP         ; MOVE VALUE TO EXPECTED STORAGE
529 015604 005137 002456    COM      EXP            ; COMPLIMENT EXPECTED
530                          ;
531                          ; NOW CHECK THE VALUES
532                          ;
533 015610          40$:
534 015610 012137 002454    MOV      (R1)+,VAL      ; MOVE READ VALUE TO VAL
535 015614 023737 002456 002454  CMP      EXP,VAL        ; COMPARE EXPECTED WITH VALUE
536 015622 001433          BEQ      60$            ; BRANCH IF OK
537                          ;
538                          ; ERROR CONDITION, BAD RAM VALUE
539                          ;
540 015624 005237 002462    INC      ERROR1         ; INCREMENT ERROR COUNTER
541 015630 022737 000001 002462  CMP      #1,ERROR1      ; IS IT THE FIRST ERROR?
542 015636 001004          BNE      50$            ; IF NOT FIRST ERROR, BRANCH PRINT HEADER
543 015640          ERRHRD  10.,MSG4,ERR4 ; PRINT ERROR HEADER ONLY
    015640 104456          TRAP   C#ERRHD
    015642 000012          .WORD  10
    015644 006130          .WORD  MSG4
    015646 007012          .WORD  ERR4
544 015650 022737 000011 002462 50$:  CMP      #9.,ERROR1     ; SEE IF TOO MANY ERRORS TO PRINT
545 015656 100415          BMI      60$            ; BRANCH IF TOO MANY ERRORS TO PRINT
546 015660          PRINTX  #FOR5,R2,VAL,EXP ; PRINT DATA MESSAGE
    015660 013746 002456    MOV      EXP,-(SP)
    015664 013746 002454    MOV      VAL,-(SP)
    015670 010246          MOV      R2,-(SP)
    015672 012746 003574    MOV      #FOR5,-(SP)
    015676 012746 000004    MOV      #4,-(SP)
    015702 010600          MOV      SP,R0
    015704 104415          TRAP   C#PNTX
    015706 062706 000012    ADD      #12,SP
547                          ;
548                          ; END ERROR CONDITION
549                          ;
550 015712          60$:
551 015712 005202          INC      R2              ; INCREMENT ADDRESS VALUE AND LOOP COUNTER
552 015714 010237 002456    MOV      R2,EXP         ; MOVE VALUE TO EXPECTED STORAGE
553 015720 005137 002456    COM      EXP            ; COMPLIMENT EXPECTED
554 015724 020227 002000    CMP      R2,#2000       ; IS IT DONE YET
555 015730 001327          BNE      40$            ; IF NOT DONE, GO CHECK NEXT VALUE
556                          ;
557                          ; TEST DONE, PRINT FINAL ERROR TOTALS IF ANY
558                          ;
559 015732 005737 002462    TST      ERROR1         ; ANY ERRORS
560 015736 001414          BEQ      70$            ; BRANCH IF NO ERRORS
561                          ;
562                          ; PRINT FINAL ERROR TOTALS
563                          ;
564 015740          PRINTB  #FOR6,ERROR1 ; PRINT # OF ERRORS
    015740 013746 002462    MOV      ERROR1,-(SP)

```

TEST 6: VIEW REGISTER COMPLIMENTED READ

```

015744 012746 003622      MOV    #FOR6, (SP)
015750 012746 000002      MOV    #2, -(SP)
015754 010600              MOV    SP,R0
015756 104414              TRAP   C$PNTB
015760 062706 000006      ADD    #6, SP
565 015764 004737 013116  CALL   ENDERR                ; GO PRINT OUT ENDING TERMINATOR
566
567                      ; SET QCMD CMDACK TO COMPLETE HANDSHAKE
568
569 015770 052777 000001 164144 701:  BIS    @CMDACK,@QCMD        ; SET CMDACK
570 015776 012701 000005      MOV    #5,,R1              ; TEST END NORMALLY FLAG FOR 5 SECS
571 016002
572 016002 000207      RTS    PC                  ; EXIT TEST TO MAINLINE

```



TEST 7: VIEW REGISTER COMPLIMENTED WRITE

```

574 .SBTTL TEST 7: VIEW REGISTER COMPLIMENTED WRITE
575 ; .....
576 ; ..
577 ; TEST DESCRIPTION-
578 ; WRITES COMPLIMENTED DATA TO VAX VIEW REGISTER
579 ;
580 ; TEST STEPS-
581 ; U CMD QIO USING VIEW REGISTER BUFFER
582 ; Q LOAD ADDRESS VALUES IN VIEW REGISTER
583 ; U VNT QIO USING VIEW REGISTER BUFFER
584 ; Q SET VNT PRS
585 ; U CHECK VIEW REGISTER BUFFER
586 ; --
587 ; .....
588 ;
589 016004 STATST
016004 TEST7::
016004 012737 000007 002114 MOV #7,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
016012 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
590 ;
591 ; SET UP TO WRITE TO RAM ON VAX SIDE
592 ;
593 016016 042777 000020 164124 BIC #QVREN,#QCTL ; CLEAR READ ENABLE
594 016024 012777 000000 164134 MOV #0,#QVAD ; STARTING ADDRESS
595 016032 012701 177777 MOV #1,R1 ; STARTING VALUE TO WRITE
596 ;
597 ; WRITE TO U-RAM
598 ;
599 016036 010177 164126 100: MOV R1,#QVDA ; WRITE TO RAM
600 016042 005301 DEC R1 ; DECREMENT VALUE
601 016044 022701 175777 CMP #C2000,R1 ; IS IT DONE YET?
602 016050 001372 BNE 100 ; IF NOT DONE, BRANCH
603 ;
604 ; TELL VAX RAM DATA WAS WRITTEN
605 ;
606 016052 052777 000001 164064 BIS #VNTPRS,#QVNT ; SET EVENT PRESENT
607 016060 013701 002532 MOV DEFAULT,R1 ; NUMBER OF SECONDS TO TIME OUT
608 016064 012702 026000 150: MOV #TIMEOUT,R2 ; TIME OUT VALUE TO R2
609 016070 032777 002000 200: BIT #VNTACK,#QINT ; COMMAND PRESENT?
610 016076 001020 BNE 300 ; IF COMMAND PRESENT, BRANCH
611 016100 BREAK ; ALLOW CONTROL C
016100 104422 TRAP C#BRK
612 016102 005302 DEC R2 ; DECREMENT TIMEOUT REGISTER
613 016104 001371 BNE 200 ; IF NOT IN TIMEOUT, TRY AGAIN
614 016106 005301 DEC R1 ; DECREMENT SECOND TIMER
615 016110 001365 BNE 150 ; BRANCH IF NOT DONE
616 ;
617 ; TIMEOUT ERROR CONDITION
618 ;
619 016112 012701 000001 MOV #1,R1 ; REGISTER # TO R1
620 016116 ERRHRD 11...ERR2 ; TELL OPERATOR TIMEOUT
016116 104456 TRAP C#ERRD
016120 000013 .WORD 11
016122 000000 .WORD 0
016124 006506 .WORD ERR2
621 016126 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
622 016132 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
    
```

TEST 7: VIEW REGISTER COMPLIMENTED WRITE

```
623 016136 000402          BR      1008          ; ABORT TEST
624
625          ; END TIMEOUT ERROR CONDITION
626          ;
627          ;
628 016140 012701 000005    308:   MOV      #5.,R1          ; TEST END NORMALLY WITH 5 SEC DELAY
629 016144
630 016144 000207          1008:  RTS      PC          ; EXIT TEST TO MAINLINE
```

TEST 8: COMMAND INTERRUPTS

```

632 .SBTTL TEST 8: COMMAND INTERRUPTS
633 ;*****
634 ;**
635 ;TEST DESCRIPTION-
636 ; SETS CMD PRS, CKD ACK INTERRUPTS
637 ;
638 ;TEST STEPS-
639 ; U CMD QIO WITH INTERRUPT
640 ; Q WAIT FOR INTERRUPT ON TIME-OUT
641 ; Q SET CMD ACK
642 ; U WAIT FOR INTERRUPT ON TIME-OUT
643 ;--
644 ;*****
645
646 016146 STATST
016146 TEST8::
016146 012737 000010 002114 MOV #8,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
016154 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
647 016160 005037 002450 CLR TRPFLG ; CLEAR INTERRUPT FLAG
648 016164 CLRVEC VECTOR ; CLEAR ERR VECTOR
016164 013700 002444 MOV VECTOR,R0
016170 104436 TRAP C#CVEC
649 016172 SETVEC VECTOR,#TRAP4,#PRI07 ; SET UP FOR INTERRUPT
016172 012746 000340 MOV #PRI07,-(SP)
016176 012746 012644 MOV #TRAP4,-(SP)
016202 013746 002444 MOV VECTOR,-(SP)
016206 012746 000003 MOV #3,-(SP)
016212 104437 TRAP C#SVEC
016214 062706 000010 ADD #10,SP
650 016220 052777 000100 163714 BIS #PRSI0,BQCMD ; ENABLE COMMAND INTERRUPT
651 016226 013701 002532 MOV DEFAULT,R1 ; # OF SECONDS TO WAIT
652 016232 012702 026000 10#: MOV #TIMEOUT,R2 ; TIME OUT DELAY TO R2
653 016236 20#: BREAK ; ALLOW CONTROL C
016236 104422 TRAP C#BRK
654 016240 005737 002450 TST TRPFLG ; DID INTERRUPT OCCUR
655 016244 001020 BNE 30# ; IF YES INTERRUPT, BRANCH
656 016246 005302 DEC R2 ; DECREMENT TIMEOUT
657 016250 001372 BNE 20# ; IF NOT DONE LOOP
658 016252 005301 DEC R1 ; DECREMENT SECOND COUNTER
659 016254 001366 BNE 10# ; IF NOT DONE, LOOP
660
661 ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
662 ;
663 016256 052777 000001 163656 BIS #CMDACK,BQCMD ; SET COMMAND ACKNOWLEDGE INTERRUPT
664 016264 ERRHRD 12,,,ERR6 ; TELL OPERATOR TIMEOUT
016264 104456 TRAP C#ERRRD
016266 000014 .WORD 12
016270 000000 .WORD 0
016272 007320 .WORD ERR6
665 016274 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
666 016300 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
667 016304 000447 BR 100# ; BRANCH TO END OF TEST
668
669 ; END ERROR CONDITION
670 ;
671 ; INTERRUPT OCCURRED, FIND OUT WHICH ONE
672 ;

```

## TEST 8: COMMAND INTERRUPTS

```

673 016306
674 016306 017737 163634 002454 300:
675 016314 042737 000677 002454      MOV      @QINT,VAL      ; READ Q-INTERRUPT REGISTER
676 016322 012737 004000 002456      BIC      #'B0000000110111111,VAL ; GET RID OF UNWANTED BITS
677 016330 033737 002456 002454      MOV      #'BIT11,EXP    ; EXPECTED VALUE TO EXP
678 016336 001025                BIT      EXP,VAL        ; COMPARE EXPECTED WITH VALUE
679
680                ;
681                ; ERROR CONDITION, WRONG INTERRUPT BIT SET
682 016340 012701 000002                ;
683 016344                MOV      #2,R1          ; REGISTER # TO R1
        016344 104456                ERRMRD   13,MSG6,ERR3   ; TELL OPERATOR ERROR
        016346 000015                TRAP    C:ERRMRD
        016350 006202                .WORD   13
        016352 006652                .WORD   MSG6
684 016354                .WORD   ERR3
        016354 010146                SETBITS QINT,INTBIT,VAL
        016356 010546                MOV     R1,-(SP)       ;SAVE REGISTERS
        016360 013701 002454                MOV     R5,-(SP)
        016364 004537 012130                MOV     VAL,R1        ;GET VAL REGISTER TO READ
        016370 000012                JSR     R5,BITS#      ;SUBROUTINE TO OUTPUT VALUES
        016372 002672                .WORD   #LEN
        016374 121 111 116                .WORD   INTBIT
        016402 012605                .ASCIZ  /QINT/
        016404 012601                MOV     (SP)+,R5      ;RESTORE REGISTERS
685 016406 004737 013116                MOV     (SP)+,R1
        016406 004737 013116                CALL    ENDERR        ; GO PRINT OUT ENDING TERMINATOR
686
687                ;
688                ; END ERROR CONDITION, SEND INTERRUPT BACK
689 016412 400:
690 016412 052777 000001 163522          BIS      @CMDACK,@QCMD ; SET COMMAND ACKNOWLEDGE INTERRUPT
691 016420 012701 000000                MOV     #0,R1        ; TEST END NORMALLY WITH DEFAULT DELAY
692 016424 042777 000100 163510 1000:    BIC      #'PRISIE,@QCMD ; DISABLE COMMAND INTERRUPT
693 016432                CLRVEC   VECTOR      ; RESTORE INTERRUPT VECTOR
        016432 013700 002444                MOV     VECTOR,R0
        016436 104436                TRAP    C:CLRVEC
694 016440                SETVEC   VECTOR,@ERRINT,@PRI07 ; SET UP FOR ERROR INTERRUPT
        016440 012746 000340                MOV     #'PRI07,-(SP)
        016444 012746 012652                MOV     @ERRINT,-(SP)
        016450 013746 002444                MOV     VECTOR,-(SP)
        016454 012746 000003                MOV     #3,-(SP)
        016460 104437                TRAP    C:SETVEC
        016462 062706 000010                ADD     #10,SP
695 016466 052777 000100 163452          BIS      @ERRIE,@QINT  ; MAKE SURE ERROR INTERRUPT IS ENABLED
696
697 016474 000207                RTS     PC            ; EXIT TEST TO MAINLINE

```

TEST 9: EVENT INTERRUPTS

```

699
700
701
702
703
704
705
706
707
708
709
710
711
712
713 016476
    016476
    016476 012737 000011 002114
    016504 004737 013034
714 016510 005037 002450
715 016514
    016514 013700 002444
    016520 104436
716 016522
    016522 012746 000340
    016526 012746 012644
    016532 013746 002444
    016536 012746 000003
    016542 104437
    016544 062706 000010
717 016550 052777 000100 163366
718 016556 052777 000001 163360
719 016564 013701 002532
720 016570 012702 026000
721 016574
    016574 104422
722 016576 005737 002450
723 016602 001015
724 016604 005302
725 016606 001372
726 016610 005301
727 016612 001366
728
729
730
731 016614
    016614 104456
    016616 000016
    016620 000000
    016622 007634
732 016624 012701 013630
733 016630 004737 013116
734 016634 000450
735
736
737
738
739
    
```

```

.SBTTL TEST 9: EVENT INTERRUPTS
;*****
;
;TEST DESCRIPTION-
; SETS VNT PRS, VNT ACK INTERRUPTS
;
;TEST STEPS-
; Q   SET VNT PRS
; U   WAIT FOR INTERRUPT ON TIME-OUT
; U   VNT QIO WITH INTERRUPT
; Q   WAIT FOR INTERRUPT ON TIME-OUT
;--
;*****

TEST9::
STATST
MOV     #9,L#TEST           ;TEST NUMBER FOR ERROR MESSAGES
CALL   BMSG                ;CHECK FOR PNT FLAG
CLR    TRPFLG              ; CLEAR INTERRUPT FLAG
CLRVEC VECTOR              ; CLEAR ERR VECTOR
MOV    VECTOR,R0
TRAP   C#VEC
SETVEC VECTOR,#TRAP4,#PRI07 ; SET UP FOR INTERRUPT
MOV    #PRI07,-(SP)
MOV    #TRAP4,-(SP)
MOV    VECTOR,-(SP)
MOV    #3,-(SP)
TRAP   C#SVEC
ADD    #10,SP
BIS    #ACKIE,#QVNT        ; ENABLE EVENT INTERRUPT
BIS    #VNTPRS,#QVNT      ; SET EVENT PRESENT
MOV    DEFAULT,R1         ; # OF SECONDS TO WAIT
10#:   MOV    #TIMEOUT,R2  ; TIME OUT DELAY TO R2
20#:   BREAK
TRAP   C#BRK
TST    TRPFLG             ; DID INTERRUPT OCCUR
BNE    30#                ; IF YES INTERRUPT, BRANCH
DEC    R2                 ; DECREMENT TIMEOUT
BNE    20#                ; IF NOT DONE LOOP
DEC    R1                 ; DECREMENT SECOND COUNTER
BNE    10#                ; IF NOT DONE, LOOP

;
; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
;
ERRHRD 14...ERR9          ; TELL OPERATOR TIMEOUT
TRAP   C#ERRHD
.WORD  14
.WORD  0
.WORD  ERR9
MOV    #ABORT,R1         ; ABORT CODE TO R1
CALL   ENDERR            ; GO PRINT OUT ENDING TERMINATOR
BR     100#              ; BRANCH TO END OF TEST

;
; END ERROR CONDITION
;
; INTERRUPT OCCURRED, FIND OUT WHICH ONE
;
    
```

## TEST 9: EVENT INTERRUPTS

```

740 016636
741 016636 017737 163304 002454 300: MOV @QINT,VAL ; READ Q-INTERRUPT REGISTER
742 016644 042737 000677 002454 BIC @B0000000110111111,VAL ; GET RID OF UNWANTED BITS
743 016652 032737 100000 002454 BIT @ERR,VAL ; ERR INTERRUPT?
744 016660 001007 BNE 350 ; BRANCH IF ERROR INTERRUPT
745 016662 012737 002000 002456 MOV @BIT10,EXP ; EXPECTED VALUE TO EXP
746 016670 033737 002456 002454 BIT EXP,VAL ; COMPARE EXPECTED WITH VALUE
747 016676 001025 BNE 400 ; BRANCH IF OK
748
749 ; ERROR CONDITION, WRONG INTERRUPT BIT SET
750
751 016700 012701 000002 350: MOV @2,R1 ; REGISTER # TO R1
752 016704 016704 104456 ERRHRD 15,MSG6,ERR3 ; TELL OPERATOR ERROR
016706 000017 TRAP C@ERRHRD
016710 006202 .WORD 15
016712 006652 .WORD MSG6
753 016714 016714 010146 SETBITS QINT,INTBIT,VAL
016716 010546 MOV R1,-(SP) ;SAVE REGISTERS
016720 013701 002454 MOV R5,-(SP)
016724 004537 012130 MOV VAL,R1 ;GET VAL REGISTER TO READ
016730 000012 JSR R5,BITS ;SUBROUTINE TO OUTPUT VALUES
016732 002672 .WORD @LEN
016734 121 111 116 .WORD INTBIT
016742 012605 .ASCIZ /QINT/
016744 012601 MOV (SP)+,R5 ;RESTORE REGISTERS
754 016746 004737 013116 MOV (SP)+,R1
CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
755
756 ; END ERROR CONDITION
757
758 016752 012701 000000 400: MOV @0,R1 ; TEST END NORMALLY WITH DEFAULT DELAY
759 016756 042777 000100 163160 1000: BIC @ACKIE,@QVNT ; DISABLE EVENT INTERRUPT
760 016764 016764 013700 002444 CLRVEC VECTOR ; RESTORE INTERRUPT VECTOR
016770 104436 MOV VECTOR,R0
761 016772 016772 012746 000340 TRAP C@CVEC
016776 012746 012652 SETVEC VECTOR,@ERRINT,@PRI07 ; SET UP FOR ERROR INTERRUPT
017002 013746 002444 MOV @PRI07,-(SP)
017006 012746 000003 MOV @ERRINT,-(SP)
017012 104437 MOV VECTOR,-(SP)
017014 062706 000010 MOV @3,-(SP)
762 017020 052777 000100 163120 TRAP C@SVEC
763 017026 000207 ADD @10,SP
BIS @ERRIE,@QINT ; MAKE SURE ERROR INTERRUPT IS ENABLED
764 RTS PC ; EXIT TEST TO MAINLINE

```

TEST 10: VIEW CONTENTION

```

766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788 017030
    017030
    017030 012737 000012 002114
    017036 004737 013034
789 017042 012737 000001 002526
790 017050 012737 000001 002530
791 017056 012737 000145 002524
792 017064 005037 002474
793
794
795
796 017070 013701 002502
797 017074 012702 001777
798 017100 010221
799 017102 005302
800 017104 100375
801
802
803
804 017106
805 017106 013701 002532
806 017112 005237 002474
807 017116 012702 026000
808 017122
    017122 104422
809 017124 032777 004000 163014
810 017132 001032
811 017134 005302
812 017136 001371
813 017140 005301
814 017142 001365
815
816
817
818 017144 012701 000000
    
```

```

.SBTTL TEST 10: VIEW CONTENTION
;*****
;**
;TEST DESCRIPTION
; CHECK VIEW FOR CONTENTION WITH SIMULTANEOUS WRITES AND READS
;
;TEST STEPS-
; LOAD BUFFER WITH DECREMENTING VALUE STARTING AT 1777
; WAIT FOR CMDPRS ON TIME OUT
; START VARIABLE DELAY LOOP
; LOAD VIEW PAM (MINUS LAST ADDRESS) WITH BUFFER
; WAIT FOR CMDPRS ON TIME OUT
; CLEAR READ BUFFER
; SET VNTPRS
; START VARIABLE DELAY LOOP
; READ VIEW INTO BUFFER
; WAIT FOR VNTACK ON TIME OUT
; CHECK DATA
; INCREMENT VARIABLE DELAY LOOP VALUE
; LOOP 101. TIMES
;
;*****
;          STATST
TEST10::
    MOV     #10,L#TEST           ;TEST NUMBER FOR ERROR MESSAGES
    CALL   BMSG                 ;CHECK FOR PNT FLAG
    MOV     #1.,DLY1            ; INITIAL WRITE DELAY LOOP VALUE
    MOV     #1.,DLY2            ; INITIAL READ DELAY LOOP VALUE
    MOV     #101.,LOOPS         ; STORE NUMBER OF LOOPS
    CLR     COUNTER              ; CLEAR LOOP COUNTER
;
; FIRST LOAD BUFFER WITH DECREMENT VALUE
;
    MOV     FREMEM,R1            ; POINT R1 TO BUFFER
    MOV     #1777,R2            ; VALUE IN R2
10$:      MOV     R2,(R1)+        ; PLACE VALUE IN BUFFER
    DEC     R2                  ; DECREMENT VALUE
    BPL     10$                 ; BRANCH IF NOT DONE
;
; NOW DO A COMMAND SYNC (POLLED)
;
DLYST:
    MOV     DEFAULT,R1          ; NUMBER OF SECONDS TO TIMEOUT
    INC     COUNTER              ; INCREMENT LOOP COUNTER
11$:      MOV     #TIMOUT,R2     ; TIME OUT VALUE TO R2
12$:      BREAK
    TRAP   C#BRK                ; ALLOW CONTROL C
    BIT    #CMDPRS,@QINT        ; COMMAND PRESENT?
    BNE    15$                  ; IF COMMAND PRESENT, BRANCH
    DEC    R2                   ; DECREMENT TIMEOUT REGISTER
    BNE    12$                  ; IF NOT IN TIMEOUT, TRY AGAIN
    DEC    R1                   ; DECREMENT SECOND TIMER
    BNE    11$                  ; IF NOT TIMEOUT, TRY AGAIN
;
; TIMEOUT ERROR CONDITION
;
    MOV     #0,R1                ; REGISTER NUMBER TO R1
    
```

## TEST 10: VIEW CONTENTION

```

819 017150          ERRHRD 16,ERR2          ; TELL OPERATOR TIMEOUT
    017150 104456   TRAP    C#ERRHRD
    017152 000020   .WORD  16
    017154 000000   .WORD  0
    017156 006506   .WORD  ERR2
820 017160          PRINTX #FOR23,COUNTER      ; TELL WHICH LOOP NUMBER
    017160 013746 002474  MOV    COUNTER,-(SP)
    017164 012746 005314  MOV    #FOR23,-(SP)
    017170 012746 000002  MOV    #2,-(SP)
    017174 010600   MOV    SP,R0
    017176 104415   TRAP  C#PNTX
    017200 062706 000006  ADD    #6,SP
821 017204 012701 013630  MOV    #ABORT,R1          ; ABORT CODE TO R1
822 017210 004737 013116  CALL  ENDERR              ; GO PRINT OUT ENDING TERMINATOR
823 017214 000137 020130  JMP    ECONT              ; ABORT TEST
824
825 ; END TIMEOUT ERROR CONDITION
826
827 ; VARIABLE DELAY LOOP
828
829 017220 012777 000001 162714 15#:  MOV    #CMDACK,@QCMD          ; SEND COMMAND ACK
830 017226 013701 002526   MOV    DLY1,R1            ; DELAY VALUE INTO R1
831 017232          BREAK                                ; ALLOW CONTROL C
    017232 104422   TRAP  C#BRK
832 017234 077102   SOB    R1,20#            ; DECREMENT
833
834 ; LOAD VIEW RAM
835
836 017236 042777 000020 162704   BIC    #QVREN,@QCTL      ; CLEAR READ ENABLE
837 017244 012777 000000 162714   MOV    #0,@QVAD          ; ZERO ADDRESS
838 017252 012701 001777   MOV    #1777,R1          ; LOOP COUNTER TO R1
839 017256 013702 002170   MOV    QVDA,R2           ; ADDRESS OF QVDA TO R2
840 017262 013703 002502   MOV    FREMEM,R3        ; BUFFER TO R3
841 017266 012312 30#:  MOV    (R3)+,(R2)        ; LOAD VIEW
842 017270 077102   SOB    R1,30#           ; DEC AND BRANCH IF NOT DONE
843
844 ; WAIT FOR CMDPRS ON TIMEOUT
845
846 017272 013701 002532   MOV    DEFAULT,R1       ; NUMBER OF SECONDS TO TIMEOUT
847 017276 012702 026000 40#:  MOV    #TIMOUT,R2        ; TIME OUT VALUE TO R2
848 017302          BREAK                                ; ALLOW CONTROL C
    017302 104422   TRAP  C#BRK
849 017304 032777 004000 162634   BIT    #CMDPRS,@QINT    ; COMMAND PRESENT?
850 017312 001032   BNE    60#              ; IF COMMAND PRESENT, BRANCH
851 017314 005302   DEC    R2               ; DECREMENT TIMEOUT REGISTER
852 017316 001371   BNE    50#              ; IF NOT IN TIMEOUT, TRY AGAIN
853 017320 005301   DEC    R1               ; DECREMENT SECOND TIMER
854 017322 001365   BNE    40#              ; IF NOT TIMEOUT, TRY AGAIN
855
856 ; TIMEOUT ERROR CONDITION
857
858 017324 012701 000000   MOV    #0,R1            ; REGISTER NUMBER TO R1
859 017330          ERRHRD 17,ERR2          ; TELL OPERATOR TIMEOUT
    017330 104456   TRAP  C#ERRHRD
    017332 000021   .WORD  17
    017334 000000   .WORD  0
    017336 006506   .WORD  ERR2

```



## TEST 10: VIEW CONTENTION

```

860 017340          PRINTX  #FOR23,COUNTER          ; TELL WHICH LOOP NUMBER
      017340 013746 002474      MOV      COUNTER,-(SP)
      017344 012746 005314      MOV      #FOR23,-(SP)
      017350 012746 000002      MOV      #2,-(SP)
      017354 010600          MOV      SP,R0
      017356 104415          TRAP     C#PNTX
      017360 062706 000006      ADD      #6,SP
861 017364 004737 013116      CALL     ENDERR          ; GO PRINT OUT ENDING TERMINATOR
862 017370 012701 013630      MOV      #ABORT,R1      ; ABORT CODE TO R1
863 017374 000137 020130      JMP      ECONT          ; ABORT TEST
864
865          ; END TIMEOUT ERROR CONDITION
866
867 017400 052777 000001 162534 60#:  BIS      #CMDACK,#QCMD          ; ACKNOWLEDGE COMMAND
868
869          ; CLEAR READ BUFFER
870
871 017406 013701 002502          MOV      FREMEM,R1          ; POINT TO FREE MEMORY
872 017412 062701 004000          ADD      #4000,R1          ; POINT TO READ BUFFER
873 017416 012702 001777          MOV      #1777,R2          ; SIZE OF BUFFER
874 017422 005021          65#:  CLR      (R1)+          ; CLEAR BUFFER
875 017424 077202          SOB      R2,65#          ; LOOP IF NOT DONE
876
877          ; TELL VAX READY
878
879 017426 052777 000001 162510          BIS      #VNTPRS,#QVNT          ; TELL VAX READY TO START VIEW READ
880
881          ; VARIABLE DELAY LOOP
882
883 017434 013701 002530          ;
884 017440          70#:  MOV      DLY2,R1          ; DELAY VALUE INTO R1
      017440 104422          BREAK     ; ALLOW CONTROL C
      017442 077102          TRAP     C#BRK
885 017442 077102          SOB      R1,70#          ; DECREMENT
886
887          ; NOW READ VIEW
888
889 017444 052777 000020 162476          BIS      #QVREN,#QCTL          ; READ ENABLE
890 017452 012777 000000 162506          MOV      #0,#QVAD          ; ZERO VIEW ADDRESS REGISTER
891 017460 013701 002502          MOV      FREMEM,R1          ; POINT TO START OF WORK BUFFER
892 017464 062701 004000          ADD      #4000,R1          ; POINT TO READ BUFFER
893 017470 013702 002170          MOV      QVDA,R2          ; POINT R2 TO DATA REGISTER
894 017474 012703 001777          MOV      #1777,R3          ; LOOP COUNTER TO R3
895 017500 011221          80#:  MOV      (R2),(R1)+          ; READ VIEW DATA
896 017502 077302          SOB      R3,80#          ; DEC AND BRANCH IF NOT DONE
897
898          ; WAIT FOR EVENTACK ON TIMEOUT
899
900 017504 013701 002532          ;
901 017510 012702 026000          90#:  MOV      DEFAULT,R1          ; NUMBER OF SECONDS TO TIMEOUT
902 017514          100#:  MOV      #TIMOUT,R2          ; TIME OUT VALUE TO R2
      017514 104422          BREAK     ; ALLOW CONTROL C
      017516 032777 002000 162422          TRAP     C#BRK
903 017516 032777 002000 162422          BIT      #VNTACK,#QINT          ; EVENT ACKNOWLEDGE PRESENT?
904 017524 001032          BNE      110#          ; IF PRESENT, BRANCH
905 017526 005302          DEC      R2          ; DECREMENT TIMEOUT REGISTER
906 017530 001371          BNE      100#          ; IF NOT IN TIMEOUT, TRY AGAIN
907 017532 005301          DEC      R1          ; DECREMENT SECOND TIMER
908 017534 001365          BNE      90#          ; IF NOT TIMEOUT, TRY AGAIN

```

## TEST 10: VIEW CONTENTION

