

TSV05

TSV05 CTRL LT2  
CVTSBA0

AH-T096A-MC  
FICHE 1 OF 2

SEP 1982  
COPYRIGHT © 1982  
MADE IN USA



The main body of the document is a large grid of approximately 15 columns and 15 rows of small, dense tables. Each cell in the grid contains a small table with multiple columns and rows of text, likely representing data points or parameters for a specific component or function. The text is too small to be legible, but the overall structure is a comprehensive data matrix.



TSV05

TSV05 CTRL LT2  
CVTSBA0

AH-T096A-MC  
FICHE 2 OF 2

SEP 1982  
COPYRIGHT © 1982  
MADE IN USA



Table with multiple columns and rows of data, likely a control or status table. The text is very faint and difficult to read, but appears to be organized in a grid format.





.REM\_  
IDENTIFICATION

ID:        AC-T095A-MC  
PRODUCT TITLE: CVTSBA0 TSV05 CTRL LT2  
AUTHOR:        DICK GORDON  
MAINTAINER:    SCOTT SNOWDON  
DATE:        MARCH 08, 1982

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

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

COPYRIGHT (C) 1982,1982 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL  
DEC

PDP  
DECUS

UNIBUS  
DECTAPE

MASSBUS

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
  
- 7.0 MAINTENANCE HISTORY



## 1.0 GENERAL INFORMATION

### 1.1 PROGRAM ABSTRACT

THIS IS A PDP-11/23 RESIDENT DIAGNOSTIC WHICH CHECKS THE FUNCTIONALITY OF A TSV05 MAGTAPE SUBSYSTEM WHILE CONNECTED TO A PDP-11/23 SYSTEM (Q-BUS). THE PROGRAM PROVIDES ERROR MESSAGES WHICH IDENTIFY FAILING FUNCTIONS THAT AID IN THE REPAIR OF THE DEVICE. THIS DIAGNOSTIC CONSIST OF TWELVE TEST. TEST 1-9 ARE EXECUTED IN SEQUENCE. TEST 10-12 ARE STAND ALONE TEST WHICH ALLOW THE OPERATOR TO PERFORM SPECIFIC FUNCTIONAL TEST ON SCOPE LOOPS ON CERTAIN FUNCTIONS.

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

PDP-11/23 PROCESSOR AND MEMORY  
CAUTION:DIAGNOSTIC REQUIRES 32K WORDS OF MEMORY  
(28K USEABLE AND 4K RESERVED FOR I/O PAGE)  
TSV05 MAGTAPE SUBSYSTEM (DRIVE AND CONTROLLER)  
CONSOLE TERMINAL  
PDP-11 DIAGNOSTIC SUPERVISOR (HSAAA.SYS VERSION 34 OR LATER)  
PDP-11 DIAGNOSTIC LOADER/MONITOR (XXDP+)

### 1.3 RELATED DOCUMENTS AND STANDARDS

#### DIGITAL EQUIPMENT CORPORATION DOCUMENTS:

1. CHQUS XXDP+ USERS GUIDE; DOCUMENT NUMBER AC-F348E-MC  
DATE: 14 JULY 1980.
2. TSV05 TRANSPORT SUBSYSTEM USER'S GUIDE; DOCUMENT NUMBER EK-TSV05-UG-001  
DATE: AUGUST 1982
3. TSV05 TRANSPORT SUBSYSTEM TECHNICAL MANUAL; DOCUMENT NUMBER EK-TSV05-TM-001  
DATE: AUGUST 1982
4. TSV05 TRANSPORT SUBSYSTEM INSTALLATION MANUAL; DOCUMENT NUMBER EK-TSV05-IN-001  
DATE: AUGUST 1982

### 1.4 DIAGNOSTIC HIERARCY PREREQUISITES

FUNCTIONAL PDP-11/23 CENTRAL PROCESSOR AND MEMORY  
FUNCTIONAL CONSOLE TERMINAL  
FUNCTIONAL STANDALONE DIAGNOSTIC SUPERVISOR



## FUNCTIONAL DIAGNOSTIC LOADER/MONITOR (XXDP+)

## 1.5 ASSUMPTIONS

ALL HARDWARE EXCEPT THE HARDWARE UNDER TEST IS ASSUMED TO WORK PROPERLY OR FALSE ERRORS CAN BE REPORTED.  
 THE TAPE BEING USED ON THE TS05 TRANSPORT IS A KNOWN GOOD REEL OF TAPE.  
 CVTSAA HAS RUN SUCCESSFULLY.

## 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".

## 2.1.1 OPERATOR COMMANDS

THE TSV05 DIAGNOSTIC IS A PDP-11/23 DIAGNOSTIC SUPERVISOR COMPATIBLE PROGRAM. ALL LOADING AND RUNTIME INSTRUCTIONS CAN BE REFERENCED IN THE CHQUS XXDP+ USERS GUIDE, DOCUMENT NUMBER AC-F348E-MC. THE USER ENTRY IS IN QUOTES.

## BOOT THE DIAGNOSTIC MEDIA

.R VTSB??  
 DIAG. RUN-TIME SERVICES REV D. APR 79  
 CVTSB-A-0



\*\*\*\*TSV05 LOGIC DIAGNOSTIC\*\*\*\*  
 UNIT IS TSV05  
 >DR

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 'DDDD'.

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:DDDD	EXECUTE DDDDD PASSES (DDDD = 1 TO 64000)
/FLAGS:FLGS	SET SPECIFIED FLAGS. FLAGS ARE DESCRIBED IN SECTION 2.3.
/EOP:DDDD	REPORT END OF PASS MESSAGE AFTER EVERY DDDDD PASSES ONLY. (DDDD = 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
IBR*	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

\*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 14 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).

AFTER INITIAL STARTING OF THE PROGRAM (START COMMAND TO THE DIAGNOSTIC SUPERVISOR), THE PROGRAM WILL ISSUE THE "CHANGE HW?" QUESTION TO ASK IF THE HARDWARE PARAMETERS ARE TO BE CHANGED (BY THE OPERATOR).

ON A 'N' (NO) RESPONSE TO THE 'CHANGE HW?' QUESTION, THE DIAGNOSTIC WILL RUN USING THE DEFAULT VALUES FOR ALL QUESTIONS. THE DEFAULT ADDRESS AND VECTOR ARE:

TSBA/TSDB = 172520, VECTOR = 224

ON A 'Y' (YES) RESPONSE TO THE QUESTION, THE FOLLOWING QUESTIONS WILL THEN BE ASKED TO ALLOW THE OPERATOR TO SELECT THE UNITS TO BE TESTED. A VALUE, IF PRESENT, LOCATED TO THE LEFT OF THE QUESTION MARK IS THE DEFAULT VALUE THAT WILL BE TAKEN IF ONLY A CARRIAGE RETURN IS TYPED AS A RESPONSE. A '(D)' IN A QUESTION INDICATES THAT A DECIMAL NUMBER IS REQUIRED AS A RESPONSE. AN '(O)' INDICATES AN OCTAL NUMBER IS BEING SOLICITED. AN '(L)' INDICATES THAT A LOGICAL RESPONSE IS TO BE MADE: 'Y' FOR YES, 'N' FOR NO.

# UNITS (D) ? <ENTER THE NUMBER OF M7196 CONTROLLERS  
PRESENT TO BE TESTED>

UNIT 0

DEVICE ADDRESS (O) 172520 ? <ENTER THE ADDRESS OF THE  
TSBA/TSDB REGISTER>

VECTOR (O) 224 ? <ENTER ADDRESS OF INTERRUPT  
VECTOR>

THE ADDRESS AND VECTOR QUESTIONS WILL BE ASKED FOR EACH OF THE NUMBER OF UNITS (CONTROLLERS) SPECIFIED IN THE '# UNITS?' QUESTION. LOGICAL UNIT NUMBERS ARE ASSIGNED IN ORDER, BEGINNING AT 0. UP TO FOUR UNITS CAN BE SELECTED FOR TESTING AS FOLLOWS:

UP TO 4 TSV05 CONTROLLERS PER 11/23 AND UP TO 2 DRIVES PER CONTROLLER

## 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).

THE FOLLOWING QUESTIONS ARE ASKED ON A START, RESTART, OR CONTINUE. THEY ALLOW FLEXIBILITY IN THE WAY THE PROGRAM BEHAVES.

CHANGE SW (L) ? <TYPE Y TO CAUSE THE FOLLOWING  
QUESTIONS TO BE ASKED>

INHIBIT ITERATIONS (L) N ? <TYPE 'Y' TO PREVENT MULTIPLE  
ITERATIONS OF CERTAIN TESTS.  
THIS CAUSES EACH TEST PASS TO  
RUN AS QUICKLY AS POSSIBLE.  
ONLY QUICK-RUNNING LOGIC  
TESTS USE MULTIPLE  
ITERATIONS.>



## 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 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 (O) ? 160000<CR>
SUB-DEVICE # (O) ? 0<CR>
Q-FACTOR (O) 0 ? 1<CR>

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

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

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

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

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

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

UNIT 8
CSR ADDRESS (O) 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
7. ANSWER THE 'CHANGE SW' QUESTION WITH 'N'

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 'IBR' 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', 'IBR' OR 'IXR' 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

BELOW ARE SAMPLE ERROR MESSAGES. EACH ERROR MESSAGE REPRESENTS DIFFERENT TYPES OF ERRORS DETECTED BY THIS DIAGNOSTIC.

## ERROR MESSAGE EXAMPLE 1

THIS ERROR IS INDICATIVE OF AN INCORRECT REGISTER OR STATUS WORD RETURNED TO THE DIAGNOSTIC. THE FIRST PART DEFINES THE TEST FUNCTION AND UNIT THAT FAILED. THE SECOND PART PROVIDES THE REGISTER BITS AND THEIR MNEMONICS FOR THE INCORRECT REGISTER OR STATUS WORDS. THE THIRD PART IS THE EXPECTED AND RECEIVED DATA.

TST: 016 FIFO EXERCISER TEST  
CVTSB HRD ERR 01610 ON UNIT 00 TST 016 SUB 002 PC: 040624  
FIFO STATUS (IN WORD 9) INCORRECT AFTER WRITE FIFO

TAPE BUS SIGNALS IN WORD #8: - DESIGNATOR <BIT #>  
PARERR<15> IEOT <12> IFMK <9> IRDY<6> IRWD<2>  
IRESV2<14> IIDENT<11> IHER <8> IONL<5> IFBY<1>  
IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>

TAPE BUS SIGNALS IN WORD #9:  
DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>

MESSAGE BUFFER ADDRESS = 047352

MESSAGE BUFFER CONTENTS:

WORD #0	EXPD: 100020	RECV: 100020	XOR: 000000
WORD #1	EXPD: 000012	RECV: 000012	XOR: 000000
WORD #2	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #3	EXPD: 000010	RECV: 000010	XOR: 000000
WORD #4	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #5	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #6	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #7	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #8	EXPD: 070217	RECV: 070217	XOR: 000000
WORD #9	EXPD: 000074	RECV: 000034	XOR: 000040

## ERROR MESSAGE EXAMPLE 2

THIS ERROR SHOWS A FATAL FUNCTION ERROR FROM THE TAPE DRIVE. IN THIS INSTANCE A UNRECOVERABLE ERROR OCCURED WHICH INDICATES THAT THE CONTROLLER MAY BE DEFECTIVE.

CVTSB HRD ERR 00159 ON UNIT 00 TST 001 SUB 005 PC: 026202  
TSSR NOT CORRECT AFTER SPACE RECORDS COMMAND

TSSR = 100214

TSSR BITS SET: SC,SSR

TERMINATION CLASS CODE = UNRECOVERABLE ERROR

PACKET ADDRESS = 026420

PACKET WORD # = 140010

PACKET WORD # = 000010

PACKET WORD # = 000000

PACKET WORD # = 000024



## ERROR MESSAGE EXAMPLE 3

THIS ERROR SHOWS THAT THE MOTION BIT DID NOT GET SET WHILE DOING A REWIND WITH EXTENDED FEATURES MODE ENABLED.

CVTSB HRD ERR 00121 ON UNIT 00 TST 001 SUB 002 PC: 023306  
MOT BIT (XST0) NOT SET DURING REWIND (EXTENDED FEATURES MODE)  
EXPD: 000312 RECV: 000112 XOR: 000200

## 4.0 PERFORMANCE AND PROGRESS REPORTS

AT THE END OF EACH PASS, THE PASS COUNT IS GIVEN ALONG WITH THE TOTAL NUMBER OF ERRORS REPORTED SINCE THE DIAGNOSTIC WAS STARTED. THE 'EOP' SWITCH CAN BE USED TO CONTROL HOW OFTEN THE END OF PASS MESSAGE IS PRINTED. SECTION 2.2 DESCRIBES SWITCHES.

## SUCCESSFUL RUN EXAMPLE (PDP-11/23)

DR>STA/FLA:PNT:HOE:UAM

UNITS (D) ? 1

UNIT 0

DEVICE ADDRESS (0) 172520 ? <CR>

VECTOR (0) 224 ? <CR>

CHANGE SW (L) ? N<CR>

THE ABOVE COMMAND WILL START THE DIAGNOSTIC. THE COMMAND HAS THREE SWITCHES ON WHICH ARE 'PRINT EACH TEST NBR AS EXECUTED', 'HALT ON ERROR' AND 'RUN IN UNATTENDED MODE'.

NOTE: THE UAM FLAG SHOULD BE USED TO PREVENT TEST 10-12 FROM BEING EXECUTED UNLESS THE OPERATOR WANTS THESE SPECIFIC TEST.

TST: 001 INITIALIZE #3 TEST  
TST: 002 BASIC WRITE SUBSYSTEM MEMORY TEST  
TST: 003 DMA MEMORY ADDRESSING TEST  
TST: 004 RAM EXERCISER TEST  
TST: 005 FIFO EXERCISER TEST  
TST: 006 STATIC TRANSPORT BUS CHECK  
TST: 007 TRANSPORT BUS INTERFACE CHECK VIA LOOPBACK TEST  
TST: 008 READ/WRITE DATA PARITY CHECK TEST  
TST: 009 MISCELLANEOUS LOGIC CHECKS TEST  
TST: 010 STAND-ALONE MANUAL INTERVENTION NOT EXECUTED TEST  
TST: 011 STAND-ALONE CONFIGURATION TYPEOUT NOT EXECUTED TEST  
TST: 012 STAND-ALONE SCOPE LOOPS NOT EXECUTED TEST

0 ERRORS

NOTE: THE DIAGNOSTIC WILL RUN CONTINUOUSLY UNLESS A PASS LIMIT HAS BEEN SPECIFIED WITH THE "/PASS:" SWITCH.

PROGRAM RUN TIMES

THE AVERAGE RUN TIMES OF THE PROGRAM ARE LISTED BELOW. THESE FIGURES ARE TO BE USED AS A GUIDE. THE TIMING WAS DONE ON A PDP-11/23 PROCESSOR WITH A LA-34 CONSOLE.

THE PROGRAM RUNS IN TWO MODES; NO ITERATIONS AND DEFAULT MODE. IN THE NO ITERATIONS MODE, EACH TEST IS RUN ONCE, WITH NO ITERATIONS. IN THE DEFAULT MODE EACH TEST IS REPEATED BY THE NUMBER OF TIMES INDICATED BY THE ITERATION COUNT. NO ITERATIONS MODE IS SELECTED BY ANSWERING THE INHIBIT ITERATIONS QUESTION WITH A "Y" (YES).

TEST NUMBER	N/I SECS.	ITER SECS	DEF SECS.
1	15	50	35
2	1	6	5
3	1	1	0
4	110	540	430
5	1	10	9
6	10	120	110
7	1	3	2
8	15	15	12
9	17	17	13

THE TIMES REQUIRED TO RUN TESTS 1 THROUGH 9 IN ONE COMMAND:

Q.V. 2 MINS 19 SECONDS  
 DEFAULT 11 MINS 35 SECONDS

5.0 DEVICE INFORMATION TABLES

WHENEVER THE PROGRAM IS STARTED, VIA THE STA(RT) COMMAND, THE SUPERVISOR REQUESTS THE FOLLOWING P-TABLES PARAMETER CHANGES:

CHANGE HW (L) ?

# UNITS (D) ? <ENTER THE NUMBER OF M7196 CONTROLLERS PRESENT TO BE TESTED>

UNIT 0

DEVICE ADDRESS (O) 172520 ? <ENTER THE ADDRESS OF THE TSBA/TSDB REGISTER>

VECTOR (O) 224 ? <ENTER ADDRESS OF INTERRUPT VECTOR>

THE ADDRESS AND VECTOR QUESTIONS WILL BE ASKED FOR EACH OF THE NUMBER OF

UNITS (CONTROLLERS) SPECIFIED IN THE '# UNITS?' QUESTION. LOGICAL UNIT NUMBERS ARE ASSIGNED IN ORDER, BEGINNING AT 0. UP TO FOUR UNITS CAN BE SELECTED FOR TESTING.

IN ADDITION, ON A START, RESTART OR CONTINUE THE SUPERVISOR REQUESTS CHANGES TO THE SOFTWARE OPERATING PARAMETERS, AS FOLLOWS:

CHANGE SW (L) ?

6.0 TEST SUMMARIES

TEST 1: INITIALIZE AFTER WRITE CHARACTERISTICS

TEST DESCRIPTION:

THIS TEST VERIFIES THAT A HARDWARE INITIALIZE COMMAND INVOKED AFTER A WRITE CHARACTERISTICS COMMAND SETS UP THE COMMAND, MESSAGE AND CHARACTERISTIC IMAGE BLOCKS IN THE CONTROLLER RAM CORRECTLY.

TEST 2: BASIC WRITE SUBSYSTEM MEMORY COMMAND

THIS TEST VERIFIES THAT THE WRITE SUBSYSTEM MEMORY COMMAND WITH A BSEL0 SELECT CODE OF 0 (NO-OP) EXECUTES CORRECTLY. IT ALSO VERIFIES THAT A WRITE SUBSYSTEM MEMORY COMMAND WITH A NON-ZERO MODE FIELD IS REJECTED. THE TEST FURTHER VERIFIES MICROPROGRAM COMMAND DECODING AND HANDLING SEQUENCES.

TEST 3: DMA MEMORY ADDRESSING

THIS TEST VERIFIES THAT THE CONTROLLER CAN PROPERLY ADDRESS AND ACCESS ALL AVAILABLE CPU MEMORY (OTHER THAN THAT OCCUPIED BY THE DIAGNOSTIC AND DIAGNOSTIC SUPERVISOR CODE) FOR BOTH READING (DATI) AND WRITING (DATO). VERIFIED ARE THE LSI-11 BUS DRIVERS FOR ALL AVAILABLE ADDRESS LINES. UP TO THIS POINT ONLY 16 BITS HAVE BEEN USED FOR DMA TRANSFERS.

\*\*\*\*\*

CAUTION

THE LSI BUS DRIVERS FOR ALL AVAILABLE ADDRESS LINES ARE ONLY CHECKED WHEN RUNNING ON A 11/23B SYSTEM WITH MORE THAN 128K WORDS OF MEMORY!

\*\*\*\*\*

TEST 4: RAM EXERCISER TEST

THIS TEST USES THE READ AND WRITE RAM (BOTH SINGLE AND 256 LOCATIONS) SELECT CODES OF THE WRITE SUBSYSTEM MEMORY COMMAND TO EXERCISE THE CONTROLLER'S RAM MEMORY AND DMA LOGIC

TEST 5: EXTENDED FEATURES SWITCH AND TIMERS A,B

TEST DESCRIPTION:

THIS TEST VERIFIES THE INVERT EXTENDED FEATURES FUNCTION CAN LOGICALLY INVERT THE EXTENDED FEATURES SWITCH AND THAT THE INTERNAL TIMERS A AND B OPERATE CORRECTLY.

TEST 6: FIFO EXERCISER

TEST DESCRIPTION:

THIS TEST USES THE WRITE SUBSYSTEM MEMORY COMMAND TO VERIFY THE CONTROLLER'S FIFO AND ASSOCIATED STATUS AND CONTROL LOGIC.

TEST 7: STATIC TRANSPORT BUS INTERFACE TEST

TEST DESCRIPTION:

WRITE TO TSSR REGISTER TO SOFT INITIALIZE THE CONTROLLER  
DO WRITE CHARACTERISTICS TO CHECK FOR EXTENDED FEATURES SWITCH  
IF EXTENDED FEATURES HARDWARE SWITCH CLEAR THEN:  
DO WRITE SUBSYSTEM WRITE MISCELLANEOUS TO SET EXTENDED FEATURES.  
DO WRITE CHARACTERISTICS TO SELECT RESERVED UNIT 7  
DO A WRITE SUBSYSTEM READ STATUS  
IF ANY TRANSPORT INTERFACE SIGNALS ARE ASSERTED THEN PRINT ERROR

TEST 8: TRANSPORT BUS INTERFACE LOOPBACK TEST

TEST DESCRIPTION:

THIS TEST VERIFIES THE CONTROLLER'S TRANSPORT BUS DRIVERS, RECEIVERS, AND SIGNAL LOOPBACK LOGIC. NOTE THAT THE STATIC TRANSPORT BUS TEST MUST HAVE RUN CORRECTLY FOR THIS TEST TO PROVIDE MEANINGFUL RESULTS.

TEST 9: READ/WRITE DATA PARITY TEST

TEST DESCRIPTION:

THIS TEST VERIFIES THAT THE WRITE DATA PARITY GENERATOR AND THE READ DATA PARITY CHECKER OPERATE PROPERLY. THE TRANSPORT BUS SIGNAL LOOPBACK MODE IS ENABLED AND A SET WRONG PARITY FUNCTION IS EXECUTED. THEN VARIOUS WRITE SUBSYSTEM MEMORY FUNCTIONS ARE PERFORMED TO WRITE DATA TO AND FROM THE FIFO IN LOOPBACK MODE. THE PROGRAM THEN CHECKS TO INSURE A READ DATA PARITY ERROR OCCURRED.  
A RESET FIFO IS DONE AND THE READ DATA PARITY



ERROR BIT IS AGAIN TESTED TO INSURE IT CLEARED.  
FINALLY A CLEAR WRONG PARITY FUNCTION IS DONE  
AND IT IS VERIFIED THE DATA WORD CAN PASS IN LOOPBACK  
MODE WITHOUT SETTING READ DATA PARITY ERROR.

#### TEST 10: MANUAL INTERVENTION

THE MANUAL INTERVENTION TEST IS A STANDALONE ROUTINE (NOT REALLY A "TEST") THAT ALLOWS THE OPERATOR TO CHECK OUT VARIOUS ELEMENTS AND FUNCTIONS OF THE SUBSYSTEM THAT CANNOT BE MANIPULATED BY THE PROGRAM ALONE. WHEN THIS ROUTINE IS STARTED, IT FIRST PRINTS OUT A MENU OF SELECTABLE SUBTESTS AND THEN WAITS FOR THE OPERATOR TO TYPE IN A SELECTION CODE. THE ONLY WAYS TO EXIT THIS ROUTINE AND RETURN TO THE DIAGNOSTIC SUPERVISOR ARE BY TYPING <CTRL-C> OR SELECTING CODE 6. SELECTION CODES AND SUBROUTINES ARE:

CODE	ROUTINE
0	HELP. PRINTS THIS MENU.
1	TURN ON ALL M7196 LED INDICATORS
2	TURN OFF ALL M7196 LED INDICATORS
3	OFFLINE/OFFLINE ATTENTION TEST
4	WRITE-PROTECT TEST
5	PRINT EXTENDED TRANSPORT STATUS
6	EXIT (RETURN TO SUPERVISOR)

#### TEST 11: CONFIGURATION TYPEOUT

THIS IS A STANDALONE ROUTINE THAT PRINTS OUT ON THE CONSOLE TERMINAL THE CONFIGURATION OF THE M7196 MODULE AND TSV05 SUBSYSTEM. SPECIFICALLY, THE FOLLOWING INFORMATION IS PRESENTED:

- 1.0 STATE OF THE EXTENDED FEATURES SWITCH ON THE M7196: ON (EXTENDED FEATURES ENABLED) OR OFF (EXTENDED FEATURES DISABLED),
- 2.0 STATE OF THE BUFFERING ENABLE SWITCH ON THE M7196: ON (BUFFERING ENABLED) OR OFF (BUFFERING DISABLED),
- 3.0 MICROCODE REVISION LEVEL OF THE M7196,
- 4.0 NUMBER OF TAPE TRANSPORTS CONNECTED TO THE CONTROLLER,
- 5.0 UNIT SELECT CODE AND STATE (ONLINE/OFFLINE, WRITE ENABLED/PROTECTED) OF EACH CONNECTED TRANSPORT. IN ADDITION, THE PROGRAM WILL INDICATE, FOR EACH ON-LINE TRANSPORT, WHETHER OR NOT IT IS EQUIPPED WITH THE EXTENDED TAPE STATUS READOUT FEATURE.

#### TEST 12: SCOPE LOOPS

THIS IS A STANDALONE ROUTINE PROVIDING A NUMBER OF TIGHT "SCOPE LOOPS" USEFUL FOR DEBUGGING BASIC REGISTER ACCESS PROBLEMS WITH THE M7196 MODULE. THESE SCOPE LOOPS CAN BE USED WHEN THE NORMAL "LOOP ON ERROR" OR "LOOP ON TEST (SUBTEST)" FACILITIES DON'T

SEEM TO ALLOW THE OPERATOR TO ZERO IN A PROBLEM IN THE EARLY TESTS (I.E, THE HARDWARE MAY NOT BE RESPONDING TO A REGISTER ACCESS, CAUSING A BUS ERROR TRAP, EVEN THOUGH THE DEVICE ADDRESS SELECTED BY THE PROGRAM MATCHES THE CONFIGURATION SET UP IN THE HARDWARE DIP SWITCHES). THE FOLLOWING MENU OF SCOPE LOOPS ARE AVAILABLE:

CODE	SCOPE LOOP
0	HELP. PRINT THIS MENU.
1	TSBA READ ACCESS
2	TSSR READ ACCESS
3	INITIALIZE (TSSR WRITE ACCESS)
4	TSDB HIGH BYTE WRITE ACCESS
5	TSDB LOW BYTE WRITE ACCESS
6	TSDB MAINTENANCE-MODE WORD WRITE ACCESS
7	TSDBX (TSSR HIGH BYTE) WRITE ACCESS (EXTENDED FEATURES SWITCH MUST BE ON TO USE SELECTION CODE 7)
8	EXIT (RETURN TO SUPERVISOR)

FOR SCOPE LOOPS THAT WRITE INTO REGISTERS, THE PROGRAM PROMPTS THE OPERATOR FOR THE DATA TO BE WRITTEN, LIMITS ON THE DATA PATTERNS ARE 0-377. TYPING <RETURN> CAUSES AN EXIT FROM THE SCOPE LOOP BACK TO MENU LEVEL.

#### 7.0 MAINTENANCE HISTORY

REVISION A - MARCH 1982

```

2          .TITLE TSV2 - PROGRAM HEADER
3          .SBTTL PROGRAM HEADER
4
10         .MCALL SVC
11 000000 SVC ; INITIALIZE SUPERVISOR MACROS
12         .ENABLE LC
13         .NLIST BEX,CND
19 000000 .ENABL ABS,AMA
20         =2000
21 002000 BGNMOD V2
002000
TSV2.:
:++
: THE PROGRAM HEADER IS THE INTERFACE BETWEEN
: THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
:--
22
23
24
25
26
27
28
29 002000 POINTER BGNSW,BGNSFT,BGNAU,BGNDU,BGNRPT
30 002000 HEADER CVTSB,A,0,655.,0
002000 L$NAME:: ;DIAGNOSTIC NAME
002000 .ASCII /C/
002001 .ASCII /V/
002002 .ASCII /T/
002003 .ASCII /S/
002004 .ASCII /B/
002005 .BYTE 0
002006 .BYTE 0
002007 .BYTE 0
002010 L$REV:: ;REVISION LEVEL
002010 101 .ASCII /A/
002011 L$DEPO:: ;0
002011 060 .ASCII /O/
002012 L$UNIT:: ;NUMBER OF UNITS
002012 000000 .WORD 0
002014 L$TIML:: ;LONGEST TEST TIME
002014 001217 .WORD 655.
002016 L$HPCP:: ;PTR. TO H.W. QUES.
002016 101342 .WORD L$HARD
002020 L$SPCP:: ;PTR. TO S.W. QUES.
002020 101474 .WORD L$SOFT
002022 L$HPTP:: ;PTR. TO DEF. H.W. PTABLE
002022 002156 .WORD L$HW
002024 L$SPTP:: ;PTR. TO S.W. PTABLE
002024 002166 .WORD L$SW
002026 L$LADP:: ;DIAG. END ADDRESS
002026 102004 .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 000000 .WORD 0
002036 L$APT:: ;APT EXPANSION
002036 000000 .WORD 0
002040 L$DTP:: ;PTR. TO DISPATCH TABLE
002040 002124 .WORD L$DISPATCH

```

002042		LSPRIO::		;DIAGNOSTIC RUN PRIORITY
002042	000000	.WORD	0	
002044		LSENV1::		;FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000	.WORD	0	
002046		LSEXP1::		;EXPANSION WORD
002046	000000	.WORD	0	
002050		LSMREV::		;SVC REV AND EDIT #
002050	003	.BYTE	CSREVISION	
002051	003	.BYTE	CSREDIT	
002052		LSEF::		;DIAG. EVENT FLAGS
002052	000000	.WORD	0	
002054	000000	.WORD	0	
002056		LSSPC::		
002056	000000	.WORD	0	
002060		LSDEVP::		; POINTER TO DEVICE TYPE LIST
002060	003402	.WORD	LSDVTYP	
002062		LSREPP::		;PTR. TO REPORT CODE
002062	022664	.WORD	LSRPT	
002064		LSEXP4::		
002064	000000	.WORD	0	
002066		LSEXP5::		
002066	000000	.WORD	0	
002070		LSAUT::		;PTR. TO ADD UNIT CODE
002070	022352	.WORD	LSAU	
002072		LSDUT::		;PTR. TO DROP UNIT CODE
002072	022450	.WORD	LSDU	
002074		LSLUN::		;LUN FOR EXERCISERS TO FILL
002074	000000	.WORD	0	
002076		LSDESP::		;POINTER TO DIAG. DESCRIPTION
002076	003410	.WORD	LSDESC	
002100		LSLOAD::		;GENERATE SPECIAL AUTOLOAD EMT
002100	104035	EMT	ESLOAD	
002102		LSETP::		;POINTER TO ERR_TBL
002102	000000	.WORD	0	
002104		LSICP::		;PTR. TO INIT CODE
002104	021556	.WORD	LSINIT	
002106		LSCCP::		;PTR. TO CLEAN-UP CODE
002106	022636	.WORD	LSCLEAN	
002110		LSACP::		;PTR. TO AUTO CODE
002110	022556	.WORD	LSAUTO	
002112		LSPRT::		;PTR. TO PROTECT TABLE
002112	021546	.WORD	LSPROT	
002114		LSTEST::		;TEST NUMBER
002114	000000	.WORD	0	
002116		LSDLY::		;DELAY COUNT
002116	000000	.WORD	0	
002120		LSHIME::		;PTR. TO HIGH MEM
002120	000000	.WORD	0	



32  
33  
34  
35  
36  
37  
38  
39

.SBTTL DISPATCH TABLE

:++  
: THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.  
: IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.  
:--

002122  
002122 000014  
002124  
002124 023446  
002126 024426  
002130 026420  
002132 031774  
002134 034564  
002136 040356  
002140 050470  
002142 051750  
002144 062576  
002146 066646  
002150 074510  
002152 077662

DISPATCH 12  
.WORD 12  
L\$DISPATCH::  
.WORD T1  
.WORD T2  
.WORD T3  
.WORD T4  
.WORD T5  
.WORD T6  
.WORD T7  
.WORD T8  
.WORD T9  
.WORD T10  
.WORD T11  
.WORD T12

40

```
42          .SBTTL  DEFAULT HARDWARE P-TABLE
43
44          :++
45          : THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
46          : THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
47          : IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
48          :--
49 002154      BGNHW  DFPTBL      ;DEFAULT HARD-P-TABLE
      002154 000003      .WORD  L10000-L$HW/2
      002156      L$HW::
      002156      DFPTBL::
50
51 002156 172520      .WORD  172520      ; 1ST (OF 2) REGISTERS.
52 002160 000224      .WORD  224        ; INTERRUPT VECTOR
53 002162 000200      .WORD  PRI04      ; INTERRUPT PRIORITY.
54 002164      ENDHW
      002164      L10000:
```

```
56                                     .SBTTL SOFTWARE P-TABLE
57
58                                     :++
59                                     : THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
60                                     : PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
61                                     :--
62 002164                                BGNSW  SFPTBL
   002164 000004                          .WORD L10001-LSSW/2
   002166
   002166
63
64 002166 000000                          TRANSTST:: .WORD 0 ; ENABLE TEST OF TRANSPORT(S) IF =1
65 002170 000000                          NOITS:: .WORD 0 ; INHIBIT ITERATION OPTION.
66
67
68 002172 000017                          LERRMAX:: .WORD 15. ; ... 0 = ITERATE.
69 002174 000310                          GERRMAX:: .WORD 200. ; ...NZ = INHIBIT ITERATE.
70 002176
   002176
71
72 002176                                ENDSW
                                       L10001:
                                       ENDMOD
```

7  
8  
13  
19  
20 002176  
002176  
21  
22  
23  
24  
25  
26  
27  
28  
29  
33 002176

.TITLE TSV3 - GLOBAL AREAS  
.SBTTL GLOBAL EQUATES SECTION

BGNMOD TSV3  
TSV3::

.SBTTL GLOBAL EQUATES SECTION

:+  
: THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT  
: ARE USED IN MORE THAN ONE TEST.  
:--

EQUALS ; GET STANDARD EQUATES.