```

909
910 ; TIMEOUT ERROR CONDITION
911 ;
912 017536 012701 000001      MOV      #1,R1      ; REGISTER NUMBER TO R1
913 017542      ERRHRD 18.,ERR2 ; TELL OPERATOR TIMEOUT
    017542 104456      TRAP  C#ERRHD
    017544 000022      .WORD 18
    017546 000000      .WORD 0
    017550 006506      .WORD ERR2
914 017552      PRINTX #FOR23,COUNTER ; TELL WHICH LOOP NUMBER
    017552 013746 002474      MOV      COUNTER,-(SP)
    017556 012746 005314      MOV      #FOR23,-(SP)
    017562 012746 000002      MOV      #2,-(SP)
    017566 010600      MOV      SP,R0
    017570 104415      TRAP  C#PNTX
    017572 062706 000006      ADD      #6,SP
915 017576 004737 013116      CALL    ENDERR      ; GO PRINT OUT ENDING TERMINATOR
916 017602 012701 013630      MOV      #ABORT,R1 ; ABORT CODE TO R1
917 017606 000137 020130      JMP      ECONT      ; ABORT TEST
918
919 ; END TIMEOUT ERROR CONDITION
920 ;
921 ; NOW CHECK DATA
922 ;
923 017612      110$:      CLR      ERROR1      ; INITIAL ERROR COUNT
924 017612 005037 002462      MOV      #0,ADD      ; INITIAL RAM ADDRESS
925 017616 012737 000000 002452      MOV      FREMEM,R1 ; START OF BUFFER TO R1
926 017624 013701 002502      ADD      #4000,R1 ; ADJUST TO POINT TO READ BUFFER
927 017630 062701 004000      MOV      #1,EXP      ; LOOP COUNTER AND EXPECTED VALUE
928 017634 012737 000001 002456
929
930 ; NOW CHECK THE VALUES
931 ;
932 017642      120$:      MOV      (R1)+,VAL ; MOVE READ VALUE TO VAL
933 017642 012137 002454      CMP      EXP,VAL ; COMPARE EXPECTED WITH VALUE
934 017646 023737 002456 002454      BEQ      140$ ; BRANCH IF OK
935 017654 001446
936
937 ; ERROR CONDITION, BAD RAM VALUE
938 ;
939 017656 005237 002462      INC      ERROR1 ; INCREMENT ERROR COUNTER
940 017662 022737 000001 002462      CMP      #1,ERROR1 ; IS IT THE FIRST ERROR?
941 017670 001016      BNE      130$ ; IF NOT FIRST ERROR, BRANCH PRINT HEADER
942 017672      ERRHRD 19.,MSG4,ERR4 ; PRINT ERROR HEADER ONLY
    017672 104456      TRAP  C#ERRHD
    017674 000023      .WORD 19
    017676 006130      .WORD MSG4
    017700 007012      .WORD ERR4
943 017702      PRINTB #FOR23,COUNTER ; PRINT LOOP NUMBER
    017702 013746 002474      MOV      COUNTER,-(SP)
    017706 012746 005314      MOV      #FOR23,-(SP)
    017712 012746 000002      MOV      #2,-(SP)
    017716 010600      MOV      SP,R0
    017720 104414      TRAP  C#PNTB
    017722 062706 000006      ADD      #6,SP
944 017726 022737 000011 002462 130$:      CMP      #9.,ERROR1 ; SEE IF TOO MANY ERRORS TO PRINT
945 017734 100416      BMI      140$ ; BRANCH IF TOO MANY ERRORS TO PRINT

```

TEST 10: VIEW CONTENTION

```

946 017736          PRINTX  #FOR5,ADD,VAL,EXP      ; PRINT DATA MESSAGE
    017736 013746 002456
    017742 013746 002454
    017746 013746 002452
    017752 012746 003574
    017756 012746 000004
    017762 010600
    017764 104415
    017766 062706 000012
947
948
949
950 017772          ;
    017772 005237 002452      140$: INC ADD ; INCREMENT ADDRESS
    017776 005237 002456      INC EXP ; INCREMENT EXPECTED VALUE AND LOOP COUNTER
    020002 023727 002456 002000 CMP EXP,#2000 ; IS IT DONE YET
    020010 001314          BNE 120$ ; IF NOT DONE, GO CHECK NEXT VALUE
955
956
957
958 020012 005737 002462
959 020016 001427
960
961
962
963 020020          ;
    020020 013746 002462      PRINTB #FOR6,ERROR1 ; PRINT # OF ERRORS
    020024 012746 003622      MOV ERROR1,-(SP)
    020030 012746 000002      MOV #FOR6,-(SP)
    020034 010600          MOV #2,-(SP)
    020036 104414          MOV SP,R0
    020040 062706 000006      TRAP C#PNTB
    020044 012746 005534      ADD #6,SP
    020050 012746 000001      PRINTB #ABO ; PRINT ABORT MESSAGE
    020054 010600          MOV #ABO,-(SP)
    020056 104414          MOV #1,-(SP)
    020060 062706 000004      MOV SP,R0
    020064 004737 013116      TRAP C#PNTB
    020070 012701 013630      ADD #4,SP
    020074 000415          CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
965
966
967
968
969
970
971 020076 062737 000001 002526 150$: ADD #1,DLY1 ; INCREMENT WRITE DELAY VALUE
    020104 062737 000001 002530 ADD #1,DLY2 ; INCREMENT READ DELAY VALUE
    020112 005337 002524      DEC LOOPS ; DECRMENT LOOP VALUE
    020116 001402          BEQ 160$ ; BRANCH IF DONE TO END OF TEST
    020120 000137 017106      JMP DLYST ; GO DO IT AGAIN
    020124 012701 000000      MOV #0,R1 ; OK RETURN CODE
    020130 000207          ECONT:
978

```

RETURN

TEST 11: PDP-11 NXM INTERRUPT

```

980 .SBTTL TEST 11: PDP-11 NXM INTERRUPT
981 ;*****
982 ;**
983 ;TEST DESCRIPTION-
984 ; STARTS DMA WITH NON-EXISTENT PDP-11 ADDRESS
985 ;
986 ;TEST STEPS-
987 ; U SET READ QIO WITH LEGAL VAX ADDRESS
988 ; Q SET CMD ACK
989 ; Q START DMA WITH PDP-11 ADDRESS = 760000
990 ; Q CHECK FOR QNEX ERROR
991 ; Q SEND VNT PRS WITH ERROR CONDITION
992 ; U CHECK ERROR STATUS
993 ;--
994 ;*****
995
996 020132          STATST
020132          TEST11::
020132 012737 000013 002114      MOV     #11,L#TEST      ;TEST NUMBER FOR ERROR MESSAGES
020140 004737 013034           CALL    BMSG           ;CHECK FOR PNT FLAG
997 020144 013701 002532           MOV     DEFAULT,R1    ; SECONDS OF TIME OUT
998 020150 012702 026000      8#:    MOV     #TIMOUT,R2  ; TIME OUT VALUE TO R2
999 020154           10#:   BREAK
020154 104422           TRAP   C#BRK
1000 020156 032777 004000 161762  BIT     #CMDPRS,#QINT ; COMMAND PRESENT?
1001 020164 001020           BNE    20#           ; IF COMMAND PRESENT, BRANCH
1002 020166 005302           DEC     R2           ; DECREMENT TIMEOUT REGISTER
1003 020170 001371           BNE    10#           ; IF NOT IN TIMEOUT, TRY AGAIN
1004 020172 005301           DEC     R1           ; DECREMENT SECOND TIMER
1005 020174 001365           BNE    8#           ; IF NOT TIMEOUT, AGAIN
1006
1007 ;
1008 ; TIMEOUT ERROR CONDITION
1009 ;
1009 020176 012701 000000      MOV     #0,R1         ; REGISTER # TO R1
1010 020202           ERRHRD 20,,,ERR2 ; TELL OPERATOR TIMEOUT#
020202 104456           TRAP   C#ERRHRD
020204 000024           .WORD 20
020206 000000           .WORD 0
020210 006506           .WORD ERR2
1011 020212 012701 013630      MOV     #ABORT,R1    ; ABORT CODE TO R1
1012 020216 004737 013116      CALL   ENDERR        ; GO PRINT OUT ENDING TERMINATOR
1013 020222 000137 020664      JMP    ENXM          ; ABORT TEST
1014
1015 ;
1016 ; END TIMEOUT ERROR CONDITION
1017 ;
1017 ; COMMAND PRESENT WAS REALLY THERE
1018 ;
1019 20#:
1020 020226 052777 000020 161714  BIS     #QVREN,#QCTL ; READ ENABLE
1021 020234 012777 000000 161724  MOV     #0,#QVAD     ; CLEAR VIEW REGISTER
1022 020242 017737 161722 002514  MOV     #QVDA,LOW16  ; READ LOW 16 BITS OF UNIBUS ADDRESS
1023 020250 013777 002514 161676  MOV     LOW16,#QUBA  ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
1024 020256 017737 161706 002516  MOV     #QVDA,MI2    ; READ MI 2 BITS OF UNIBUS ADDRESS
1025 020264 032737 177757 002516  BIT     #+CBIT4!BIT5,MI2 ; SEE IF ONLY BITS 4 AND 5 ARE SET
1026 020272 001412           BEQ    30#           ; BRANCH IF OK
1027
1028 ;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS

```

## TEST 11: PDP-11 NXM INTERRUPT

```

1029
1030 020274      ;
      020274 104456      ERRHRD 21...ERR10      ; TELL OPERATOR ERROR
      020276 000025      TRAP    C:ERRHRD
      020300 000000      .WORD  21
      020302 007740      .WORD  0
1031 020304 012701 013630      .WORD  ERR10
1032 020310 004737 013116      MOV    @ABORT,R1      ; ABORT CODE TO R1
1033 020314 000137 020664      CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
      JMP    ENXM        ; END TEST
1034
1035      ; END ERROR CONDITION
1036
1037 020320      ;
1038 020320 053777 002516 161624      300:  BIS    HI2,BQDMA      ; SET UP EXTENDED BITS IN EXP
1039 020326 017737 161636 002512      MOV    BQVDA,WORD,COUNT      ; READ WORD COUNT FROM VIEW
1040 020334 013777 002512 161616      MOV    WORD,COUNT,BQWC      ; MOVE WORD COUNT INTO QWC REGISTER
1041 020342 012777 160000 161606      MOV    @160000,BQBA      ; SET UP LOW 16 BITS OF Q-BUS ADDRESS
1042 020350 052777 001400 161574      BIS    @BIT8:BIT9,BQDMA      ; SET UP EXTENDED BITS
1043 020356 042777 000002 161566      BIC    @DMAWT,BQDMA      ; SET DMA UP TO WRITE
1044 020364      CLRVEC VECTOR      ; CLEAR ERR VECTOR
      020364 013700 002444      MOV    VECTOR,R0
      020370 104436      TRAP  C:VEC
1045 020372      SETVEC VECTOR,@CONT3,@PRI07      ; SET UP INTERRUPT VECTOR
      020372 012746 000340      MOV    @PRI07,-(SP)
      020376 012746 020552      MOV    @CONT3,-(SP)
      020402 013746 002444      MOV    VECTOR,-(SP)
      020406 012746 000003      MOV    @3,-(SP)
      020412 104437      TRAP  C:SVEC
      020414 062706 000010      ADD    @10,SP
1046 020420 052777 000100 161524      BIS    @RDYIE,BQDMA      ; SET UP TO ALLOW DMA RDY INTERRUPT
1047 020426 052777 000100 161512      BIS    @ERRIE,BQINT      ; MAKE SURE ERR ENABLE IS SET
1048 020434 052777 000001 161510      BIS    @GO,BQDMA      ; START UP DMA
1049
1050      ; WAIT FOR RDY OR ERR INTERRUPT ON TIMEOUT
1051
1052 020442 013701 002532      ;
1053 020446 012702 026000      400:  MOV    DEFAULT,R1      ; SECONDS OF TIME OUT
1054 020452      500:  MOV    @TIMEOUT,R2      ; TIME OUT VALUE TO R2
      020452 104422      BREAK
1055 020454 005302      TRAP  C:BRK
1056 020456 001375      DEC    R2      ; DECREMENT TIMEOUT REGISTER
1057 020460 005301      BNE   500      ; IF NOT IN TIMEOUT, TRY AGAIN
1058 020462 001371      DEC    R1      ; DECREMENT SECOND TIMER
1059      BNE   400      ; IF NOT TIMEOUT, AGAIN
1060
1061      ; TIMEOUT ERROR CONDITION
1062 020464      ;
      020464 104456      ERRHRD 22...ERR12      ; TELL OPERATOR TIMEOUT
      020466 000026      TRAP  C:ERRHRD
      020470 000000      .WORD  22
      020472 010146      .WORD  0
1063 020474 012701 013630      .WORD  ERR12
1064 020500      MOV    @ABORT,R1      ; ABORT CODE TO R1
      020500 010146      SETBITS QINT,INTBIT,@QINT      ;SAVE REGISTERS
      020502 01054E      MOV    R1,-(SP)
      020504 017701 161436      MOV    R5,-(SP)
      020510 004537 012130      MOV    @QINT,R1      ;GET @QINT REGISTER TO READ
      JSR  R5,BITS@      ;SUBROUTINE TO OUTPUT VALUES

```

TEST 11: PDP-11 NXM INTERRUPT

```

020514 000012          .WORD  $LEN
020516 002672          .WORD  INTBIT
020520      121      111      116      .ASCIZ  /QINT/
020526 012605          MOV    (SP),R5          ;RESTORE REGISTERS
020530 012601          MOV    (SP),R1
1065 020532 004737 013116          CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1066 020536 042777 000100 161406          BIC   #RDYIE,#QDMA          ; DISABLE RDY INTERRUPT
1067 020544 012701 013630          MOV   #ABORT,R1
1068 020550 000445          BR    ENXM          ; ABORT TEST
1069
1070          ; END TIMEOUT ERROR CONDITION
1071          ;
1072          ; INTERRUPT WAS RECEIVED, CHECK FOR WHICH QNEX
1073          ;
1074 020552          BGNSRV  CONT3
020552          CONT3::
1075 020552 042777 000100 161372          BIC   #RDYIE,#QDMA          ; DISABLE RDY INTERRUPT
1076 020560 052777 000001 161354          BIS   #CMDACK,#QCMD          ; SEND COMMAND ACK TO MASTER
1077 020566 012716 020574          MOV   #CONT4,(SP)          ; CORRECT STACK
1078 020572          ENDSRV
020572          L10035:
1079 020574 017737 161346 002454          CONT4: RTI
1080 020602 032737 020000 002454          MOV   #QINT,VAL          ; READ INTERRUPT REGISTER
1081 020610 001023          BIT   #QNEX,VAL          ; CHECK FOR QNEX
1082          BNE   60$          ; BRANCH IF OK
1083          ;
1084          ; ERROR CONDITION, QNEX BIT NOT SET
1085          ;
020612          ERRMRD  23,,,ERR13          ; TELL OPERATOR ERROR
020612 104456          TRAP  C#ERRMRD
020614 000027          .WORD  23
020616 000000          .WORD  0
020620 010252          .WORD  ERR13
1086 020622          SETBITS  QINT,INTBIT,VAL          ; SHOW WHAT WAS SET IN QINT REGISTER
020622 010146          MOV   R1,-(SP)          ;SAVE REGISTERS
020624 010546          MOV   R5,-(SP)
020626 013701 002454          MOV   VAL,R1          ;GET VAL REGISTER TO READ
020632 004537 012130          JSR   R5,BITS$          ;SUBROUTINE TO OUTPUT VALUES
020636 000012          .WORD  $LEN
020640 002672          .WORD  INTBIT
020642      121      111      116      .ASCIZ  /QINT/
020650 012605          MOV    (SP),R5          ;RESTORE REGISTERS
020652 012601          MOV    (SP),R1
1087 020654 004737 013116          CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1088
1089          ; END ERROR CONDITION
1090          ;
1091 020660          60$:
1092 020660 012701 000000          ENXM: MOV   #0,R1          ; EXIT NORMALLY CODE
1093 020664 005077 161256          CLR   #QINT          ; CLEAR INTERRUPT REGISTER
1094 020670          CLRVEC VECTOR
020670 013700 002444          MOV   VECTOR,R0
020674 104436          TRAP  C#CVEC
1095 020676          SETVEC VECTOR,#ERRINT,#PRI07          ; SET UP INTERRUPT VECTOR
020676 012746 000340          MOV   #PRI07,-(SP)
020702 012746 012652          MOV   #ERRINT,-(SP)
020706 013746 002444          MOV   VECTOR,-(SP)
    
```

TEST 11: PDP-11 NXM INTERRUPT

	020712	012746	000003		MOV	#3, -(SP)	
	020716	104437			TRAP	C#SVEC	
	020720	062706	000010		ADD	#10, SP	
1096	020724	052777	000100	161214	BIS	#ERRIE, #QINT	; ENABLE ERROR INTERRUPT
1097							
1098	020732	000207			RTS	PC	

TEST 12: DMA 2 WORDS, VAX TO PDP-11

```

1100 .SBTTL TEST 12: DMA 2 WORDS, VAX TO PDP-11
1101 ;*****
1102 ;
1103 ;TEST DESCRIPTION-
1104 ; WRITE 2 WORDS FROM COMMAND BRIDGE TO CONTROL
1105 ; CHECK DATA
1106 ;TEST STEPS-
1107 ;
1108 ;--
1109 ;*****
1110
1111          020734          STATST
          020734          TEST12::
          020734 012737 000014 002114      MOV    #12,L#TEST          ;TEST NUMBER FOR ERROR MESSAGES
          020742 004737 013034          CALL  BMSG                ;CHECK FOR PNT FLAG
1112 020746 013701 002532          MOV    DEFAULT,R1        ; SECONDS OF TIME OUT
1113 020752 012702 026000      8#:   MOV    #TIMEOUT,R2    ; TIME OUT VALUE TO R2
1114 020756          10#:   BREAK
          020756 104422          TRAP  C#BRK
1115 020760 032777 004000 161160      BIT    #CMDPRS,#QINT     ; COMMAND PRESENT?
1116 020766 001020          BNE   20#                ; IF COMMAND PRESENT, BRANCH
1117 020770 005302          DEC   R2                 ; DECREMENT TIMEOUT REGISTER
1118 020772 001371          BNE   10#                ; IF NOT IN TIMEOUT, TRY AGAIN
1119 020774 005301          DEC   R1                 ; DECREMENT SECOND TIMER
1120 020776 001365          BNE   8#                 ; IF NOT TIMEOUT, AGAIN
1121
1122 ; TIMEOUT ERROR CONDITION
1123 ;
1124 021000 012701 000000          MOV    #0,R1              ; REGISTER # TO R1
1125 021004          ERRHRD 24...ERR2    ; TELL OPERATOR TIMEOUT#
          021004 104456          TRAP  C#ERRHD
          021006 000030          .WORD 24
          021010 000000          .WORD 0
          021012 006506          .WORD ERR2
1126 021014 004737 013116          CALL  ENDERR              ; GO PRINT OUT ENDING TERMINATOR
1127 021020 012701 013630          MOV    #ABORT,R1        ; ABORT CODE TO R1
1128 021024 000137 021706          JMP   EN                  ; ABORT TEST
1129
1130 ; END TIMEOUT ERROR CONDITION
1131 ;
1132 ; COMMAND PRESENT WAS REALLY THERE
1133 ;
1134 021030      20#:   BIS    #QVREN,#QCTL      ; READ ENABLE
1135 021030 052777 000020 161112      MOV    #0,#QVAD          ; CLEAR VIEW REGISTER
1136 021036 012777 000000 161122      MOV    #QVDA,LOW16       ; READ LOW 16 BITS OF UNIBUS ADDRESS
1137 021044 017737 161120 002514      MOV    LOW16,#QUBA       ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
1138 021052 013777 002514 161074      MOV    #QVDA,HI2        ; READ HI 2 BITS OF UNIBUS ADDRESS
1139 021060 017737 161104 002516      BIT    #+CBIT4!BIT5,HI2  ; SEE IF ONLY BITS 4 AND 5 ARE SET
1140 021066 032737 177757 002516      BEQ   30#                ; BRANCH IF OK
1141 021074 001412
1142
1143 ; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
1144 ;
1145 021076          ERRHRD 25...ERR10    ; TELL OPERATOR ERROR
          021076 104456          TRAP  C#ERRHD
          021100 000031          .WORD 25
          021102 000000          .WORD 0

```



TEST 12: DMA 2 WORDS, VAX TO PDP-11

```

1146 021104 007740          .WORD  ERR10
1147 021106 004737 013116  CALL   ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1148 021112 012701 013630  MOV    #ABORT,R1      ; ABORT CODE TO R1
1149 021116 000137 021706  JMP    EN             ; END TEST
1150                                     ;
1151                                     ; END ERROR CONDITION
1152 021122                                     ;
1153 021122 053777 002516 161022 300:   BIS    M12,@QDMA      ; SET UP EXTENDED BITS IN EXP
1154 021130 017737 161034 002512  MOV    @QVDA,WORD.COUNT ; READ WORD COUNT FROM VIEW
1155                                     ;
1156                                     ; CLEAR BUFFER
1157                                     ;
1158 021136 013701 002512          MOV    WORD.COUNT,R1    ; GET WORD.COUNT IN R1
1159 021142 005401          NEG    R1               ; CHANGE FROM 2'S COMP
1160 021144 013702 002502          MOV    FREMEM,R2       ; STARTING ADDRESS OF BUFFER
1161 021150 005022          350:  CLR    (R2)+         ; CLEAR BYTE IN BUFFER
1162 021152 005301          DEC    R1              ; DECREMENT COUNTER
1163 021154 001375          BNE   350             ; BRANCH BACK IF NOT DONE
1164                                     ;
1165                                     ; BUFFER CLEARED, SET UP FOR DMA
1166                                     ;
1167 021156 013777 002512 160774  MOV    WORD.COUNT,@QWC ; MOVE WORD COUNT INTO QWC REGISTER
1168 021164 013777 002502 160764  MOV    FREMEM,@QBA     ; START ADDRESS OF BUFFER TO QBA
1169 021172 042777 037400 160752  BIC    #37400,@QDMA    ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
1170 021200 042777 000002 160744  BIC    @DMAWT,@QDMA    ; SET UP FOR VAX TO PDP DMA TRANSFER
1171 021206          CLRVEC VECTOR
1172 021206 013700 002444          MOV    VECTOR,R0
1173 021212 104436          TRAP  C1CVEC
1174 021214          SETVEC VECTOR,#C01,#PRI07 ; SET UP NEW INTERRUPT VECTOR
1175 021214 012746 000340          MOV    #PRI07,-(SP)
1176 021220 012746 021340          MOV    #C01,-(SP)
1177 021224 013746 002444          MOV    VECTOR,-(SP)
1178 021230 012746 000003          MOV    #3,-(SP)
1179 021234 104437          TRAP  C1SVEC
1180 021236 062706 000010          ADD   #10,SP
1181 021242 052777 000100 160676  BIS    #ERRIE,@QINT    ; ENABLE ERROR INTERRUPT
1182 021250 052777 000100 160674  BIS    #RDYIE,@QDMA    ; ENABLE RDY INTERRUPT
1183 021256 052777 000001 160666  BIS    #GO,@QDMA       ; INITIATE DMA TRANSFER
1184                                     ;
1185                                     ; WAIT FOR INTERRUPT
1186                                     ;
1187 021264 013701 002532          MOV    DEFAULT,R1     ; NUMBER OF SECONDS OF TIMEOUT
1188 021270 012702 026000          400:  MOV    #TIMOUT,R2    ; TIME OUT DELAY TO R2
1189 021274          450:  BREAK          ; ALLOW CONTROL C
1190 021274 104422          TRAP  C1BRK
1191 021276 005302          DEC    R2             ; DECREMENT TIMEOUT
1192 021300 001375          BNE   450            ; IF NOT DONE LOOP
1193 021302 005301          DEC    R1             ; DECREMENT SECOND COUNTER
1194 021304 001371          BNE   400            ; IF NOT DONE, LOOP
1195                                     ;
1196                                     ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
1197                                     ;
1198 021306          ERRHRD 26,,,ERR7    ; TELL OPERATOR TIMEOUT
1199 021306 104456          TRAP  C1ERRHD
1200 021310 000032          .WORD 26
1201 021312 000000          .WORD 0

```

## TEST 12: DMA 2 WORDS, VAX TO PDP-11

```

1190 021314 007424          .WORD  ERR7
1191 021316 004737 013116  CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1192 021322 042777 000100 160622 BIC   #RDYIE,#QDMA    ; CLEAR INTERRUPT BIT
1193 021330 012701 013630     MOV   #ABORT,R1      ; ABORT CODE TO R1
1194 021334 000137 021706     JMP   #              ; BRANCH TO END OF TEST
1195                                     ;
1196                                     ; END ERROR CONDITION
1197                                     ;
1198                                     ; DMA INTERRUPT SHOULD GO HERE
1199 021340          BGNSRV  C01
1200 021340 042777 000100 160604 C01:: BIC   #RDYIE,#QDMA    ; CLEAR INTERRUPT BIT
1201 021346 012716 021354     MOV   #C02,(SP)     ; CORRECT STACK
1202 021352          ENDSRV
1203 021352 000002          L10036: RTI
1204 021354 032777 100000 160564 C02:  BIT   #ERR,#QINT    ; ERROR INTERRUPT?
1205 021362 001412          BEQ   504              ; BRANCH IF NO ERROR INTERRUPT
1206                                     ;
1207                                     ; ERROR CONDITION, ERROR INTERRUPT OCCURED
1208                                     ;
1209 021364          ERRHRD  27,,ERR11      ; TELL OPERATOR ERROR INTERRUPT OCCURED
1210 021364 104456          TRAP  C#ERRHRD
1211 021366 000033          .WORD  27
1212 021370 000000          .WORD  0
1213 021372 010030          .WORD  ERR11
1214 021374 004737 013116  CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1215 021400 012701 013630     MOV   #ABORT,R1      ; ABORT CODE TO R1
1216 021404 000137 021706     JMP   EN              ; BRANCH TO END OF TEST
1217                                     ;
1218                                     ; END ERROR CONDITION
1219 021410          504:
1220 021410 052777 000001 160524 BIS   #CMDACK,#QCMD    ; TELL MASTER DAM DONE
1221 021416          CLRVEC  VECTOR          ; CLEAR VECTOR
1222 021416 013700 002444     MOV   VECTOR,R0
1223 021422 104436          TRAP  C#CVEC
1224 021424          SETVEC  VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
1225 021424 012746 000340     MOV   #PRI07,-(SP)
1226 021430 012746 012652     MOV   #ERRINT,-(SP)
1227 021434 013746 002444     MOV   VECTOR,-(SP)
1228 021440 012746 000003     MOV   #3,-(SP)
1229 021444 104437          TRAP  C#SVEC
1230 021446 062706 000010     ADD   #10,SP
1231 021452 052777 000100 160466 BIS   #ERRIE,#QINT    ; ENABLE ERROR INTERRUPT
1232                                     ;
1233                                     ; NOW CHECK DATA
1234 021460 013701 002502     MOV   FREMEM,R1      ; R1 TO START OF BUFFER
1235 021464 013702 002512     MOV   WORD.COUNT,R2 ; R2 GETS WORD COUNT
1236 021470 005402          NEG   R2              ; CHANGE WORD COUNT FROM 2'S COMP
1237                                     ;
1238                                     ; NOW CHECK BUFFER AGAINST PATTERN
1239                                     ;
1240 021472 005037 002462     CLR   ERROR1         ; CLEAR ERROR COUNTER

```

TEST 12: DMA 2 WORDS, VAX TO PDP-11