: 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
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    HOE== 100000
```

34  
35 002176

```
          ; DEFINE MEMORY MANAGEMENT REGISTERS
          ;
          .SBTTL  KT11 MEMORY MANAGEMENT DEFINITIONS
          ;*KT11 VECTOR ADDRESS
000250    MMVEC= 250
          ;*KT11 STATUS REGISTER ADDRESSES
177572    SR0= 177572
177574    SR1= 177574
177576    SR2= 177576
172516    SR3= 172516
          .IF NB
          ;*USER 'I' PAGE DESCRIPTOR REGISTERS
UIPDR0= 177600
UIPDR1= 177602
UIPDR2= 177604
UIPDR3= 177606
UIPDR4= 177610
UIPDR5= 177612
UIPDR6= 177614
UIPDR7= 177616
          .IF NB
          ;*USER 'D' PAGE DESCRIPTOR REGISTERS
UDPDR0= 177620
UDPDR1= 177622
UDPDR2= 177624
UDPDR3= 177626
UDPDR4= 177630
UDPDR5= 177632
UDPDR6= 177634
UDPDR7= 177636
```

```
.ENDC
;*USER 'I' PAGE ADDRESS REGISTERS
UIPAR0= 177640
UIPAR1= 177642
UIPAR2= 177644
UIPAR3= 177646
UIPAR4= 177650
UIPAR5= 177652
UIPAR6= 177654
UIPAR7= 177656
. IF NB
;*USER 'D' PAGE ADDRESS REGISTERS
UDPAR0= 177660
UDPAR1= 177662
UDPAR2= 177664
UDPAR3= 177666
UDPAR4= 177670
UDPAR5= 177672
UDPAR6= 177674
UDPAR7= 177676
.ENDC
.ENDC
. IF NB
;*SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS
SIPDR0= 172200
SIPDR1= 172202
SIPDR2= 172204
SIPDR3= 172206
SIPDR4= 172210
SIPDR5= 172212
SIPDR6= 172214
SIPDR7= 172216
. IF NB
;*SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS
SDPDR0= 172220
SDPDR1= 172222
SDPDR2= 172224
SDPDR3= 172226
SDPDR4= 172230
SDPDR5= 172232
SDPDR6= 172234
SDPDR7= 172236
.ENDC
;*SUPERVISOR 'I' PAGE ADDRESS REGISTERS
SIPAR0= 172240
SIPAR1= 172242
SIPAR2= 172244
SIPAR3= 172246
SIPAR4= 172250
SIPAR5= 172252
SIPAR6= 172254
SIPAR7= 172256
. IF NB
;*SUPERVISOR 'D' PAGE ADDRESS REGISTERS
SDPAR0= 172260
SDPAR1= 172262
SDPAR2= 172264
```

```
SDPAR3= 172266
SDPAR4= 172270
SDPAR5= 172272
SDPAR6= 172274
SDPAR7= 172276
.ENDC
.ENDC
;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS
172300 KIPDR0= 172300
172302 KIPDR1= 172302
172304 KIPDR2= 172304
172306 KIPDR3= 172306
172310 KIPDR4= 172310
172312 KIPDR5= 172312
172314 KIPDR6= 172314
172316 KIPDR7= 172316
.IF NB
;*KERNEL 'D' PAGE
DESCRIPTOR REGISTERS
KDPDR0= 172320
KDPDR1= 172322
KDPDR2= 172324
KDPDR3= 172326
KDPDR4= 172330
KDPDR5= 172332
KDPDR6= 172334
KDPDR7= 172336
.ENDC
;*KERNEL 'I' PAGE ADDRESS REGISTERS
172340 KIPAR0= 172340
172342 KIPAR1= 172342
172344 KIPAR2= 172344
172346 KIPAR3= 172346
172350 KIPAR4= 172350
172352 KIPAR5= 172352
172354 KIPAR6= 172354
172356 KIPAR7= 172356
.IF NB
;*KERNEL 'D' PAGE ADDRESS REGISTERS
KDPAR0= 172360
KDPAR1= 172362
KDPAR2= 172364
KDPAR3= 172366
KDPAR4= 172370
KDPAR5= 172372
KDPAR6= 172374
KDPAR7= 172376
.ENDC
```

```

40          .SBTTL  TSV05 REGISTER AND PACKET DEFINITIONS
41
42          :
43          : SOME GENERAL EQUATES.
44          :
45
46          000004  ERRVEC==      4          ; POINTER TO ERROR VECTOR FOR BUS TIME OUT.
47          000060  TTIVEC==     60          ; INTERRUPT VECTOR FOR CONSOLE INPUT
48          177560  TTICSR==    177560       ; BUS ADDRESS OF CONSOLE INPUT
49          177562  TTIBFR==    177562       ; CONSOLE INPUT DATA BUFFER
50          177520  BDVPCR==    177520       ; BDV11 PAGE CONTROL REGISTER
51
52          :+
53          :BIT DEFINITIONS FOR TSSR REGISTER
54          :-
55
56          100000  SC=      BIT15          ;SPECIAL CONDITION
57          040000  BIE=     BIT14          ;BUS INTERFACE ERROR
58          020000  SCE=     BIT13          ;SANITY CHECK ERROR
59          010000  RMR=     BIT12          ;MODIFICATION REFUSED
60          004000  NXM=     BIT11          ;NONEXISTANT MEMORY ERROR
61          002000  NBA=     BIT10          ;NEED BUFFER ADDRESS
62          001400  HIADDR= BIT9!BIT8      ;EXTENDED ADDRESS BITS
63          000200  SSR=     BIT7           ;SUB SYSTEM READY
64          000100  OFL=     BIT6           ;OFF LINE BIT
65          000060  FATERR= BIT4!BIT5      ;FATAL TERMINATION ERROR CODES
66          000016  TERCLS= BIT3!BIT2!BIT1 ;TERMINATION CODES
67
68          :+
69          :BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 0
70          : (XST0)
71          :
72          :
73          :
74          :-
75
76          100000  XSOTMK= BIT15          ;TAPE MARK DETECTED
77          040000  XSORLS= BIT14          ;RECORD LENGTH SHORT
78          020000  XSOLET= BIT13          ;LOGICAL END OF TAPE
79          010000  XSORLL= BIT12          ;RECORD LENGTH 'ONG
80          004000  XSOWLE= BIT11          ;WRITE LOCK ERROR
81          002000  XSONEF= BIT10          ;NON EXECUTABLE FUNCTION
82          001000  XSOILC= BIT9           ;ILLEGAL COMMAND
83          000400  XSOILA= BIT8           ;ILLEGAL ADDRESS
84          000200  XSOMOT= BIT7           ;TAPE IN MOTION
85          000100  XSOONL= BIT6           ;TRANSPORT ON LINE
86          000040  XSOIE=  BIT5           ;INTERRUPT ENABLE
87          000020  XSOVCK= BIT4           ;VOLUME CHECK BIT
88          000010  XSOPED= BIT3           ;PHASE ENCODED DRIVE
89          000004  XSOWLK= BIT2           ;WRITE LOCKED
90          000002  XSOBOT= BIT1           ;BEGINNING OF TAPE
91          000001  XSOEOT= BIT0           ;END OF TAPE
92
93          :+
94          :BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 1
95          : (XST1)
96
    
```



```

97
98      100000      :-
99      040000      X1.DLT = BIT15      ;DATA LATE
100     020000      X1.SPARE= BIT14      ;NOT USED
101     017375      X1.COR = BIT13      ;CORRECTABLE DATA ERROR
102     000400      X1.MBZ = BIT12+BIT11+BIT10+BIT9+BIT7+BIT6+BIT5+BIT4+BIT3+BIT2+BIT0 ;ALWAYS 0
103     000002      X1.RBP = BIT8      ;READ BUS PARITY ERROR
104                                     X1.UNC = BIT1      ;UNCORRECTABLE DATA OR HARD ERROR
105
106     :-
107     ;+
108     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 2
109     ;(XST2)
110     :-
111     100000      X2.OPM = BIT15      ;OPERATION IN PROGRESS (TAPE MOVING)
112     040000      X2.RCE = BIT14      ;RAM CHECKSUM ERROR
113     035400      X2.SPARE= BIT13+BIT12+BIT11+BIT9+BIT8 ;NOT USED BY TSV05 (ALWAYS=0)
114     002000      X2.WCF = BIT10      ;WRITE CLOCK FAILURE (FIFO NOT EMPTIED BY TRANSPORT)
115     000200      X2.EXTF = BIT7      ;IF WRITE CHAR CMD THEN = EXTENDED FEATURES ENABLED
116     000100      X2.BUFE = BIT6      ;IF WRITE CHAR CMD THEN = BUFFERING ENABLED
117     000077      X2.REV = 000077    ;IF WRITE CHAR CMD THEN = MICROCODE REVISION LEVEL
118     000007      X2.UNIT = BIT2+BIT1+BIT0 ;IF GET STATUS THEN = CURRENTLY SELECTED UNIT NO.
119
120     ;+
121     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 3
122     ;(XST3)
123     :-
124     177400      X3.MDE = 177400    ;MICRO-DIAGNOSTIC ERROR CODE
125     000200      X3.SPARE= BIT7      ;NOT USED BY TSV05
126     000100      X3.OPI = BIT6      ;OPERATION INCOMPLETE
127     000040      X3.REV = BIT5      ;REVERSE
128     000020      X3.TRF = BIT4      ;TRANSPORT RESPONSE FAILURE
129     000010      X3.DCK = BIT3      ;DENSITY CHECK
130     000006      X3.MBZ =BIT2+BIT1    ;NOT USED ALWAYS 0
131     000001      X3.RIB = BIT0      ;REVERSE INTO BOT
132
133     ;+
134     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 4
135     ;(XST4)
136     :-
137     100000      X4.HSP = BIT15      ;HIGH SPEED
138     040000      X4.RCE = BIT14      ;RETRY COUNT EXCEEDED
139     020000      X4.TSM = BIT13      ;TRANSPORT SPECIAL MODE
140     017400      X4.MBZ = BIT12+BIT11+BIT10+BIT9+BIT8 ;NOT USED ALWAYS 0
141     000377      X4.WRC = 000377    ;WRITE RETRY COUNT FIELD
142
143     ;+
144     ;TSSR TERMINATION CODES (BIT 0-2)
145     :-
146
147
148     000006      TSREJ= 3*2      ;COMMAND REJECTED
149     000006      UNREC= 6      ;UNRECOVERABLE ERROR
150
151     ;+
152     ;DEVICE REGISTER OFFSETS
153

```

```

154      :-
155      :-
156      :-
157      000000      TSBA== 0
158      000000      TSDB== 0      ;TSDB/TSBA REGISTER
159      000001      TSBAH== 1
160      000001      TSDBH== 1      ;TSDB/TSBA REGISTER HIGH BYTE
161      000002      TSSR== 2      ;TSSR REGISTER
162      000003      TSSRH== 3      ;TSSR REGISTER HIGH BYTE
163
164      :-+
165      :-+ TSDB ADDRESS BIT DEFINITIONS
166      :-
167      000003      A1716 = BIT1+BIT0      ;ADDRESS BITS 17:16 ARE IN 1:0
168
169      :-+
170      :-+ COMMAND DEFINITIONS
171      :-
172      000017      P.GETSTAT      = 17      ;GET STATUS
173      000013      P.INIT          = 13      ;INITIALIZE
174      000012      P.CONTROL       = 12      ;CONTROL COMMANDS
175      000011      P.FORMAT        = 11      ;FORMAT
176      000010      P.POSITION      = 10      ;POSITION
177      000006      P.WRTSUB        = 6       ;SUBSYSTEM WRITE
178      000005      P.WRITE         = 5       ;WRITE
179      000004      P.WRTCHAR       = 4       ;WRITE CHARACTERISTICS
180      000001      P.READ          = 1       ;READ
181
182      :-+
183      :-+ COMMAND PACKET HEADER WORD BIT DEFINITIONS
184      :-
185      100000      P.ACK          = BIT15      ;BUFFER AVAIL FOR CONTROLLER
186      040000      P.CVC          = BIT14      ;CLEAR VOLUME CHECK
187      020000      P.OPP          = BIT13      ;REVERSE SEQUENCE OF DATA BITS
188      010000      P.SWB          = BIT12      ;SWAP BYTES IN MEMORY
189      007400      P.MODE         = BIT11!BIT10!BIT9!BIT8 ;EXTENDED COMMAND MODE FIELD
190      000200      P.IE           = BIT7       ;INTERRUPT ENABLE
191      000140      P.FMT= BIT6!BITS  ;PACKET HEADER TYPE (ALWAYS=0)
192      000037      P.CMD          = 37        ;MAJOR COMMAND FIELD
193
194      :-+
195      :-+ CONTROL COMMAND MODE CODES
196      :-
196      000000      PC.RELEASE     = 0*256.    ;RELEASE BUFFER
197      000400      PC.REWIND      = 1*256.    ;REWIND
198      001000      PC.NOOP        = 2*256.    ;NO-OP
199      002000      PC.IEREW       = 4*256.    ;REWIND IMMEDIATE INTERRUPT
200      002400      PC.ERASE       = 5*256.    ;SECURITY ERASE
201
202      :-+
203      :-+ CONTROLLER RAM DEFINITIONS
204      :-
205      000167      RMCHBEG = 167      ;CHARACTERISTICS IO DATA BEGIN RAM ADDRESS
206      000200      RMCHEND = 200     ;CHARACTERISTICS IO DATA END RAM ADDRESS
207      000201      RMPKTBEG= 201     ;COMMAND PACKET BEGIN RAM ADDRESS
208      000210      RMPKTEND= 210     ;COMMAND PACKET END RAM ADDRESS
209      000215      RMMMSGBEG= 215    ;MESSAGE BUFFER BEGIN RAM ADDRESS
210      000234      RMMMSGEND= 234    ;MESSAGE BUFFER END RAM ADDRESS

```

```

211      :+
212      :
213      :REGISTER DEFINITIONS IN THE MESSAGE BUFFER
214      :
215      :-
216
217      000006      XST0== 6      :EXTENDED STATUS REGISTER 0 (WORD 4)
218      000010      XST1== 8      :EXTENDED STATUS REGISTER 1 (WORD 5)
219      000012      XST2== 10     :EXTENDED STATUS REGISTER 2 (WORD 6)
220      000014      XST3== 12     :EXTENDED STATUS REGISTER 3 (WORD 7)
221      000016      XST4== 14     :EXTENDED STATUS REGISTER 4 (WORD 8)
222
223
224      :+
225      :
226      :OFFSETS TO WORD LOCATIONS IN PACKET DEFINITIONS
227      :
228      :-
229
230      000002      PKLOW  = 2      :LOW ORDER CHARACTERISTIC DATA POINTER
231      000004      PKHI   = 4      :HIGH ORDER CHARACTERISTIC DATA POINTER
232      000006      PKBCNT = 6      :NUMBER OF BYTES IN DATA PACKET
233
234      000010      EXBCNT=10     :NUMBER OF BYTES IN EXTENDED DATA PACKET
235
236      :+
237      :DATA PACKET OFFSETS FOR WRITE SUBSYSTEM COMMAND
238      :
239      :
240      000001      BSEL0  = 0      :BYTE 0
241      000002      BSEL1  = 1      :BYTE 1
242      000004      SEL2   = 2      :WORD 2
243      000004      SELDATA = 4      :WORD 3
244
245      :+
246      :BSEL0 SELECT CODES FOR WRITE SUBSYSTEM COMMAND
247      :
248      000000      PW.NOP   = 0      :NO-OP
249      000001      PW.RDRAM = 1      :READ RAM
250      000002      PW.WTRAM = 2      :WRITE RAM
251      000003      PW.RFIFO = 3      :READ FIFO
252      000004      PW.WFIFO = 4      :WRITE FIFO
253      000005      PW.RDSTAT = 5     :READ STATUS
254      000006      PW.WCTL  = 6     :WRITE TAPE CONTROL
255      000007      PW.WFMT  = 7     :WRITE TAPE FORMAT
256      000010      PW.WMISC = 10     :WRITE MISCELLANEOUS
257      000011      PW.WNPR  = 11     :WRITE NPR CONTROL
258      000020      PW.D22   = 20     :DO MICROTEST 22
259      000021      PW.D11   = 21     :DO MICROTEST 11
260      000022      PW.D13   = 22     :DO MICROTEST 13
261      000023      PW.NO1311 = 23    :DISABLE MICROTEST 11 AND 13
262      000024      PW.RDXT   = 24     :READ EXT. TAPE STATUS (NOT SUPPORTED BY ALL TRANSPORTS)
263
264      :+
265      :BSEL1 CODES FOR WRITE TAPE CONTROL
266      :
267      000200      WC.IFAD   = BIT7   :IFAD - FORMATTER ADDRESS
267      000100      WC.IOTAD = BIT6   :ITADO - TRANSPORT ADDRESS BIT 0
    
```

```

268      000040      WC.I1TAD      = BIT5      :ITAD1 - TRANSPORT ADDRESS BIT 1
269      000020      WC.I5RESV     = BIT4      :IRESV5 - RESERVED #5
270      000010      WC.IREW       = BIT3      :IREW   - REWIND
271      000004      WC.IRWU      = BIT2      :IRWU   - REWIND AND UNLOAD
272      000002      WC.IFEN      = BIT1      :IFEN   - FORMATTER ENABLE
273      000001      WC.IGO       = BIT0      :GO
274
275      :+
276      :BSEL1 CODES FOR WRITE FORMAT
277      :-
278      000200      WF.IHISP     = BIT7      :IHISP  - HIGH SPEED
279      000100      WF.IWRT     = BIT6      :IWRT   - WRITE
280      000040      WF.IREV     = BIT5      :IREV   - REVERSE
281      000020      WF.IWFM     = BIT4      :IWFM   - WRITE FILE MARK
282      000010      WF.IEDIT    = BIT3      :IEDIT  - EDIT
283      000004      WF.IERASE   = BIT2      :IERASE - ERASE
284      000002      WF.I3RESV   = BIT1      :IRESV3 - RESERVED #3
285      000001      WF.I4RESV   = BIT0      :IRESV4 - RESERVED #4
286
287
288      :+
289      :BSEL1 CODES FOR WRITE MISCELLANEOUS SUBCOMMAND
290      :-
291      000200      MS.EXT      = BIT7      :INVERT SENSE OF EXTENDED FEATURES SWITCH
292      000020      MS.RSFIFO    = BIT4      :RESET FIFO AND INPUT PARITY ERRORR
293      000010      MS.RSTAPE    = BIT3      :RESET TAPE STATUS IN 2 FLIP-FLOPS
294      000006      MS.ATTN     = BIT2!BIT1 :ATTENTION TRIGGER FIELD
295      000001      MS.RSD      = BIT0      :RESET TIMER A,B THEN DELAY TIMES IN SEL2
296
297      :+
298      : MS.ATTN SUBCODES
299      :-
300      000000      MSA.NOP     = 0*2      :NO-OP (NOTHING TRIGGERED)
301      000002      MSA.VOL     = 1*2      :SIMULATE ON-LINE/OFF-LINE TRANSISTION
302      000004      MSA.NRAM    = 2*2      :FORCE NON-FATAL RAM ERROR (FORCES ERRCODE 54)
303      000006      MSA.FRAME   = 3*2      :FORCE FATAL RAM ERROR (CAUSES SCE TO SET)
304
305      :+
306      : WRITE SUBSYSTEM WRITE NPR BSEL1 BIT DEFINITIONS
307      :-
308      000200      NP.IR       = BIT7      :INTERRUPT REQUEST (0-1 TRANSITION)
309      000100      NP.OUT      = BIT6      :TAPE DATA DIRECTION OUT (0= IN)
310      000040      NP.LOOP     = BIT5      :ENABLE TRANSPORT LOOPBACK
311      000020      NP.WRP      = BIT4      :WRITE CORRECT PARITY (SET=0 TO WRITE WRONG)
312
313      :+
314      : READ STATUS MESSAGE BUFFER BIT DEFINITIONS
315      :-
316      000200      S2.DIM      = BIT7      :WORD #9 BYTE 2 DATA IN MISS
317      000100      S2.ILW      = BIT6      :ILW H
318      000040      S2.OUTRDY    = BIT5      :OUT RDY H
319      000020      S2.INRDY    = BIT4      :IN RDY H
320      000010      S2.ATIMR    = BIT3      :TIMER A FLAG H
321      000004      S2.BTIMR    = BIT2      :TIMER B FLAG H
322      000003      S2.UNDEF    = BIT1+BIT0 : (UNDEFINED)
323      100000      S1.PARIN    = BIT15     :WORD #8 BYTE 1 PARIN H
324      040000      S1.I2RESV   = BIT14     :IRESV2
325      020000      S1.I1RESV   = BIT13     :IRESV1
326      010000      S1.IEOT     = BIT12     :IEOT L

```

325	004000	S1.IIDENT	= BIT11	:	IIDENT H
326	002000	S1.ICER	= BIT10	:	ICER H
327	001000	S1.IFMK	= BIT9	:	IFMK H
328	000400	S1.IHER	= BIT8	:	IHER H
329	000200	SO.ISPEED	= BIT7	:WORD #8 BYTE 0	ISPEED H
330	000100	SO.IRDY	= BIT6	:	IRDY L
331	000040	SO.IONL	= BIT5	:	IONL L
332	000020	SO.ILDP	= BIT4	:	ILDP L
333	000010	SO.IDBY	= BIT3	:	IDBY L
334	000004	SO.IRWD	= BIT2	:	IRWD L
335	000002	SO.IFBY	= BIT1	:	IFBY L
336	000001	SO.IFPT	= BIT0	:	IFPT L
337				:	
338				:	