```

1231 021476 012704 000002      100:  MOV    #2,R4           ; # OF WORDS IN PATTERN
1232 021502 012703 021752      MOV    #DP1,R3          ; R3 GETS PATTERN ADDRESS
1233 021506 012137 002454      110:  MOV    (R1),VAL        ; GET WORD FROM BUFFER
1234 021512 012337 002456      MOV    (R3),EXP        ; GET EXPECTED PATTERN IN EXP
1235 021516 023737 002454 002456  CMP    VAL,EXP         ; COMPARE VALUE WITH EXPECTED
1236 021524 001442              BEQ    130:            ; IF OK, BRANCH
1237
1238 ;
1239 ; ERROR CONDITION, NOT WHAT EXPECTED IN BUFFER
1240 021526 005737 002462      ;
1241 021532 001004              TST    ERROR1          ; ANY OTHER ERRORS OCCURED
1242 021534 104456              BNE    120:            ; SKIP HEADER ERROR IF ALREADY PRINTED
      021534 000034              ERRHRD 28,,ERR8       ; PRINT ERROR HARD MESSAGE
      021536 000000              TRAP  C#ERRHRD
      021540 000000              .WORD 28
      021542 007530              .WORD 0
1243 021544 005237 002462      120:  .WORD ERR8
1244 021550 022737 000012 002462  INC    ERROR1          ; INCREMENT ERROR COUNTER
1245 021556 103425              CMP    #10,,ERROR1    ; SEE IF TOO MANY MESSAGES PRINTED
1246 021560 013737 002512 002474  BCS   130:            ; IF TOO MANY, BRANCH
1247 021566 005437 002474      MOV    WORD,COUNT,COUNTER ; CALCULATE WORD NUMBER
1248 021572 160237 002474      NEG    COUNTER         ; CHANGE WORD COUNT FROM 2'S COMP
1249 021576              SUB    R2,COUNTER
      021576 013746 002456      PRINTX #FOR10,COUNTER,VAL,EXP ; PRINT DETAIL LINE
      021602 013746 002454      MOV    EXP,-(SP)
      021606 013746 002474      MOV    VAL,-(SP)
      021612 012746 004230      MOV    COUNTER,-(SP)
      021616 012746 C00004      MOV    #FOR10,-(SP)
      021622 010600      MOV    #4,-(SP)
      021624 104415      MOV    SP,R0
      021626 062706 000012      TRAP  C#PNTX
      ADD    #12,SP
1250 ;
1251 ; END ERROR CONDITION
1252 ;
1253 021632              130:
1254 021632 005302              DEC    R2              ; DECREMENT WORD COUNT
1255 021634 001403              BEQ    140:            ; BRANCH OUT IF DONE
1256 021636 005304              DEC    R4              ; DECREMENT PATTERN WORD COUNT
1257 021640 001322              BNE    110:            ; BRANCH IF NOT DONE WITH PATTERN
1258 021642 000715              BR    100:            ; BRANCH IF DONE WITH PATTERN
1259 ;
1260 ; BUFFER CHECKED, PRINT TOTAL ERROR COUNT IF NECESSARY
1261 ;
1262 021644              140:
1263 021644 005737 002462      TST    ERROR1          ; ANY ERRORS
1264 021650 001414              BEQ    150:            ; IF NO ERRORS, BRANCH
1265 021652              PRINTB #FOR11,ERROR1  ; TELL TOTAL ERROR COUNT
      021652 013746 002462      MOV    ERROR1,-(SP)
      021656 012746 004253      MOV    #FOR11,-(SP)
      021662 012746 000002      MOV    #2,-(SP)
      021666 010600      MOV    SP,R0
      021670 104414      TRAP  C#PNTB
      021672 062706 000006      ADD    #6,SP
1266 021676 004737 013116      CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1267 ;
1268 ; END OF TEST
1269 ;

```

TEST 12: DMA 2 WORDS, VAX TO PDP-11

```

1270 021702 012701 000000      1508:  MOV    #0,R1           ; END TEST NORMALLY
1271 021706 013700 002444      EN:    CLRVEC VECTOR      ; CLEAR VECTOR
      021706 013700 002444      MOV    VECTOR,R0
      021712 104436              TRAP   C#CVEC
1272 021714 012746 000340      SETVEC VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
      021714 012746 000340      MOV    #PRI07,-(SP)
      021720 012746 012652      MOV    #ERRINT,-(SP)
      021724 013746 002444      MOV    VECTOR,-(SP)
      021730 012746 000003      MOV    #3,-(SP)
      021734 104437              TRAP   C#SVEC
      021736 062706 000010      ADD    #10,SP
1273 021742 052777 000100      BIS    #ERRIE,#QINT        ; ENABLE ERROR INTERRUPT
1274                                ;
1275 021750 000207              RTS    PC                   ; RETURN TO MAINLINE
1276                                ;
1277                                ; DATA PATTERN FOR TEST
1278                                ;
1279 021752 055132              DP1:  .WORD 55132
1280 021754 122645              .WORD 122645
1281

```

TEST 13: DMA 2 WORDS, PDP-11 TO VAX

```

1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293 021756
      021756
      021756 012737 000015 002114
      021764 004737 013034
1294 021770 013701 002532
1295 021774 012702 026000
1296 022000
      022000 104422
1297 022002 032777 004000 160136
1298 022010 001020
1299 022012 005302
1300 022014 001371
1301 022016 005301
1302 022020 001365
1303
1304
1305
1306 022022 012701 000000
1307 022026
      022026 104456
      022030 000035
      022032 000000
      022034 006506
1308 022036 004737 013116
1309 022042 012701 013630
1310 022046 000137 022530
1311
1312
1313
1314
1315
1316 022052
1317 022052 052777 000020 160070
1318 022060 012777 000000 160100
1319 022066 017737 160076 002514
1320 022074 013777 002514 160052
1321 022102 017737 160062 002516
1322 022110 032737 177757 002516
1323 022116 001412
1324
1325
1326
1327 022120
      022120 104456
      022122 000036
      022124 000000
      022126 007740

```

```

.SBTTL TEST 13: DMA 2 WORDS, PDP-11 TO VAX
;*****
;**
;TEST DESCRIPTION-
; WRITE 2 WORDS FROM CONTROL BRIDGE TO COMMAND BRIDGE
;TEST STEPS-
;
;--
;*****
;
;          STATST
TEST13::
      MOV     #13,L#TEST           ; TEST NUMBER FOR ERROR MESSAGES
      CALL   BMSG                  ; CHECK FOR PNT FLAG
      MOV     DEFAULT,R1           ; SECONDS OF TIME OUT
      MOV     #TIMOUT,R2          ; TIME OUT VALUE TO R2
8$:
10$:
      BREAK
      TRAP   C#BRK
      BIT    #CMDPRS,#QINT        ; COMMAND PRESENT?
      BNE   20$                   ; IF COMMAND PRESENT, BRANCH
      DEC   R2                     ; DECREMENT TIMEOUT REGISTER
      BNE   10$                   ; IF NOT IN TIMEOUT, TRY AGAIN
      DEC   R1                     ; DECREMENT SECOND TIMER
      BNE   8$                    ; IF NOT TIMEOUT, AGAIN
;
; TIMEOUT ERROR CONDITION
;
      MOV     #0,R1                ; REGISTER # TO R1
      ERHRD  29,,,ERR2            ; TELL OPERATOR TIMEOUT#
      TRAP   C#ERHRD
      .WORD  29
      .WORD  0
      .WORD  ERR2
      CALL   ENDERR                ; GO PRINT OUT ENDING TERMINATOR
      MOV     #ABORT,R1            ; ABORT CODE TO R1
      JMP    EQ                    ; ABORT TEST
;
; END TIMEOUT ERROR CONDITION
;
; COMMAND PRESENT WAS REALLY THERE
;
20$:
      BIS    #QVREN,#QCTL          ; READ ENABLE
      MOV     #0,#QVAD             ; CLEAR VIEW REGISTER
      MOV     #QVDA,LOW16          ; READ LOW 16 BITS OF UNIBUS ADDRESS
      MOV     LOW16,#QUBA          ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
      MOV     #QVDA,HI2           ; READ HI 2 BITS OF UNIBUS ADDRESS
      BIT    #+CBIT4!BIT5,HI2     ; SEE IF ONLY BITS 4 AND 5 ARE SET
      BEQ    30$                  ; BRANCH IF OK
;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
;
      ERHRD  30,,,ERR10           ; TELL OPERATOR ERROR
      TRAP   C#ERHRD
      .WORD  30
      .WORD  0
      .WORD  ERR10

```

TEST 13: DMA 2 WORDS, PDP-11 TO VAX

```

1328 022130 004737 013116          CALL      ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1329 022134 012701 013630          MOV       #ABORT,R1      ; ABORT CODE TO R1
1330 022140 000137 022530          JMP      EQ              ; END TEST
1331                                ;
1332                                ; END ERROR CONDITION
1333                                ;
1334 022144                                30$:
1335 022144 053777 002516 160000    BIS       HI2,@QDMA      ; SET UP EXTENDED BITS IN EXP
1336 022152 017737 160012 002512    MOV      @QVDA,WORD.COUNT ; READ WORD COUNT FROM VIEW
1337                                ;
1338                                ; MOVE DATA PATTERN TO BUFFER
1339                                ;
1340 022160 013703 002502          MOV      FREMEM,R3      ; GET STARTING ADDRESS OF BUFFER IN R3
1341 022164 013702 002512          MOV      WORD.COUNT,R2 ; GET WORD COUNT TO R2
1342 022170 005402                    NEG      R2              ; CHANGE WORD COUNT FROM 2'S COMP
1343 022172 012704 000002          40$: MOV      #2,R4        ; # OF WORDS IN PATTERN
1344 022176 012701 022532          MOV      #DP2,R1       ; POINT TO PATTERN
1345 022202 012123          50$: MOV      (R1)+,(R3)+ ; MOVE PATTERN WORD TO BUFFER
1346 022204 005302          DEC      R2              ; DECREMENT WORD COUNT
1347 022206 001403          BEQ     55$             ; BRANCH IF DONE
1348 022210 005304          DEC      R4              ; DECREMENT PATTERN WORD COUNT
1349 022212 001373          BNE     50$             ; IF NOT LAST PATTERN WORD, LOOP
1350 022214 000766          BR      40$             ; START ALL OVER AGAIN
1351                                ;
1352                                ; BUFFER TAKEN CARE OF, SET AND WAIT FOR INTERRUPT
1353                                ;
1354 022216                                55$: CLRVEC  VECTOR          ; CLEAR ERR INTERRUPT VECTOR
      022216 013700 002444          MOV      VECTOR,R0
      022222 104436          TRAP    C#CVEC
1355 022224          SETVEC VECTOR,#CO01,#PRI07 ; SET UP INTERRUPT VECTOR
      022224 012746 000340          MOV      #PRI07,-(SP)
      022230 012746 022400          MOV      #CO01,-(SP)
      022234 013746 002444          MOV      VECTOR,-(SP)
      022240 012746 000003          MOV      #3,-(SP)
      022244 104437          TRAP    C#SVEC
      022246 062706 000010          ADD     #10,SP
1356 022252 013777 002512 157700    MOV      WORD.COUNT,@QWC ; MOVE WORD COUNT INTO QWC REGISTER
1357 022260 013777 002502 157670    MOV      FREMEM,@QBA    ; START ADDRESS OF BUFFER TO QBA
1358 022266 042777 037400 157656    BIC     #37400,@QDMA    ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
1359 022274 052777 000002 157650    BIS     #DMAWT,@QDMA    ; SET UP FOR PDP TO VAX DMA TRANSFER
1360 022302 052777 000100 157642    BIS     #RDYIE,@QDMA   ; SET RDY ENABLE BIT
1361 022310 052777 000100 157630    BIS     #ERRIE,@QINT   ; MAKE SURE ERR INTERRUPT IS ENABLED
1362 022316 052777 000001 157626    BIS     #GO,@QDMA      ; INITIATE DMA TRANSFER
1363                                ;
1364                                ; WAIT FOR INTERRUPT
1365                                ;
1366 022324 013701 002532          MOV      DEFAULT,R1    ; NUMBER OF SECONDS OF TIMEOUT
1367 022330 012702 026000          60$: MOV      #TIMOUT,R2 ; TIME OUT DELAY TO R2
1368 022334                                65$: BREAK
      022334 104422          TRAP    C#BRK          ; ALLOW CONTROL C
1369 022336 005302          DEC      R2              ; DECREMENT TIMEOUT
1370 022340 001375          BNE     65$             ; IF NOT DONE LOOP
1371 022342 005301          DEC      R1              ; DECREMENT SECOND COUNTER
1372 022344 001371          BNE     60$             ; IF NOT DONE, LOOP
1373                                ;
1374                                ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
1375                                ;

```



TEST 13: DMA 2 WORDS, PDP-11 TO VAX

1414 022530 000207  
1415  
1416  
1417  
1418  
1419 022532 055132  
1420 022534 122645

RTS      PC

; RETURN TO MAINLINE

; DATA PATTERN FOR TEST

DP2:    .WORD 55132  
         .WORD 122645



TEST 14: DMA VAX TO PDP-11

```

1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433 022536
      022536
      022536 012737 000016 002114
      022544 004737 013034
1434 022550 013701 002532
1435 022554 012702 026000
1436 022560
      022560 104422
1437 022562 032777 004000 157356
1438 022570 001020
1439 022572 005302
1440 022574 001371
1441 022576 005301
1442 022600 001365
1443
1444
1445
1446 022602 012701 000000
1447 022606
      022606 104456
      022610 000041
      022612 000000
      022614 006506
1448 022616 004737 013116
1449 022622 012701 013630
1450 022626 000137 023452
1451
1452
1453
1454
1455
1456 022632
1457 022632 052777 000020 157310
1458 022640 012777 000000 157320
1459 022646 017737 157316 002514
1460 022654 013777 002514 157272
1461 022662 017737 157302 002516
1462 022670 032737 177757 002516
1463 022676 001412
1464
1465
1466
1467 022700
      022700 104456
      022702 000042
      022704 000000

```

```

.SBTTL TEST 14: DMA VAX TO PDP-11
;*****
;
;TEST DESCRIPTION-
; WRITE LARGE BUFFER FROM COMMAND BRIDGE TO CONTROL BRIDGE
; CHECK DATA
;TEST STEPS-
;
;--
;*****
;
;          STATST
TEST14::
      MOV     #14,L#TEST           ;TEST NUMBER FOR ERROR MESSAGES
      CALL   BMSG                 ;CHECK FOR PNT FLAG
      MOV     DEFAULT,R1          ; SECONDS OF TIME OUT
      MOV     #TIMOUT,R2         ; TIME OUT VALUE TO R2
8#:
10#:
      BREAK
      TRAP   C#BRK
      BIT    #CMDPRS,#QINT       ; COMMAND PRESENT?
      BNE    20#                 ; IF COMMAND PRESENT, BRANCH
      DEC    R2                  ; DECREMENT TIMEOUT REGISTER
      BNE    10#                 ; IF NOT IN TIMEOUT, TRY AGAIN
      DEC    R1                  ; DECREMENT SECOND TIMER
      BNE    8#                  ; IF NOT TIMEOUT, AGAIN
;
; TIMEOUT ERROR CONDITION
;
      MOV     #0,R1               ; REGISTER # TO R1
      ERHRD  33...,ERR2         ; TELL OPERATOR TIMEOUT#
      TRAP   C#ERHRD
      .WORD  33
      .WORD  0
      .WORD  ERR2
      CALL   ENDERR              ; GO PRINT OUT ENDING TERMINATOR
      MOV     #ABORT,R1          ; ABORT CODE TO R1
      JMP    EW                  ; ABORT TEST
;
; END TIMEOUT ERROR CONDITION
;
; COMMAND PRESENT WAS REALLY THERE
;
20#:
      BIS    #GVREN,#QCTL        ; READ ENABLE
      MOV     #0,#QVAD           ; CLEAR VIEW REGISTER
      MOV     #QVDA,LOW16        ; READ LOW 16 BITS OF UNIBUS ADDRESS
      MOV     LOW16,#QUBA        ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
      MOV     #QVDA,HI2         ; READ HI 2 BITS OF UNIBUS ADDRESS
      BIT    #+CBIT4!BITS5,HI2  ; SEE IF ONLY BITS 4 AND 5 ARE SET
      BEQ    30#                 ; BRANCH IF OK
;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
;
      ERHRD  34...,ERR10        ; TELL OPERATOR ERROR
      TRAP   C#ERHRD
      .WORD  34
      .WORD  0

```

## TEST 14: DMA VAX TO PDP-11

1468	022706	007740			.WORD	ERR10		
1469	022710	004737	013116		CALL	ENDERR		: GO PRINT OUT ENDING TERMINATOR
1470	022714	012701	013630		MOV	#ABORT,R1		: ABORT CODE TO R1
1471	022720	000137	023452		JMP	EW		: END TEST
1472								
1473								
1474	022724							
1475	022724	053777	002516	157220	300:	BIS	M12, @QDMA	: SET UP EXTENDED BITS IN EXP
1476	022732	017737	157232	002512		MOV	@QVDA, WORD.COUNT	: READ WORD FROM VIEW
1477								
1478								
1479								
1480	022740	013701	002512			MOV	WORD.COUNT, R1	: GET WORD.COUNT IN R1
1481	022744	005401				NEG	R1	: CONVERT WORD COUNT FROM 2'S COMP
1482	022746	013702	002502			MOV	FREMEM, R2	: STARTING ADDRESS OF BUFFER
1483	022752	005022			350:	CLR	(R2)	: CLEAR WORD IN BUFFER
1484	022754	005301				DEC	R1	: DECREMENT COUNTER
1485	022756	001375				BNE	350	: BRANCH BACK IF NOT DONE
1486								
1487								
1488								
1489	022760	013777	002512	157172		MOV	WORD.COUNT, @QMC	: MOVE WORD COUNT INTO QMC REGISTER
1490	022766	013777	002502	157162		MOV	FREMEM, @QBA	: START ADDRESS OF BUFFER TO QBA
1491	022774	042777	037400	157150		BIC	#37400, @QDMA	: MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
1492	023002	042777	000002	157142		BIC	@DMAWT, @QDMA	: SET UP FOR VAX TO PDP DMA TRANSFER
1493	023010					CLRVEC	VECTOR	
	023010	013700	002444			MOV	VECTOR, R0	
	023014	104436				TRAP	C1CVEC	
1494	023016					SETVEC	VECTOR, @COP1, @PRI07	: SET UP NEW INTERRUPT VECTOR
	023016	012746	000340			MOV	@PRI07, -(SP)	
	023022	012746	023142			MOV	@COP1, -(SP)	
	023026	013746	002444			MOV	VECTOR, -(SP)	
	023032	012746	000003			MOV	#3, -(SP)	
	023036	104437				TRAP	C1SVEC	
	023040	062706	000010			ADD	#10, SP	
1495	023044	052777	000100	157074		BIS	@ERRIE, @QINT	: ENABLE ERROR INTERRUPT
1496	023052	052777	000100	157072		BIS	@RDYIE, @QDMA	: ENABLE RDY INTERRUPT
1497	023060	052777	000001	157064		BIS	@GO, @QDMA	: INITIATE DMA TRANSFER
1498								
1499								
1500								
1501	023066	013701	002532			MOV	DEFAULT, R1	: NUMBER OF SECONDS OF TIMEOUT
1502	023072	012702	026000		400:	MOV	@TIMEOUT, R2	: TIME OUT DELAY TO R2
1503	023076				450:	BREAK		: ALLOW CONTROL C
	023076	104422				TRAP	C1BRK	
1504	023100	005302				DEC	R2	: DECREMENT TIMEOUT
1505	023102	001375				BNE	450	: IF NOT DONE LOOP
1506	023104	005301				DEC	R1	: DECREMENT SECOND COUNTER
1507	023106	001371				BNE	400	: IF NOT DONE, LOOP
1508								
1509								
1510								
1511	023110					ERRHRD	35, ERR7	: TELL OPERATOR TIMEOUT
	023110	104456				TRAP	C1ERRRD	
	023112	000043				.WORD	35	
	023114	000000				.WORD	0	

## TEST 14: DMA VAX TO PDP-11

```

1512 023116 007424          .WORD  ERR7
1513 023120 004737 013116 CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1514 023124 042777 000100 157020 BIC   #RDYIE,#QDMA    ; CLEAR INTERRUPT BIT
1515 023132 012701 013630      MOV   #ABORT,R1      ; ABORT CODE TO R1
1516 023136 000137 023452      JMP   EW             ; BRANCH TO END OF TEST
1517
1518 ; END ERROR CONDITION
1519 ; DMA INTERRUPT SHOULD GO HERE
1520
1521 023142          BGNSRV  COP1
1522 023142          COP1:: BIC   #RDYIE,#QDMA    ; CLEAR INTERRUPT BIT
1523 023150 042777 000100 157002      MOV   #COP2,(SP)    ; CORRECT STACK
1524 023154          ENDSRV
1525 023154 000002          L10040: RTI
1526 023156 032777 100000 156762 COP2: BIT   #ERR,#QINT    ; ERROR INTERRUPT?
1527 023164 001414          BEQ   500             ; BRANCH IF NO ERROR INTERRUPT
1528
1529 ; ERROR CONDITION, ERROR INTERRUPT OCCURED
1530
1531 023166          ERRHRD  36,,ERR11      ; TELL OPERATOR ERROR INTERRUPT OCCURED
1532 023166 104456          TRAP  CERRHRD
1533 023170 000044          .WORD  36
1534 023172 000000          .WORD  0
1535 023174 010030          .WORD  ERR11
1536 023176 004737 013116      CALL  ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1537 023202 005077 156740      CLR   #QINT         ; CLEAR INTERRUPT REGISTER
1538 023206 012701 013630      MOV   #ABORT,R1      ; ABORT CODE TO R1
1539 023212 000137 023452      JMP   EW             ; BRANCH TO END OF TEST
1540
1541 ; END ERROR CONDITION
1542
1543 023216          500:   BIS   #CMDACK,#QCMD    ; TELL MASTER DAM DONE
1544 023216 052777 000001 156716
1545 ; NOW CHECK DATA
1546
1547      MOV   FREMEM,R1    ; R1 TO START OF BUFFER
1548      MOV   WORD.COUNT,R2 ; R2 GETS WORD COUNT
1549      NEG   R2           ; CONVERT WORD COUNT FROM 2'S COMP.
1550
1551 ; NOW CHECK BUFFER AGAINST PATTERN
1552
1553 023236 005037 002462          CLR   ERROR1        ; CLEAR ERROR COUNTER
1554 023242 012704 000002 1000:  MOV   #2,R4         ; # OF WORDS IN PATTERN
1555 023246 012703 023516          MOV   #DP3,R3       ; R3 GETS PATTERN ADDRESS
1556 023252 012137 002454 1100:  MOV   (R1),VAL       ; GET WORD FROM BUFFER
1557 023256 012337 002456          MOV   (R3),EXP      ; GET EXPECTED PATTERN IN EXP
1558 023262 023737 002454 002456      CMP   VAL,EXP       ; COMPARE VALUE WITH EXPECTED
1559 023270 001442          BEQ   1300         ; IF OK, BRANCH
1560
1561 ; ERROR CONDITION, NOT WHAT EXPECTED IN BUFFER
1562
1563 023272 005737 002462          TST   ERROR1        ; ANY OTHER ERRORS OCCURED

```

## TEST 14: DMA VAX TO PDP-11

```

1561 023276 001004      BNE      1200      ; SKIP HEADER ERROR IF ALREADY PRINTED
1562 023300      ERRHRD  37.,,ERR8      ; PRINT ERROR HARD MESSAGE
      023300 104456      TRAP      C#ERRHRD
      023302 000045      .WORD      37
      023304 000000      .WORD      0
      023306 007530      .WORD      ERR8
1563 023310 005237 002462 1200: INC      ERROR1      ; INCREMENT ERROR COUNTER
1564 023314 022737 000012 002462 CMP      #10.,ERROR1 ; SEE IF TOO MANY MESSAGES PRINTED
1565 023322 103425      BCS      1300      ; IF TOO MANY, BRANCH
1566 023324 013737 002512 002474 MOV      WORD,COUNT,COUNTER ; CALCULATE WORD NUMBER
1567 023332 005437 002474 NEG      COUNTER      ; CONVERT WORD COUNT FROM 2'S COMP
1568 023336 160237 002474 SUB      R2,COUNTER
1569 023342      PRINTX  #FOR10,COUNTER,VAL,EXP ; PRINT DETAIL LINE
      023342 013746 002456 MOV      EXP,-(SP)
      023346 013746 002454 MOV      VAL,-(SP)
      023352 013746 002474 MOV      COUNTER,-(SP)
      023356 012746 004230 MOV      #FOR10,-(SP)
      023362 012746 000004 MOV      #4,-(SP)
      023366 010600      MOV      SP,R0
      023370 104415      TRAP      C#PNTX
      023372 062706 000012 ADD      #12,SP

1570      ;
1571      ; END ERROR CONDITION
1572      ;
1573 023376      1300:
1574 023376 005302      DEC      R2      ; DECREMENT WORD COUNT
1575 023400 001403      BEQ      1400      ; BRANCH OUT IF DONE
1576 023402 005304      DEC      R4      ; DECREMENT PATTERN WORD COUNT
1577 023404 001322      BNE      1100      ; BRANCH IF NOT DONE WITH PATTERN
1578 023406 000715      BR       1000      ; BRANCH IF DONE WITH PATTERN
1579
1580      ; BUFFER CHECKED, PRINT TOTAL ERROR COUNT IF NECESSARY
1581      ;
1582 023410      1400:
1583 023410 005737 002462 TST      ERROR1      ; ANY ERRORS
1584 023414 001414      BEQ      1500      ; IF NO ERRORS, BRANCH
1585 023416      PRINTB  #FOR11,ERROR1 ; TELL TOTAL ERROR COUNT
      023416 013746 002462 MOV      ERROR1,-(SP)
      023422 012746 004253 MOV      #FOR11,-(SP)
      023426 012746 000002 MOV      #2,-(SP)
      023432 010600      MOV      SP,R0
      023434 104414      TRAP      C#PNTB
      023436 062706 000006 ADD      #6,SP
1586 023442 004737 013116 CALL     ENDERR      ; GO PRINT OUT ENDING TERMINATOR
1587
1588      ;
1589      ; END OF TEST
1590 023446 012701 000000 1500: MOV      #0,R1      ; END TEST NORMALLY
1591 023452      EW:   CLRVEC  VECTOR      ; CLEAR VECTOR
      023452 013700 002444 MOV      VECTOR,R0
      023456 104436      TRAP      C#CVEC
1592 023460      SETVEC  VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
      023460 012746 000340 MOV      #PRI07,-(SP)
      023464 012746 012652 MOV      #ERRINT,-(SP)
      023470 013746 002444 MOV      VECTOR,-(SP)
      023474 012746 000003 MOV      #3,-(SP)
      023500 104437      TRAP      C#SVEC

```

TEST 14: DMA VAX TO PDP-11

```

1593 023502 062706 000010          ADD    #10,SP
1594 023506 052777 000100 156432    BIS    #ERRIE,#QINT          ; ENABLE ERROR INTERRUPT
1595 023514 000207                    RTS    PC                    ; RETURN TO MAINLINE
1596
1597
1598          ; DATA PATTERN FOR TEST
1599          ;
1600 023516 000000                    DP3:  .WORD 0
1601 023520 177777                    .WORD 177777
1602

```

TEST 15: DMA PDP-11 TO VAX

```

1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615 023522
      023522
      023522 012737 000017 002114
      023530 004737 013034
1616 023534 013701 002532
1617 023540 012702 026000
1618 023544
      023544 104422
1619 023546 032777 004000 156372
1620 023554 001020
1621 023556 005302
1622 023560 001371
1623 023562 005301
1624 023564 001365
1625
1626
1627
1628 023566 012701 000000
1629 023572
      023572 104456
      023574 000046
      023576 000000
      023600 006506
1630 023602 004737 013116
1631 023606 012701 013630
1632 023612 000137 024232
1633
1634
1635
1636
1637
1638 023616
1639 023616 052777 000020 156324
1640 023624 012777 000000 156334
1641 023632 017737 156332 002514
1642 023640 013777 002514 156306
1643 023646 017737 156316 002516
1644 023654 032737 177757 002516
1645 023662 001412
1646
1647
1648
1649 023664
      023664 104456
      023666 000047
      023670 000000

```