```

340                .SBTTL SPECIAL MACROS AND OPDEFS.
341
342
343                :+
344                :SAVE GENERAL REGS 1 TO 5
345                :-
346
347                .MACRO SAVREG
348                JSR     R5,REGSAV
349                .ENDM
350
351                :+
352                : MACRO TO FORCE AN ERROR
353                :-
354                .MACRO FORCERROR TAG,NOTSSR
355                .NLIST
356                .IIF NDF LISTALL, .NLIST
357                .LIST
358                .IF B NOTSSR
359                MOV     TSSR(R5),R1      ;READ TSSR
360                .ENDC
361                MOV     FORCER,FORCER   ;IS FORCER SET? (LEAVE C BIT ALONE)
362                BNE     TAG             ;BR IF YES
363                .NLIST
364                .IIF NDF LISTALL, .LIST
365                .LIST
366                .ENDM
367
368                :+
369                : MACRO TO FORCE AN EXIT TO AVOID SECTION ITERATIONS
370                : WILL EXIT TO A LABEL IF FORCER IS NEGATIVE
371                : SO TO FORCE ERRORS AND EXIT ON 1 ERROR SET
372                : FORCER TO 177777
373                : TO FORCE ERRORS AND ITERATIONS SET FORCER TO 1.
374                :-
375                .MACRO FORCEEXIT TAG
376                .NLIST
377                .IIF NDF LISTALL, .NLIST
378                .LIST
379                MOV     FORCER,FORCER   ;IS FORCER NEGATIVE?
380                BMI     TAG             ;BR IF YES
381                .NLIST
382                .IIF NDF LISTALL, .LIST
383                .LIST
384                .ENDM
385                :+
386                : MACRO TO INCREMENT ERROR COUNTS
387                :-
388                .MACRO NEXT.ERRNO
389                .NLIST
390                :::: .IIF NDF LISTALL, .NLIST
391                : ERRNO=ERRNO+1
392                :::: .IIF NDF LISTALL, .LIST
393                .LIST
394                .ENDM
395
396                :+
    
```

397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420

:MACRO TO PERFORM XOR  
:-

.MACRO XOR A,B  
MOV A,-(SP)  
BIC B,(SP)  
BIC A,B  
BIS (SP)+,B  
.ENDM

000000

EN=0 ; INITIALIZE ERROR NUMBER  
.SBTTL FORCER - FORCE ERROR FLAG

:  
: THE FOLLOWING LOCATIONS MAY BE PATCHED BY THE USER  
: TO OBTAIN THE RESULTS DESCRIBED FOR EACH.  
:

002176 000000

FORCER:: 0 ; FORCE TYPE ALL HARD ERRORS (THE ONES CALLED -  
: - BY THE MACRO 'IFERROR'). AN ERROR NEED NOT -  
: - EXIST, JUST ASSUME AND TYPE THE MESSAGE.



.SBTTL GLOBAL DATA SECTION

```

422
423
424
425
426
427
428
429
430
431
432
433 002200 000000
434 002202 000000
435 002204 000000
436 002206 000000
437 002210 000224
438 002212 000200
439 002214 000000
440 002216 000000
441 002220 000000
442 002222 000000
443 002224 000000
444 002226 000000
445 002230 000000
446 002232 000000
447 002234 000000
448 002236 000000
449 002240 000000
450 002242
451 002302 000000
452 002304 000000
453 002306 000000
454 002310 000000
455 002312 000000
456 002314 000000
457 002316 000000
458 002320 000000
459 002322
460 002466
461 002632

      :++
      :THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
      :IN MORE THAN ONE TEST.
      :--

      :
      :THE FOLLOWING DATA ARE SET FOR EACH UNIT AT INIT TIME.
      :SINGLE UNIT DEFAULTS (LISTED) ARE IN THE DEFAULT P-TABLE.
      :
      EPRTSW::      .WORD      0      :PRINT SWITCH
      UNITN::      .WORD      0      :UNIT # UNDER TEST.
      QVP::        .WORD      0      :QUICK VERIFY FLAG.
      CSRADDR::    .WORD      0      :ADDRESS OF CSR FOR CURRENT DEVICE
      IVEC::       .WORD      224    :INTERRUPT VECTOR
      IPRI::       .WORD      PRI04  :INTERRUPT PRIORITY.
      TSTCNT::     .WORD      0      :NUMBER OF TESTS RUN IN THIS PASS
      LOOPCNT::    .WORD      0      :REMAINING ITERATION COUNT FOR TEST
      DEVCNT::     .WORD      0      :NUMBER OF DEVICE UNDER TEST
      FATFLG::     .WORD      0      :SET IF FATAL ERROR IS DETECTED IN TEST
      INTRECV::   .WORD      0      :SET IF TAPE INTERRUPT WAS RECEIVED
      EXTFEA::    .WORD      0      :EXTENDED FEATURES SOFTWARE SW 0=OFF;1=ON
      BENBSW::    .WORD      0      :BUFFER ENABLE SWITCH SW 0=OFF;1=ON
      EXPD::      .WORD      0      :EXPECTED RAM DATA FOR PRAMPKT ROUTINE
      RECV::      .WORD      0      :RECEIVED RAM DATA FOR PRAMPKT ROUTINE
      ERRHI::     .WORD      0      :HIGH ADDRESS MEMORY ERROR
      ERRLO::     .WORD      0      :LOW ADDRESS MEMORY ERROR
      RAMDATA::   .BLKW     16.     :DATA READ FROM RAM PACKET OR MESSAGE BUF AREA
      RAMSIZ::    .WORD      0      :RAM DATA SIZE FOR PRAMPKT ROUTINE
      RCVHIADD::  .WORD      0      :RECEIVED BUFFER HIGH ADDRESS
      RCVLOADD::  .WORD      0      :RECEIVED BUFFER LOW ADDRESS
      COUNT::    .WORD      0      :TEST COUNT PATTERN
      DATA::     .WORD      0      :TEST DATA
      TSTFLAG::   .WORD      0      :TEST FLAG WORD
      TSTPTR::    .WORD      0      :TSTBLK POINTER
      PRMNO::     .WORD      0      :PRINT ROUTINE TEMP
      EXPMSG::    .BLKB     100.    :EXPECTED MESSAGE BUFFER DATA
      RECMSG::    .BLKB     100.    :RECEIVED MESSAGE BUFFER DATA
      TMPBFR::    .BLKB     80.     :TEMPORARY STORAGE FOR PRINT

```

463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479 002752  
480 002752 000000  
481 002754 177777  
482 002756 000001  
483 002760 000002  
484 002762 000004  
485 002764 000010  
486 002766 000020  
487 002770 000040  
488 002772 000100  
489 002774 000200  
490 002776 000400  
491 003000 001000  
492 003002 002000  
493 003004 004000  
494 003006 010000  
495 003010 020000  
496 003012 040000  
497 003014 100000  
498 003016 177776  
499 003020 177775  
500 003022 177773  
501 003024 177767  
502 003026 177757  
503 003030 177737  
504 003032 177677  
505 003034 177577  
506 003036 177377  
507 003040 176777  
508 003042 175777  
509 003044 173777  
510 003046 167777  
511 003050 157777  
512 003052 137777  
513 003054 077777  
514 003056 125252  
515 003060 052525  
516 003062 003062

.SBTTL TSTBLK - TEST DATA TABLE

```

: +
: THIS TABLE CONTAINS TEST DATA USED IN SEVERAL TESTS
: IN SEQUENCE THE DATA IS:
:
:     ALL ZEROS
:     ALL ONES
:     WALKING ONES
:     WALKING ZEROS
:     ALTERNATING ONES AND ZEROS
: -

```

TSTBLK::

```

.WORD 0 ;ALL ZEROS
.WORD 177777 ;ALL ONES
.WORD BIT0 ;DATA FOR WALKING ONES
.WORD BIT1
.WORD BIT2
.WORD BIT3
.WORD BIT4
.WORD BIT5
.WORD BIT6
.WORD BIT7
.WORD BIT8
.WORD BIT9
.WORD BIT10
.WORD BIT11
.WORD BIT12
.WORD BIT13
.WORD BIT14
.WORD BIT15
.WORD ^CBIT0 ;DATA FOR WALKING ZEROS
.WORD ^CBIT1
.WORD ^CBIT2
.WORD ^CBIT3
.WORD ^CBIT4
.WORD ^CBIT5
.WORD ^CBIT6
.WORD ^CBIT7
.WORD ^CBIT8
.WORD ^CBIT9
.WORD ^CBIT10
.WORD ^CBIT11
.WORD ^CBIT12
.WORD ^CBIT13
.WORD ^CBIT14
.WORD ^CBIT15
.WORD 125252 ;ALTERNATING ONES, ZEROS
.WORD 052525 ;ALTERNATING ONES, ZERO OPPOSITE FROM ABOVE

```

TBLEND==.

```

518 .SBTTL GLOBAL ENVIRONMENT STORAGE
519
520 ; STORAGE FOR DEVICE REGISTERS
521
522 003062 000000 100000 000000 DUMMY: 0,100000,0,0 ; DUMMY DEVICE REGISTERS...
523 003072 000000 000000 000000 0,0,0,0,0,0,0,0 ; ...FOR MULTI-UNIT CHECKOUT.
524
525
526
527 003112 000000 DUFLG:: .WORD 0 ; 'DROPPED UNIT' FLAG.
528 ; INHIBITS CODE IN 'CLEAN-UP'.
529 003114 000000 NODEV:: .WORD 0 ; FLAG TO SAY NO DEVICE.
530
531 003116 000000 TEMP1:: .WORD 0 ; SOME TEMP LOCATIONS.
532 003120 000000 TEMP2:: .WORD 0
533 003122 000000 XXCOMM:: .WORD 0 ; XXDP+ COMM BLOCK POINTER.
534 003124 000000 FREE:: .WORD 0 ; 1ST FREE MEMORY ADDRESS...
535 003126 000000 FRESIZ:: .WORD 0 ; ...AND SIZE (IN WORDS).
536 003130 000000 FREEHI: .WORD 0 ; LAST WORD IN FREE SPACE
537 003132 000000 KTFLG:: .WORD 0 ; KT11, MEM AVAIL FLAG -
538 ; - .WORD 0 = <24K OR NO KT -
539 ; - NZ = >24K AND KT.
540 003134 000000 KTENABLE:: .WORD 0 ; SET BY TEST ROUTINES TO FLAG >28K UNDER TEST
541 003136 000000 NXMFLG:: .WORD 0 ; SET IF WE CAN TEST CLEARED OTHERWISE
542 003140 000000 NXMLO:: .WORD 0 ; NXM LO ADDRESS BITS
543 003142 000000 NXMHI:: .WORD 0 ; NXM HI ADDRESS BITS FOR DAL'S 16-21
544 003144 000000 T23A:: .WORD 0 ; 11/23A FLAG
545 003146 000000 T23B:: .WORD 0 ; 11/23B FLAG
546 003150 000000 T3BFLG:: .WORD 0 ; TEST 3B FLAG ^0
547 003152 002000 PST32W:: .WORD 2000 ; 32W BLOCK ADDRESS FOR 32K START
548 003154 000000 SIFLAG:: .WORD 0
549 003156 000000 BADDAT:: .WORD 0 ; ACTUAL DATA
550 003160 000000 GDDAT:: .WORD 0 ; EXPECTED DATA
551 003162 000000 LOOPFL:: .WORD 0
552 003164 CTAB:: .WORD 0 ; CONFIGURATION TABLES.
553 003164 000000 CTABM:: .WORD 0 ; CONFIG WORK.
554 003166 000000 .WORD 0
555 003170 000000 .WORD 0
556 003172 000000 .WORD 0
557 003174 177777 .WORD -1 ; END OF MEM TABLE.
558 003176
559 ; ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:
560
561 0 = UNIT NOT TESTED
562 100000 = UNIT ONLINE, NO ERRORS
563 10XXXX = UNIT ONLINE, ENCOUNTERED XXXX ERRORS
564 160000 = UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
565 160001 = UNIT DROPPED, NOT IDLE AT START
566 14XXXX = UNIT DROPPED, ENCOUNTERED XXXX ERRORS
567
568 003176 ERTABL: .BLKW 64.
569 003376 000000 ERTABE: .WORD 0
570
571 003400 000000 SKIPT: .WORD 0 ; 1=SKIP SUBTEST 0=NO SKIP OF SUBTEST
    
```

```

573          .SBTTL GLOBAL TEXT MESSAGES
574          :++
575          : THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
576          : MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
577          : MORE THAN ONE TEST.
578          :--
579
580
581
582          :+
583          :NAMES OF DEVICES SUPPORTED
584          :--
585
586 003402      DEVTYP <TSV05>
003402      LSDVTYP::
003402      .ASCIZ /TSV05/
          .EVEN

587
595          :+
596          :TEST DESCRIPTION
597          :--
598 003410      DESCRIPT <**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****>
003410      LSDESC::
003410      .ASCIZ /**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****/
          .EVEN

599
613
614
615
616
617          :+
618          :BIT TO ASCII CONVERSION FOR TSSR REGISTER
619          :--
620 003502      003542 003545 003551 TSSRBIT::      .WORD 1$,2$,3$,4$,5$,6$,7$,8$
621 003522      003603 003607 003613      .WORD 9$,10$,11$,12$,13$,14$,15$,16$
622 003542      123 103 000 1$: .ASCIZ 'SC'
623 003545      102 111 105 2$: .ASCIZ 'BIE'
624 003551      123 103 105 3$: .ASCIZ 'SCE'
625 003555      122 115 122 4$: .ASCIZ 'RMR'
626 003561      116 130 115 5$: .ASCIZ 'NXM'
627 003565      116 102 101 6$: .ASCIZ 'NBA'
628 003571      102 111 124 7$: .ASCIZ 'BIT9'
629 003576      102 111 124 8$: .ASCIZ 'BIT8'
630 003603      123 123 122 9$: .ASCIZ 'SSR'
631 003607      117 106 114 10$: .ASCIZ 'OFL'
632 003613      102 111 124 11$: .ASCIZ 'BIT5'
633 003620      102 111 124 12$: .ASCIZ 'BIT4'
634 003625      102 111 124 13$: .ASCIZ 'BIT3'
635 003632      102 111 124 14$: .ASCIZ 'BIT2'
636 003637      102 111 124 15$: .ASCIZ 'BIT1'
637 003644      102 111 124 16$: .ASCIZ 'BIT0'
638          .EVEN
639 003652      124 123 123 SFIERR: .ASCIZ 'TSSR ERROR AFTER SOFT INIT'
640 003705      124 123 123 SFHERR: .ASCIZ 'TSSR ERROR AFTER BUS RESET'
641 003740      040 040 116 NXR: .ASCIZ / NON-EXISTANT DEVICE REGISTER/
642 003777      045 101 040 NXR: .ASCIZ /% ADDRESS: %06/
643 004020      045 101 040 TSSX: .ASCII /% TSBA,TSSR EXP'D: %06%,%06%/

```

```
644 004060 045 101 040 .ASCIZ /%A TSBA,TSSR REC'D: %06%A,%06/
645 004117 045 116 045 FUSI: .ASCII /%N%A/
646 004123 040 040 125 USI: .ASCIZ / UNEXPECTED INTERRUPT/
647 004152 040 040 111 NSI: .ASCIZ / INTERRUPT EXPECTED, NOT RECEIVED/
648 004215 045 116 045 FNOINTR: .ASCII /%N%A/
649 004221 040 040 116 NOINTR: .ASCIZ / NO INTERRUPT WAS GENERATED/
650 004256 040 040 111 IFAULT: .ASCIZ / INTERRUPT FAULT/
651 004300 045 101 040 INTX: .ASCIZ /%A CPU PC: %06%A TSBA: %06/
652 004335 040 040 042 NOINIT: .ASCIZ / 'BUS-INIT' DIDN'T INITIALIZE CONTROLLER/
653 004407 040 040 042 NSINIT: .ASCIZ / 'SOFT-INIT' DIDN'T INITIALIZE THE DPU/
654 004457 040 040 042 BRINIT: .ASCIZ / 'BUS-RESET' DIDN'T INITIALIZE THE DPU/
655
656 004527 000 NUL: .ASCIZ //
657 004530 045 116 000 NULCR: .ASCIZ /%N/
658 004533 045 101 040 EXPGOT: .ASCIZ /%A EXP'D: %06%A, REC'D: %06/
659 004567 045 116 045 EXPGT2: .ASCIZ /%N%A EXP'D: %06%A, %06%N%A REC'D: %0%A, %06/
660 004643 045 101 040 DUAD12: .ASCIZ /%A REG(W) WRITTEN TO: %06%A REG(R) READ; EXP'D: %06%A, REC'D: %06/
661 004745 122 101 115 PKTRAM:: .ASCIZ 'RAM Contents Do Not Match Packet Sent'
662 005013 040 040 103 SCME: .ASCIZ / CONFIG DOESN'T MATCH MFG. MASTER/
663 005056 127 122 111 WRTMSG: .ASCIZ 'WRITE CHARACTERISTICS Failed'
664 005113 124 123 123 WRTERR: .ASCIZ 'TSSR Incorrect After WRITE Command, More Bits Set Than SSR'
665 005206 124 123 123 RDERR: .ASCIZ 'TSSR Incorrect After READ Command, More Bits Set Than SSR'
666 005300 106 101 124 SCHERR: .ASCIZ 'FATAL ERROR IN SUBTEST - CHECK TAPE,CABLES,TRANSPORT etc.'
667 005372 105 122 122 RETERR: .ASCIZ 'ERROR IN SUBTEST - WRITE DATA RETRY FIVE TIMES FAILED'
668 005460 045 116 045 NOMEM: .ASCIZ '%N%A ***** NO NXM ADDRESS--CANNOT TEST NXM TIMEOUT. *****%N'
669 005554 045 116 045 M8186: .ASCIZ '%N%A ***** 11/23A SYSTEM *****%N'
670 005645 045 116 045 M8189: .ASCIZ '%N%A ***** 11/23B SYSTEM *****%N'
671
672
673
674
```

```

676
677
678
679
680
681
682
683
684 005736
    005736
685 005736
    005736 013746 003114
    005742 012746 003777
    005746 012746 000002
    005752 010600
    005754 104415
    005756 062706 000006
686 005762 004737 005770
687 005766
    005766
    005766 104423
688
689
690
691
692
693
694 005770 005727
695 005772 000000
696 005774 001402
697 005776 004777 177770
698 006002
    006002 012746 004530
    006006 012746 000001
    006012 010600
    006014 104415
    006016 062706 000004
699 006022 000207
    
```

.SBTTL GLOBAL ERROR REPORT SECTION

```

:++
: THE GLOBAL ERROR REPORT SECTION CONTAINS THE PRINTB AND PRINTX
: CALLS THAT ARE USED IN MORE THAN ONE TEST.
: ASCII TEXT STRINGS ARE FOUND IN THE GLOBAL TEXT SECTION.
:--
    
```

```

        BGNMSG  NXRERR          ;NON-EXISTANT DEVICE REGISTER.
NXRETR: PRINTX  #NXRX,NODEV     ;NODEV = NEXM ADDRESS.
        MOV     NODEV,-(SP)
        MOV     #NXRX,-(SP)
        MOV     #2,-(SP)
        MOV     SP,R0
        TRAP   C$PNTX
        ADD    #6,SP
        JSR    PC,EXTEND       ; PRINT EXTENSION IF REQUIRED.
        ENDMSG
L10002: TRAP    C$MSG
    
```

```

:
: THIS ROUTINE APPENDS A UNIQUE EXTENSION (IF REQUIRED)
: TO ANY OF THE ABOVE ERROR SIGNATURES.
:
    
```

```

EXTEND: TST     (PC)+
EXTA:  0        ; 0 = NO EXTENSION.
        BEQ    1$
        JSR    PC,@EXTA       ; APPEND EXTENSION TEXT.
1$:    PRINTX  #NULCR          ; PRINT A BLANK LINE
        MOV     #NULCR,-(SP)
        MOV     #1,-(SP)
        MOV     SP,R0
        TRAP   C$PNTX
        ADD    #4,SP
        RTS    PC
    
```

702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719

.SBTTL PRITSSR - PRINT TSSR CONTENTS

```

:ROUTINE TO DISPLAY THE CONTENTS, AND BIT DEFINITIONS, OF
:THE TSSR REGISTER. THIS ROUTINE IS NORMALLY CALLED ONLY
:BY A MESSAGE PRINTING ROUTINE
    
```

:INPUTS:

R1 CONTENTS OF TSSR

:SUBORDINATE ROUTINES:

CHKAMB CHECK FOR AMBIGUOUS CONTENTS

:-

720 006024  
 721 006024  
 722 006030 010104  
 723 006032  
     006032 010446  
     006034 012746 006415  
     006040 012746 000002  
     006044 010600  
     006046 104414  
     006050 062706 000006  
 724 006054 010400  
 725 006056 004737 016044  
 726 006062 103410  
 727 006064  
     006064 012746 006635  
     006070 012746 000001  
     006074 010600  
     006076 104415  
     006100 062706 000004  
 728 006104 010403  
 729 006106 042703 001476  
 730 006112 001434  
 731 006114 012702 002632  
 732 006120 012701 003502  
 733 006124 005703  
 734 006126 001413  
 735 006130 000241  
 736 006132 006103  
 737 006134 103006  
 738 006136 011100  
 739 006140 112022  
 740 006142 001376  
 741 006144 112762 000054 177777  
 742 006152 005721  
 743 006154 000763  
 744 006156 105042  
 745 006160  
     006160 012746 002632  
     006164 012746 006606

PRITSSR:

```

SAVREG          :SAVE GENERAL REGISTERS
MOV             R1,R4          :SAVE THE TSSR CONTENTS
PRINTB         #TSSRFOR,R4    :PRINT THE CONTENTS OF TSSR
MOV            R4,-(SP)
MOV            #TSSRFOR,-(SP)
MOV            #2,-(SP)
MOV            SP,R0
TRAP          C$PNTB
ADD            #6,SP
MOV            R4,R0          :GET TSSR BACK FOR CHKAMB
JSR            PC,CHKAMB      :ARE CONTENTS AMBIGUOUS ?
BCS            5$            :BRANCH IF NOT
PRINTX         #AMBTSSR       :SHOW CONTENTS ARE AMBIGUOUS
MOV            #AMBTSSR,-(SP)
MOV            #1,-(SP)
MOV            SP,R0
TRAP          C$PNTX
ADD            #4,SP
5$: MOV         R4,R3          :CONTENTS OF TSSR
BIC            #HIADDR!FATERR!TERCLS,R3 :CLEAR ALL MULTIPLE BIT FIELDS
BEQ           20$            :NO BITS ARE SET
MOV            #TMPBFR,P2     :TEMPORARY ASCII BUFFER
MOV            #TSSRBIT,R1   :ASCII EQUIVALENT OF BITS
10$: TST         R3           :REMAINING BITS TO CONVERT
BEQ           15$            :BRANCH WHEN ALL ARE DONE
CLC            :CLEAR CARRY FOR SHIFT
ROL            R3            :SHIFT NEXT BIT TO CARRY
BCC           13$            :BRANCH IF BIT NOT SET
MOV            (R1),R0        :POINTER TO BIT DEFINITION
11$: MOVB       (R0)+,(R2)+   :MOVE ASCII TO BUFFER
BNE           11$            :MOVE ALL BITS
MOVB          #' , -1(R2)    :INSERT A COMMA TO TERMINATE
13$: TST         (R1)+       :POINT TO NEXT DESCRIPTION
BR            10$            :GET THE REMAINING BITS
15$: CLRB       -(R2)        :TERMINATE THE LINE
PRINTX         #TSSDEF,#TMPBFR :PRINT THE BIT DEFINITIONS
MOV            #TMPBFR,-(SP)
MOV            #TSSDEF,-(SP)
    
```



```

006170 012746 000002      MOV      #2,-(SP)
006174 010600      MOV      SP,R0
006176 104415      TRAP    C$PNTX
006200 062706 000006      ADD      #6,SP
746
747 006204 010403      20$:    MOV      R4,R3          ;GET THE TSSR CONTENTS
748 006206 042703 177761      BIC      #^CTERCLS,R3    ;CLEAR ALL BUT TERMINATION
749 006212 016303 006676      MOV      TCOCOD(R3),R3   ;GET THE TERMINATION CODE MEANING
750 006216      PRINTX #TCOASC,R3      ;PRINT THE TERMINATION CODE
      006216 010346      MOV      R3,-(SP)
      006220 012746 006476      MOV      #TCOASC,-(SP)
      006224 012746 000002      MOV      #2,-(SP)
      006230 010600      MOV      SP,R0
      006232 104415      TRAP    C$PNTX
      006234 062706 000006      ADD      #6,SP
751 006240 010403      MOV      R4,R3          ;TSSR CONTENTS AGAIN
752 006242 042703 177717      BIC      #^CFATERR,R3   ;CLEAR ALL BUT FATAL TERMINATION
753 006246 001416      BEQ     25$            ;DON'T PRINT IF ZERO
754 006250 006203      ASR     R3
755 006252 006203      ASR     R3
756 006254 006203      ASR     R3
757 006256 016303 007236      MOV      TSFCOD(R3),R3   ;ALINE TERMINATION CODE FOR INDEX
758 006262      PRINTX #TFCASC,R3      ;GET THE FATAL TERMINATION CODE
      006262 010346      MOV      R3,-(SP)      ;PRINT THE FATAL TERMINATION CODE
      006264 012746 006537      MOV      #TFCASC,-(SP)
      006270 012746 000002      MOV      #2,-(SP)
      006274 010600      MOV      SP,R0
      006276 104415      TRAP    C$PNTX
      006300 062706 000006      ADD      #6,SP
759 006304 042704 176377      25$:    BIC      #^CHIADDR,R4   ;CLEAR ALL BUT EXTENDED ADDRESS
760 006310 001411      BEQ     30$            ;DON'T PRINT IF ZERO
761 006312      PRINTX #TEXASC,R4      ;PRINT THE EXTENDED ADDRESS BITS
      006312 010446      MOV      R4,-(SP)
      006314 012746 006435      MOV      #TEXASC,-(SP)
      006320 012746 000002      MOV      #2,-(SP)
      006324 010600      MOV      SP,R0
      006326 104415      TRAP    C$PNTX
      006330 062706 000006      ADD      #6,SP
762 006334 013703 002200      30$:    MOV      EPRTSW,R3      ;PRINT MEASGE BUFFER ADDRESS
763 006340      PRINTX R3              ;PRINT PROPER MESSAGE
      006340 010346      MOV      R3,-(SP)
      006342 012746 000001      MOV      #1,-(SP)
      006346 010600      MOV      SP,R0
      006350 104415      TRAP    C$PNTX
      006352 062706 000004      ADD      #4,SP
764 006356 000207      RTS      PC              ;RETURN TO CALLER
765
771 006360      EPRT2:
772 006360      045      116      045      EPRT1:  .ASCIZ  '%N% *****REPLACE M7196*****'
773
783 006415      045      116      045      TSSRFOR: .ASCIZ  '%N% TSSR = %06'
784 006435      045      116      045      TEXASC:  .ASCIZ  '%N% Extended Address Bits = %06'
785 006476      045      116      045      TCOASC:  .ASCIZ  '%N% Termination Class Code = %T'
786 006537      045      116      045      TFCASC:  .ASCIZ  '%N% Fatal Termination Class Code = %T'
787 006606      045      116      045      TSSDEF:  .ASCIZ  '%N% TSSR Bits Set: %T'
788 006635      045      116      045      AMBTSSR: .ASCIZ  '%N% TSSR Contents Are Ambiguous'
789
      .EVEN

```

790	006676	006716	006741	006767	TCOCOD:	.WORD	1\$,2\$,3\$,4\$,5\$,6\$,7\$,8\$
791	006716	116	157	162	1\$:	.ASCIZ	'Normal Termination'
792	006741	124	145	162	2\$:	.ASCIZ	'Termination Condition'
793	006767	124	141	160	3\$:	.ASCIZ	'Tape Status Alert'
794	007011	106	165	156	4\$:	.ASCIZ	'Function Reject'
795	007031	122	145	143	5\$:	.ASCIZ	'Recoverable Error - Tape Position One Record Down'
796	007113	122	145	143	6\$:	.ASCIZ	'Recoverable Error - Tape Was Not Moved'
797	007162	125	156	162	7\$:	.ASCIZ	'Unrecoverable Error'
798	007206	106	141	164	8\$:	.ASCIZ	'Fatal Controller Error'
799						.EVEN	
800							
801	007236	007246	007302	007313	TSFCOD:	.WORD	1\$,2\$,3\$,4\$
802	007246	111	156	164	1\$:	.ASCIZ	'Internal Diagnostic Failure'
803	007302	122	145	163	2\$:	.ASCIZ	'Reserved'
804	007313	102	165	163	3\$:	.ASCIZ	'Bus Interface or Sanity Check Error'
805	007357	122	145	163	4\$:	.ASCIZ	'Reserved'
806						.EVEN	

```

808 .SBTTL PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET
809
810
811 :+
812 :THIS ROUTINE PRINTS THE ADDRESS AND CONTENTS OF A COMMAND PACKET.
813 :THIS ROUTINE IS NORMALLY ONLY CALLED FROM A PRINT ROUTINE.
814 :INPUT:
815
816 : R0 NUMBER OF WORDS IN PACKET
817 : R3 HIGH ORDER COMMAND PACKET ADDRESS
818 : R4 ADDRESS OF COMMAND PACKET
819
820 :NOTE: R3 IS IGNORED IF THE KTENABLE FLAG IS CLEAR.
821 :-
822
823 PRIPKT::
824 SAVREG ;SAVE THE REGISTERS
825 MOV R0,R5 ;SAVE NO. OF WORDS IN PACKET
826 TST KTENABLE ;ABOVE 28K UNDER TEST?
827 BNE 10$ ;BR IF YES
828 CLR R3 ;SET HIGH ORDER ADDRESS TO 0
829 10$: MOV R3,R1 ;COPY HIGH ORDER ADDRESS
830 MOV R4,R0 ;GET LOWER ADDRESS
831 ROL R0 ;SHIFT BIT 15 INTO C BIT
832 ROL R1 ;AND INTO HIGH ORDER.
833 PRINTB #PKTADD,R1,R4 ;PRINT PACKET ADDRESS
      MOV R4,-(SP)
      MOV R1,-(SP)
      MOV #PKTADD,-(SP)
      MOV #3,-(SP)
      MOV SP,R0
      TRAP C$PNTB
834 15$: ADD #10,SP
835 MOV R3,R0 ;GET HIGH ORDER ADDRESS
836 BEQ 20$ ;BR IF NOT ABOVE 28K.
837 MOV R4,R1 ;GET LOW ORDER ADDRESS
838 JSR PC,SETMAP ;SETUP PAR6 MAPPING FOR 18 BIT ADDRESS
839 20$: MOV R0,R4 ;GET RETURNED PAR6 ADDRESS BIAS
840 25$: CLR R1 ;SAVE WORD NUMBER
841 MOV (R4)+,R2 ;GET PACKET CONTENTS
      PRINTB #PKTFRM,R1,R2 ;PRINT THE DATA
      MOV R2,-(SP)
      MOV R1,-(SP)
      MOV #PKTFRM,-(SP)
      MOV #3,-(SP)
      MOV SP,R0
      TRAP C$PNTB
842 000010: ADD #10,SP
843 INC R1 ;NEXT WORD NUMBER
844 CMP R1,R5 ;DONE ALL PACKET WORDS?
845 BLT 25$ ;LOOP TILL ALL DONE
846 RTS PC ;RETURN
847 007516 045 116 045 PKTFRM: .ASCIZ '%N% Packet Word #%D1% = %06%'
848 007554 045 116 045 PKTADD: .ASCIZ '%N% Packet Address = %01%05%'
849 .EVEN
850

```

```

852                                    .SBTTL PRIBXOR - PRINT EXPD, RECV AND XOR BYTE
853
854                                    :
855                                    :
856                                    :PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE DATA BYTE
857                                    :THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
858                                    :
859                                    :INPUTS:
860                                    :
861                                    :      R1       RECEIVED DATA
862                                    :      R2       EXPECTED DATA
863                                    :
864                                    :OUTPUT:
865                                    :
866                                    :      R0       XOR OF EXPECTED/RECEIVED DATA
867                                    :
868                                    :-
869
870 007612                            PRIBXOR::
871 007612                            SAVREG                                ;SAVE THE REGISTERS
872 007616   010203                   MOV       R2,R3                        ;EXPECTED DATA
873 007620                           XOR       R1,R3                        ;FORM THE EXCLUSIVE OR
874 007630   012700   177400           MOV       #^C<377>,R0                ;BYTE MASK
875 007634   040001                   BIC       R0,R1                        ;SAVE LOW BYTE RECV
876 007636   040002                   BIC       R0,R2                        ;SAVE LOW BYTE EXPD
877 007640   040003                   BIC       R0,R3                        ;SAVE LOW BYTE XOR
878 007642                            PRINTB   #XORBFOR,R2,R1,R3 ;PRINT THE MESSAGE
                                      MOV       R3,-(SP)
                                      MOV       R1,-(SP)
                                      MOV       R2,-(SP)
                                      MOV       #XORBFOR,-(SP)
                                      MOV       #4,-(SP)
                                      MOV       SP,R0
                                      TRAP     CSPNTB
879 007670   010300                   ADD       #12,SP
880 007672   000207                   MOV       R3,R0                        ;R0 HAS XOR ON RETURN
                                      RTS       PC                         ;RETURN TO CALLER
881
882 007674       045       116       045 XORBFOR:    .ASCIZ  'XN% EXPD: %03% RECV: %03% XOR: %03'
883                                    .EVEN
884

```