```

.SBTTL TEST 15: DMA PDP-11 TO VAX
;*****
;
;TEST DESCRIPTION-
; WRITE LARGE BUFFER FROM CONTROL BRIDGE TO COMMAND BRIDGE.
; CHECK DATA
;TEST STEPS-
;
;--
;*****
;
;          STATST
TEST15::
      MOV     #15,L#TEST           ;TEST NUMBER FOR ERROR MESSAGES
      CALL   BMSG                 ;CHECK FOR PNT FLAG
      MOV     DEFAULT,R1          ; SECONDS OF TIME OUT
      MOV     #TIMOUT,R2         ; TIME OUT VALUE TO R2
8#:
10#:
      BREAK
      TRAP   C#BRK
      BIT    #CMDPRS,#QINT       ; COMMAND PRESENT?
      BNE    20#                 ; IF COMMAND PRESENT, BRANCH
      DEC    R2                  ; DECREMENT TIMEOUT REGISTER
      BNE    10#                 ; IF NOT IN TIMEOUT, TRY AGAIN
      DEC    R1                  ; DECREMENT SECOND TIMER
      BNE    8#                  ; IF NOT TIMEOUT, AGAIN
;
; TIMEOUT ERROR CONDITION
;
      MOV     #0,R1               ; REGISTER # TO R1
      ERRHRD 38,,,ERR2           ; TELL OPERATOR TIMEOUT#
      TRAP   C#ERHRD
      .WORD  38
      .WORD  0
      .WORD  ERP2
      CALL   ENDERR              ; GO PRINT OUT ENDING TERMINATOR
      MOV     #ABORT,R1          ; ABORT CODE TO R1
      JMP    EZ                  ; ABORT TEST
;
; END TIMEOUT ERROR CONDITION
;
; COMMAND PRESENT WAS REALLY THERE
;
20#:
      BIS     #QVREN,#QCTL       ; READ ENABLE
      MOV     #0,#QVAD           ; CLEAR VIEW REGISTER
      MOV     #QVDA,LOW16        ; READ LOW 16 BITS OF UNIBUS ADDRESS
      MOV     LOW16,#QUBA        ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
      MOV     #QVDA,HI2         ; READ HI 2 BITS OF UNIBUS ADDRESS
      BIT     #+CBIT4!BITS5,HI2 ; SEE IF ONLY BITS 4 AND 5 ARE SET
      BEQ    30#                 ; BRANCH IF OK
;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
;
      ERRHRD 39,,,ERR10          ; TELL OPERATOR ERROR
      TRAP   C#ERHRD
      .WORD  39
      .WORD  0

```

TEST 15: DMA PDP-11 TO VAX

```

1650 023672 007740          .WORD  ERR10
1650 023674 004737 013116  CALL   ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1651 023700 012701 013630  MOV    #ABORT,R1      ; ABORT CODE TO R1
1652 023704 000137 024232  JMP    EZ              ; END TEST
1653
1654                      ; END ERROR CONDITION
1655
1656 023710          30$:
1657 023710 053777 002516 156234  BIS    M12,@QDMA      ; SET UP EXTENDED BITS IN EXP
1658 023716 017737 156246 002512  MOV    @QVDA,WORD.COUNT ; READ WORD FROM VIEW
1659
1660                      ; MOVE DATA PATTERN TO BUFFER
1661
1662 023724 013703 002502          MOV    FREMEM,R3      ; GET STARTING ADDRESS OF BUFFER IN R3
1663 023730 013702 002512          MOV    WORD.COUNT,R2 ; GET WORD COUNT TO R2
1664 023734 005402          NEG    R2             ; CONVERT WORD COUNT FROM 2'S COMP
1665 023736 012704 000002 40$:  MOV    #2,R4          ; # OF WORDS IN PATTERN
1666 023742 012701 024276          MOV    #DP4,R1       ; POINT TO PATTERN
1667 023746 012123 50$:  MOV    (R1)+,(R3)+    ; MOVE PATTERN WORD TO BUFFER
1668 023750 005302          DEC    R2             ; DECREMENT WORD COUNT
1669 023752 001403          BEQ    55$           ; BRANCH IF DONE
1670 023754 005304          DEC    R4             ; DECREMENT PATTERN WORD COUNT
1671 023756 001373          BNE    50$           ; IF NOT LAST PATTERN WORD, LOOP
1672 023760 000766          BR     40$           ; START ALL OVER AGAIN
1673
1674                      ; BUFFER TAKEN CARE OF, SET AND WAIT FOR INTERRUPT
1675
1676 023762 013777 002512 156170 55$:  MOV    WORD.COUNT,@QWC ; MOVE WORD COUNT INTO QWC REGISTER
1677 023770 013777 002502 156160  MOV    FREMEM,@QBA    ; START ADDRESS OF BUFFER TO QBA
1678 023776 042777 037400 156146  BIC    #37400,@QDMA   ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
1679 024004 052777 000002 156140  BIS    #DMAWT,@QDMA   ; SET UP FOR PDP TO VAX DMA TRANSFER
1680 024012          CLRVEC VECTOR
1680 024012 013700 002444  MOV    VECTOR,R0
1680 024016 104436  TRAP  C#CVEC
1681 024020          SETVEC VECTOR,#COX1,#PRI07 ; SET UP NEW INTERRUPT VECTOR
1681 024020 012746 000340  MOV    #PRI07,-(SP)
1681 024024 012746 024144  MOV    #COX1,-(SP)
1681 024030 013746 002444  MOV    VECTOR,-(SP)
1681 024034 012746 000003  MOV    #3,-(SP)
1681 024040 104437  TRAP  C#SVEC
1681 024042 062706 000010  ADD    #10,SP
1682 024046 052777 000100 156072  BIS    #ERRIE,@QINT   ; ENABLE ERROR INTERRUPT
1683 024054 052777 000100 156070  BIS    #RDYIE,@QDMA   ; ENABLE RDY INTERRUPT
1684 024062 052777 000001 156062  BIS    #GO,@QDMA      ; INITIATE DMA TRANSFER
1685
1686                      ; WAIT FOR INTERRUPT
1687
1688 024070 013701 002532          MOV    DEFAULT,R1    ; NUMBER OF SECONDS OF TIMEOUT
1689 024074 012702 026000 60$:  MOV    #TIMOUT,R2    ; TIME OUT DELAY TO R2
1690 024100          65$:  BREAK
1690 024100 104422  TRAP  C#BRK          ; ALLOW CONTROL C
1691 024102 005302          DEC    R2             ; DECREMENT TIMEOUT
1692 024104 001375          BNE    65$           ; IF NOT DONE LOOP
1693 024106 005301          DEC    R1             ; DECREMENT SECOND COUNTER
1694 024110 001371          BNE    60$           ; IF NOT DONE, LOOP
1695
1696                      ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT

```

TEST 15: DMA PDP-11 TO VAX

```

1697
1698 024112      ;
      024112 104456      ERRHRD 40,,,ERR7      ; TELL OPERATOR TIMEOUT
      024114 000050      TRAP    C#ERRHRD
      024116 000000      .WORD  40
      024120 007424      .WORD  0
1699 024122 004737 013116      .WORD  ERR7
1700 024126 042777 000100 156016      CALL  ENDERR      ; GO PRINT OUT ENDING TERMINATOR
1701 024134 012701 013630      BIC   #RDYIE,#QDMA      ; CLEAR INTERRUPT BIT
1702 024140 000137 024232      MOV   #ABORT,R1      ; ABORT CODE TO R1
      JMP   EZ      ; BRANCH TO END OF TEST
1703
1704      ; END ERROR CONDITION
1705
1706      ; DMA INTERRUPT SHOULD GO HERE
1707
1708 024144      ;
      024144      BGNSRV  COX1
1709 024144 042777 000100 156000      COX1:: BIC   #RDYIE,#QDMA      ; CLEAR INTERRUPT BIT
1710 024152 012716 024160      MOV   #COX2,(SP)      ; CORRECT STACK
1711 024156      ENDSRV
      024156 000002      L10041: RTI
1712 024160      COX2:
1713 024160 032777 100000 155760      BIT   #ERR,#QINT      ; ERROR INTERRUPT?
1714 024166 001414      BEQ   70#      ; BRANCH IF NO ERROR INTERRUPT
1715
1716      ; ERROR CONDITION, ERROR INTERRUPT OCCURED
1717
1718 024170      ;
      024170 104456      ERRHRD 41,,,ERR11      ; TELL OPERATOR ERROR INTERRUPT OCCURED
      024172 000051      TRAP    C#ERRHRD
      024174 000000      .WORD  41
      024176 010030      .WORD  0
1719 024200 004737 013116      .WORD  ERR11
1720 024204 005077 155736      CALL  ENDERR      ; GO PRINT OUT ENDING TERMINATOR
1721 024210 012701 013630      CLR   #QINT      ; CLEAR INTERRUPT REGISTER
1722 024214 000137 024232      MOV   #ABORT,R1      ; ABORT CODE TO R1
      JMP   EZ      ; BRANCH TO END OF TEST
1723
1724      ; END ERROR CONDITION
1725
1726 024220      ;
1727 024220 052777 000001 155714      70#: BIS   #CMDACK,#QCMD      ; TELL MASTER DAM DONE
1728
1729      ; END OF TEST
1730
1731 024226 012701 000000      ;
1732 024232      EZ: MOV   #0,R1      ; END TEST NORMALLY
      024232 013700 002444      CLRVEC VECTOR      ; CLEAR VECTOR
      024236 104436      MOV   VECTOR,R0
1733 024240      TRAP  C#CVEC
      024240 012746 000340      SETVEC VECTOR,#ERRINT,#PRI07      ; SET UP INTERRUPT VECTOR
      024244 012746 012652      MOV   #PRI07,-(SP)
      024250 013746 002444      MOV   #ERRINT,-(SP)
      024254 012746 000003      MOV   VECTOR,-(SP)
      024260 104437      MOV   #3,-(SP)
      024262 062706 000010      TRAP  C#SVEC
1734 024266 052777 000100 155652      ADD   #10,SP
      BIS   #ERRIE,#QINT      ; ENABLE ERROR INTERRUPT

```



TEST 15: DMA PDP-11 TO VAX

1735  
 1736 024274 000207  
 1737  
 1738  
 1739  
 1740  
 1741 024276 000000  
 1742 024300 177777  
 1743

RTS PC ; RETURN TO MAINLINE  
 ;  
 ; DATA PATTERN FOR TEST  
 ;  
 DP4: .WORD 0  
 .WORD 177777

TEST 16: DMA VAX (BUFFERED DATA PATH) TO PDP-11

1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756

```
.SBTTL TEST 16: DMA VAX (BUFFERED DATA PATH) TO PDP-11
;*****
;
;TEST DESCRIPTION-
; WRITE LARGE BUFFER FROM COMMAND BRIDGE TO CONTROL BRIDGE
; THE VAX USES A BUFFERED DATA PATH
; CHECK DATA
;TEST STEPS-
;
;--
;*****
```

1757 024302  
024302  
024302 012737 000020 002114  
024310 004737 013034  
1758 024314 013701 002532  
1759 024320 012702 026000  
1760 024324  
024324 104422  
1761 024326 032777 004000 155612  
1762 024334 001020  
1763 024336 005302  
1764 024340 001371  
1765 024342 005301  
1766 024344 001365  
1767  
1768  
1769  
1770 024346 012701 000000  
1771 024352  
024352 104456  
024354 000051  
024356 000000  
024360 006506  
1772 024362 004737 013116  
1773 024366 012701 013630  
1774 024372 000137 025216  
1775  
1776  
1777  
1778  
1779  
1780 024376  
1781 024376 052777 000020 155544  
1782 024404 012777 000000 155554  
1783 024412 017737 155552 002514  
1784 024420 013777 002514 155526  
1785 024426 017737 155536 002516  
1786 024434 032737 177757 002516  
1787 024442 001412  
1788  
1789  
1790  
1791 024444  
024444 104456  
024446 000053

```
STATST
TEST16::
MOV #16,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
CALL BMSG ;CHECK FOR PNT FLAG
MOV DEFAULT,R1 ; SECONDS OF TIME OUT
MOV #TIMOUT,R2 ; TIME OUT VALUE TO R2
8#:
10#:
BREAK
TRAP C#BRK
BIT #CMDPRS,@QINT ; COMMAND PRESENT?
BNE 20# ; IF COMMAND PRESENT, BRANCH
DEC R2 ; DECREMENT TIMEOUT REGISTER
BNE 10# ; IF NOT IN TIMEOUT, TRY AGAIN
DEC R1 ; DECREMENT SECOND TIMER
BNE 8# ; IF NOT TIMEOUT, AGAIN

;
; TIMEOUT ERROR CONDITION
;
MOV #0,R1 ; REGISTER # TO R1
ERRHRD 41,,,ERR2 ; TELL OPERATOR TIMEOUT
TRAP C#ERHRD
.WORD 41
.WORD 0
.WORD ERR2
CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
MOV #ABORT,R1 ; ABORT CODE TO R1
JMP EXW ; ABORT TEST

;
; END TIMEOUT ERROR CONDITION
;
; COMMAND PRESENT WAS REALLY THERE
;
20#:
BIS #QVREN,@QCTL ; READ ENABLE
MOV #0,@QVAD ; CLEAR VIEW REGISTER
MOV @QVDA,LOW16 ; READ LOW 16 BITS OF UNIBUS ADDRESS
MOV LOW16,@QUBA ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
MOV @QVDA,HI2 ; READ HI 2 BITS OF UNIBUS ADDRESS
BIT #+CBIT4!BITS,HI2 ; SEE IF ONLY BITS 4 AND 5 ARE SET
BEQ 30# ; BRANCH IF OK

;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
;
ERRHRD 43,,,ERR10 ; TELL OPERATOR ERROR
TRAP C#ERHRD
.WORD 43
```

## TEST 16: DMA VAX (BUFFERED DATA PATH) TO PDP-11

```

024450 000000          .WORD 0
024452 007740          .WORD ERR10
1792 024454 004737 013116 CALL ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1793 024460 012701 013630 MOV  #ABORT,R1        ; ABORT CODE TO R1
1794 024464 000137 025216 JMP  EXW              ; END TEST
1795
1796          ; END ERROR CONDITION
1797
1798 024470          30$:
1799 024470 053777 002516 155454 BIS  HI2,@QDMA        ; SET UP EXTENDED BITS IN EXP
1800 024476 017737 155466 002512 MOV  @QVDA,WORD.COUNT ; READ WORD FROM VIEW
1801
1802          ; CLEAR BUFFER
1803
1804 024504 013701 002512          MOV  WORD.COUNT,R1    ; GET WORD.COUNT IN R1
1805 024510 005401          NEG  R1                ; CONVERT WORD COUNT FROM 2'S COMP
1806 024512 013702 002502          MOV  FREM,R2          ; STARTING ADDRESS OF BUFFER
1807 024516 005022          35$: CLR  (R2)+            ; CLEAR WORD IN BUFFER
1808 024520 005301          DEC  R1                ; DECREMENT COUNTER
1809 024522 001375          BNE  35$              ; BRANCH BACK IF NOT DONE
1810
1811          ; BUFFER CLEARED, SET UP FOR DMA
1812
1813 024524 013777 002512 155426 MOV  WORD.COUNT,@QWC  ; MOVE WORD COUNT INTO QWC REGISTER
1814 024532 013777 002502 155416 MOV  FREM,@QBA       ; START ADDRESS OF BUFFER TO QBA
1815 024540 042777 037400 155404 BIC  #37400,@QDMA    ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
1816 024546 042777 000002 155376 BIC  #DMAWT,@QDMA    ; SET UP FOR VAX TO PDP DMA TRANSFER
1817 024554
1818 024554 013700 002444          CLRVEC VECTOR
1819 024560 104436          MOV  VECTOR,R0
1820 024562          TRAP  C:CVFC
1821 024562          SETVEC VECTOR,@CXP1,@PRI07 ; SET UP NEW INTERRUPT VECTOR
1822 024562 012746 000340          MOV  @PRI07,-(SP)
1823 024566 012746 024706          MOV  @CXP1,-(SP)
1824 024572 013746 002444          MOV  VECTOR,-(SP)
1825 024576 012746 000003          MOV  #3,-(SP)
1826 024602 104437          TRAP  C:SVEC
1827 024604 062706 000010          ADD  #10,SP
1828 024610 052777 000100 155330 BIS  @ERRIE,@QINT    ; ENABLE ERROR INTERRUPT
1829 024616 052777 000100 155326 BIS  @RDYIE,@QDMA    ; ENABLE RDY INTERRUPT
1830 024624 052777 000001 155320 BIS  @GO,@QDMA       ; INITIATE DMA TRANSFER
1831
1832          ; WAIT FOR INTERRUPT
1833
1834          ;
1835 024632 013701 002532          MOV  DEFAULT,R1     ; NUMBER OF SECONDS OF TIMEOUT
1836 024636 012702 026000          40$: MOV  @TIMOUT,R2   ; TIME OUT DELAY TO R2
1837 024642          45$: BREAK          ; ALLOW CONTROL C
1838 024642 104422          TRAP  C:BRK
1839 024644 005302          DEC  R2                ; DECREMENT TIMEOUT
1840 024646 001375          BNE  45$              ; IF NOT DONE LOOP
1841 024650 005301          DEC  R1                ; DECREMENT SECOND COUNTER
1842 024652 001371          BNE  40$              ; IF NOT DONE, LOOP
1843
1844          ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
1845
1846 024654          ERRHRD 44...ERR7 ; TELL OPERATOR TIMEOUT
1847 024654 104456          TRAP  C:ERHRD
1848 024656 000054          .WORD 44

```

TEST 16: DMA VAX (BUFFERED DATA PATH) TO PDP-11

```

024660 000000 .WORD 0
024662 007424 .WORD ERR7
1836 024664 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR.
1837 024670 042777 000100 155254 BIC #RDYIE,%QDMA ; CLEAR INTERRUPT BIT
1838 024676 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
1839 024702 000137 025216 JMP EXW ; BRANCH TO END OF TEST
1840 ;
1841 ; END ERROR CONDITION
1842 ;
1843 ; DMA INTERRUPT SHOULD GO HERE
1844 ;
1845 024706 BGNSRV CXP1
024706 CXP1::
1846 024706 042777 000100 155236 BIC #RDYIE,%QDMA ; CLEAR INTERRUPT BIT
1847 024714 012716 024722 MOV #CXP2,(SP) ; CORRECT STACK
1848 024720 ENDSRV
024720 L10042:
024720 000002 RTI
1849 024722 CXP2:
1850 024722 032777 100000 155216 BIT #ERR,%QINT ; ERROR INTERRUPT?
1851 024730 001414 BEQ 500 ; BRANCH IF NO ERROR INTERRUPT
1852 ;
1853 ; ERROR CONDITION, ERROR INTERRUPT OCCURED
1854 ;
1855 024732 ERRHRD 45...ERR11 ; TELL OPERATOR ERROR INTERRUPT OCCURED
024732 104456 TRAP C%ERRHD
024734 000055 .WORD 45
024736 000000 .WORD 0
024740 010030 .WORD ERR11
1856 024742 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
1857 024746 005077 155174 CLR %QINT ; CLEAR INTERRUPT REGISTER
1858 024752 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
1859 024756 000137 025216 JMP EXW ; BRANCH TO END OF TEST
1860 ;
1861 ; END ERROR CONDITION
1862 ;
1863 024762 500:
1864 024762 052777 000001 155152 BIS #CMDACK,%QCMD ; TELL MASTER DAM DONE
1865 ;
1866 ; NOW CHECK DATA
1867 ;
1868 024770 013701 002502 MOV FREMEM,R1 ; R1 TO START OF BUFFER
1869 024774 013702 002512 MOV WORD.COUNT,R2 ; R2 GETS WORD COUNT
1870 025000 005402 NEG R2 ; CONVERT WORD COUNT FROM 2'S COMP.
1871 ;
1872 ; NOW CHECK BUFFER AGAINST PATTERN
1873 ;
1874 025002 005037 002462 CLR ERROR1 ; CLEAR ERROR COUNTER
1875 025006 012704 000002 1000: MOV #2,R4 ; # OF WORDS IN PATTERN
1876 025012 012703 025262 MOV #DPX3,R3 ; R3 GETS PATTERN ADDRESS
1877 025016 012137 002454 1100: MOV (R1)+,VAL ; GET WORD FROM BUFFER
1878 025022 012337 002456 MOV (R3)+,EXP ; GET EXPECTED PATTERN IN EXP
1879 025026 023737 002454 002456 CMP VAL,EXP ; COMPARE VALUE WITH EXPECTED
1880 025034 001442 BEQ 1300 ; IF OK, BRANCH
1881 ;
1882 ; ERROR CONDITION, NOT WHAT EXPECTED IN BUFFER
1883 ;

```

TEST 16: DMA VAX (BUFFERED DATA PATH) TO PDP-11

```

1884 025036 005737 002462          TST      ERROR1          ; ANY OTHER ERRORS OCCURED
1885 025042 001004          BNE      120$           ; SKIP HEADER ERROR IF ALREADY PRINTED
1886 025044          ERRHRD  46...ERR8          ; PRINT ERROR HARD MESSAGE
      025044 104456          TRAP     C$ERRHRD
      025046 000056          .WORD   46
      025050 000000          .WORD   0
      025052 007530          .WORD   ERR8
1887 025054 005237 002462          INC      ERROR1          ; INCREMENT ERROR COUNTER
1888 025060 022737 000012 002462 120$:  CMP      #10.,ERROR1      ; SEE IF TOO MANY MESSAGES PRINTED
1889 025066 103425          BCS      130$           ; IF TOO MANY, BRANCH
1890 025070 013737 002512 002474  MOV      WORD,COUNT,COUNTER ; CALCULATE WORD NUMBER
1891 025076 005437 002474          NEG      COUNTER        ; CONVERT WORD COUNT FROM 2'S COMP
1892 025102 160237 002474          SUB      R2,COUNTER
1893 025106          PRINTX  #FOR10,COUNTER,VAL,EXP ; PRINT DETAIL LINE
      025106 013746 002456          MOV      EXP,-(SP)
      025112 013746 002454          MOV      VAL,-(SP)
      025116 013746 002474          MOV      COUNTER,-(SP)
      025122 012746 004230          MOV      #FOR10,-(SP)
      025126 012746 000004          MOV      #4,-(SP)
      025132 010600          MOV      SP,R0
      025134 104415          TRAP     C$PNTX
      025136 062706 000012          ADD      #12,SP
1894          ;
1895          ; END ERROR CONDITION
1896          ;
1897 025142          130$:
1898 025142 005302          DEC      R2              ; DECREMENT WORD COUNT
1899 025144 001403          BEQ      140$           ; BRANCH OUT IF DONE
1900 025146 005304          DEC      R4              ; DECREMENT PATTERN WORD COUNT
1901 025150 001322          BNE      110$           ; BRANCH IF NOT DONE WITH PATTERN
1902 025152 000715          BR       100$           ; BRANCH IF DONE WITH PATTERN
1903          ;
1904          ; BUFFER CHECKED, PRINT TOTAL ERROR COUNT IF NECESSARY
1905          ;
1906 025154          140$:
1907 025154 005737 002462          TST      ERROR1          ; ANY ERRORS
1908 025160 001414          BEQ      150$           ; IF NO ERRORS, BRANCH
1909 025162          PRINTB  #FOR11,ERROR1      ; TELL TOTAL ERROR COUNT
      025162 013746 002462          MOV      ERROR1,-(SP)
      025166 012746 004253          MOV      #FOR11,-(SP)
      025172 012746 000002          MOV      #2,-(SP)
      025176 010600          MOV      SP,R0
      025200 104414          TRAP     C$PNTB
      025202 062706 000006          ADD      #6,SP
1910 025206 004737 013116          CALL    ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1911          ;
1912          ; END OF TEST
1913          ;
1914 025212 012701 000000          150$:  MOV      #0,R1          ; END TEST NORMALLY
1915 025216          EXW:  CLRVEC  VECTOR          ; CLEAR VECTOR
      025216 013700 002444          MOV      VECTOR,R0
      025222 104436          TRAP     C$CVEC
1916 025224          SETVEC  VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
      025224 012746 000340          MOV      #PRI07,-(SP)
      025230 012746 012652          MOV      #ERRINT,-(SP)
      025234 013746 002444          MOV      VECTOR,-(SP)
      025240 012746 000003          MOV      #3,-(SP)
    
```

TEST 16: DMA VAX (BUFFERED DATA PATH) TO PDP-11

```

      025244 104437
      025246 062706 000010
1917 025252 052777 000100 154666
1918
1919 025260 000207
1920
1921
1922
1923
1924 025262 000000
1925 025264 177777

```

TRAP    C#SVEC  
 ADD    #10,SP  
 BIS    #ERRIE,#QINT      ; ENABLE ERROR INTERRUPT  
  
 RTS    PC                ; RETURN TO MAINLINE  
  
 ; DATA PATTERN FOR TEST  
 ;  
 DPX3:  .WORD  0  
  .WORD  177777

TEST 17: DMA PDP-11 TO VAX (BUFFERED DATA PATH)

```

1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939 025266
      025266
      025266 012737 000021 002114
      025274 004737 013034
1940 025300 013701 002532
1941 025304 012702 026000
1942 025310
      025310 104422
1943 025312 032777 004000 154626
1944 025320 001020
1945 025322 005302
1946 025324 001371
1947 025326 005301
1948 025330 001365
1949
1950
1951
1952 025332 012701 000000
1953 025336
      025336 104456
      025340 000057
      025342 000000
      025344 006506
1954 025346 004737 013116
1955 025352 012701 013630
1956 025356 000137 025776
1957
1958
1959
1960
1961
1962 025362
1963 025362 052777 000020 154560
1964 025370 012777 000000 154570
1965 025376 017737 154566 002514
1966 025404 013777 002514 154542
1967 025412 017737 154552 002516
1968 025420 032737 177757 002516
1969 025426 001412
1970
1971
1972
1973 025430
      025430 104456
      025432 000060

```

```

.SBTTL TEST 17: DMA PDP-11 TO VAX (BUFFERED DATA PATH)
;*****
;
;TEST DESCRIPTION-
; WRITE LARGE BUFFER FROM CONTROL BRIDGE TO COMMAND BRIDGE.
; THE VAX USES A BUFFERED DATA PATH
; CHECK DATA
;TEST STEPS-
;
;--
;*****
;
;          STATST
TEST17::
      MOV      #17,L#TEST          ; TEST NUMBER FOR ERROR MESSAGES
      CALL    BMSG                 ; CHECK FOR PNT FLAG
      MOV     DEFAULT,R1           ; SECONDS OF TIME OUT
80:    MOV     @TIMEOUT,R2         ; TIME OUT VALUE TO R2
100:   BREAK
      TRAP    C#BRK
      BIT     @CMDPRS,@QINT        ; COMMAND PRESENT?
      BNE    200                  ; IF COMMAND PRESENT, BRANCH
      DEC    R2                   ; DECREMENT TIMEOUT REGISTER
      BNE    100                  ; IF NOT IN TIMEOUT, TRY AGAIN
      DEC    R1                   ; DECREMENT SECOND TIMER
      BNE    80                   ; IF NOT TIMEOUT, AGAIN
;
; TIMEOUT ERROR CONDITION
;
      MOV     #0,R1                ; REGISTER # TO R1
      ERHRD  47...ERR2            ; TELL OPERATOR TIMEOUT
      TRAP   C#ERHRD
      .WORD  47
      .WORD  0
      .WORD  ERR2
      CALL   ENDERR                ; GO PRINT OUT ENDING TERMINATOR
      MOV    @ABORT,R1             ; ABORT CODE TO R1
      JMP   EQZ                    ; ABORT TEST
;
; END TIMEOUT ERROR CONDITION
;
; COMMAND PRESENT WAS REALLY THERE
;
200:   BIS     @QVREN,@QCTL         ; READ ENABLE
      MOV     #0,@QVAD             ; CLEAR VIEW REGISTER
      MOV     @QVDA,LOW16          ; READ LOW 16 BITS OF UNIBUS ADDRESS
      MOV     LOW16,@QUBA          ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
      MOV     @QVDA,HI2           ; READ HI 2 BITS OF UNIBUS ADDRESS
      BIT     @+CBIT4!BIT5,HI2    ; SEE IF ONLY BITS 4 AND 5 ARE SET
      BEQ    300                  ; BRANCH IF OK
;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
;
      ERHRD  48...ERR10           ; TELL OPERATOR ERROR
      TRAP   C#ERHRD
      .WORD  48

```

## TEST 17: DMA PDP-11 TO VAX (BUFFERED DATA PATH)