```

886 .SBTTL PRI XOR - PRINT EXPD, RECV AND XOR
887
888
889
890 :PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE TWO
891 :THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
892
893 :INPUTS:
894
895 R1 RECEIVED DATA
896 R2 EXPECTED DATA
897
898 :OUTPUT:
899
900 R0 XOR OF EXPECTED/RECEIVED DATA
901
902 :-
903
904 007742
905 007742
906 007746 010203
907 007750
908 007760
    007760 010346
    007762 010146
    007764 010246
    007766 012746 010012
    007772 012746 000004
    007776 010600
    010000 104414
    010002 062706 000012
909 010006 010300
910 010010 000207
911
912 010012 045 116 045 XORFOR: .ASCIZ '%N% EXPD: %06% RECV: %06% XOR: %06%'
913 .EVEN
    
```

```

: +
:PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE TWO
:THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
:INPUTS:
R1 RECEIVED DATA
R2 EXPECTED DATA
:OUTPUT:
R0 XOR OF EXPECTED/RECEIVED DATA
:-
    
```

```

PRI XOR::
    SAVREG                :SAVE THE REGISTERS
    MOV R2,R3              :EXPECTED DATA
    XOR R1,R3              :FORM THE EXCLUSIVE OR
    PRINTB #XORFOR,R2,R1,R3 :PRINT THE MESSAGE
    MOV R3,-(SP)
    MOV R1,-(SP)
    MOV R2,-(SP)
    MOV #XORFOR,-(SP)
    MOV #4,-(SP)
    MOV SP,R0
    TRAP C$PNTB
    ADD #12,SP
    MOV R3,R0              :R0 HAS XOR ON RETURN
    RTS PC                 :RETURN TO CALLER
    
```

```

915 .SBTTL PRIEQU - PRINT BIT NUMBERS AS ASCII EQUIVALENT
916
917
918
919 :ROUTINE TO CONVERT BIT VALUES TO ASCII AND PRINT THE STRING
920 :THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
921
922 :INPUTS:
923
924 R0 OCTAL VALUE TO CONVERT
925 R1 TABLE OF POINTERS TO ASCII EQUIVALENT
926
927
928
929 010060 PRIEQU: SAVREG ;SAVE THE REGISTERS
930 010060 RTS PC ;RETURN TO CALLER
931 010064 000207
932
933
934
935
936 .SBTTL PRIRAM - PRINT RAM ADDRESS
937
938
939 :PRINT CONTROLLER RAM ADDRESS.
940 :THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
941
942 :INPUTS:
943
944 R4 RAM ADDRESS
945
946
947 010066 PRIRAM: SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
948 010066 PRINTB #RAMFOR,R4 ;PRINT RAM ADDRESS IN ERROR
949 010072 MOV R4,-(SP)
    010072 010446 MOV #RAMFOR,-(SP)
    010074 012746 010116 MOV #2,-(SP)
    010100 012746 000002 MOV SP,R0
    010104 010600 TRAP C$PNTB
    010106 104414 ADD #6,SP
    010110 062706 000006 RTS PC ;RETURN
    010114 000207
950
951
952 010116 045 116 045 RAMFOR: .ASCIZ '%N%A CONTROLLER RAM ADDRESS = %06'
953 .EVEN
954
955
956 .SBTTL PRIADD - PRINT MEMORY ERROR ADDRESS
957
958
959 :PRINT MEMORY ADDRESS
960 :THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
961
962 :IMPLICIT INPUTS
963
964 ERRHI - HIGH ORDER ADDRESS
965 ERRLO - LOW ORDER ADDRESS
    
```

```

966
967
968 010160
969 010160
970 010164 013700 002236
971 010170 013701 002240
972 010174 010102
973 010176 006101
974 010200 006100
975 010202
    010202 010246
    010204 010046
    010206 012746 010230
    010212 012746 000003
    010216 010600
    010220 104414
    010222 062706 000010
976 010226 000207
977
978 010230 045 116 045 PRIA0: .ASCIZ '%X%A MEMORY ERROR ADDRESS = %01X05'
979 .EVEN
980
981
982 .SBTTL PRITADD - PRINT MEMORY TEST ADDRESS
983
984
985 .PRINT MEMORY ADDRESS
986 .THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
987
988 .IMPLICIT INPUTS
989
990 .ERRHI - HIGH ORDER ADDRESS
991 .ERRLO - LOW ORDER ADDRESS
992
993
994 010274
995 010274
996 010300 013702 002236
997 010304 013701 002240
998
999
1000
1001 010310
    010310 010146
    010312 012746 010356
    010316 012746 000002
    010322 010600
    010324 104414
    010326 062706 000006
1002 010332
    010332 010246
    010334 012746 010421
    010340 012746 000002
    010344 010600
    010346 104414
    010350 062706 000006
1003 010354 000207

```

```

:
:
: PRIADD:
: SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
: MOV ERRHI,R0 ;GET HIGH ADDRESS
: MOV ERRLO,R1 ;GET LOW ADDRESS
: MOV R1,R2 ;COPY LOW ADDRESS
: ROL R1 ;SHIFT BIT 15 TO C BIT
: ROL R0 ;SHIFT INTO HIGH ORDER
: PRINTB #PRIA0,R0,R2 ;PRINT MEMORY ADDRESS IN ERROR
: MOV R2,-(SP)
: MOV R0,-(SP)
: MOV #PRIA0,-(SP)
: MOV #3,-(SP)
: MOV SP,R0
: TRAP C$PNTB
: ADD #10,SP
: RTS PC ;RETURN

:
: PRIA0: .ASCIZ '%X%A MEMORY ERROR ADDRESS = %01X05'
: .EVEN

:
: .SBTTL PRITADD - PRINT MEMORY TEST ADDRESS
:
: .PRINT MEMORY ADDRESS
: .THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
:
: .IMPLICIT INPUTS
:
: .ERRHI - HIGH ORDER ADDRESS
: .ERRLO - LOW ORDER ADDRESS
:
:
: PRITADD:
: SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
: MOV ERRHI,R2 ;GET HIGH ADDRESS
: MOV ERRLO,R1 ;GET LOW ADDRESS
: MOV R1,R2 ;COPY LOW ADDRESS
: ROL R1 ;SHIFT BIT 15 TO C BIT
: ROL R0 ;SHIFT INTO HIGH ORDER
: PRINTB #PRIT0,R1 ;PRINT MEMORY ADDRESS LOW IN ERROR
: MOV R1,-(SP)
: MOV #PRIT0,-(SP)
: MOV #2,-(SP)
: MOV SP,R0
: TRAP C$PNTB
: ADD #6,SP
: PRINTB #PRIT1,R2 ;PRINT MEMORY ADDRESS HIGH IN ERROR
: MOV R2,-(SP)
: MOV #PRIT1,-(SP)
: MOV #2,-(SP)
: MOV SP,R0
: TRAP C$PNTB
: ADD #6,SP
: RTS PC ;RETURN

```





1012 .SBTTL SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND

1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046

```

:ROUTINE TO ISSUE A SPACE RECORDS
:COMMAND (FORWARD OR REVERSE)
:
:INPUT:
:
:   R3      NUMBER OF RECORDS TO BE SPACED OVER
:           BIT15 CONTROLS DIRECTION
:           BIT15 = 0 IS FORWARD
:           BIT15 = 1 IS REVERSE
:   R5      FIRST DEVICE UNIBUS ADDRESS
:
:   REQUIRES A WRITE CHARACTERISTICS DONE PREVIOUSLY
:
:OUTPUT:
:
:   CARRY   SET - SPACE RECORDS COMMAND OK
:           CLR - SPACE RECORDS FAILED
:
:   R0      THE CONTENTS OF R4 IS MOVED TO R0
:
:IMPLICIT OUTPUT:
:
:   TAPE HAS BEEN MOVED
:
:SIDE EFFECTS:
:
:-
    
```

1047 010466  
 1048 010466  
 1049 010472 012737 000764 010660  
 1050 010500 012737 140010 010650  
 1051 010506 005703  
 1052 010510 100403  
 1053 010512 010337 010652  
 1054 010516 000407  
 1055 010520 042703 100000  
 1056 010524 010337 010652  
 1057 010530 052737 000400 010650  
 1058 010536 012704 010650  
 1059 010542 010465 000000  
 1060 010546 004737 016250  
 1061 010552 103420  
 1062 010554  
     010554 012727 000250  
     010560 000000  
     010562 013727 002116  
     010566 000000  
     010570 005367 177772  
     010574 001375

```

SPACE::
:SAVREG
:MOV #500,SDELAY ;SAVE THE GENERAL REGISTERS
:MOV #140010,80$ ;SET UP DELAY
:TST R3 ;SET UP COMMAND, SPACE FORWARD
:BMI 5$ ;CHECK FOR DIRECTION
:MOV R3,90$ ;BR, IF REVERSE INDICATED
:BR 10$ ;LOAD UP NUMBER OF RECORDS TO SPACE
:MOV R3,90$ ;GO DO COMMAND
:BIC #BIT15,R3 ;CLEAR DIRECTION BIT
:MOV R3,90$ ;LOAD UP NUMBER OF RECORDS TO SPACE
:BIS #BIT8,80$ ;SET REVERSE BIT IN COMMAND PACKET
:MOV #80$,R4 ;SET UP R4 WITH PACKET ADDRESS
:MOV R4,TSDB(R5) ;SEND OUT COMMAND
:JSR PC,WAITF ;WAIT FOR SSR
:BCS 20$ ;BR, IF SSR IS SET AND OK
:DELAY 250 ;DELAY ABOUT .25 SECONDS
:MOV #250,(PC)+
:.WORD 0
:MOV L$DLY,(PC)+
:.WORD 0
:DEC -6(PC)
:BNE -4
    
```

	010576	005367	177756		DEC	-22(PC)	
	010602	001367			BNE	.-20	
1063	010604	005337	010660		DEC	SDELAY	:BUMP DELAY COUNTER DOWN
1064	010610	001356			BNE	15\$	:BR, IF MORE DELAY
1065	010612	000411			BR	60\$	:BR IF TROUBLE CARRY = CLEAR
1066	010614	016507	000002	20\$:	MOV	TSSR(R5),R1	:READ TSSR
1067	010620	012702	000200		MOV	#SSR,R2	:SET UP EXPECTED
1068	010624	020201		25\$:	CMP	R2,R1	:ARE THEY OK
1069	010626	001401			BEQ	40\$	:BR, IF EQUAL = OK
1070	010630	000402			BR	60\$	:TROUBLE EXIT
1071	010632	000261		40\$:	SEC		:SET CARRY NO TROUBLE
1072	010634	000401			BR	70\$	:EXIT
1073	010636	000241		60\$:	CLC		:CARRY CLEAR = ERROR
1074	010640			70\$:			
1075	010640	010400			MOV	R4,R0	:PASS PACKET ADDRESS
1076	010642	000207			RTS	PC	:RETURN

1078			:		
1079			:		
1080			:		
1081			:	PACKET FOR SPACE COMMAND	
1082			:		
1084	010650		:	.=<.+10>&177770	
1086			:		
1087			:	COMMAND WORD	
1088	010650	000000	:	80\$: .WORD	
1089			:	:NUMBER OF RECORDS TO BE SPACED OVER WORD	
1090	010652	000000	:	90\$: .WORD	
1091	010654	000000	:	.WORD	
1092	010656	000000	:	.WORD	
1093	010660	000000	:	SDELAY: .WORD 0	:DELAY COUNTER
1094			:	.EVEN	

```

1096 .SBTTL WRTCHR - WRITE CHARACTERISTICS COMMAND
1097
1098 :+
1099 :ROUTINE TO ISSUE A WRITE CHARACTERISTICS
1100 :COMMAND SO THAT OTHER COMMANDS WILL BE ACCEPTED
1101
1102 :INPUT:
1103
1104 R4 ADDRESS OF PACKET FROM TEST
1105 R5 FIRST DEVICE UNIBUS ADDRESS
1106 REQUIRES A CALL TO SOFINIT BE DONE PREVIOUSLY
1107
1108 :OUTPUT:
1109
1110 R0 TSSR CONTENTS
1111 CARRY SET - WRITE CHARACTERISTICS COMMAND OK
1112 CLR - WRITE CHARACTERISTICS FAILED
1113
1114 :IMPLICIT OUTPUT:
1115
1116 MESSAGE BUFFER AND OTHER BUFFERS ALL SET UP
1117 SOFTWARE SWITCHES SET AS FOLLOWS:
1118 EXTFEA = EXTENDED FEATURES PRESENT
1119 BENBSW = BUFFER ENABLE SWITCH ON OR OFF
1120
1121 :SIDE EFFECTS:
1122
1123 :-
1124
1125
1126
1127
1128 010662 WRTCHR::
1129 010662 SAVREG ;SAVE THE GENERAL REGISTERS
1130 010666 005037 002230 CLR BENBSW ;CLEAR BUFFER ENABLE SWITCH
1131 010672 005037 002226 CLR EXTFEA ;CLEAR EXTENDED FEATURES SW SWITCH
1132 010676 010465 000000 10$: MOV R4,TSDB(R5) ;SEND OUT COMMAND
1133 010702 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR
1134 010706 103401 BCS 20$ ;BR, IF SSR IS SET AND OK
1135 010710 000435 BR 60$ ;BR IF TROUBLE CARRY = CLEAR
1136 010712 016501 000002 20$: MOV TSSR(R5),R1 ;READ TSSR
1137 010716 012702 000200 MOV #SSR,R2 ;SET UP EXPECTED
1138 010722 032701 000100 BIT #OFL,R1 ;WAS OFF LINE SET IN TSSR
1139 010726 001402 BEQ 25$ ;BR, IF NO OFL SET
1140 010730 052702 000100 BIS #OFL,R2 ;MAKE THEM LOOK ALIKE
1141 010734 02020? 25$: CMP R2,R1 ;ARE THEY OK
1142 010736 001401 BEQ 40$ ;BR, IF EQUAL = OK
1143 010740 000421 BR 60$ ;TROUBLE EXIT
1144 010742 062704 000010 40$: ADD #8.,R4 ;POINT TO WRT CHARA DATA PACKET
1145 010746 011403 MOV (R4),R3 ;GET ADDRESS OF MESSAGE BUFFER
1146 010750 032763 000200 000012 BIT #X2.EXTF,XST2(R3) ;EXTENDED FEATURES BIT SET?
1147 010756 001402 BEQ 45$ ;BR IF NO
1148 010760 005237 002226 INC EXTFEA ;SET EXTENDED FEATURES SW SWITCH
1149 010764 45$:
1150 010764 032763 000100 000012 BIT #X2.BUFE,XST2(R3) ;BUFFER ENABLE SWITCH SET
1151 010772 001402 BEQ 50$ ;BR, IF SWITCH NOT SET
1152 010774 005237 002230 INC BENBSW ;SET SOFTWARE SWITCH FOR ENABLED
    
```

1153	011000		50\$:			
1154	011000	000261		SEC		:SET CARRY NO TROUBLE
1155	011002	000401		BR	70\$	:EXIT
1156	011004	000241	60\$:	CLC		:CARRY CLEAR = ERROR
1157	011006	016500	70\$:	MOV	TSSR(R5),R0	:RETURN TSSR CONTENTS
1158	011012	000207		RTS	PC	:RETURN
1159						
1160						

1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1206  
1208  
1209  
1210  
1211  
1212

.SBTTL REWIND - POSITION TAPE (REWIND) COMMAND

```

+
: THIS ROUTINE WILL REWIND THE SELECTED TAPE.
: CAUTION: THE ROUTINE DOES NOT WAIT FOR BOT
:           TO ARRIVE. ALSO THE CALLER MUST CHECK FOR
:           SSR TO SET IN THE TSSR
    
```

```

: CALLING SEQUENCE:
: DO A SOFT INIT
: DO A WRITE CHARACTERISTICS
: JSR PC,REWIND
    
```

```

: INPUT:
: R5 FIRST DEVICE UNIBUS ADDRESS
    
```

```

: OUTPUT
: R0 THE CONTENTS OF R4 IS PASSED TO R0
    
```

```

REWIND::
: SAVREG
: SAVE R1-R5 UNTIL NEXT RETURN
MOV #RWPACK,R4 : GET PACKET ADDRESS
MOV R4,TSDB(R5) : SEND PACKET ADDRESS TO EXECUTE
MOV #360.,R3 : ENOUGH TIME FOR 2400' REEL TO REWIND
10$: JSR PC,WAITF : WAIT FOR SSR TO SET
BCS 20$ : LEAVE WHEN SSR IS SET
DELAY 250. : WAIT FOR .25 SECONDS
MOV #250.,(PC)+
.WORD 0
MOV LSDLY,(PC)+
.WORD 0
DEC -6(PC)
BNE -.4
DEC -22(PC)
BNE -.20
DEC R3 : BUMP COUNTER DOWN
BNE 10$ : KEEP GOING
CLC : CLEAR CARRY TO SET ERROR
20$: MOV R4,R0 : PASS THE PACKET ADDRESS
RTS PC : RETURN
    
```

```

RWPACK: .=<.+10>&177770
.WORD 102010 : POSTION COMMAND (REWIND)
.WORD 0 : NOT USED
    
```

```

011014
011014
012704 011110
010465 000000
012703 000550
004737 016250
103417
012727 000372
000000
013727 002116
000000
005367 177772
001375
005367 177756
001367
005303
001357
000241
010400
000207
011110
011110 102010
011112 000000
    
```



1213  
1214  
1215

1217 .SBTTL CKRAM - COMPARE RAM TO I/O PACKET

```

1218
1219
1220
1221 :ROUTINE TO READ THE FIRST 8 BYTES FROM RAM
1222 :MEMORY AND COMPARE THIS DATA TO A COMMAND PACKET.
1223
1224 :INPUT:
1225
1226 R4 ADDRESS OF THE COMMAND PACKET
1227 R5 FIRST DEVICE UNIBUS ADDRESS
1228
1229 :OUTPUT:
1230
1231 CARRY SET - RAM MATCHES PACKET
1232 CLR - RAM DOES NOT MATCH PACKET
1233
1234 :IMPLICIT OUTPUT:
1235
1236 THE TABLE RAMDATA IS FILLED WITH THE
1237 DATA HELD IN RAM.
1238 RAMSIZ IS SET TO 8. FOR PRAMPKT ROUTINE
1239
1240 :SIDE EFFECTS:
1241
1242 THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
1243
1244
1245

```

```

1246 011114 CKRAM:: SAVREG :SAVE THE GENERAL REGISTERS
1247 011114 MOV #RAMDATA,R1 :ADDRESS TO SAVE THE RAM DATA
1248 011120 012701 002242 MOV #RMPKTBEG,R2 :BYTE ADDRESS OF FIRST RAM DATA
1249 011124 012702 000201 CLR R3 :CLEAR THE ERROR FLAG
1250 011130 005003 JSR PC,CHKTSSR :WAIT FOR SSR
1251 011132 004737 016336 MOVB #0,TSDB(R5) :SET MAINTENANCE MODE
1252 011136 112765 000000 000000 10$: JSR PC,CHKTSSR :WAIT FOR SSR TO SET
1253 011144 004737 016336 MOV R2,TSDB(R5) :SELECT NEXT RAM ADDRESS
1254 011150 010265 000000 JSR PC,CHKTSSR :WAIT FOR SSR TO SET
1255 011154 004737 016336 MOVB TSBA(R5),(R1) :READ THE RAM DATA
1256 011160 116511 000000 CMPB (R1)+,(R4)+ :COMPARE TO EXPECTED
1257 011164 122124 BEQ 20$ :BRANCH IF OK
1258 011166 001401 INC R3 :SET ERROR FLAG
1259 011170 005203 INC R2 :ADDRESS OF NEXT RAM LOCATION
1260 011172 005202 20$: INC R2 :REACHED END YET ?
1261 011174 020227 000210 CMP R2,#RMPKTEND :BRANCH TILL ALL READ
1262 011200 003761 BLE 10$ :WAS AN ERROR FOUND ?
1263 011202 005703 TST R3 :BRANCH IF NOT
1264 011204 001402 BEQ 30$ :CLEAR CARRY TO SHOW ERROR
1265 011206 000241 CLC :AND EXIT
1266 011210 000401 BR 50$ :SHOW GOOD COMPARE
1267 011212 000261 30$: SEC :SETUP RAMSIZ FOR PRAMPKT ROUTINE
1268 011214 012737 000010 002302 50$: MOV #8.,RAMSIZ :RETURN
1269 011222 000207 RTS
1270

```

1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299

.SBTTL CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA

: ROUTINE TO READ THE FIRST 8 OR 10 BYTES FROM RAM  
 : MEMORY AND COMPARE THIS DATA TO A CHARACTERISTICS DATA BLOCK.

: INPUT:

R4 ADDRESS OF THE CHARACTERISTICS DATA  
 R5 FIRST DEVICE UNIBUS ADDRESS

: OUTPUT:

CARRY SET - RAM MATCHES PACKET  
 CLR - RAM DOES NOT MATCH PACKET

: IMPLICIT OUTPUT:

THE TABLE RAMDATA IS FILLED WITH THE  
 DATA HELD IN RAM.  
 RAMSIZ IS SET TO 8. OR 10. FOR PRAMPKT ROUTINE

: SIDE EFFECTS:

THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE

: -

1300 011224  
1301 011224  
1302 011230 012701 002242  
1303 011234 012702 000167  
1304 011240 005003  
1305 011242 004737 016336  
1306 011246 112765 000000 000000  
1307 011254 004737 016336 10\$:  
1308 011260 010265 000000  
1309 011264 004737 016336  
1310 011270 116511 000000  
1311 011274 122124  
1312 011276 001401  
1313 011300 005203  
1314 011302 005202 20\$:  
1315 011304 012737 000010 002302  
1316 011312 005737 002226  
1317 011316 001407  
1318 011320 012737 000012 002302  
1319 011326 020227 000200  
1320 011332 003750  
1321 011334 000403  
1322 011336 020227 000176 25\$:  
1323 011342 003744  
1324 011344 005703 27\$:  
1325 011346 001402  
1326 011350 000241  
1327 011352 000401  
1328 011354 000261 30\$:

CKRAM2::  
 SAVREG :SAVE THE GENERAL REGISTERS  
 MOV #RAMDATA,R1 :ADDRESS TO SAVE THE RAM DATA  
 MOV #RMCHBEG,R2 :BYTE ADDRESS OF FIRST RAM DATA  
 CLR R3 :CLEAR THE ERROR FLAG  
 JSR PC,CHKTSSR :WAIT FOR SSR  
 MOVB #0,TSDB(R5) :SET MAINTENANCE MODE  
 JSR PC,CHKTSSR :WAIT FOR SSR TO SET  
 MOV R2,TSDB(R5) :SELECT NEXT RAM ADDRESS  
 JSR PC,CHKTSSR :WAIT FOR SSR TO SET  
 MOVB TSBA(R5),(R1) :READ THE RAM DATA  
 CMPB (R1)+,(R4)+ :COMPARE TO EXPECTED  
 BEQ 20\$ :BRANCH IF OK  
 INC R3 :SET ERROR FLAG  
 INC R2 :ADDRESS OF NEXT RAM LOCATION  
 MOV #8,RAMSIZ :ASSUME EXTFEA NOT SET  
 TST EXTFEA :IS THE SOFTWARE EXTENDED FEATURES SET  
 BEQ 25\$ :BR, IF NOT SET  
 MOV #10,RAMSIZ :SET RAMSIZ FOR EXTEND FEATURES  
 CMP R2,#RMCHEND :AT END OF EXTENDED BUFFER  
 BLE 10\$ :BR, IF NOT AT END YET  
 BR 27\$ :AT END BRANCH  
 CMP R2,#RMCHEND-2 :REACHED END YET ?  
 BLE 10\$ :BRANCH TILL ALL READ  
 TST R3 :WAS AN ERROR FOUND ?  
 BEQ 30\$ :BRANCH IF NOT  
 CLC :CLEAR CARRY TO SHOW ERROR  
 BR 50\$ :AND EXIT  
 SEC :SHOW GOOD COMPARE

1329 011356 000207  
1330

50\$:    RTS    PC

;RETURN

```

1332 .SBTTL CKMSG - COMPARE WRITE CHAR. MESSAGE BUFFERS
1333
1334
1335 :ROUTINE TO COMPARE A WRITE CHARACTERISTICS EXPD AND RECV
1336 :BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
1337 :ERROR PRINT ROUTINES.
1338
1339 :INPUT:
1340
1341 R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS
1342 R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS
1343 R2 EXPD MESSAGE BUFFER ADDRESS
1344 :OUTPUT:
1345
1346 CARRY SET - MESSAGE BUFFERS MATCH
1347 CLR -MESSAGE BUFFERS DON'T MATCH
1348
1349 :IMPLICIT OUTPUT:
1350
1351 EXPMSG BUFFER IS SET TO EXPD DATA
1352 RECMG MSG BUFFER IS SET TO RECV DATA
1353 RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
1354 RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
1355
1356
1357 011360 CKMSG:: SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1358 011360 MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
1359 011364 010037 002304 MOV R1,RCVLOAD ;SAVE RECV LOW ADDRESS
1360 011370 010137 002306 TST KTENABLE ;TESTING ABOVE 28K?
1361 011374 005737 003134 BEQ 10$ ;BR IF NO
1362 011400 001403 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
1363 011402 004737 017316 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
1364 011406 010001 10$: CLR R4 ;WORD IN BUFFER
1365 011410 005004 CLR R3 ;CLEAR ERROR SEEN FLAG
1366 011412 005003 MOV R2,R5 ;GET EXPD BUFFER ADDRESS
1367 011414 010205 15$: MOV (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
1368 011416 011264 002322 MOV (R1),RCMSG(R4) ;SAVE RECV FOR ERROR REPORT
1369 011422 011164 002466 CMP (R2)+,(R1)+ ;EXPD EQUAL RECV?
1370 011426 022221 BEQ 25$ ;BR IF YES
1371 011430 001401 INC R3 ;SET ERROR SEEN FLAG
1372 011432 005203 25$: ADD #2,R4 ;POINT TO NEXT WORD ADDRESS
1373 011434 062704 000002 CMP R4,#14 ;DONE FIRST 7 WORDS?
1374 011440 020427 000014 BLE 15$ ;BR IF NO
1375 011444 003764 BIT #X2.EXTF,XST2(R5) ;IS EXTENDED FEATURES SET IN EXPD?
1376 011446 032765 000200 000012 BEQ 50$ ;BR IF NO
1377 011454 001403 CMP R4,#16 ;DONE EXTENDED FEATURES WORD?
1378 011456 020427 000016 BLE 15$ ;BR IF NO
1379 011462 003755 50$: TST R3 ;ANY ERRORS SEEN?
1380 011464 005703 BEQ 55$ ;BR IF NO
1381 011466 001402 CLC ;SET FAILURE
1382 011470 000241 BR 60$
1383 011472 000401 55$: SEC ;SET SUCCESS
1384 011474 000261 60$: RTS PC ;RETURN
1385 011476 000207
1386

```

1388 .SBTTL CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

1389  
 1390  
 1391 :ROUTINE TO COMPARE AN EXPECTED AND RECEIVED MESSAGE  
 1392 :BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR  
 1393 :ERROR PRINT ROUTINES.

1394  
 1395 :INPUT:

1396  
 1397 R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS  
 1398 R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS  
 1399 R2 EXPD MESSAGE BUFFER ADDRESS  
 1400 R3 NUMBER OF BYTES TO COMPARE

1401  
 1402 :OUTPUT:

1403  
 1404 CARRY SET - MESSAGE BUFFERS MATCH  
 1405 CLR - MESSAGE BUFFERS DON'T MATCH

1406  
 1407 :IMPLICIT OUTPUT:

1408  
 1409 EXPMSG BUFFER IS SET TO EXPD DATA  
 1410 RECVMSG BUFFER IS SET TO RECV DATA  
 1411 RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV  
 1412 RCVLOADD SET TO LOW ORDER ADDRESS OF RECV  
 1413

1414  
 1415 011500  
 1416 011500  
 1417 011504 020327 000144  
 1418 011510 003412  
 1419 011512 012703 000144  
 1420 011516  
 011516 012746 011632  
 011522 012746 000001  
 011526 010600  
 011530 104417  
 011532 062706 000004  
 1421 011536 010037 002304  
 1422 011542 010137 002306  
 1423 011546 005737 003134  
 1424 011552 001403  
 1425 011554 004737 017316  
 1426 011560 010001  
 1427 011562 005004  
 1428 011564 005005  
 1429 011566 111264 002322  
 1430 011572 111164 002466  
 1431 011576 122221  
 1432 011600 001401  
 1433 011602 005205  
 1434 011604 062704 000001  
 1435 011610 020403  
 1436 011612 002001  
 1437 011614 000764  
 1438 011616 005705  
 1439 011620 001402

CKMSG2::  
 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN  
 CMP R3,#RECVMSG-EXPMSG;@AD IS COUNT ABOVE MAX ALLOWED?  
 BLE 5\$ ;@AD BR IF NO  
 MOV #REC SG-EXPMSG,R3;@AD  
 PRINTF #DEBUGMSG ;@AD  
 MOV #DEBUGMSG,-(SP)  
 MOV #1,-(SP)  
 MOV SP,R0  
 TRAP C\$PNTF  
 ADD #4,SP  
 5\$: MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS  
 MOV R1,RCVLOAD ;SAVE RECV LOW ADDRESS  
 TST KTENABLE ;TESTING ABOVE 28K?  
 BEQ 10\$ ;BR IF NO  
 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0  
 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6  
 10\$: CLR R4 ;WORD IN BUFFER  
 CLR R5 ;CLEAR ERROR SEEN FLAG  
 15\$: MOVB (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT  
 MOVB (R1),RECVMSG(R4) ;SAVE RECV FOR ERROR REPORT  
 CMPB (R2)+,(R1)+ ;EXPD EQUAL RECV?  
 BEQ 25\$ ;BR IF YES  
 INC R5 ;SET ERROR SEEN FLAG  
 25\$: ADD #1,R4 ;POINT TO NEXT BYTE  
 CMP R4,R3 ;DONE ALL BYTES?  
 BGE 50\$ ;BR IF YES  
 BR 15\$ ;DO NEXT BYTE  
 50\$: TST R5 ;ANY ERRORS SEEN?  
 BEQ 55\$ ;BR IF NO

```
1440 011622 000241                    CLC                    ;SET FAILURE
1441 011624 000401                    BR                    60$                    ;
1442 011626 000261                    55$:                  SEC                    ;SET SUCCESS
1443 011630 000207                    60$:                  RTS                    ;RETURN
1444
1445 011632        120        122        117 DEBUGMSG:        .ASCIZ 'PROGRAM INTERNAL ERROR -CKMSG2 MESSAGE BUFFER EXCEEDED-';@@D
1446 011722        045        116        045 FERCM:        .ASCII /%N%A ***/
1447 011733        040        040        124 ERCM:        .ASCIZ / TSSR ERROR CODE REC'D = /
1448 011766        056        056        056 SIMSG:        .ASCIZ /... AFTER DOING SOFT INIT/
1449 012021        124        105        123 TINERR:        .ASCIZ /TEST: .../
1450                                    .EVEN
```

1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501

012034  
012034  
012034 004737 006024  
012040 004737 017202  
012044  
012044 104423  
012046  
012046  
012046 004737 006024  
012052 012700 000004  
012056 004737 007370  
012062  
012062  
012062 104423  
012064  
012064

```

: +
: PRINT ROUTINE TO FATAL SOFT INIT ERRORS
: INPUT:
:       R1       CONIENTS OF TSSR AT ERROR
: SIDE EFFECTS:
:       EXECUTES DROP UNIT TO CEASE TESTING
: -

      BGNMSG  SFIMSG
SFIMSG::
      JSR     PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
      JSR     PC,CKDROP      ;DROP UNIT, IF ALLOWED
      ENDMSG
L10003:
      TRAP    C$MSG

: +
: PRINT ROUTINE TO PRINT THE CONTENTS OF
: TSSR AND A COMMAND PACKET OTHER THAN GET STATUS COMMAND PACKET.
: INPUTS:
:       R1       TSSR CONTENTS
:       R4       ADDRESS OF COMMAND PACKET
: -

      BGNMSG  PKTSSR
PKTSSR::
      JSR     PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
      MOV     #4,R0           ;NO. OF WORDS IN PACKET
      JSR     PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
      ENDMSG
L10004:
      TRAP    C$MSG

: +
: PRINT ROUTINE TO PRINT THE CONTENTS OF
: TSSR AND A GET STATUS COMMAND PACKET.
: INPUTS:
:       R1       TSSR CONTENTS
:       R4       ADDRESS OF COMMAND PACKET
: -

      BGNMSG  PKTGETS
PKTGETS::
```



```

1502 012064 004737 006024      JSR    PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
1503 012070 012700 000002      MOV    #2,R0           ;NO. OF WORDS IN GET STATUS PACKET
1504 012074 004737 007370      JSR    PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
1505 012100      ENDMSG
      012100
      012100 104423      L10005: TRAP    C$MSG

1506
1507
1508
1509      ;+
1510      ;PRINT TSSR ERRORS FOR INITIALIZATION TESTS
1511      ;INPUTS:
1512      ;
1513      ;       R1      TSSR CONTENTS
1514      ;       R4      ADDRESS OF COMMAND PACKET
1515      ;-
1516
1517 012102      BGNMSG  SFFMSG
      012102      SFFMSG:: JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
1518 012102 004737 006024      JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
1519 012106      ENDMSG
      012106      L10006: TRAP    C$MSG
      012106 104423

1520
1521
1522      .SBTTL  PKTMES - PRINT TSSR AND MESSAGE BUFFER
1523      ;+
1524      ;PRINT ROUTINE TO PRINT THE CONTENTS OF TSSR AND MESSAGE
1525      ;BUFFER FOR ERROR REPORTS
1526      ;INPUTS:
1527      ;
1528      ;       R1      CONTENTS OF TSSR
1529      ;       R2      LOW ORDER MESSAGE BUFFER
1530      ;       R3      HIGH ORDER MESSAGE BUFFER ADDRESS
1531      ;       NOTE: R3 IS IGNORED IF KTENABLE FLAG IS CLEAR
1532      ;-
1533
1534
1535 012110      BGNMSG  PKTMES
      012110      PKTMES:: JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR
1536 012110 004737 006024      MOV    R2,R0           ;LOW ORDER ADDRESS
1537 012114 010200      MOV    R3,R1           ;HIGH ORDER ADDRESS
1538 012116 010301      JSR    PC,PRMESS      ;PRINT THE MESSAGE BUFFER
1539 012120 004737 014242      ENDMSG
1540 012124      L10007: TRAP    C$MSG
      012124
      012124 104423

1541
  
```

1543  
 1544  
 1545  
 1546  
 1547  
 1548  
 1549  
 1550  
 1551  
 1552  
 1553  
 1554  
 1555 012126  
 012126  
 1556 012126 004737 010274  
 1557 012132 016501 000002  
 1558 012136 004737 006024  
 1559 012142  
 012142  
 012142 104423  
 1560  
 1561  
 1562  
 1563  
 1564  
 1565  
 1566  
 1567  
 1568  
 1569  
 1570  
 1571  
 1572  
 1573  
 1574 012144  
 012144  
 1575 012144 012700 000007  
 1576 012150 005737 002226  
 1577 012154 001402  
 1578 012156 012700 000010  
 1579 012162 004737 014552  
 1580 012166  
 012166  
 012166 104423  
 1581  
 1582