```

025434 000000          .WORD 0
025436 007740          .WORD ERR10
1974 025440 004737 013116 CALL ENDERR          ; GO PRINT OUT ENDING TERMINATOR
1975 025444 012701 013630 MOV  @ABORT,R1        ; ABORT CODE TO R1
1976 025450 000137 025776 JMP  EQZ              ; END TEST
1977
1978 ; END ERROR CONDITION
1979
1980 025454          300:
1981 025454 053777 002516 154470 BIS  HI2,@QDMA        ; SET UP EXTENDED BITS IN EXP
1982 025462 017737 154502 002512 MOV  @QVDA,WORD.COUNT ; READ WORD FROM VIEW
1983
1984 ; MOVE DATA PATTERN TO BUFFER
1985
1986 025470 013703 002502          MOV  FREMEM,R3        ; GET STARTING ADDRESS OF BUFFER IN R3
1987 025474 013702 002512          MOV  WORD.COUNT,R2    ; GET WORD COUNT TO R2
1988 025500 005402          NEG  R2              ; CONVERT WORD COUNT FROM 2'S COMP
1989 025502 012704 000002 400: MOV  @2,R4            ; # OF WORDS IN PATTERN
1990 025506 012701 026042          MOV  @DPX4,R1         ; POINT TO PATTERN
1991 025512 012123 500: MOV  (R1),.(R3).    ; MOVE PATTERN WORD TO BUFFER
1992 025514 005302          DEC  R2              ; DECREMENT WORD COUNT
1993 025516 001403          BEQ  550             ; BRANCH IF DONE
1994 025520 005304          DEC  R4              ; DECREMENT PATTERN WORD COUNT
1995 025522 001373          BNE  500             ; IF NOT LAST PATTERN WORD, LOOP
1996 025524 000766          BR   400             ; START ALL OVER AGAIN
1997
1998 ; BUFFER TAKEN CARE OF, SET AND WAIT FOR INTERRUPT
1999
2000 025526 013777 002512 154424 550: MOV  WORD.COUNT,@QWC    ; MOVE WORD COUNT INTO QWC REGISTER
2001 025534 013777 002502 154414 MOV  FREMEM,@QBA     ; START ADDRESS OF BUFFER TO QBA
2002 025542 042777 037400 154402 BIC  @37400,@QDMA    ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
2003 025550 052777 000002 154374 BIS  @DMAWT,@QDMA    ; SET UP FOR PDP TO VAX DMA TRANSFER
2004 025556
    025556 013700 002444          CLRVEC VECTOR
    025562 104436          MOV  VECTOR,R0
2005 025564          TRAP  C0CVEC
    025564 012746 000340          SETVEC VECTOR,@CXX1,@PRI07 ; SET UP NEW INTERRUPT VECTOR
    025570 012746 025710          MOV  @PRI07,-(SP)
    025574 013746 002444          MOV  @CXX1,-(SP)
    025600 012746 000003          MOV  VECTOR,-(SP)
    025604 104437          MOV  @3,-(SP)
    025606 062706 000010          TRAP  C0SVEC
    025612 052777 000100 154326 ADD  @10,SP
2006 025612 052777 000100 154326 BIS  @ERRIE,@QINT    ; ENABLE ERROR INTERRUPT
2007 025620 052777 000100 154324 BIS  @RDYIE,@QDMA    ; ENABLE RDY INTERRUPT
2008 025626 052777 000001 154316 BIS  @GO,@QDMA       ; INITIATE DMA TRANSFER
2009
2010 ; WAIT FOR INTERRUPT
2011
2012 025634 013701 002532          MOV  DEFAULT,R1     ; NUMBER OF SECONDS OF TIMEOUT
2013 025640 012702 026000 600: MOV  @TIMOUT,R2    ; TIME OUT DELAY TO R2
2014 025644          650: BREAK
    025644 104422          TRAP  C0BRK        ; ALLOW CONTROL C
2015 025646 005302          DEC  R2              ; DECREMENT TIMEOUT
2016 025650 001375          BNE  650             ; IF NOT DONE LOOP
2017 025652 005301          DEC  R1              ; DECREMENT SECOND COUNTER
2018 025654 001371          BNE  600             ; IF NOT DONE, LOOP
2019

```



## TEST 17: DMA PDP-11 TO VAX (BUFFERED DATA PATH)

```

2020 ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
2021 ;
2022 025656 ; ERRHRD 49...ERR7 ; TELL OPERATOR TIMEOUT
      025656 104456 TRAP C#ERRHRD
      025660 000061 .WORD 49
      025662 000000 .WORD 0
      025664 007424 .WORD ERR7
2023 025666 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
2024 025672 042777 000100 154252 BIC #RDYIE,#QDMA ; CLEAR INTERRUPT BIT
2025 025700 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
2026 025704 000137 025776 JMP EQZ ; BRANCH TO END OF TEST
2027 ;
2028 ; END ERROR CONDITION
2029 ;
2030 ; DMA INTERRUPT SHOULD GO HERE
2031 ;
2032 025710 ; BGNSRV CXX1
      025710 CXX1::
2033 025710 042777 000100 154234 BIC #RDYIE,#QDMA ; CLEAR INTERRUPT BIT
2034 025716 012716 025724 MOV #CXX2,(SP) ; CORRECT STACK
2035 025722 ENDSRV
      025722 L10043:
      025722 000002 RTI
2036 025724 CXX2:
2037 025724 032777 100000 154214 BIT #ERR,#QINT ; ERROR INTERRUPT?
2038 025732 001414 BEQ 70# ; BRANCH IF NO ERROR INTERRUPT
2039 ;
2040 ; ERROR CONDITION, ERROR INTERRUPT OCCURED
2041 ;
2042 025734 ; ERRHRD 50...ERR11 ; TELL OPERATOR ERROR INTERRUPT OCCURED
      025734 104456 TRAP C#ERRHRD
      025736 000062 .WORD 50
      025740 000000 .WORD 0
      025742 010030 .WORD ERR11
2043 025744 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
2044 025750 005077 154172 CLR #QINT ; CLEAR INTERRUPT REGISTER
2045 025754 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
2046 025760 000137 025776 JMP EQZ ; BRANCH TO END OF TEST
2047 ;
2048 ; END ERROR CONDITION
2049 ;
2050 025764 ; 70#:
2051 025764 052777 000001 154150 BIS #CMDACK,#QCMD ; TELL MASTER DAM DONE
2052 ;
2053 ; END OF TEST
2054 ;
2055 025772 012701 000000 MOV #0,R1 ; END TEST NORMALLY
2056 025776 EQZ: CLRVEC VECTOR ; CLEAR VECTOR
      025776 013700 002444 MOV VECTOR,R0
      026002 104436 TRAP C#CVEC
2057 026004 SETVEC VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
      026004 012746 000340 MOV #PRI07,-(SP)
      026010 012746 012652 MOV #ERRINT,-(SP)
      026014 013746 002444 MOV VECTOR,-(SP)
      026020 012746 000003 MOV #3,-(SP)
      026024 104437 TRAP C#SVEC
      026026 062706 000010 ADD #10,SP

```



TEST 18: DMA PDP-11 HI MEMORY TO VAX

```

2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079 026046
      026046
      026046 012737 000022 002114
2080 026054 004737 013034
      026060 013701 002532
2081 026064 012702 026000
2082 026070
      026070 104422
2083 026072 032777 004000 154046
2084 026100 001020
2085 026102 005302
2086 026104 001371
2087 026106 005301
2088 026110 001365
2089
2090
2091
2092 026112 012701 000000
2093 026116
      026116 104456
      026120 000063
      026122 000000
      026124 006506
2094 026126 004737 013116
2095 026132 012701 013630
2096 026136 000137 026706
2097
2098
2099
2100
2101
2102 026142
2103 026142 052777 000020 154000
2104 026150 012777 000000 154010
2105 026156 017737 154006 002514
2106 026164 013777 002514 153762
2107 026172 017737 153772 002516
2108 026200 032737 177757 002516
2109 026206 001412
2110
2111
2112
2113 026210
      026210 104456
      026212 000064
      026214 000000

```

```

.SBTTL TEST 18: DMA PDP-11 HI MEMORY TO VAX
;*****
;..
;TEST DESCRIPTION-
; WRITE 2 WORD BUFFER FROM PDP-AA MAX MEMORY TO VAX
; CHECK DATA
;TEST STEPS-
;
;--
;*****
          STATST
TEST18::
          MOV     #18,L#TEST           ;TEST NUMBER FOR ERROR MESSAGES
          CALL   BMSG                  ;CHECK FOR PNT FLAG
          MOV     DEFAULT,R1           ; SECONDS OF TIME OUT
          MOV     #TIMOUT,R2          ; TIME OUT VALUE TO R2
8$:
10$:
          BREAK
          TRAP   C#BRK
          BIT    #CMDPRS,#QINT        ; COMMAND PRESENT?
          BNE    20$                   ; IF COMMAND PRESENT, BRANCH
          DEC    R2                    ; DECREMENT TIMEOUT REGISTER
          BNE    10$                   ; IF NOT IN TIMEOUT, TRY AGAIN
          DEC    R1                    ; DECREMENT SECOND TIMER
          BNE    8$                    ; IF NOT TIMEOUT, AGAIN
;
; TIMEOUT ERROR CONDITION
;
          MOV     #0,R1                ; REGISTER # TO R1
          ERRHRD 51,,,ERR2            ; TELL OPERATOR TIMEOUT#
          TRAP   C#ERHRD
          .WORD  51
          .WORD  0
          .WORD  ERR2
          CALL   ENDERR                ; GO PRINT OUT ENDING TERMINATOR
          MOV     #ABORT,R1            ; ABORT CODE TO R1
          JMP    EXZ                   ; ABORT TEST
;
; END TIMEOUT ERROR CONDITION
;
; COMMAND PRESENT WAS REALLY THERE
;
20$:
          BIS     #QVREN,#QCTL         ; READ ENABLE
          MOV     #0,#QVAD             ; CLEAR VIEW REGISTER
          MOV     #QVDA,LOW16         ; READ LOW 16 BITS OF UNIBUS ADDRESS
          MOV     #QVDA,HI2          ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
          BIT     #CBIT4!BITS,HI2    ; READ HI 2 BITS OF UNIBUS ADDRESS
          BEQ    30$                  ; SEE IF ONLY BITS 4 AND 5 ARE SET
; BRANCH IF OK
;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
;
          ERRHRD 52,,,ERR10           ; TELL OPERATOR ERROR
          TRAP   C#ERHRD
          .WORD  52
          .WORD  0

```

## TEST 18: DMA PDP-11 HI MEMORY TO VAX

```

026216 007740 .WORD ERR10
2114 026220 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
2115 026224 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
2116 026230 000137 026706 JMP EXZ ; END TEST
2117 ;
2118 ; END ERROR CONDITION
2119 ;
2120 026234 30#: BIS HI2,@QDMA ; SET UP EXTENDED BITS IN EXP
2121 026234 053777 002516 153710 MOV @QVDA,WORD.COUNT ; READ WORD FROM VIEW
2122 026242 017737 153722 002512 ;
2123 ; SET UP MEMORY MANAGEMENT
2124 ;
2125 ;
2126 026250 012737 000000 002520 MOV #NO,MAPPING ; DEFAULT IS NO MAPPING IN FLAG
2127 026256 013701 002120 MOV L#HIMEM,R1 ; GET PAR FORMAT MEM SIZE IN R1
2128 026262 020127 001600 CMP R1,#1600 ; SEE IF THERE IS A HI MEMORY
2129 026266 100415 BMI 35# ; BRANCH IF NO HI MEMORY
2130 026270 012737 000001 002520 MOV #YES,MAPPING ; SET MAPPING FLAG
2131 026276 004737 012754 JSR PC,#MINIT ; INITIALIZE MEMORY MANAGEMENT
2132 026302 010137 172350 MOV R1,APR4 ; SET UP HI MEMORY IN APR4
2133 026306 012703 100000 MOV #100000,R3 ; SET UP BUFFER ADDRESS
2134 026312 012737 000001 177572 MOV #BIT0,#M#R0 ; TURN ON MEMORY MANAGEMENT
2135 026320 000402 BR 38# ; GO MOVE DATA PATTERN
2136 ;
2137 ; MOVE DATA PATTERN TO BUFFER
2138 ;
2139 026322 013703 002502 35#: MOV FREMEM,R3 ; GET STARTING ADDRESS OF BUFFER IN R3
2140 026326 013702 002512 38#: MOV WORD.COUNT,R2 ; GET WORD COUNT TO R2
2141 026332 005402 NEG R2 ; CONVERT WORD COUNT FROM 2'S COMP
2142 026334 012704 000002 40#: MOV #2,R4 ; # OF WORDS IN PATTERN
2143 026340 012701 026752 MOV #DP5,R1 ; POINT TO PATTERN
2144 026344 012123 50#: MOV (R1)+,(R3)+ ; MOVE PATTERN WORD TO BUFFER
2145 026346 005302 DEC R2 ; DECREMENT WORD COUNT
2146 026350 001403 BEQ 52# ; BRANCH IF DONE
2147 026352 005304 DEC R4 ; DECREMENT PATTERN WORD COUNT
2148 026354 001373 BNE 50# ; IF NOT LAST PATTERN WORD, LOOP
2149 026356 000766 BR 40# ; START ALL OVER AGAIN
2150 ;
2151 ; BUFFER TAKEN CARE OF, SET AND WAIT FOR INTERRUPT
2152 ;
2153 026360 013777 002512 153572 52#: MOV WORD.COUNT,@QWC ; GET WORD.COUNT INTO QWC REGISTER
2154 026366 022737 000001 002520 CMP #YES,MAPPING ; ARE WE MAPPING
2155 026374 001407 BEQ 55# ; BRANCH IF YES
2156 026376 013777 002502 153552 MOV FREMEM,@QBA ; START ADDRESS OF BUFFER TO QBA
2157 026404 042777 037400 153540 BIC #37400,@QDMA ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
2158 026412 000422 BR 58#
2159 ;
2160 ; SET UP PDP ADDRESS WITH MAPPING ENABLED
2161 ;
2162 ; FIRST SET UP HIGH BITS 16 & 17
2163 ;
2164 026414 013703 002120 55#: MOV L#HIMEM,R3 ; GET PAR VALUE IN R3
2165 026420 006203 ASR R3 ; SET UP TO PUT IN QDMA
2166 026422 006203 ASR R3
2167 026424 042703 176377 BIC #C1400,R3 ; MASK OUT UNWANTED BITS
2168 026430 050377 153516 BIS R3,@QDMA ; SET UP QBUS EXTENDED BITS
2169 ;

```

## TEST 18: DMA PDP-11 HI MEMORY TO VAX

```

2170 ; NOW SET UP LOW 16 BITS
2171 ;
2172 026434 013703 002120 ; MOV L#HIMEM,R3 ; GET PAR VALUE IN R3
2173 026440 006303 ; ASL R3 ; MOVE OVER INTO PLACE
2174 026442 006303 ; ASL R3
2175 026444 006303 ; ASL R3
2176 026446 006303 ; ASL R3
2177 026450 006303 ; ASL R3
2178 026452 006303 ; ASL R3
2179 026454 010377 153476 ; MOV R3,QBA ; LOAD LOW 16 BITS INTO QBA
2180 ;
2181 ; NOW SET DIRECTION BIT AND GO
2182 ;
2183 026460 052777 000002 153464 58: ; BIS #DMAWT,@QDMA ; SET UP FOR PDP TO VAX DMA TRANSFER
2184 026466 ; CLRVEC VECTOR
; 026466 013700 002444 ; MOV VECTOR,R0
; 026472 104436 ; TRAP C#CVEC
2185 026474 ; SETVEC VECTOR,#COL1,#PRI07 ; SET UP NEW INTERRUPT VECTOR
; 026474 012746 000340 ; MOV #PRI07,-(SP)
; 026500 012746 026620 ; MOV #COL1,-(SP)
; 026504 013746 002444 ; MOV VECTOR,-(SP)
; 026510 012746 000003 ; MOV #3,-(SP)
; 026514 104437 ; TRAP C#SVEC
; 026516 062706 000010 ; ADD #10,SP
2186 026522 052777 000100 153416 ; BIS #ERRIE,@QINT ; ENABLE ERROR INTERRUPT
2187 026530 052777 000100 153414 ; BIS #RDYIE,@QDMA ; ENABLE RDY INTERRUPT
2188 026536 052777 000001 153406 ; BIS #GO,@QDMA ; INITIATE DMA TRANSFER
2189 ;
2190 ; WAIT FOR INTERRUPT
2191 ;
2192 026544 013701 002532 ; MOV DEFAULT,R1 ; NUMBER OF SECONDS OF TIMEOUT
2193 026550 012702 026000 60: ; MOV #TIMOUT,R2 ; TIME OUT DELAY TO R2
2194 026554 ; 65: ; BREAK ; ALLOW CONTROL C
; 026554 104422 ; TRAP C#BRK
2195 026556 005302 ; DEC R2 ; DECREMENT TIMEOUT
2196 026560 001375 ; BNE 65: ; IF NOT DONE LOOP
2197 026562 005301 ; DEC R1 ; DECREMENT SECOND COUNTER
2198 026564 001371 ; BNE 60: ; IF NOT DONE, LOOP
2199 ;
2200 ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
2201 ;
2202 026566 ; ERRHRD 53,,ERR7 ; TELL OPERATOR TIMEOUT
; 026566 104456 ; TRAP C#ERRHD
; 026570 000065 ; .WORD 53
; 026572 000000 ; .WORD 0
; 026574 007424 ; .WORD ERR7
2203 026576 004737 013116 ; CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
2204 026602 042777 000100 153342 ; BIC #RDYIE,@QDMA ; CLEAR INTERRUPT BIT
2205 026610 012701 013630 ; MOV #ABORT,R1 ; ABORT CODE TO R1
2206 026614 000137 026706 ; JMP EXZ ; BRANCH TO END OF TEST
2207 ;
2208 ; END ERROR CONDITION
2209 ;
2210 ; DMA INTERRUPT SHOULD GO HERE
2211 ;
2212 026620 ; BGNSRV COL1
026620 COL1::

```

TEST 18: DMA PDP-11 HI MEMORY TO VAX

```

2213 026620 042777 000100 153324      BIC      #RDYIE,#QDMA      ; CLEAR INTERRUPT BIT
2214 026626 012716 026634              MOV      #COL2,(SP)      ; CORRECT STACK
2215 026632              ENDSRV
      026632              L10044:
      026632 000002              RTI
2216 026634              COL2:
2217 026634 032777 100000 153304      BIT      #ERR,#QINT      ; ERROR INTERRUPT?
2218 026642 001414              BEQ      70#            ; BRANCH IF NO ERROR INTERRUPT
2219
2220      ; ERROR CONDITION, ERROR INTERRUPT OCCURED
2221
2222      ;
      026644              ERRHRD 54...ERR11      ; TELL OPERATOR ERROR INTERRUPT OCCURED
      026644 104456              TRAP   C#ERHRD
      026646 000066              .WORD 54
      026650 000000              .WORD 0
      026652 010030              .WORD ERR11
2223 026654 004737 013116              CALL   ENDERR          ; GO PRINT OUT ENDING TERMINATOR
2224 026660 005077 153262              CLR    #QINT          ; CLEAR INTERRUPT REGISTER
2225 026664 012701 013630              MOV    #ABORT,R1      ; ABORT CODE TO R1
2226 026670 000137 026706              JMP    EXZ            ; BRANCH TO END OF TEST
2227
2228      ; END ERROR CONDITION
2229
2230      ;
2231 026674 052777 000001 153240      70#:  BIS    #CMDACK,#QCMD      ; TELL MASTER DAM DONE
2232
2233      ; END OF TEST
2234
2235 026702 012701 000000              MOV    #0,R1          ; END TEST NORMALLY
2236 026706              EXZ:  CLRVEC VECTOR      ; CLEAR VECTOR
      026706 013700 002444              MOV    VECTOR,R0
      026712 104436              TRAP  C#CVEC
2237 026714              SETVEC VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
      026714 012746 000340              MOV    #PRI07,-(SP)
      026720 012746 012652              MOV    #ERRINT,-(SP)
      026724 013746 002444              MOV    VECTOR,-(SP)
      026730 012746 000003              MOV    #3,-(SP)
      026734 104437              TRAP  C#SVEC
      026736 062706 000010              ADD    #10,SP
2238 026742 052777 000100 153176      BIS    #ERRIE,#QINT    ; ENABLE ERROR INTERRUPT
2239
2240 026750 000207              RTS    PC            ; RETURN TO MAINLINE
2241
2242      ;
2243      ; DATA PATTERN FOR TEST
2244
2245 026752 055132              DP5:  .WORD 055132
2246 026754 122645              .WORD 122645

```

TEST 19: DMA VAX TO PDP-11 (32K BOUNDARY)

```

2248 .SBTTL TEST 19: DMA VAX TO PDP-11 (32K BOUNDARY)
2249 ;:*****
2250 ;**
2251 ;TEST DESCRIPTION-
2252 ; WRITE 2 WORD BUFFER FROM VAX COMMAND BRIDGE TO PDP CONTROL BRIDGE
2253 ; USING PDP-11 ADDRESS VALUES 177776-20000
2254 ; CHECK DATA
2255 ;TEST STEPS-
2256 ;
2257 ;--
2258 ;:*****
2259
2260 026756          STATST
      026756          TEST19::
      026756 012737 000023 002114  MOV    #19,L#TEST      ;TEST NUMBER FOR ERROR MESSAGES
      026764 004737 013034          CALL   BMSG           ;CHECK FOR PNT FLAG
2261 026770 013701 002532          MOV    DEFAULT,R1     ; SECONDS OF TIME OUT
2262 026774 012702 026000 8#:    MOV    #TIMOUT,R2   ; TIME OUT VALUE TO R2
2263 027000          10#:   BREAK
      027000 104422          TRAP   C#BRK
2264 027002 032777 004000 153136  BIT    #CMDPRS,#QINT  ; COMMAND PRESENT?
2265 027010 001020          BNE    20#            ; IF COMMAND PRESENT, BRANCH
2266 027012 005302          DEC    R2                ; DECREMENT TIMEOUT REGISTER
2267 027014 001371          BNE    10#            ; IF NOT IN TIMEOUT, TRY AGAIN
2268 027016 005301          DEC    R1                ; DECREMENT SECOND TIMER
2269 027020 001365          BNE    8#              ; IF NOT TIMEOUT, AGAIN
2270 ;
2271 ; TIMEOUT ERROR CONDITION
2272 ;
2273 027022 012701 000000          MOV    #0,R1          ; REGISTER # TO R1
2274 027026          ERRHRD 55...ERR2    ; TELL OPERATOR TIMEOUT#
      027026 104456          TRAP   C#ERHRD
      027030 000067          .WORD  55
      027032 000000          .WORD  0
      027034 006506          .WORD  ERR2
2275 027036 004737 013116          CALL   ENDERR        ; GO PRINT OUT ENDING TERMINATOR
2276 027042 012701 013630          MOV    #ABORT,R1     ; ABORT CODE TO R1
2277 027046 000137 030026          JMP    EWM           ; ABORT TEST
2278 ;
2279 ; END TIMEOUT ERROR CONDITION
2280 ;
2281 ; COMMAND PRESENT WAS REALLY THERE
2282 ;
2283 027052          20#:   BIS    #QVREN,#QCTL    ; READ ENABLE
2284 027052 052777 000020 153070  MOV    #0,#QVAD      ; CLEAR VIEW REGISTER
2285 027060 012777 000000 153100  MOV    #QVDA,LOW16   ; READ LOW 16 BITS OF UNIBUS ADDRESS
2286 027066 017737 153076 002514  MOV    LOW16,#QUBA   ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
2287 027074 013777 002514 153052  MOV    #QVDA,HI2     ; READ HI 2 BITS OF UNIBUS ADDRESS
2288 027102 017737 153062 002516  BIT    #+CBIT4!BIT5,HI2 ; SEE IF ONLY BITS 4 AND 5 ARE SET
2289 027110 032737 177757 002516  BEQ    30#           ; BRANCH IF OK
2290 027116 001412
2291 ;
2292 ; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
2293 ;
2294 027120          ERRHRD 56...ERR10    ; TELL OPERATOR ERROR
      027120 104456          TRAP   C#ERHRD
      027122 000070          .WORD  56

```

TEST 19: DMA VAX TO PDP-11 (32K BOUNDARY)

```

027124 000000          .WORD 0
027126 007740          .WORD ERR10
2295 027130 004737 013116 CALL ENDERR          ; GO PRINT OUT ENDING TERMINATOR
2296 027134 012701 013630 MOV  #ABORT,R1        ; ABORT CODE TO R1
2297 027140 000137 030026 JMP  EWM              ; END TEST
2298
2299          ; END ERROR CONDITION
2300
2301 027144          30$:
2302 027144 053777 002516 153000 BIS  HI2,@QDMA        ; SET UP EXTENDED BITS IN EXP
2303 027152 017737 153012 002512 MOV  @QVDA,WORD.COUNT ; READ WORD FROM VIEW
2304
2305          ; MAKE SURE ENOUGH MEMORY IN
2306
2307 027160 012737 000000 002520 MOV  #NO,MAPPING      ; DEFAULT NO MAPPING
2308 027166 023727 002120 002000 CMP  L#HIMEM,#2000    ; CHECK FOR MEMORY SIZE
2309 027174 100420                      BMI  33$              ; IF NOT ENOUGH MEMORY, BRANCH
2310
2311          ; THERE IS MEMORY AT THE 32KW BOUNDRY, SET UP MEMORY MANAGEMENT
2312
2313 027176 012737 000001 002520 MOV  #YES,MAPPING     ; SET MAPPING FLAG
2314 027204 004737 012754                      CALL M#INIT          ; INITIALIZE MEMORY MANAGEMENT
2315 027210 012737 001777 172350 MOV  #1777,APR4       ; SET UP APR4 TO 177700
2316 027216 012702 100076                      MOV  #100076,R2      ; POINT R2 TO ADDRESS
2317 027222 012705 177776                      MOV  #177776,R5      ; PUT PHYSICAL ADDRESS IN R5
2318 027226 012737 000001 177572 MOV  #BIT0,M#RO       ; TURN ON MEMORY MANAGEMENT
2319 027234 000403 BR    34$              ; GO CLEAR BUFFER
2320
2321          ; CLEAR BUFFER
2322
2323 027236 013702 002502 33$: MOV  FREMEM,R2        ; STARTING ADDRESS OF BUFFER
2324 027242 010205                      MOV  R2,R5           ; SAVE STARTING PHYSICAL ADDRESS
2325 027244 013701 002512 34$: MOV  WORD.COUNT,R1     ; GET WORD.COUNT IN R1
2326 027250 005401                      NEG  R1              ; CONVERT WORD COUNT FROM 2'S COMP
2327 027252 005022 35$: CLR  (R2)+          ; CLEAR WORD IN BUFFER
2328 027254 005301                      DEC  R1              ; DECREMENT COUNTER
2329 027256 001375 BNE  35$              ; BRANCH BACK IF NOT DONE
2330
2331          ; BUFFER CLEARED, SET UP FOR DMA
2332
2333 027260 013777 002512 152672 MOV  WORD.COUNT,@QWC  ; MOVE WORD COUNT INTO QWC REGISTER
2334 027266 010577 152664 MOV  R5,@QBA         ; START ADDRESS OF BUFFER TO QBA
2335 027272 042777 037400 152652 BIC  #37400,@QDMA    ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
2336 027300 042777 000002 152644 BIC  @DMAWT,@QDMA    ; SET UP FOR VAX TO PDP DMA TRANSFER
2337 027306
2338 027306 013700 002444 CLRVEC VECTOR        ;
2338 027312 104436 TRAP C#CVEC          ;
2338 027314 SETVEC VECTOR,#COY1,#PRI07 ; SET UP NEW INTERRUPT VECTOR
2338 027314 012746 000340 MOV  #PRI07,-(SP)
2338 027320 012746 027440 MOV  #COY1,-(SP)
2338 027324 013746 002444 MOV  VECTOR,-(SP)
2338 027330 012746 000003 MOV  #3,-(SP)
2338 027334 104437 TRAP C#SVEC          ;
2338 027336 062706 000010 ADD  #10,SP
2339 027342 052777 000100 152576 BIS  #ERRIE,@QINT    ; ENABLE ERROR INTERRUPT
2340 027350 052777 000100 152574 BIS  #RDYIE,@QDMA    ; ENABLE RDY INTERRUPT
2341 027356 052777 000001 152566 BIS  #GO,@QDMA       ; INITIATE DMA TRANSFER

```



TEST 19: DMA VAX TO PDP-11 (32K BOUNDARY)

```

2342
2343 ; WAIT FOR INTERRUPT
2344 ;
2345 027364 013701 002532      MOV     DEFAULT,R1      ; NUMBER OF SECONDS OF TIMEOUT
2346 027370 012702 026000    40$:   MOV     @TIMOUT,R2  ; TIME OUT DELAY TO R2
2347 027374      104422    45$:   BREAK      ; ALLOW CONTROL C
      027374      005302    TRAP     C#BRK
2348 027376 005302      DEC     R2              ; DECREMENT TIMEOUT
2349 027400 001375      BNE     45$            ; IF NOT DONE LOOP
2350 027402 005301      DEC     R1              ; DECREMENT SECOND COUNTER
2351 027404 001371      BNE     40$            ; IF NOT DONE, LOOP
2352 ;
2353 ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
2354 ;
2355 027406      104456    ERRHRD   57...ERR7      ; TELL OPERATOR TIMEOUT
      027406      000071    TRAP     C#ERHRD
      027410      000000    .WORD   57
      027412      000000    .WORD   0
      027414      007424    .WORD   ERR7
2356 027416 004737 013116    CALL    ENDERR          ; GO PRINT OUT ENDING TERMINATOR
2357 027422 042777 000100 152522  BIC     @RDYIE,@QDMA    ; CLEAR INTERRUPT BIT
2358 027430 012701 013630    MOV     @ABORT,R1      ; ABORT CODE TO R1
2359 027434 000137 030026    JMP     EWW            ; BRANCH TO END OF TEST
2360 ;
2361 ; END ERROR CONDITION
2362 ;
2363 ; DMA INTERRUPT SHOULD GO HERE
2364 ;
2365 027440      000100 152504    COY1:: BGNSRV   COY1
      027440      012716 027454    BIC     @RDYIE,@QDMA    ; CLEAR INTERRUPT BIT
2366 027440 042777 000100 152504    MOV     @COY2,(SP)     ; CORRECT STACK
2367 027446 012716 027454    ENDSRV
2368 027452      000002    L10045: RTI
      027452      100000 152464    COY2:   BIT     @ERR,@QINT  ; ERROR INTERRUPT?
2369 027454      001414      BEQ     50$            ; BRANCH IF NO ERROR INTERRUPT
2370 027454 032777 100000 152464
2371 027462 001414
2372 ;
2373 ; ERROR CONDITION, ERROR INTERRUPT OCCURED
2374 ;
2375 027464      104456    ERRHRD   58...ERR11    ; TELL OPERATOR ERROR INTERRUPT OCCURED
      027464      000072    TRAP     C#ERHRD
      027466      000000    .WORD   58
      027470      000000    .WORD   0
      027472      010030    .WORD   ERR11
2376 027474 004737 013116    CALL    ENDERR          ; GO PRINT OUT ENDING TERMINATOR
2377 027500 005077 152442    CLR     @QINT          ; CLEAR INTERRUPT REGISTER
2378 027504 012701 013630    MOV     @ABORT,R1      ; ABORT CODE TO R1
2379 027510 000137 030026    JMP     EWW            ; BRANCH TO END OF TEST
2380 ;
2381 ; END ERROR CONDITION
2382 ;
2383 027514      000001 152420    50$:   BIS     @CMDACK,@QCMD  ; TELL MASTER DAM DONE
2384 027514 052777 000001 152420    CLRVEC VECTOR          ; CLEAR VECTOR
2385 027522      013700 002444    MOV     VECTOR,R0

```

TEST 19: DMA VAX TO PDP-11 (32K BOUNDARY)

```

2386 027526 104436 TRAP C#CVEC
027530 SETVEC VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
027530 012746 000340 MOV #PRI07,-(SP)
027534 012746 012652 MOV #ERRINT,-(SP)
027540 013746 002444 MOV VECTOR,-(SP)
027544 012746 000003 MOV #3,-(SP)
027550 104437 TRAP C#SVEC
027552 062706 000010 ADD #10,SP
2387 027556 052777 000100 152362 BIS #ERRIE,@QINT ; ENABLE ERROR INTERRUPT
2388 ;
2389 ; NOW CHECK DATA
2390 ;
2391 027564 013701 002502 MOV FREMEM,R1 ; R1 TO START OF BUFFER
2392 027570 022737 000000 002520 CMP #NO,MAPPING ; WAS MAPPING USED?
2393 027576 001402 BEQ 60$ ; BRANCH IF NO MAPPING
2394 ;
2395 ; MAPPING WAS USED
2396 ;
2397 027600 012701 100076 MOV #100076,R1
2398 027604 013702 002512 60$: MOV WORD.COUNT,R2 ; R2 GETS WORD COUNT
2399 027610 005402 NEG R2 ; CONVERT WORD COUNT FROM 2'S COMP.
2400 ;
2401 ; NOW CHECK BUFFER AGAINST PATTERN
2402 ;
2403 027612 005037 002462 CLR ERROR1 ; CLEAR ERROR COUNTER
2404 027616 012704 000002 100$: MOV #2,R4 ; # OF WORDS IN PATTERN
2405 027622 012703 030112 MOV #DP6,R3 ; R3 GETS PATTERN ADDRESS
2406 027626 012137 002454 110$: MOV (R1)+,VAL ; GET WORD FROM BUFFER
2407 027632 012337 002456 MOV (R3)+,EXP ; GET EXPECTED PATTERN IN EXP
2408 027636 023737 002454 002456 CMP VAL,EXP ; COMPARE VALUE WITH EXPECTED
2409 027644 001442 BEQ 130$ ; IF OK, BRANCH
2410 ;
2411 ; ERROR CONDITION, NOT WHAT EXPECTED IN BUFFER
2412 ;
2413 027646 005737 002462 TST ERROR1 ; ANY OTHER ERRORS OCCURED
2414 027652 001004 BNE 120$ ; SKIP HEADER ERROR IF ALREADY PRINTED
2415 027654 ERRHRD 59,,ERR8 ; PRINT ERROR HARD MESSAGE
027654 104456 TRAP C#ERRHRD
027656 000073 .WORD 59
027660 000000 .WORD 0
027662 007530 .WORD ERR8
2416 027664 005237 002462 120$: INC ERROR1 ; INCREMENT ERROR COUNTER
2417 027670 022737 000012 002462 CMP #10.,ERROR1 ; SEE IF TOO MANY MESSAGES PRINTED
2418 027676 103425 BCS 130$ ; IF TOO MANY, BRANCH
2419 027700 013737 002512 002474 MOV WORD.COUNT,COUNTER ; CALCULATE WORD NUMBER
2420 027706 005437 002474 NEG COUNTER ; CONVERT WORD COUNT FROM 2'S COMP
2421 027712 160237 002474 SUB R2,COUNTER
2422 027716 PRINTX #FOR10,COUNTER,VAL,EXP ; PRINT DETAIL LINE
027716 013746 002456 MOV EXP,-(SP)
027722 013746 002454 MOV VAL,-(SP)
027726 013746 002474 MOV COUNTER,-(SP)
027732 012746 004230 MOV #FOR10,-(SP)
027736 012746 000004 MOV #4,-(SP)
027742 010600 MOV SP,R0
027744 104415 TRAP C#PNTX
2423 027746 062706 000012 ADD #12,SP

```

TEST 19: DMA VAX TO PDP-11 (32K BOUNDARY)

```

2424          ; END ERROR CONDITION
2425          ;
2426 027752    ;
2427 027752 005302 130$: DEC R2          ; DECREMENT WORD COUNT
2428 027754 001403    BEQ 140$      ; BRANCH OUT IF DONE
2429 027756 005304    DEC R4          ; DECREMENT PATTERN WORD COUNT
2430 027760 001322    BNE 110$      ; BRANCH IF NOT DONE WITH PATTERN
2431 027762 000715    BR 100$       ; BRANCH IF DONE WITH PATTERN
2432          ;
2433          ; BUFFER CHECKED, PRINT TOTAL ERROR COUNT IF NECESSARY
2434          ;
2435 027764    ;
2436 027764 005737 002462 140$: TST ERROR1      ; ANY ERRORS
2437 027770 001414    BEQ 150$      ; IF NO ERRORS, BRANCH
2438 027772    PRINTB #FOR11,ERROR1 ; TELL TOTAL ERROR COUNT
      027772 013746 002462    MOV ERROR1,-(SP)
      027776 012746 004253    MOV #FOR11,-(SP)
      030002 012746 000002    MOV #2,-(SP)
      030006 010600          MOV SP,R0
      030010 104414          TRAP C:PNTB
      030012 062706 000006    ADD #6,SP
2439 030016 004737 013116    CALL ENDERR      ; GO PRINT OUT ENDING TERMINATOR
2440          ;
2441          ; END OF TEST
2442          ;
2443 030022 012701 000000 150$: MOV #0,R1      ; END TEST NORMALLY
2444 030026    EWW: CLRVEC VECTOR      ; CLEAR VECTOR
      030026 013700 002444    MOV VECTOR,R0
      030032 104436          TRAP C:CVEC
2445 030034    SETVEC VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
      030034 012746 000340    MOV #PRI07,-(SP)
      030040 012746 012652    MOV #ERRINT,-(SP)
      030044 013746 002444    MOV VECTOR,-(SP)
      030050 012746 000003    MOV #3,-(SP)
      030054 104437          TRAP C:SVEC
      030056 062706 000010    ADD #10,SP
2446 030062 052777 000100 152056 BIS #ERRIE,@QINT ; ENABLE ERROR INTERRUPT
2447 030070 022737 000000 002520 CMP #NO,MAPPING ; WAS MAPPING USED?
2448 030076 001404          BEQ 160$      ; BRANCH IF NO MAPPING
2449          ;
2450          ; MAPPING WAS USED, RESTORE MAPPING REGISTERS AND TURN OFF MAPPING
2451          ;
2452 030100 004737 012754    JSR PC,MINIT      ; RESTORE MM TO NORMAL
2453 030104 005037 177572    CLR MMRO         ; DISABLE MEMORY MANAGEMENT
2454 030110          ;
2455 030110 000207 160$: RTS PC          ; RETURN TO MAINLINE
2456          ;
2457          ; DATA PATTERN FOR TEST
2458          ;
2459          ;
2460 030112 055132    DP6: .WORD 055132
2461 030114 122645    .WORD 122645

```

TEST 20: WATCHDOG TIMER

```

2463 .SBTTL TEST 20: WATCHDOG TIMER
2464 ;:.....
2465 ;..
2466 ;TEST DESCRIPTION-
2467 ; ENABLE WATCHDOG TIMER AND CHECKS FOR INTERRUPT
2468 ; REPEATS FOR ALL TIMER INTERRUPTS
2469 ;
2470 ;TEST STEPS-
2471 ; U QMD QIO WITH INTERRUPT ENABLED; WAIT WITH TIME-OUT
2472 ; Q SET UP INTERVAL AND ENABLE TIMER
2473 ; U REPEAT FOR ALL TIMER INTERVALS
2474 ;--
2475 ;:.....
2476 030116 STATST
030116 TEST20::
030116 012737 000024 002114 MOV #20,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
030124 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
2477 ;
2478 ; SET UP STARTING VALUES
2479 ;
2480 030130 012705 030412 MOV #WDTBUF,R5 ; POINT TO FIRST TIME PATTERN
2481 030134 005037 002524 CLR LOOPS ; INITIALIZE LOOP COUNTER
2482 ;
2483 ; SET UP AND ENABLE WATCHDOG TIMER
2484 ;
2485 030140 005237 002524 100: INC LOOPS ; INCREMENT LOOP COUNTER
2486 030144 012577 152000 MOV (R5),@QCTL ; SET UP TIMER CODE
2487 030150 052777 000002 151772 BIS #TMRENB,@QCTL ; HIT TIMER ENABLE
2488 ;
2489 ; NOW WAIT FOR EVENT ACK
2490 ;
2491 030156 012701 000005 MOV #5.,R1 ; NUMBER OF SECONDS TO TIME OUT ! VB1 DFP
2492 030162 012702 026000 300: MOV #TIMOUT,R2 ; TIME OUT VALUE TO R2
2493 030166 030166 104422 400: BREAK ; ALLOW CONTROL C
TRAP C#BRK
2494 030170 032777 002000 151750 BIT #VNTACK,@QINT ; EVENT PRESENT?
2495 030176 001033 BNE 500 ; IF EVENT PRESENT, BRANCH
2496 030200 005302 DEC R2 ; DECREMENT TIMEOUT REGISTER
2497 030202 001371 BNE 400 ; IF NOT IN TIMEOUT, TRY AGAIN
2498 030204 005301 DEC R1 ; DECREMENT SECOND TIMER
2499 030206 001365 BNE 300 ; IF NOT TIMEOUT, AGAIN
2500 ;
2501 ; TIMEOUT ERROR CONDITION
2502 ;
2503 030210 012701 000001 MOV #1,R1 ; REGISTER NUMBER TO R1
2504 030214 030214 104456 ERPHRD 60...ERR2 ; TELL OPERATOR TIMEOUT
TRAP C#ERHRD
030216 000074 .WORD 60
030220 000000 .WORD 0
030222 006506 .WORD ERR2
2505 030224 030224 013746 002524 PRINTX #FOR23,LOOPS ; TELL OPERATOR LOOP #
MOV LOOPS,-(SP)
030230 012746 005314 MOV #FOR23,-(SP)
030234 012746 000002 MOV #2,-(SP)
030240 010600 MOV SP,R0
030242 104415 TRAP C#PNTX
030244 062706 0C0006 ADD #6,SP

```

## TEST 20: WATCHDOG TIMER

```

2506 030250 004737 013116          CALL    ENDERR          ; GO PRINT OUT ENDING TERMINATOR
2507 030254 012701 013630          MOV     @ABORT,R1      ; ABORT CODE TO R1
2508 030260 005077 151664          CLR    @QCTL          ; CLEAR THE CONTROL REGISTER
2509 030264 000451                   BR     100#           ; ABORT TEST
2510
2511          ;
2512          ; END TIMEOUT ERROR CONDITION
2513          ;
2514          ; EVENT ACK RECEIVED, WAIT FOR VNTACK CLEAR
2515
2515 030266          50#:
2516 030266 013701 002532          MOV     DEFAULT,R1    ; NUMBER OF SECONDS TO TIME OUT
2517 030272 012702 026000          60#: MOV     @TIMOUT,R2 ; TIME OUT VALUE TO R2
2518 030276          70#: BREAK   ; ALLOW CONTROL C
      030276 104422          TRAP   C#BRK
2519 030300 032777 002000 151640    BIT    @VNTACK,@QINT ; EVENT PRESENT CLEAR?
2520 030306 001431                   BEQ    80#           ; IF EVENT PRESENT, BRANCH
2521 030310 005302                   DEC    R2            ; DECREMENT TIMEOUT REGISTER
2522 030312 001371                   BNE   70#           ; IF NOT IN TIMEOUT, TRY AGAIN
2523 030314 005301                   DEC    R1            ; DECREMENT SECOND TIMER
2524 030316 001365                   BNE   60#           ; IF NOT TIMEOUT, AGAIN
2525
2526          ;
2527          ; TIMEOUT ERROR CONDITION
2528 030320          ;
      030320 104456          ERRHRD 61...ERR16      ; TELL OPERATOR TIMEOUT
      030322 000075          TRAP   C#ERRHRD
      030324 000000          .WORD 61
      030326 010552          .WORD 0
2529 030330          PRINTX @FOR23,COUNTER ; TELL OPERATOR LOOP #
      030330 013746 002474          MOV    COUNTER,-(SP)
      030334 012746 005314          MOV    @FOR23,-(SP)
      030340 012746 000002          MOV    @2,-(SP)
      030344 010600          MOV    SP,R0
      030346 104415          TRAP   C#PNTX
      030350 062706 000006          ACD   @6,SP
2530 030354 004737 013116          CALL    ENDERR          ; GO PRINT OUT ENDING TERMINATOR
2531 030360 012701 013630          MOV     @ABORT,R1      ; ABORT CODE TO R1
2532 030364 005077 151560          CLR    @QCTL          ; CLEAR THE CONTROL REGISTER
2533 030370 000407                   BR     100#           ; ABORT TEST
2534
2535          ;
2536          ; END TIMEOUT ERROR CONDITION, SET UP NEXT TIME OR EXIT
2537
2537 030372          80#:
2538 030372 005077 151552          CLR    @QCTL          ; CLEAR THE CONTROL REGISTER
2539 030376 022737 000004 002524    CMP    @4,LOOPS       ; IS LOOP COUNTER AT MAXIMUM?
2540 030404 001255                   BNE   10#           ; BRANCH BACK IF NOT DONE
2541 030406 005000                   CLR    R0            ; END OK CODE
2542 030410          100#:
2543 030410 000207                   RETURN
2544
2545          ;
2546          ; WATCH DOG TIMER TIME PATTERNS
2547
2547 030412 000014          WDTBUF: TIME4
2548 030414 000010          TIME2
2549 030416 000004          TIME1
2550 030420 000000          TIME.5

```

TEST 21: DMA THROTTLE

```

2552 .SBTTL TEST 21: DMA THROTTLE
2553 ;*****
2554 ;**
2555 ;TEST DESCRIPTION-
2556 ; SETS DMA THROTTLE
2557 ; DMA'S TO PDP-11
2558 ; CHECKS DATA
2559 ;
2560 ;TEST STEPS-
2561 ; U CTL QIO TO SET THROTTLE
2562 ; Q WAIT FOR INTERRUPT
2563 ; U WRITE QIO WITH 256 WORDS
2564 ; Q CHECK DATA
2565 ; REPEAT FOR ALL THROTTLE VALUES
2566 ;--
2567 ;*****
2568 030422 STATST
030422 TEST21::
030422 012737 000025 002114 MOV #21,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
030430 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
2569 030434 005037 002524 CLR LOOPS ; # OF PASSES COUNTER
2570 030440 TST19:
2571 030440 005237 002524 INC LOOPS ; INCREMENT PASS COUNTER
2572 030444 013701 002532 MOV DEFAULT,R1 ; SECONDS OF TIME OUT
2573 030450 012702 026000 8#: MOV #TIMOUT,R2 ; TIME OUT VALUE TO R2
2574 030454 104422 104422 104422 104422 104422 104422 104422 104422 104422
030454 032777 004000 151462 TRAP C#BRK
2575 030456 032777 004000 151462 BIT #CMDPRS,#QINT ; COMMAND PRESENT?
2576 030464 001032 BNE 20# ; IF COMMAND PRESENT, BRANCH
2577 030466 005302 DEC R2 ; DECREMENT TIMEOUT REGISTER
2578 030470 001371 BNE 10# ; IF NOT IN TIMEOUT, TRY AGAIN
2579 030472 005301 DEC R1 ; DECREMENT SECOND TIMER
2580 030474 001365 BNE 8# ; IF NOT TIMEOUT, AGAIN
2581 ;
2582 ; TIMEOUT ERROR CONDITION
2583 ;
2584 030476 012701 000000 MOV #0,R1 ; REGISTER # TO R1
2585 030502 ERRHRD 62...ERR2 ; TELL OPERATOR TIMEOUT
030502 104456 TRAP C#ERRHD
030504 000076 .WORD 62
030506 000000 .WORD 0
030510 006506 .WORD ERR2
2586 030512 PRINTX #FOR24,LOOPS ; TELL OPERATOR WHICH LOOP
030512 013746 002524 MOV LOOPS,-(SP)
030516 012746 005343 MOV #FOR24,-(SP)
030522 012746 000002 MOV #2,-(SP)
030526 010600 MOV SP,R0
030530 104415 TRAP C#PNTX
030532 062706 000006 ADD #6,SP
2587 030536 004737 013116 CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
2588 030542 012701 013630 MOV #ABORT,R1 ; ABORT CODE TO R1
2589 030546 000137 031436 JMP E19 ; ABORT TEST
2590 ;
2591 ; END TIMEOUT ERROR CONDITION
2592 ;
2593 ; COMMAND PRESENT WAS REALLY THERE
2594 ;

```

## TEST 21: DMA THROTTLE

```

2595 030552
2596 030552 052777 000020 151370
2597 030560 012777 000000 151400
2598 030566 017737 151376 002514
2599 030574 013777 002514 151352
2600 030602 017737 151362 002516
2601 030610 032737 177757 002516
2602 030616 001412
2603
2604
2605
2606 030620
      030620 104456
      030622 000077
      030624 000000
      030626 007740
2607 030630 004737 013116
2608 030634 012701 013630
2609 030640 000137 031436
2610
2611
2612
2613 030644
2614 030644 053777 002516 151300
2615 030652 017737 151312 002512
2616
2617
2618
2619 030660 013701 002512
2620 030664 005401
2621 030666 013702 002502
2622 030672 005022
2623 030674 005301
2624 030676 001375
2625
2626
2627
2628 030700 013777 002512 151252
2629 030706 013777 002502 151242
2630 030714 042777 037400 151230
2631 030722 042777 000002 151222
2632 030730
      030730 013700 002444
      030734 104436
2633 030736
      030736 012746 000340
      030742 012746 031062
      030746 013746 002444
      030752 012746 000003
      030756 104437
      030760 062706 000010
2634 030764 052777 000100 151154
2635 030772 052777 000100 151152
2636 031000 052777 000001 151144
2637
2638
2639

```

```

204:
      BIS      #QVREN,@QCTL      ; READ ENABLE
      MOV      #0,@QVAD          ; CLEAR VIEW REGISTER
      MOV      @QVDA,LOW16       ; READ LOW 16 BITS OF UNIBUS ADDRESS
      MOV      LOW16,@QUBA       ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
      MOV      @QVDA,HI2         ; READ HI 2 BITS OF UNIBUS ADDRESS
      BIT      #'CBIT4!BITS,HI2  ; SEE IF ONLY BITS 4 AND 5 ARE SET
      BEQ      304               ; BRANCH IF OK
;
; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
;
      ERRHRD   63,,,ERR10        ; TELL OPERATOR ERROR
      TRAP    C#ERRHRD
      .WORD   63
      .WORD   0
      .WORD   ERR10
      CALL    ENDERR              ; GO PRINT OUT ENDING TERMINATOR
      MOV     #ABORT,R1           ; ABORT CODE TO R1
      JMP     E19                 ; END TEST
;
; END ERROR CONDITION
;
304:
      BIS      HI2,@QDMA          ; SET UP EXTENDED BITS IN EXP
      MOV      @QVDA,WORD.COUNT  ; READ WORD COUNT FROM VIEW
;
; CLEAR BUFFER
;
      MOV      WORD.COUNT,R1     ; GET WORD.COUNT IN R1
      NEG     R1                  ; CHANGE FROM 2'S COMP
      MOV      FREMEM,R2         ; STARTING ADDRESS OF BUFFER
354:
      CLR     (R2)+              ; CLEAR BYTE IN BUFFER
      DEC     R1                  ; DECREMENT COUNTER
      BNE     354                ; BRANCH BACK IF NOT DONE
;
; BUFFER CLEARED, SET UP FOR DMA
;
      MOV      WORD.COUNT,@QMC   ; MOVE WORD COUNT INTO QMC REGISTER
      MOV      FREMEM,@QBA      ; START ADDRESS OF BUFFER TO QBA
      BIC     #37400,@QDMA      ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
      BIC     #DMAWT,@QDMA      ; SET UP FOR VAX TO PDP DMA TRANSFER
      CLRVEC  VECTOR
      MOV      VECTOR,R0
      TRAP    C#CVEC
      SETVEC  VECTOR,#C19.1,#PRI07 ; SET UP NEW INTERRUPT VECTOR
      MOV     #PRI07,-(SP)
      MOV     #C19.1,-(SP)
      MOV     VECTOR,-(SP)
      MOV     #3,-(SP)
      TRAP    C#SVEC
      ADD     #10,SP
      BIS     #ERRIE,@QINT      ; ENABLE ERROR INTERRUPT
      BIS     #RDYIE,@QDMA     ; ENABLE RDY INTERRUPT
      BIS     #GO,@QDMA        ; INITIATE DMA TRANSFER
;
; WAIT FOR INTERRUPT
;

```

## TEST 21: DMA THROTTLE

```

2640 031006 013701 002532      MOV      DEFAULT,R1      ; NUMBER OF SECONDS OF TIMEOUT
2641 031012 012702 026000      40$:    MOV      @TIMOUT,R2  ; TIME OUT DELAY TO R2
2642 031016      104422      45$:    BREAK     ; ALLOW CONTROL C
      031016 104422      TRAP     C#BRK
2643 031020 005302      DEC      R2              ; DECREMENT TIMEOUT
2644 031022 001375      BNE     45$             ; IF NOT DONE LOOP
2645 031024 005301      DEC      R1              ; DECREMENT SECOND COUNTER
2646 031026 001371      BNE     40$             ; IF NOT DONE, LOOP
2647
2648      ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
2649      ;
2650 031030      ERRHRD  64...ERR7      ; TELL OPERATOR TIMEOUT
      031030 104456      TRAP     C#ERHRD
      031032 000100      .WORD   64
      031034 000000      .WORD   0
      031036 007424      .WORD   ERR7
2651 031040 004737 013116      CALL    ENDERR           ; GO PRINT OUT ENDING TERMINATOR
2652 031044 042777 000100 151100      BIC     @RDYIE,@QDMA     ; CLEAR INTERRUPT BIT
2653 031052 012701 013630      MOV     @ABORT,R1       ; ABORT CODE TO R1
2654 031056 000137 031436      JMP     E19             ; BRANCH TO END OF TEST
2655
2656      ; END ERROR CONDITION
2657      ;
2658      ; DMA INTERRUPT SHOULD GO HERE
2659      ;
2660 031062      BGNSRV  C19.1
      031062      C19.1::
2661 031062 042777 000100 151062      BIC     @RDYIE,@QDMA     ; CLEAR INTERRUPT BIT
2662 031070 012716 031076      MOV     @C19.2,(SP)     ; CORRECT STACK
2663 031074      ENDSRV
      031074      L10046:
      031074 000002      RTI
2664 031076      C19.2:
2665 031076 032777 100000 151042      BIT     @ERR,@QINT      ; ERROR INTERRUPT?
2666 031104 001412      BEQ     50$             ; BRANCH IF NO ERROR INTERRUPT
2667
2668      ; ERROR CONDITION, ERROR INTERRUPT OCCURED
2669      ;
2670 031106      ERRHRD  65...ERR11     ; TELL OPERATOR ERROR INTERRUPT OCCURED
      031106 104456      TRAP     C#ERHRD
      031110 000101      .WORD   65
      031112 000000      .WORD   0
      031114 010030      .WORD   ERR11
2671 031116 004737 013116      CALL    ENDERR           ; GO PRINT OUT ENDING TERMINATOR
2672 031122 012701 013630      MOV     @ABORT,R1       ; ABORT CODE TO R1
2673 031126 000137 031436      JMP     E19             ; BRANCH TO END OF TEST
2674
2675      ; END ERROR CONDITION
2676      ;
2677      ;
2678 031132      50$:
2679 031132 052777 000001 151002      BIS     @CMDACK,@QCMD    ; TELL MASTER DAM DONE
      031140 013700 002444      CLRVEC  VECTOR         ; CLEAR VECTOR
      031144 104436      MOV     VECTOR,R0
2680 031146      TRAP     C#VEC
      031146 012746 000340      SETVEC  VECTOR,@ERRINT,@PRI07 ; SET UP INTERRUPT VECTOR
      031152 012746 012652      MOV     @PRI07,-(SP)
      MOV     @ERRINT,-(SP)

```



## TEST 21: DMA THROTTLE

```

031156 013746 002444      MOV     VECTOR,-(SP)
031162 012746 000003      MOV     #3,-(SP)
031166 104437              TRAP   C#SVEC
031170 062706 000010      ADD     #10,SP
2681 031174 052777 000100 150744      BIS     #ERRIE,@QINT      ; ENABLE ERROR INTERRUPT
2682
2683      ; NOW CHECK DATA
2684
2685 031202 013701 002502      MOV     FREMEM,R1      ; R1 TO START OF BUFFER
2686 031206 013702 002512      MOV     WORD.COUNT,R2  ; R2 GETS WORD COUNT
2687 031212 005402              NEG     R2              ; CHANGE WORD COUNT FROM 2'S COMP
2688
2689      ; NOW CHECK BUFFER AGAINST PATTERN
2690
2691 031214 005037 002462      CLR     ERROR1         ; CLEAR ERROR COUNTER
2692 031220 005037 002474      CLR     COUNTER        ; CLEAR COUNTER
2693 031224 012704 000002      100#:  MOV     #2,R4       ; # OF WORDS IN PATTERN
2694 031230 012703 031502      MOV     #DP19,R3       ; R3 GETS PATTERN ADDRESS
2695 031234 012137 002454      110#:  MOV     (R1)+,VAL     ; GET WORD FROM BUFFER
2696 031240 012337 002456      MOV     (R3)+,EXP     ; GET EXPECTED PATTERN IN EXP
2697 031244 005237 002474      INC     COUNTER        ; INCREMENT ADDRESS COUNTER
2698 031250 023737 002454 002456      CMP     VAL,EXP        ; COMPARE VALUE WITH EXPECTED
2699 031256 001433              BEQ     130#           ; IF OK, BRANCH
2700
2701      ; ERROR CONDITION, NOT WHAT EXPECTED IN BUFFER
2702
2703 031260 005737 002462      TST     ERROR1         ; ANY OTHER ERRORS OCCURED
2704 031264 001004              BNE     120#           ; SKIP HEADER ERROR IF ALREADY PRINTED
2705 031266              ERRHRD 66,,ERR14      ; PRINT ERROR HARD MESSAGE
031266 104456              TRAP   C#ERRHD
031270 000102              .WORD 66
031272 000000              .WORD 0
031274 010336              .WORD ERR14
2706 031276 005237 002462      120#:  INC     ERROR1         ; INCREMENT ERROR COUNTER
2707 031302 022737 000012 002462      CMP     #10,,ERROR1   ; SEE IF TOO MANY MESSAGES PRINTED
2708 031310 103416              BCS     130#           ; IF TOO MANY, BRANCH
2709 031312              PRINTX #FOR10,COUNTER,VAL,EXP ; PRINT DETAIL LINE
031312 013746 002456      MOV     EXP,-(SP)
031316 013746 002454      MOV     VAL,-(SP)
031322 013746 002474      MOV     COUNTER,-(SP)
031326 012746 004230      MOV     #FOR10,-(SP)
031332 012746 000004      MOV     #4,-(SP)
031336 010600              MOV     SP,R0
031340 104415              TRAP   C#PNTX
031342 062706 000012      ADD     #12,SP
2710
2711      ; END ERROR CONDITION
2712
2713 031346              130#:
2714 031346 005302      DEC     R2              ; DECREMENT WORD COUNT
2715 031350 001403      BEQ     140#           ; BRANCH OUT IF DONE
2716 031352 005304      DEC     R4              ; DECREMENT PATTERN WORD COUNT
2717 031354 001327      BNE     110#           ; BRANCH IF NOT DONE WITH PATTERN
2718 031356 000722      BR      100#           ; BRANCH IF DONE WITH PATTERN
2719
2720      ; BUFFER CHECKED, PRINT TOTAL ERROR COUNT IF NECESSARY
2721

```

TEST 21: DMA THROTTLE

```

2722 031360
2723 031360 005737 002462
2724 031364 001414
2725 031366
      031366 013746 002462
      031372 012746 004253
      031376 012746 000002
      031402 010600
      031404 104414
      031406 062706 000006
2726 031412 004737 013116
2727
2728
2729
2730 031416 022737 000004 002524
2731 031424 001402
2732 031426 000137 030440
2733 031432 012701 000000
2734 031436
      031436 013700 002444
      031442 104436
2735 031444
      031444 012746 000340
      031450 012746 012652
      031454 013746 002444
      031460 012746 000003
      031464 104437
      031466 062706 000010
2736 031472 052777 000100 150446
2737
2738 031500 000207
2739
2740
2741
2742 031502 000000
2743 031504 177777

      140$: TST ERROR1 ; ANY ERRORS
           BEQ 150$ ; IF NO ERRORS, BRANCH
           PRINTB #FOR11,ERROR1 ; TELL TOTAL ERROR COUNT
           MOV ERROR1,-(SP)
           MOV #FOR11,-(SP)
           MOV #2,-(SP)
           MOV SP,R0
           TRAP C#PNTB
           ADD #6,SP
           CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR

; END OF TEST
      150$: CMP #4,LOOPS ; IS IT DONE
           BEQ 160$ ; IF DONE, BRANCH
           JMP TST19 ; IF NOT DONE, AGAIN
      160$: MOV #0,R1 ; END NORMALLY CODE
E19: CLRVEC VECTOR ; CLEAR VECTOR
      MOV VECTOR,R0
      TRAP C#CVEC
      SETVEC VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
      MOV #PRI07,-(SP)
      MOV #ERRINT,-(SP)
      MOV VECTOR,-(SP)
      MOV #3,-(SP)
      TRAP C#SVEC
      ADD #10,SP
      BIS #ERRIE,#QINT ; ENABLE ERROR INTERRUPT

      RTS PC ; RETURN TO MAINLINE

; DATA PATTERN FOR TEST
      DP19: .WORD 0
           .WORD -1
    
```

TEST 22: CONVERSATION MODE

2745  
2746  
2747  
2748  
2749  
2750  
2751  
2752  
2753  
2754  
2755  
2756  
2757  
2758  
2759  
2760  
2761  
2762  
2763  
2764  
2765  
2766  
2767  
2768  
2769  
2770  
2771  
2772  
2773  
2774  
2775  
2776  
2777  
2778  
2779  
2780  
2781  
2782  
2783  
2784  
2785  
2786  
2787 031506  
2788 031506  
031506  
031506 012737 000026 002114  
031514 004737 013034  
2789 031520  
031520 012746 000340  
031524 012746 012644  
031530 013746 002444  
031534 012746 000003  
031540 104437  
031542 062706 000010  
2790 031546 005037 002450  
2791 031552 052777 000020 150370  
2792 031560 012777 000000 150400

```
.SBTTL TEST 22: CONVERSATION MODE
;*****
;
;TEST DESCRIPTION-
; GETS WORD COUNT, DIRECTION, AND DATA PATTERN FROM OPERATOR
; DOES DMA
;
;TEST STEPS-
; U PUT DIRECTION, AND DATA PATTERN CODE AND WORD COUNT IN VIEW RAM
; U COMMAND QIO
; Q READ VIEW RAM
; Q SET CMDACK
; U PUT LOW 16 BIT UNIBUS ADDRESS IN VIEW RAM ADDRESS 0
; U PUT EXTENDED UNIBUS ADDRESS BITS IN VIEW RAM BITS 4 & 5 ADDRESS 1
; U PUB WORD COUNT IN VIEW RAM ADDRESS 2
; U IF WRITE-
; LOAD BUFFER
; WRITE QIO
; IF READ-
; CLEAR BUFFER
; READ QIO
; Q IF WRITE (FROM VAX)-
; CLEAR BUFFER
; WAIT FOR CMDPRS INTERRUPT
; READ VIEW FOR ADDRESS
; LOAD ADDRESS FOR DMA
; INITIATE DMA
; SET CMDPRS
; CHECK FOR CORRECT PATTERN
; IF READ (FROM VAX)-
; LOAD BUFFER
; WAIT FOR CMDPRS INTERRUPT
; READ VIEW FOR ADDRESS
; LOAD ADDRESS FOR DMA
; INITIATE DMA
; SET CMDPRS
; SET VNTPRS
; CHECK FOR VNTACK
;
; FOR DIRECTION, YES = DMA VAX TO LSI
;--
;*****
CONVERSATION:
STATST
TEST22::
MOV #22,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
CALL BMSG ;CHECK FOR PNT FLAG
SETVEC VECTOR,#TRAP4,#PRI07 ; SET UP FOR INTERRUPT
MOV #PRI07,-(SP)
MOV #TRAP4,-(SP)
MOV VECTOR,-(SP)
MOV #3,-(SP)
TRAP C#SVEC
ADD #10,SP
CLR TRPFLG ; CLEAR INTERREPT FLAG
BIS #QVREN,#QCTL ; READ ENABLE
MOV #0,#QVAD ; ZERO ADDRESS REGISTER
```

TEST 22: CONVERSATION MODE

```

2793 031566 017737 150376 002506      MOV      @QVDA,DIR          ; GET DIRECTION
2794 031574 017737 150370 002510      MOV      @QVDA,PATTERN     ; GET PATTERN CODE
2795 031602 017737 150362 002512      MOV      @QVDA,WORD.COUNT  ; GET WORD COUNT
2796 031610 052777 000001 150324      BIS      @CMDACK,@QCMD     ; TELL MASTER DONE READING VIEW
2797 031616 052777 000100 150316      BIS      @PRISIE,@QCMD    ; ENABLE INTERRUPT
2798
2799      ; CHECK FOR AVAILABLE MEMORY SIZE AGAINST WORD COUNT
2800
2801 031624 012737 000000 002520      MOV      @NO,MAPPING       ; DEFAULT TO NO MAPPING
2802 031632 005737 002512                TST      WORD.COUNT        ; TEST FOR ZERO
2803 031636 001435                BEQ
2804 031640 013701 002504      MOV      MEMSIZ,R1         ; GET AVAILABLE MEM SIZE IN R1
2805 031644 013702 002512      MOV      WORD.COUNT,R2    ; GET WORD COUNT IN R2
2806 031650 005402      NEG      R2               ; CONVERT WORD COUNT FROM 2'S COMP
2807 031652 020102      CMP      R1,R2            ; COMPARE AVAILABLE SIZE WITH WORD COUNT
2808 031654 103040      BCC      30$             ; IF OK, BRANCH
2809
2810      ; NOT ENOUGH MEMORY AVAILABLE IN UNMAPPED MEMORY, TRY HI MEMORY WITH MAPPING
2811
2812 031656 012737 000001 002520      MOV      @YES,MAPPING     ; SET FLAG TO MAPPING
2813 031664 013700 002512      MOV      WORD.COUNT,R0    ; GET WORD COUNT IN R0
2814 031670 005400      NEG      R0               ; CONVERT WORD COUNT FROM 2'S COMP
2815 031672 012701 000005      MOV      @5,R1            ; LOOP COUNTER
2816
2817      ; CONVERT WORD.COUNT TO PAR VALUE AND CHECK
2818
2819 031676 000241 10$: CLC                    ; CLEAR CARRY FLAG
2820 031700 006000      ROR      R0               ; ROTATE RIGHT
2821 031702 077103      SOB      R1,10$          ; AGAIN IF NOT DONE
2822 031704 005200      INC      R0               ; INCREMENT TO NEXT PAR VALUE
2823 031706 013737 002120 002454      MOV      L#HIMEM,VAL      ; GET TOTAL MEMORY IN VAL
2824 031714 162737 001600 002454      SUB      @1600,VAL        ; GET TOTAL AVAILABLE HI MEM IN PAR FORMAT
2825 031722 003403      BLE
2826 031724 023700 002454      CMP      VAL,R0          ; SEE IF BUFFER LARGE ENOUGH
2827 031730 103012      BCC      30$             ; BRANCH IF ENOUGH MEMORY
2828
2829
2830      ; ERROR CONDITION, NOT ENOUGH MEMORY AVAILABLE
2831
2832 031732 20$:
2833 031732      ERRHRD  67,,ERR5        ; TELL OPERATOR ERROR
031732 104456      TRAP    C#ERRHRD
031734 000103      .WORD   67
031736 000000      .WORD   0
031740 007076      .WORD   ERR5
2834 031742 004737 013116      CALL    ENDERR           ; GO PRINT OUT ENDING TERMINATOR
2835 031746 012701 013630      MOV     @ABORT,R1        ; ABORT CODE TO R1
2836 031752 000137 033302      JMP     E                 ; EXIT TEST
2837
2838      ; END ERROR CONDITION
2839
2840      ; NOW SEE IF READ OR WRITE DMA
2841
2842 031756 30$:
2843 031756 013701 002510      MOV     PATTERN,R1       ; ASSUME NO WRITE (READ)
2844 031762 012737 000002 002456      MOV     @DMAWT,EXP       ; SET UP DIRECTION BIT NO WRITE (READ)
2845 031770 022737 000001 002506      CMP     @YES,DIR         ; DMA WRITE FROM VAX

```

TEST 22: CONVERSATION MODE

```

2846 031776 001004          BNE 40$          ; IF NO WRITE (READ), BRANCH
2847 032000 012701 000001  MOV #1,R1       ; POINT R1 TO PATTERN TO CLEAR BUFFER
2848 032004 005037 002456  CLR EXP        ; SET WORD TO WRITE
2849
2850          ; SET UP FOR BUFFER
2851
2852 032010          40$:
2853 032010 013702 002512  MOV WORD.COUNT,R2 ; GET WORD COUNT TO R2
2854 032014 005402          NEG R2          ; CONVERT WORD COUNT FROM 2'S COMP
2855 032016 006301          ASL R1          ; SET UP TO GET PATTERN
2856 032020 006301          ASL R1
2857 032022 062701 033414  ADD #DATPRN,R1   ; POINT TO PATTERN
2858 032026 022737 000001 002520  CMP #YES,MAPPING ; USE MAPPING BUFFER ROUTINE?
2859 032034 001414          BEQ 80$        ; BRANCH IF YES
2860
2861          ; SET UP BUFFER WITH NO MAPPING
2862
2863 032036          50$:
2864 032036 013703 002502  MOV FREMEM,R3    ; GET STARTING ADDRESS OF BUFFER IN R3
2865 032042 012704 000002 60$: MOV #2,R4        ; # OF WORDS IN PATTERN
2866 032046 012123          70$: MOV (R1)+,(R3)+  ; MOVE PATTERN WORD TO BUFFER
2867 032050 005302          DEC R2          ; DECREMENT WORD COUNT
2868 032052 001433          BEQ 120$       ; BRANCH IF DONE
2869 032054 005304          DEC R4          ; DECREMENT PATTERN WORD COUNT
2870 032056 001373          BNE 70$          ; IF NOT LAST PATTERN WORD, LOOP
2871 032060 162701 000004  SUB #4,R1        ; POINT R1 BACK TO START OF PATTERN
2872 032064 000766          BR 60$         ; START ALL OVER AGAIN
2873
2874          ; NO MAP BUFFER TAKEN CARE OF, SET AND WAIT FOR INTERRUPT
2875
2876
2877          ; SET UP BUFFER IF MAPPING IS USED
2878
2879 032066          80$:
2880 032066 012737 001600 002522  MOV #1600,APR    ; STORE UP INITIAL APR VALUE
2881 032074 012737 000001 177572  MOV #BITO,MMRO  ; ENABLE MAPPING
2882 032102 004737 033366          JSR PC,SETAPR   ; GO SET UP FOR APR
2883 032106 012704 000002          90$: MOV #2,R4        ; # OF WORDS IN PATTERN
2884 032112 012123          100$: MOV (R1)+,(R3)+ ; MOVE PATTERN WORD TO BUFFER
2885 032114 005305          DEC R5          ; DECREMENT APR BK LOOP COUNTER
2886 032116 001002          BNE 110$       ; BRANCH IF NEW APR IS NOT NEEDED
2887 032120 004737 033366          JSR PC,SETAPR   ; GO SET UP FOR NEXT APR
2888 032124 005302          110$: DEC R2          ; DECREMENT WORD COUNT
2889 032126 001405          BEQ 120$       ; BRANCH IF DONE
2890 032130 005304          DEC R4          ; DECREMENT PATTERN WORD COUNT
2891 032132 001367          BNE 100$       ; IF NOT LAST PATTERN WORD, LOOP
2892 032134 162701 000004  SUB #4,R1        ; POINT R1 BACK TO START OF PATTERN
2893 032140 000762          BR 90$         ; START ALL OVER AGAIN
2894
2895          ; INTERRUPT ONLY NORMAL WAY OUT OF LOOP
2896
2897 032142          120$:
2898 032142 013701 002532  MOV DEFAULT,R1   ; NUMBER OF SECONDS TO WAIT
2899 032146 012702 026000 130$: MOV #TIMOUT,R2 ; TIMEOUT VALUE TO R2
2900 032152          140$: BREAK          ; ALLOW CONTROL C
2901 032154 104422 002450  TRAP C#BRK     ; DID INTERRUPT OCCUR

```

## TEST 22: CONVERSATION MODE

```

2902 032160 001021      BNE      150$      ; BRANCH IF INTERRUPT OCCURED
2903 032162 005302      DEC      R2        ; DECREMENT TIMEOUT
2904 032164 001372      BNE      140$      ; IF NOT DONE, BRANCH
2905 032166 005301      DEC      R1        ; DECREMENT SECOND COUNTER
2906 032170 001366      BNE      130$      ; IF NOT DONE, BRANCH
2907
2908 ;
2909 ; ERROR CONDITION, TIMEOUT WAITING FOR INTERRUPT
2910 ;
2910 032172      ERRHRD  68...ERR6
      032172 104456    TRAP   C#ERRHRD
      032174 000104    .WORD  68
      032176 000000    .WORD  0
      032200 007320    .WORD  ERR6
2911 032202 004737 013116  CALL   ENDERR      ; GO PRINT OUT ENDING TERMINATOR
2912 032206 042777 000100 147726  BIC   #PRSIE,#QCMD ; CLEAR INTERRUPT
2913 032214 012701 013630  MOV   #ABORT,R1    ; ABORT CODE TO R1
2914 032220 000137 033302  JMP   E            ; JUMP TO TEST EXIT
2915
2916 ;
2917 ; END WAIT FOR INTERRUPT LOOP, INTERRUPT RECEIVED
2918 032224      150$:
2919 032224 042777 000100 147710  BIC   #PRSIE,#QCMD ; CLEAR INTERRUPT
2920
2921 ;
2922 ; SET UP FOR DMA TRANSFER
2923 032232 052777 000020 147710  BIS   #QVREN,#QCTL ; READ ENABLE
2924 032240 012777 000000 147720  MOV   #0,#QVAD     ; CLEAR VIEW REGISTER
2925 032246 017737 147716 002514  MOV   #QVDA,LOW16  ; READ LOW 16 BITS OF UNIBUS ADDRESS
2926 032254 013777 002514 147672  MOV   LOW16,#QUBA  ; LOAD LOW 16 BITS OF UNIBUS ADDRESS
2927 032262 017737 147702 002516  MOV   #QVDA,HI2    ; READ HI 2 BITS OF UNIBUS ADDRESS
2928 032270 032737 177757 002516  BIT   #↑CBIT4!BITS,HI2 ; SEE IF ONLY BITS 4 AND 5 ARE SET
2929 032276 001412      BEQ   160$        ; BRANCH IF OK
2930
2931 ;
2932 ; ERROR CONDITION, MORE THAN BITS 4 OR 5 SET FOR EXTENDED ADDRESS ON UNIBUS
2933 ;
2933 032300      ERRHRD  69...ERR10      ; TELL OPERATOR ERROR
      032300 104456    TRAP   C#ERRHRD
      032302 000105    .WORD  69
      032304 000000    .WORD  0
      032306 007740    .WORD  ERR10
2934 032310 004737 013116  CALL   ENDERR      ; GO PRINT OUT ENDING TERMINATOR
2935 032314 012701 013630  MOV   #ABORT,R1    ; ABORT CODE TO R1
2936 032320 000137 033302  JMP   E            ; END TEST
2937
2938 ;
2939 ; END ERROR CONDITION
2940 032324      160$:
2941 032324 053737 002516 002456  BIS   HI2,EXP      ; SET UP EXTENDED BITS IN EXP
2942 032332 022737 000001 002520  CMP   #YES,MAPPING ; ARE WE MAPPING?
2943 032340 001407      BEQ   170$        ; BRANCH IF YES
2944
2945 ;
2946 ; SET UP QBA IF NO MAPPING
2947 032342 013777 002502 147606  MOV   FREMEM,#QBA  ; Q-BUS ADD REG WITH BUFFER START ADD
2948 032350 042777 037400 147574  BIC   #37400,#QDMA ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
2949 032356 000406      BR    180$
2950 ;

```

## TEST 22: CONVERSATION MODE

```

2951                                     ; SET UP QBA WITH MAPPING
2952                                     ;
2953 032360 012777 160000 147570 170$: MOV    #160000,@QBA           ; START BUFFER AT 28K
2954 032366 042777 037400 147556      BIC    #37400,@QDMA         ; MAKE SURE Q-BUS EXTENDED BITS ARE ZERO
2955 032374 013777 002512 147556 180$: MOV    WORD.COUNT,@QWC       ; GET WORD COUNT IN QWC
2956 032402 052737 000100 002456      BIS    #RDYIE,EXP          ; SET UP TO ALLOW DMA RDY INTERRUPT
2957 032410 013777 002456 147534      MOV    EXP,@QDMA          ; SET UP DMA REGISTER
2958 032416                                     CLRVEC VECTOR
      032416 013700 002444      MOV    VECTOR,R0
      032422 104436      TRAP   C#CVEC
2959 032424      SETVEC VECTOR,#CONT1,#PRI07      ; SET UP NEW INTERRUPT VECTOR
      032424 012746 000340      MOV    #PRI07,-(SP)
      032430 012746 032550      MOV    #CONT1,-(SP)
      032434 013746 002444      MOV    VECTOR,-(SP)
      032440 012746 000003      MOV    #3,-(SP)
      032444 104437      TRAP   C#SVEC
      032446 062706 000010      ADD    #10,SP
2960 032452 052777 000100 147466      BIS    #ERRIE,@QINT      ; ENABLE ERROR INTERRUPT
2961 032460 052777 000100 147464      BIS    #RDYIE,@QDMA     ; ENABLE RDY INTERRUPT
2962 032466 052777 000001 147456      BIS    #GO,@QDMA        ; INITIATE DMA TRANSFER
2963                                     ;
2964                                     ; WAIT FOR INTERRUPT
2965                                     ;
2966 032474 013701 002532      MOV    DEFAULT,R1       ; NUMBER OF SECONDS OF TIMEOUT
2967 032500 012702 026000 190$: MOV    #TIMOUT,R2          ; TIME OUT DELAY TO R2
2968 032504      200$: BREAK @TIMOUT,R2          ; ALLOW CONTROL C
      032504 104422      TRAP   C#BRK
2969 032506 005302      DEC    R2                ; DECREMENT TIMEOUT
2970 032510 001375      BNE   200$              ; IF NOT DONE LOOP
2971 032512 005301      DEC    R1                ; DECREMENT SECOND COUNTER
2972 032514 001371      BNE   190$              ; IF NOT DONE, LOOP
2973                                     ;
2974                                     ; ERROR CONDITION, TIMEOUT OCCURED WAITING FOR INTERRUPT
2975                                     ;
2976 032516 042777 000100 147426      BIC    #RDYIE,@QDMA     ; CLEAR INTERRUPT BIT
2977 032524      ERRHRD 70...ERR7          ; TELL OPERATOR TIMEOUT
      032524 104456      TRAP   C#ERRHRD
      032526 000106      .WORD  70
      032530 000000      .WORD  0
      032532 007424      .WORD  ERR7
2978 032534 004737 013116      CALL  ENDERR            ; GO PRINT OUT ENDING TERMINATOR
2979 032540 012701 013630      MOV    #ABORT,R1       ; ABORT CODE TO R1
2980 032544 000137 033302      JMP    E                ; BRANCH TO END OF TEST
2981                                     ;
2982                                     ; END ERROR CONDITION
2983                                     ;
2984                                     ; DMA INTERRUPT SHOULD GO HERE
2985                                     ;
2986 032550      BGNSRV  CONT1
      032550      CONT1::
2987 032550 042777 000100 147374      BIC    #RDYIE,@QDMA     ; CLEAR INTERRUPT BIT
2988 032556 012716 032564      MOV    #CONT2,(SP)     ; CORRECT STACK
2989 032562      ENDSRV
      032562      L10047:
      032562 000002      RTI
2990 032564      CONT2:
2991 032564 032777 100000 147354      BIT    #ERR,@QINT      ; ERROR INTERRUPT?

```

TEST 22: CONVERSATION MODE

```

2992 032572 001414          BEQ      210#           ; BRANCH IF NO ERROR INTERRUPT
2993                          ;
2994                          ; ERROR CONDITION, ERROR INTERRUPT OCCURED
2995                          ;
2996 032574          ERRHRD  71.,,ERR11      ; TELL OPERATOR ERROR INTERRUPT OCCURED
      032574 104456    TRAP     C#ERRHRD
      032576 000107    .WORD   71
      032600 000000    .WORD   0
      032602 010030    .WORD   ERR11
2997 032604 004737 013116    CALL    ENDERR          ; GO PRINT OUT ENDING TERMINATOR
2998 032610 005077 147332    CLR     @QINT          ; CLEAR INTERRUPT REGISTER
2999 032614 012701 013630    MOV     @ABORT,R1     ; ABORT CODE TO R1
3000 032620 000137 033302    JMP     E             ; BRANCH TO END OF TEST
3001                          ;
3002                          ; END ERROR CONDITION
3003                          ;
3004 032624          210#:
3005 032624 052777 000001 147310    BIS     @CMDACK,@QCMD ; TELL MASTER DAM DONE
3006 032632 022737 000000 002506    CMP     @NO,DIR       ; DOES LSI NEED TO CHECK DATA
3007 032640 001002          BNE     215#           ; BRANCH IF DMA WAS VAX TO LSI
3008 032642 000137 033276          JMP     330#           ; IF NOT, END TEST
3009                          ;
3010                          ; DATA WAS FROM VAX TO LSI, CHECK DATA
3011                          ;
3012                          ; SET UP TO POINT TO DATA FIELD
3013                          ;
3014 032646 013702 002512    215#:  MOV     WORD.COUNT,R2 ; R2 GETS WORD COUNT
3015 032652 005402          NEG     R2              ; CONVERT WORD COUNT FROM 2'S COMP
3016 032654 013701 002510    MOV     PATTERN,R1    ; R1 GETS PATTERN CODE
3017 032660 006301          ASL    R1              ; POINT R1 TO PATTERN
3018 032662 006301          ASL    R1
3019 032664 062701 033414    ADD     @DATPRN,R1
3020 032670 005037 002474    CLR     COUNTER       ; CLEAR ADDRESS COUNTER
3021 032674 022737 000001 002520    CMP     @YES,MAPPING  ; WAS MAPPING USED
3022 032702 001462          BEQ     260#           ; BRANCH IF MAPPING WAS USED
3023                          ;
3024                          ; FIRST CHECK DATA IF NO MAPPING WAS USED
3025                          ;
3026 032704 013703 002502    MOV     FREMEM,R3     ; R3 TO START OF BUFFER
3027 032710 005037 002462    CLR     ERROR1        ; CLEAR ERROR COUNTER
3028 032714 012704 000002    220#:  MOV     @2,R4          ; # OF WORDS IN PATTERN
3029 032720 005237 002474    230#:  INC     COUNTER       ; INCREMENT COUNTER
3030 032724 012337 002454    MOV     (R3)+,VAL     ; GET WORD FROM BUFFER
3031 032730 012137 002456    MOV     (R1)+,EXP     ; GET EXPECTED PATTERN IN EXP
3032 032734 023737 002454 002456    CMP     VAL,EXP       ; COMPARE VALUE WITH EXPECTED
3033 032742 001433          BEQ     250#           ; BRANCH IF DATA OK
3034                          ;
3035                          ; ERROR CONDITION, NOT WHAT EXPECTED IN BUFFER
3036                          ;
3037 032744 005737 002462    TST     ERROR1        ; ANY OTHER ERRORS OCCURED
3038 032750 001004          BNE     240#           ; SKIP HEADER ERROR IF ALREADY PRINTED
3039 032752          ERRHRD  72.,,ERR8      ; PRINT ERROR HARD MESSAGE
      032752 104456    TRAP     C#ERRHRD
      032754 000110    .WORD   72
      032756 000000    .WORD   0
      032760 007530    .WORD   ERR8
3040 032762 005237 002462    240#:  INC     ERROR1        ; INCREMENT ERROR COUNTER

```





## TEST 22: CONVERSATION MODE

```

3079      033206 062706 000012      ADD      #12,SP
3080      ; END ERROR CONDITION
3081      ;
3082 033212 005305      3000: DEC      R5          ; DECREMENT APR 8K LOOP COUNTER
3083 033214 001002      BNE      3100         ; BRANCH IF NEW APR IS NOT NEEDED
3084 033216 004737 033366      JSR      PC,SETAPR   ; GO SET UP FOR NEXT APR
3085 033222 005302      3100: DEC      R2          ; DECREMENT WORD COUNT
3086 033224 001405      BEQ      3200         ; BRANCH IF DONE
3087 033226 005304      DEC      R4          ; DECREMENT PATTERN WORD COUNT
3088 033230 001323      BNE      2800         ; IF NOT LAST PATTERN WORD, LOOP
3089 033232 162701 000004      SUB      #4,R1       ; POINT R1 BACK TO START OF PATTERN
3090 033236 000716      BR       2700         ; START ALL OVER AGAIN
3091
3092
3093      ; BUFFER CHECKED, PRINT TOTAL ERROR COUNT IF NECESSARY
3094      ;
3095 033240      3200:
3096 033240 005737 002462      TST      ERROR1      ; ANY ERRORS
3097 033244 001414      BEQ      3300         ; IF NO ERRORS, BRANCH
3098 033246      PRINTB #FOR11,ERROR1 ; TELL TOTAL ERROR COUNT
3099 033272 004737 013116      MOV      ERROR1,-(SP)
3100      MOV      #FOR11,-(SP)
3101      MOV      #2,-(SP)
3102      MOV      SP,R0
3103 033276 012701 000000      TRAP    C#PNTB
3104 033302 013700 002444      ADD      #6,SP
3105 033306 104436      CALL    ENDERR      ; GO PRINT OUT ENDING TERMINATOR
3106 033310 012746 000340      ;
3107 033314 012746 012652      ; END OF TEST
3108 033320 013746 002444      3300: MOV      #0,R1      ; TEST ENDED NORMALLY TEST CODE
3109 033324 012746 000003      E:   CLRVEC VECTOR   ; CLEAR VECTOR
3110 033330 104437      MOV      VECTOR,R0
3111 033332 062706 000010      TRAP    C#CVEC
3112 033336 052777 000100 146602 SETVEC VECTOR,#ERRINT,#PRI07 ; SET UP INTERRUPT VECTOR
3113 033344 022737 000000 002520 MOV      #PRI07,-(SP)
3114 033352 001404      MOV      #ERRINT,-(SP)
3115      MOV      VECTOR,-(SP)
3116      MOV      #3,-(SP)
3117      TRAP    C#SVEC
3118      ADD      #10,SP
3119      BIS      #ERRIE,#QINT ; ENABLE ERROR INTERRUPT
3120      CMP      #NO,MAPPING ; WAS MAPPING USED
3121      BEQ      3400         ; BRANCH IF NO MAPPING USED
3122      ; MAPPING WAS USED, RESTORE MAPPING REGISTERS AND TURN OFF MAPPING
3123      ;
3124      JSR      PC,MINIT ; RESTORE MM TO NORMAL
3125      CLR      MMRO      ; DISABLE MEMORY MANAGEMENT
3126      3400:
3127      RTS      PC
3128      ;
3129      ; SUBROUTINE TO SETUP APR3 WITH NEW VALUE, AND R5 WITH NEW 4KW LOOP
3130      ; VALUE ON EXIT, R5 CONTAINS 10000 (4KW), AND APR3 HAS BEEN LOADED
3131      ; WITH POINT TO NEXT 4KW BLOCK, AND R3 CONTAINS STARTINS ADDRESS OF

```

TEST 22: CONVERSATION MODE

```

3121                                     ; MAPPED BUFFER.
3122                                     ;
3123 033366                               ; SETAPR:
3124 033366 012705 010000                 MOV    #10000,R5           ; R5 IS 8K LOOP COUNTER
3125 033372 013737 002522 172346         MOV    APR,APR3         ; LOAD APR 3 WITH INITIAL VALUE
3126 033400 012703 060000                 MOV    #60000,R3       ; GET STARTING ADDRESS OF BUFFER IN R3
3127 033404 062737 000200 002522         ADD    #200,APR        ; SET UP FOR NEXT APR
3128 033412 000207
3129
3130                                     DATPRN:
3131 033414 177777                         .WORD 177777           ; CODE 0
3132 033416 177777                         .WORD 177777
3133
3134 033420 000000                         .WORD 0                ; CODE 1
3135 033422 000000                         .WORD 0
3136
3137 033424 122645                         .WORD 122645          ; CODE 2
3138 033426 122645                         .WORD 122645
3139
3140 033430 055132                         .WORD 055132          ; CODE 3
3141 033432 055132                         .WORD 055132
3142
3143 033434 000000                         .WORD 0                ; CODE 4
3144 033436 177777                         .WORD 177777

```

TEST 23: CHECK QHALT (MANUAL)

```

3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160 033440
      033440
      033440 012737 000027 002114
      033446 004737 013034
3161 033452
      033452 013700 002444
      033456 104436
3162 033460 005077 146462
3163 033464
      033464 012746 033651
      033470 012746 000001
      033474 010600
      033476 104417
      033500 062706 000004
3164 033504
      033504 012746 033540
      033510 012746 033730
      033514 012746 000002
      033520 010600
      033522 104417
      033524 062706 000006
3165 033530 004737 013116
3166 033534
      033534 104422
3167 033536 000776
3168 033540
      033540 012746 033570
      033544 012746 000001
      033550 010600
      033552 104417
      033554 062706 000004
3169 033560 004737 013116
3170 033564 000137 013740
3171 033570 045 116
3172 033651 045 116
3173 033730 045 116
3174

```

```

.SBTTL TEST 23: CHECK QHALT (MANUAL)
;*****
;**
;TEST DESCRIPTION-
; TELLS OPERATOR TO HALT PDP-11
; CHECKS FOR QHALT BIT
; PDP-11 ENTERS ODT
;
;TEST STEPS-
; Q OPERATOR MESSAGE
; U WAIT FOR INTERRUPT
; U CHECK QHALT BIT
;--
;*****
STATST
TEST23::
MOV #23,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
CALL BMSG ;CHECK FOR PNT FLAG
CLRVEC VECTOR ; GET RID OF INTERRUPT VECTOR
MOV VECTOR,R0
TRAP C#CVEC
CLR #QINT ; CLEAR INTERRUPT REGISTER
PRINTF #MHALT1 ; PRINT MESSAGE
MOV #MHALT1,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C#PNTF
ADD #4,SP
PRINTF #MHALT2,#20# ; PRINT MESSAGE
MOV #20#,-(SP)
MOV #MHALT2,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C#PNTF
ADD #6,SP
CALL ENDERR ; PRINT MESSAGE TERMINATOR
; ALLOW CONTROL C
10# : BREAK
TRAP C#BRK
BR 10# ; BRANCH BACK
20# : PRINTF #HLTMSG ; PRINT HALT CONTINUE MESSAGE
MOV #HLTMSG,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C#PNTF
ADD #4,SP
CALL ENDERR ; PRINT MESSAGE TERMINATOR
JMP START
061 HLTMSG: .ASCIZ /#N1#ACONTINUING FROM HALT, WAITING FOR NEXT TEST/
061 MHALT1: .ASCIZ /#N1#APRESS THE HALT KEY ON THE LSI FRONT PANEL/
061 MHALT2: .ASCIZ /#N1#ATO REENTER LSI TEST FROM ODT, TYPE #06#AG/
.EVEN

```

TEST 24: CHECK Q POWER FAIL (MANUAL)

```

3176 .SBTTL TEST 24: CHECK Q POWER FAIL (MANUAL)
3177 ;:*****
3178 ;**
3179 ;TEST DESCRIPTION-
3180 ; TELLS OPERATOR TO POWER DOWN FEP
3181 ; CHECKS FOR QPWRFL BIT
3182 ; PDP-11 ENTERS BOOT PROM
3183 ;
3184 ;TEST STEPS-
3185 ; Q OPERATOR MESSAGE
3186 ; U WAIT FOR INTERRUPT
3187 ; U CHECK QPWRFL BIT
3188 ;--
3189 ;:*****
3190 034010 STATST
3191 034010 TEST24::
3192 034010 012737 000030 002114 MOV #24,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
3193 034016 004737 013034 CALL BMSG ;CHECK FOR PNT FLAG
3194 034022 013700 002444 CLRVEC VECTOR ; GET RID OF INTERRUPT VECTOR
3195 034022 104436 MOV VECTOR,R0
3196 034026 104436 TRAP C#CVEC
3197 034030 005077 146112 CLR #QINT ; CLEAR INTERRUPT REGISTER
3198 034034 012746 034064 PRINTF #MFAIL ; PRINT MESSAGE
3199 034034 012746 000001 MOV #MFAIL,-(SP)
3200 034040 010600 MOV #1,-(SP)
3201 034046 104417 MOV SP,R0
3202 034050 062706 000004 TRAP C#PNTF
3203 034054 004737 013116 ADD #4,SP
3204 034060 104422 CALL ENDERR ; PRINT MESSAGE TERMINATOR
3205 034062 000776 10# : BREAK ; ALLOW CONTROL C
3206 034062 000776 TRAP C#BRK
3207 034062 000776 BR 10# ; BRANCH BACK
3208 034064 045 116 061 MFAIL: .ASCII /#N1#APOWERR DOWN THE LSI FEP/
3209 034117 045 116 061 .ASCIZ /#N1#AREBOOT XXDP* OR MANUAL BOOT TEST TO CONTINUE/
3210 3199 .EVEN

```

TEST 25: CHECK QBOOT (MANUAL)

3201  
3202  
3203  
3204  
3205  
3206  
3207  
3208  
3209  
3210  
3211  
3212  
3213  
3214 034202  
034202  
034202 012737 000031 002114  
034210 004737 013034  
3215 034214 012701 000024  
3216 034220 012702 026000  
3217 034224  
034224 104422  
3218 034226 077202  
3219 034230 077105  
3220  
3221  
3222  
3223 034232  
034232 104456  
034234 000112  
034236 000000  
034240 010466  
3224 034242 004737 013116  
3225 034246 012701 013630  
3226 034252 000207  
3227  
3228 034254

```
.SBTTL TEST 25: CHECK QBOOT (MANUAL)
;*****
;
;TEST DESCRIPTION-
; SETS QBOOT BIT
; CHECKS FOR PDP-11 RESPONSE
;
;TEST STEPS-
; Q OPERATOR MESSAGE
; U WAIT FOR INTERRUPT
; U CHECK QHALT BIT
; --
;*****
STATST
TEST25::
MOV #25,L#TEST ;TEST NUMBER FOR ERROR MESSAGES
CALL BMSG ;CHECK FOR PNT FLAG
MOV #20.,R1 ; NUMBER OF SECONDS TO TIME OUT
10#: MOV #TIMOUT,R2 ; TIMOUT VALUE TO R2
15#: BREAK
TRAP C#BRK
SOB R2,15# ; DECREMENT TIMEOUT VALUE, BRANCH IF BNE
SOB R1,10# ; DECREMENT SECOND TIMEOUT, BRANCH IF BNE
;
; ERROR CONDITION, SHOULD HAVE BOOTED BY NOW
;
FARHRD 74...,ERR15
TRAP C#ERHRD
.WORD 74
.WORD 0
.WORD ERR15
CALL ENDERR ; GO PRINT OUT ENDING TERMINATOR
MOV #ABORT,R1 ; ABORT CODE TO R1
RETURN ; TO MAINLINE
ENDMOD
```

TEST 25: CHECK QBOOT (MANUAL)

1  
13  
14  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
66  
67  
68  
69  
70  
71  
72  
79

034254  
034254 000010  
034256  
034256 000031  
034260 034276  
034262 000000  
034264 177776  
034266  
034266 001031  
034270 034331  
034272 000000  
034274 001000

```
.ENABL AMA
.SBTTL HARDWARE PARAMETER CODING SECTION
BGNMOD
; **
; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
; WITH THE OPERATOR.
; --
```

```
BGNHRD
.WORD L10050-L#HARD/2
L#HARD::
GPRMA G1,0,0,0,177776,YES ;BASE ADDRESS
.WORD T#CODE
.WORD G1
.WORD T#LOLIM
.WORD T#HILIM
GPRMA G2,2,0,0,1000,YES ;VECTOR
.WORD T#CODE
.WORD G2
.WORD T#LOLIM
.WORD T#HILIM
```

```
ENDHRD
.EVEN
L10050:
G1: .ASCIZ /BRIDGE MODULE BASE ADDRESS/
G2: .ASCIZ /BRIDGE MODULE VECTOR /
.EVEN
```

102 122 111  
102 122 111

## SOFTWARE PARAMETER CODING SECTION

```

81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96 034364
97 034364
104
105 034564
      034564 034600
      034566 000004
      034570
106
107 034570
108 034570
      034570 000000
      034572 000002
      034574
109 034574 160500
110 034576 000500
111 034600
      034600
112 034600
113      000001

```

```

.SBTTL SOFTWARE PARAMETER CODING SECTION

```

```

; **
; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
; WITH THE OPERATOR.
; --

```

```

;SOFTWARE PARAMETER SECTION NOT NEEDED
.EVEN

```

```

$PATCH::
      .BLKW 100

```

```

      LASTAD
      .EVEN
      .WORD T#FREE
      .WORD T#SIZE

```

```

L$LAST::
;HARDCODED P-TABLE
      BGNSETUP 1
      BGNPTAB
      .WORD 0
      .WORD L10053-./2-1

```

```

L10051:
      .WORD QCMDAD
      .WORD VEC
      ENDP TAB

```

```

L10053:
      ENDSETUP

```

```

.END

```

```

;HARDWARE UNIT 0 DEFAULT BASE ADDRESS
;HARDWARE UNIT 0 DEFAULT VECTOR

```



## SYMBOL TABLE

ABO	005534	G	COL1	026620	G	C#INIT=	000011	EQZ	025776	FOR13	004364	G
ABORT	013630		COL2	026634		C#INLP=	000020	ERR	= 100000	FOR14	004435	G
ACKIE	= 000100	G	CONFIG	003217	G	C#MANI=	000050	ERRIE	= 000100	FOR14.	004477	G
ADD	002452	G	CONT1	032550	G	C#MEM =	000031	ERRINT	012652	FOR15	004536	G
ADDRES	002142	G	CONT2	032564		C#MSG =	000023	ERROR1	002462	FOR16	004612	G
ADR	= 000020	G	CONT3	020552	G	C#OPEN=	000034	ERROR2	002464	FOR17	004663	G
APR	002522	G	CONT4	020574		C#PNTB=	000014	ERR0	006270	FOR18	004734	G
APRO	= 172340	G	CONVER	031506		C#PNTF=	000017	ERR1	006316	FOR19	004770	G
APR1	= 172342	G	CO01	022400	G	C#PNTS=	000016	ERR10	007740	FOR2	003427	G
APR2	= 172344	G	CO02	022414		C#PNTX=	000015	ERR11	010030	FOR20	005104	G
APR3	= 172346	G	COP1	023142	G	C#QIO =	000377	ERR12	010146	FOR21	005134	G
APR4	= 172350	G	COP2	023156		C#RDBU=	000007	ERR13	010252	FOR22	005221	G
APR5	= 172352	G	COUNTE	002474	G	C#REFG=	000047	ERR14	010336	FOR23	005314	G
APR6	= 172354	G	COX1	024144	G	C#RESE=	000033	ERR15	010466	FOR24	005343	G
APR7	= 172356	G	COX2	024160		C#REVI=	000003	ERR16	010552	FOR25	005417	G
ASSEMB	= 000010		COY1	027440	G	C#RFLA=	000021	ERR2	006506	FOR26	005464	G
BAD	014140		COY2	027454		C#RPT =	000025	ERR3	006652	FOR3	003457	G
BADMSG	014170		CO1	021340	G	C#SEFG=	000046	ERR4	007012	FOR4	003547	G
BASEAD	002442	G	CO2	021354		C#SPRI=	000041	ERR5	007076	FOR5	003574	G
BITS#	012130	G	CTLB	011264		C#SVEC=	000037	ERR6	007320	FOR6	003622	G
BIT#	012110	G	CTLBIT	002751	G	C#TPRI=	000013	ERR7	007424	FOR7	003662	G
BIT0	= 000001	G	CXP1	024706	G	C19.1	031062	ERR8	007530	FOR8	003732	G
BIT00	= 000001	G	CXP2	024722		C19.2	031076	ERR9	007634	FOR9	003754	G
BIT01	= 000002	G	CXX1	025710	G	DATPRN	033414	EVL	= 000004	FOR9.1	004033	G
BIT02	= 000004	G	CXX2	025724		DEFAULT	002532	EW	023452	FOR9.2	004112	G
BIT03	= 000010	G	C#AU	= 000052		DFPTBL	002130	EW	030026	FOR9.3	004205	G
BIT04	= 000020	G	C#AUTO=	000061		DIAGMC=	000000	EXP	002456	FREMEM	002502	G
BIT05	= 000040	G	C#BRK =	000022		DIR	002506	EXPECT	002222	FUNCO	= 000004	G
BIT06	= 000100	G	C#BSEG=	000004		DLYST	017106	EXQBA	002236	FUNC1	= 000010	G
BIT07	= 000200	G	C#BSUB=	000002		DLY1	002526	EXQCMD	002222	FUNC2	= 000020	G
BIT08	= 000400	G	C#CEFG=	000045		DLY2	002530	EXQCTL	002230	F#AU	= 000015	
BIT09	= 001000	G	C#CLCK=	000062		DMAB	011310	EXQDMA	002232	F#AUTO=	000020	
BIT1	= 000002	G	C#CLEA=	000012		DMABIT	003015	EXQINT	002226	F#BGN =	000040	
BIT10	= 002000	G	C#CLOS=	000035		DMAWT =	000002	EXQMT0	002242	F#CLEA=	000007	
BIT11	= 004000	G	C#CLP1=	000006		DPX3	025262	EXQMT1	002244	F#DU =	000016	
BIT12	= 010000	G	C#CVEC=	000036		DPX4	026042	EXQUBA	002234	F#END =	000041	
BIT13	= 020000	G	C#DCLN=	000044		DP1	021752	EXQVAD	002246	F#HARD=	000004	
BIT14	= 040000	G	C#DDDU=	000051		DP19	031502	EXQVDA	002250	F#HW =	000013	
BIT15	= 100000	G	C#DRPT=	000024		DP2	022532	EXQVNT	002224	F#INIT=	000006	
BIT2	= 000004	G	C#DU =	000053		DP3	023516	EXQWC	002240	F#JMP =	000050	
BIT3	= 000010	G	C#EDIT=	000003		DP4	024276	EXW	025216	F#MOD =	000000	
BIT4	= 000020	G	C#ERDF=	000055		DP5	026752	EXZ	026706	F#MSG =	000011	
BIT5	= 000040	G	C#ERMR=	000056		DP6	030112	EZ	024232	F#PROT=	000021	
BIT6	= 000100	G	C#ERRO=	000060		DUMP	010656	E#END =	002100	F#PWR =	000017	
BIT7	= 000200	G	C#ERSF=	000054		E	033302	E#LOAD=	000035	F#RPT =	000012	
BIT8	= 000400	G	C#ERSO=	000057		ECONT	020130	E10#	010722	F#SEG =	000003	
BIT9	= 001000	G	C#ESCA=	000010		EDUMP	011166	E19	031436	F#SOFT=	000005	
BMSG	013034	G	C#ESG=	000005		EF.CON=	000036	E5#	010750	F#SRV =	000010	
BOE	= 000400	G	C#ESUB=	000003		EF.NEW=	000035	E6#	011006	F#SUB =	000002	
BRDGFL	002434	G	C#ETST=	000001		EF.PMR=	000034	FERR	013156	F#SW =	000014	
BRIDGE	002432	G	C#EXIT=	000032		EF.RES=	000037	FLAG	011710	F#TEST=	000001	
BRLEVE	002446	G	C#GETB=	000026		EF.STA=	000040	FORERR	005566	GO	= 000001	G
BUILD#	012152		C#GETW=	000027		EN	021706	FOR0	003306	G#CNT0=	000200	
CMACK=	000001	G	C#GMAN=	000043		ENDERR	013116	FOR1	003312	G#DELM=	000372	
CMDB	011170		C#GPHR=	000042		ENDI	013546	FOR10	004230	G#DISP=	000003	
CMDBIT	002604	G	C#GPLO=	000030		ENXM	020664	FOR11	004253	G#EXCP=	000400	
CMDBIT	002604	G	C#GPRI=	000040		EQ	022530	FOR12	004310	G#HILI=	000002	

## SYMBOL TABLE

G#LOLI=	000001	KISDRO=	172300 G	L#UNIT	002012 G	MQPFL =	000040 G	QWC	002160 G
G#NO =	000000	LOE =	040000 G	L10000	002136	MRDY =	001000 G	QXAD16=	000400 G
G#OFFS=	000400	LOGUNI	002436 G	L10001	002142	MSG1	005765 G	QXAD17=	001000 G
G#OFSI=	000376	LOOPS	002524 G	L10002	006314	MSG2	006012 G	QXAD18=	002000 G
G#PRMA=	000001	LOT =	000010 G	L10003	006504	MSG3	006061 G	QXAD19=	004000 G
G#PRMD=	000002	LOW16	002514 G	L10004	006650	MSG4	006130 G	QXAD20=	010000 G
G#PRML=	000000	L#ACP	002110 G	L10005	007010	MSG6	006202 G	QXAD21=	020000 G
G#RADA=	000140	L#APT	002036 G	L10006	007074	MT1B	011334	RAM	002466 G
G#RADB=	000000	L#AU	013730 G	L10007	007316	MT1BIT	003134 G	RDY =	000200 G
G#RADD=	000040	L#AUT	002070 G	L10010	007422	MVADR =	000002 G	RDYIE =	000100 G
G#RADL=	000120	L#AUTO	013634 G	L10011	007526	NAMES	002336 G	RPO	011564 G
G#RADO=	000020	L#CCP	002106 G	L10012	007632	NEXT	013342	RP1	011570 G
G#XFER=	000004	L#CLEA	013640 G	L10013	007736	NO =	000000 G	RP2	011660 G
G#YES =	000010	L#CO	002032 G	L10014	010026	NO#CHA	012640	RP4	011705 G
G1	034276	L#DEPO	002011 G	L10015	010144	NUM#TS	002252 G	SELDMA=	000024 G
G2	034331	L#DESC	002540 G	L10016	010250	ONEFIL=	000001	SELQCM=	000010 G
HELP =	000000	L#DESP	002076 G	L10017	010334	O#APTS=	000000	SELQEV=	000004 G
HELPI=	000000	L#DEVP	002060 G	L10020	010464	O#AU =	000000	SELRDD=	000034 G
HELP1	005654 G	L#DISP	002124 G	L10021	010550	O#BGNR=	000001	SELUBA=	000020 G
HELP2	005723 G	L#DLY	002116 G	L10022	010654	O#BGNS=	000000	SELUVA=	000014 G
HIUBM =	170370 G	L#DTP	002040 G	L10023	012650	O#DU =	000000	SELUVD=	000030 G
HI2	002516 G	L#DTYP	002034 G	L10024	012752	O#ERRT=	000000	SETAPR	033366
HLTMSG	033570	L#DU	01372\ G	L10025	013200	O#GNSW=	000000	SETUP	013334
HOE =	100000 G	L#DUT	002072 G	L10027	013632	O#POIN=	000001	SFPTBL	002140 G
HP1	002476 G	L#DVTY	002534 G	L10030	013636	O#SETU=	000001	SQBA	002206 G
HP2	002500 G	L#EF	002052 G	L10031	013716	PATTER	002510 G	SQCMD	002172 G
IBE =	010000 G	L#ENVI	002044 G	L10032	013726	PLOC	002440 G	SQCTL	002200 G
IDU =	000040 G	L#ETP	002102 G	L10033	013736	PNT =	001000 G	SQDMA	002202 G
IER =	020000 G	L#EXP1	002046 G	L10034	014216	PNTMSG	013074 G	SQINT	002176 G
INIT0	013210	L#EXP4	002064 G	L10035	020572	PRI =	002000 G	SQMT0	002212 G
INIT1	013240	L#EXP5	002066 G	L10036	021352	PRINTI	011356	SQMT1	002214 G
INIT2	013250	L#HARD	034256 G	L10037	022412	PRI00 =	000000 G	SQUBA	002204 G
INIT3	013332	L#HIME	002120 G	L10040	023154	PRI01 =	000040 G	SQVAD	002216 G
INTB	011240	L#HPCP	002016 G	L10041	024156	PRI02 =	000100 G	SQVDA	002220 G
INTBIT	002672 G	L#HPTP	002022 G	L10042	024720	PRI03 =	000140 G	SQVNT	002174 G
INTFLG	002472 G	L#HM	002130 G	L10043	025722	PRI04 =	000200 G	SQWC	002210 G
ISR =	000100 G	L#ICP	002104 G	L10044	026632	PRI05 =	000240 G	START	013740
IXE =	004000 G	L#INIT	013210 G	L10045	027452	PRI06 =	000300 G	STATU#	012642
I#AU =	000041	L#LADP	002026 G	L10046	031074	PRI07 =	000340 G	STORE#	012636
I#AUTO=	000041	L#LAST	034570 G	L10047	032562	PRISIE =	000100 G	SVCGBL =	000000
I#CLN =	000041	L#LOAD	002100 G	L10050	034276	P1	011410	SVCINS =	000000
I#DU =	000041	L#LUN	002074 G	L10051	034574	P2	011476	SVCSUB=	000000
I#HRD =	000041	L#MREV	002050 G	L10053	034600	QBA	002156 G	SVCTAG=	000000
I#INIT=	000041	L#NAME	002000 G	MAPPIN	002520 G	QBUS	003271 G	SVCTST=	000000
I#MOD =	000040	L#PRIO	002042 G	MAXUNI=	000010 G	QCMD	002142 G	S#LSYM=	010000
I#MSG =	000041	L#PROT	013202 G	MEMSIZ	002504 G	QCMDAD=	160500 G	TESTS	002254 G
I#PROT=	000040	L#PRT	002112 G	MES1#	012402 G	QCTL	002150 G	TEST1	014220 G
I#PTAB=	000041	L#REPP	002062 G	MES2#	012431 G	QDMA	002152 G	TEST10	017030 G
I#PMR =	000041	L#REV	002010 G	MFAIL	034064	QINT	002146 G	TEST11	020132 G
I#RPT =	000041	L#RPT	013170 G	MHALT1	033651	QMT0	002162 G	TEST12	020734 G
I#SEG =	000041	L#SPC	002056 G	MHALT2	033730	QMT1	002164 G	TEST13	021756 G
I#SETU=	000041	L#SPCP	002020 G	MILLI =	000013 G	QNEX =	020000 G	TEST14	022536 G
I#SRV =	000041	L#SPTP	002024 G	MIODIS=	000200 G	QUBA	002154 G	TEST15	023522 G
I#SUB =	000041	L#STA	002030 G	MINIT	012754 G	QVAD	002166 G	TEST16	024302 G
I#TST =	000041	L#SW	002140 G	M#RO =	177572 G	QVDA	002170 G	TEST17	025266 G
J#JMP =	000167	L#TEST	002114 G	M#R3 =	172516 G	QVNT	002144 G	TEST18	026046 G
KISARO=	172340 G	L#TIML	002014 G	M#HALT=	000100 G	QVREN =	000020 G	TEST19	026756 G

SYMBOL TABLE

TEST2 014240 G	TRAP4 012644 G	T\$SAVL= 177777	T\$\$RPT= 010025	WDTRST= 000001 G
TEST20 030116 G	TRPFLG 002450 G	T\$SEGL= 177777	T\$\$SRV= 010047	WDT0 = 000004 G
TEST21 030422 G	TST19 030440	T\$SIZE= 000004	T\$\$SW = 010001	WDT1 = 000010 G
TEST22 031506 G	T\$ARGC= 000001	T\$SUBN= 000000	T\$\$TES= 010034	WORD.C 002512 G
TEST23 033440 G	T\$CODE= 001031	T\$TAGL= 177777	T1 013740 G	WSPACE= 020040 G
TEST24 034010 G	T\$ERRN= 000112	T\$TAGN= 010054	UAM = 000200 G	X\$ 011124
TEST25 034202 G	T\$EXCP= 000000	T\$TEMP= 000004	UBUS 003277 G	X\$ALWA= 000000
TEST3 014474 G	T\$FLAG= 000040	T\$TEST= 000001	UNEX = 040000 G	X\$FALS= 000040
TEST4 014730 G	T\$FREE= 034600	T\$TSTM= 177777	UPWRFL= 010000 G	X\$OFFS= 000400
TEST5 015274 G	T\$GMAN= 000000	T\$TSTS= 000001	UXAD16= 000020 G	X\$TRUE= 000020
TEST6 015436 G	T\$HILI= 001000	T\$\$AU = 010033	UXAD17= 000040 G	YES = 000001 G
TEST7 016004 G	T\$LAST= 000001	T\$\$AUT= 010030	VAL 002454 G	\$HLP 011712 G
TEST8 016146 G	T\$LOLI= 000000	T\$\$CLE= 010031	VALUE 002470 G	\$HLPFO 012102 G
TEST9 016476 G	T\$LSYM= 010000	T\$\$DAT= 010053	VALUES 002172 G	\$LEN = 000012
TE\$ = 000003	T\$LTND= 000001	T\$\$DU = 010032	VEC = 000500 G	\$L1 = 000005
TIME.5= 000000 G	T\$NEST= 000000	T\$\$HAR= 010050	VECTOR 002444 G	\$MESLN 012104
TIME1 = 000004 G	T\$NS0 = 000000	T\$\$HW = 010000	VNTACK= 002000 G	\$NUM = 000031 G
TIME2 = 000010 G	T\$NS1 = 000004	T\$\$INI= 010027	VNTB 011214	\$PATCH 034364 G
TIME4 = 000014 G	T\$PCNT= 000000	T\$\$MSG= 010022	VNTBIT 002637 G	\$STORE 012106
TIMOUT= 026000 G	T\$PTAB= 010052	T\$\$PC = 000001	VNTPRS= 000001 G	\$TEMP = 000000
THRENB= 000002 G	T\$PTHV= 000001	T\$\$PRQ= 010026	WDTBUF 030412	\$TNUM = 000032
TNUM 002460 G	T\$PTNU= 000001	T\$\$PTA= 010052		

. ABS. 034600 000  
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 29352 WORDS ( 115 PAGES)  
DYNAMIC MEMORY: 20060 WORDS ( 77 PAGES)  
ELAPSED TIME: 00:08:41  
ZFPBA0.BIC,CZFPBA0/-SP=SVC/ML,ZFPBA01,ZFPBA02,ZFPBA03,ZFPBA04,ZFPBA05,ZFPBA06