```

.SBTTL ADDSSR - PRINT TEST ADDRESS AND TSSR
:
:PRINT ROUTINE TO PRINT THE CONTENTS OF
:TSSR AND A MEMORY TEST ADDRESS
:
:INPUTS:
:
:      R5      FIRST DEVICE UNIBUS ADDRESS
:      ERRHI   HIGH ORDER MEMORY TEST ADDRESS
:      ERRLO   LOW ORDER MEMORY TEST ADDRESS
:-
:
:      BGNMSG  ADDSSR
ADDSSR::
:      JSR     PC,PRITADD      ;PRINT MEMORY TEST ADDRESS
:      MOV     TSSR(R5),R1    ;GET CURRENT TSSR
:      JSR     PC,PRITSSR     ;PRINT THE CONTENTS OF TSSR REGISTER
:      ENDMSG
L10010:
:      TRAP    C$MSG

.SBTTL MSGEXP - PRINT WRITE CHAR. EXPD-RECV MESSAGE BUFFERS
:
:PRINT ROUTINE TO PRINT WRITE CHARACTERISTIC MESSAGE BUFFER
:
:IMPLICIT INPUTS:
:
:      EXPMSG  - EXPECTED MESSAGE BUFFER
:      RECMSG  - RECEIVED MESSAGE BUFFER
:      RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
:      RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
:-
:      BGNMSG  MSGEXP
MSGEXP::
:      MOV     #7,R0          ;ASSUME NO EXT FEATURES
:      TST     EXTFEA        ;EXT FEATURES SET?
:      BEQ     5$            ;BR IF NO
:      MOV     #8.,R0        ;EXT FEATURE BUFFER IS 8 WORDS
:      JSR     PC,PRMSGEXP   ;PRINT EXPD/RECV MESSAGE BUFFERS
:      ENDMSG
5$:
L10011:
:      TRAP    C$MSG
  
```

```

1584 .SBTTL FIFEXP - PRINT FIFO EXP/RECV DATA
1585
1586
1587 PRINT ROUTINE TO PRINT FIFO EXP/RECV DATA
1588
1589 R1 - BYTE COUNT
1590
1591 IMPLICIT INPUTS:
1592
1593 EXPMSG - EXPECTED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
1594 RECMSG - RECEIVED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
1595
1596 BGNMSG FIFEXP
1597 FIFEXP::
1598 PRINTX #FIF1MSG,R1 ;PRINT BYTES TRANSFERRED
1599 MOV R1,-(SP)
1600 MOV #FIF1MSG,-(SP)
1601 MOV #2,-(SP)
1602 MOV SP,R0
1603 TRAP C$PNTX
1604 ADD #6,SP
1605 PRINTX #FIF2MSG ;PRINT HEADER MSG
1606 MOV #FIF2MSG,-(SP)
1607 MOV #1,-(SP)
1608 MOV SP,R0
1609 TRAP C$PNTX
1610 ADD #4,SP
1611 MOV R1,R0 ;GET BYTE COUNT
1612 JSR PC,PRBYTEXP ;PRINT FIFO BYTES IN ERROR
1613 ENDMSG
1614
1615 L10012:
1616 TRAP C$MSG
1617 .ASCIZ '%N% NUMBER OF BYTES TRANSFERRED = %D%'
1618 .ASCIZ '%N% FIFO DATA BYTES IN ERROR:'
1619 .EVEN
  
```

1596	012170				
	012170				
1597	012170				
	012170	010146			
	012172	012746	012242		
	012176	012746	000002		
	012202	010600			
	012204	104415			
	012206	062706	000006		
1598	012212				
	012212	012746	012311		
	012216	012746	000001		
	012222	010600			
	012224	104415			
	012226	062706	000004		
1599	012232	010100			
1600	012234	004737	015122		
1601	012240				
	012240				
	012240	104423			
1602	012242	045	116	045	FIF1MSG:
1603	012311	045	116	045	FIF2MSG:
1604					
1605					

```

1607                                     .SBTTL MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS
1608                                     :+
1609                                     :PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
1610                                     :
1611                                     :IMPLICIT INPUTS:
1612                                     :
1613                                     :EXPMSG - EXPECTED MESSAGE BUFFER
1614                                     :RECMSG - RECEIVED MESSAGE BUFFER
1615                                     :RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1616                                     :RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1617                                     :
1618                                     :-
1619
1620 012350 BGNMSG MSGSTAT
1621 012350 MSGSTAT::
1622 012350 012701 012412 MOV #STATCOD,R1 ;ASCII ADDRESS TABLE
1623 012354 012100 10$: MOV (R1)+,R0 ;DONE ALL MSG LINES?
1624 012356 001410 BEQ 20$ ;BR IF YES
1625 012360 PRINTX R0 ;PRINT STATUS BIT NAMES
1626 012360 010046 MOV R0,-(SP)
1627 012362 012746 000001 MOV #1,-(SP)
1628 012366 010600 MOV SP,R0
1629 012370 104415 TRAP C$PNTX
1630 012372 062706 000004 ADD #4,SP
1631 012376 000766 BR 10$ ;DO ANOTHER MSG LINE
1632 012400 012700 000012 20$: MOV #10,R0 ;NUMBER OF WORDS IN A READ STATUS BUFFER
1633 012404 004737 014552 JSR PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
1634 012410 ENDMMSG
1635 012410 L10013:
1636 012410 104423 TRAP C$MSG
1637
1638 1629
1639 012412 012430 012472 012563 STATCOD: .WORD 1$,2$,3$,4$,5$,6$,0
1640 012430 045 116 045 1$: .ASCIZ '%N% Tape Bus Signals in Word #8:'
1641 012472 045 116 045 2$: .ASCIZ '%N% PARERR<15> IEOT <12> IFMK <9> IRDY<6> IRWD<2>'
1642 012563 045 116 045 3$: .ASCIZ '%N% IRESV2<14> IIDENT<11> IHER <8> IONL<5> IFBY<1>'
1643 012654 045 116 045 4$: .ASCIZ '%N% IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>'
1644 012745 045 116 045 5$: .ASCIZ '%N% Tape Bus Signals in Word #9:'
1645 013007 045 116 045 6$: .ASCIZ '%N% DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>'
1646
1647                                     .EVEN
1648
1649
1650                                     .SBTTL MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS
1651                                     :+
1652                                     :PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
1653                                     :
1654                                     :IMPLICIT INPUTS:
1655                                     :
1656                                     :EXPMSG - EXPECTED MESSAGE BUFFER
1657                                     :RECMSG - RECEIVED MESSAGE BUFFER
1658                                     :RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1659                                     :RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1660                                     :
1661                                     :-
1662
1663 013064 BGNMSG MSGLOOP
1664 013064 MSGLOOP::
1665 013064 012701 013126 MOV #LOOPCOD,R1 ;ASCII ADDRESS TABLE
  
```

```

1655 013070 012100          10$:  MOV      (R1)+,R0          ;DONE ALL MSG LINES?
1656 013072 001410          BEQ      20$          ;BR IF YES
1657 013074          PRINTX  R0          ;PRINT STATUS BIT NAMES
      013074 010046          MOV      R0,-(SP)
      013076 012746 000001  MOV      #1,-(SP)
      013102 010600          MOV      SP,R0
      013104 104415          TRAP    C$PNTX
      013106 062706 000004  ADD      #4,SP
1658 013112 000766          BR       10$          ;DO ANOTHER MSG LINE
1659 013114 012700 000012  20$:  MOV      #10,R0          ;NUMBER OF WORDS IN A READ STATUS BUFFER
1660 013120 004737 014552  JSR      PC,PRMSGEXP  ;PRINT EXPD/RECV MESSAGE BUFFERS
1661 013124          ENDMSG
      013124          L10014:
      013124 104423          TRAP    C$MSG
1662
1663 013126 013146 013221 013320 LOOPCOD: .WORD 1$,2$,3$,4$,5$,6$,7$,0
1664 013146          045 116 045 1$: .ASCIZ '%NZA Tape Bus Loopback Signals in Word #8:'
1665 013221          045 116 045 2$: .ASCIZ '%NZA PARERR<15> IRESV2<14> IRESV1<13>'
1666 013320          045 116 045 3$: .ASCIZ '%NZA IHISP=>IEOT<12> IWRT=>IIDENT<11> IREV =>ICER <10>'
1667 013417          045 116 045 4$: .ASCIZ '%NZA IWFM =>IFMK<09> IEDIT=>IHER <08> IFAD =>ISPEED<07>'
1668 013516          045 116 045 5$: .ASCIZ '%NZA ITAD0=>IRDY<06> ITAD1=>IONL <05> IERASE=>ILDP <04>'
1669 013615          045 116 045 6$: .ASCIZ '%NZA IREW =>IDBY<03> IRWU =>IRWD <02> IFEN =>IFBY <01>'
1670 013714          045 116 045 7$: .ASCIZ '%NZA IGO =>IFPT<00>'
1671          .EVEN
1672

```

1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687 013742  
013742  
1688 013742 012700 000012  
1689 013746 004737 014552  
1690 013752  
013752  
013752 104423  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708 013754  
013754  
1709 013754 004737 010160  
1710 013760 013701 002232  
1711 013764 013702 002234  
1712 013770 004737 007742  
1713 013774  
013774  
013774 104423  
1714

```
.SBTTL MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER
:+
:PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
:IMPLICIT INPUTS:
:
:EXPMSG - EXPECTED MESSAGE BUFFER
:RCMSG - RECEIVED MESSAGE BUFFER
:RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
:RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
:-
BGNMSG MSGSUB
MSGSUB::
MOV #10.,R0 ;SIZE OF WRITE SUBSYSTEM BUFFER
JSR PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
ENDMSG
L10015:
TRAP C$MSG
```

```
.SBTTL MEMADD - PRINT MEMORY ADDRESS DATA ERROR
:+
:PRINT ROUTINE TO PRINT MEMORY ADDRESS DATA COMPARE ERROR
:IMPLICIT INPUTS:
:
:ERRHI - MEMORY ERROR HIGH ORDER ADDRESS
:ERRLO - MEMORY ERROR LOW ORDER ADDRESS
:EXP - EXPECTED DATA
:RCV - RECEIVED DATA
:-
BGNMSG MEMADD
MEMADD::
JSR PC,PRIADD ;PRINT MEMORY ADDRESS IN ERROR
MOV EXPD,R1 ;GET EXPD DATA
MOV RECV,R2 ;GET RECEIVED DATA
JSR PC,PRIXOR ;PRINT EXPD/RCV
ENDMSG
L10016:
TRAP C$MSG
```

```

1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737 013776
1738 013776
1739 014002 012701 002242
1740 014006 005002
1741 014010 122124
1742 014012 001005
1743 014014
1744 014024 000436
1745 014026 116105 177777
1746 014032 116403 177777
1747 014036
1748 014046 042703 177400
1749 014052 116137 177777 002234
1750 014060 116437 177777 002232
1751 014066
      014066 010346
      014070 013746 002232
      014074 013746 002234
      014100 010246
      014102 012746 014156
      014106 012746 000005
      014112 010600
      014114 104414
      014116 062706 000014
1752 014122 005202
1753 014124 005737 002302
1754 014130 001404
1755 014132 020237 002302
1756 014136 003724
1757 014140 000403
1758 014142 020227 000010
1759 014146 002720
1760 014150 005037 002302
1761 014154 000207
1762
1763 014156      045      116      045 RAMASC: .ASCIZ 'XNXA BYTE: XD2XA RAM: X03XA Packet: X03XA XOR:X03'
```

```

.SBTTL PRAMPKT - PRINT RAM AND PACKET DATA
:
:PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
:WHEN THE RAM DATA DOES NOT MATCH.
:
:INPUTS:
:
:      R4      POINTER TO COMMAND PACKET
:
:IMPLICIT INPUTS:
:
:      RAMDATA  DATA AS READ FROM THE RAM
:      RAMSIZ   NUMBER OF BYTES IN PACKET
:               IF RAMSIZ=0 THEN DEFAULT TO 8.
:
:IMPLICIT OUTPUTS:
:
:      RAMSIZ  SET TO 0
:
PRAMPKT:
      SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
      MOV      #RAMDATA,R1 ;DATA FROM THE RAM
      CLR      R2      ;INIT BYTE NUMBER
5$:    CMPB     (R1)+,(R4)+ ;COMPARE EXPECTED, RECEIVED
      BNE     7$      ;BR IF NO MATCH
      FORCERROR      7$,NOTSSR
      BR     10$
7$:    MOVB    -1(R1),R5 ;GET RCV RAM DATA
      MOVB    -1(R4),R3 ;GET EXPD PACKET DATA
      XOR     R5,R3     ;XOR EXPD/RCV
      BIC     #177400,R3 ;LOW BYTE ONLY
      MOVB    -1(R1),RECV ;GET RECEIVED RAM DATA
      MOVB    -1(R4),EXPD ;GET EXPECTED RAM DATA
      PRINTB  #RAMASC,R2,RECV,EXPD,R3
      MOV     R3,-(SP)
      MOV     EXPD,-(SP)
      MOV     RECV,-(SP)
      MOV     R2,-(SP)
      MOV     #RAMASC,-(SP)
      MOV     #5,-(SP)
      MOV     SP,R0
      TRAP   C$PNTB
10$:   ADD     #14,SP
      INC     R2      ;UPDATE BYTE COUNT
      TST    RAMSIZ  ;DEFAULT TO 8.?
      BEQ    15$     ;BR IF YES
      CMP    R2,RAMSIZ ;DONE ALL BYTES?
      BLE    5$      ;BR IF NO
      BR    25$
15$:   CMP    R2,#8. ;DONE DEFAULT NUMBER OF BYTES?
20$:   BLT    5$      ;BR IF NO
25$:   CLR    RAMSIZ ;SET DEFAULT RAMSIZ
      RTS    PC      ;RETURN
```

1764  
1765  
1766

.EVEN



```

1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785 014242
1786 014246
1787 014246 010005
1788 014250 005737 003134
1789 014254 001001
1790 014256 005001
1791 014260 010103
1792 014262 006100
1793 014264 006101
1794 014266
1795 014266 010546
1796 014270 010146
1797 014272 012746 014420
1798 014276 012746 000003
1799 014302 010600
1800 014304 104415
1801 014306 062706 000010
1802 014312
1803 014312 012746 014465
1804 014316 012746 000001
1805 014322 010600
1806 014324 104415
1807 014326 062706 000004
1808 014332 005004
1809 014334 010501
1810 014336 010300
1811 014340 001403
1812 014342 004737 017316
1813 014346 010005
1814 014350
1815 014350 012546
1816 014352 010446
1817 014354 012746 014523
1818 014360 012746 000003
1819 014364 010600
1820 014366 104415
1821 014370 062706 000010
1822 014374 005204
1823 014376 020427 000007
1824 014402 003005
    
```

.SBTTL PRMESS - PRINT CONTENTS OF MESSAGE BUFFER

```

:
:
: THIS ROUTINE PRINTS THE CONTENTS OF
: THE 7 OR 8 WORD MESSAGE BUFFER RETURNED BY THE
: TSV-05.
:
    
```

: INPUT:

```

: R0 LOW ORDER ADDRESS OF MESSAGE BUFFER
: R1 HIGH ORDER ADDRESS OF MESSAGE BUFFER
: NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
    
```

: THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE

: PRMESS:

```

: SAVREG :SAVE THE REGISTERS
MOV R0,R5 :SAVE LOW ORDER ADDRESS
TST KTENABLE :ADDRESS ABOVE 28K?
BNE 10$ :BR IF YES
CLR R1 :SET HIGH ORDER ADDRESS TO 0
10$: MOV R1,R3 :SAVE HIGH ORDER ADDRESS
ROL R0 :SHIFT BIT15 TO C BIT
ROL R1 :SHIFT TO HIGH ORDER FOR PRINTOUT
PRINTX #PROASC,R1,R5 :PRINT MESSAGE BUFFER ADDRESS
MOV R5,-(SP)
MOV R1,-(SP)
MOV #PROASC,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #10,SP
PRINTX #PRIASC :PRINT HEADER FOR CONTENTS
MOV #PRIASC,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #4,SP
CLR R4 :NUMBER OF THE NEXT WORD
MOV R5,R1 :COPY LOW ORDER ADDRESS
MOV R3,R0 :COPY HIGH ORDER ADDRESS
BEQ 20$ :BR IF NOT ABOVE 28K
JSR PC,SETMAP :SETUP PAR ADDRESS IN R0
MOV R0,R5 :GET PAR FORMAT ADDRESS ABOVE 28K
20$: PRINTX #PRASC,R4,(R5)+ :PRINT THE CONTENTS OF MEMORY BUFFER
MOV (R5)+,-(SP)
MOV R4,-(SP)
MOV #PRASC,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #10,SP
INC R4 :NUMBER OF THE NEXT
CMP R4,#7 :DONE ALL YET ?
BGT 50$ :BRANCH IF ALL DONE
    
```

1806	014404	002761				BLT	20\$	:PRINT FIRST 7 WORDS
1807	014406	032763	000200	000012		BIT	#X2.EXTF,XST2(R3)	:EXTENDED FEATUTES ON ?
1808	014414	001355				BNE	20\$	:PRINT EXTENDED STATUS WORD
1809	014416	000207			50\$:	RTS	PC	:RETURN
1810								
1811	014420	045	116	045	PROASC:	.ASCIZ	'ZNA	Message Buffer Address = %01%05'
1812	014465	045	116	045	PR1ASC:	.ASCIZ	'ZNA	Message Buffer Contents:'
1813	014523	045	116	045	PRASC:	.ASCIZ	'ZNA	Word%D1%A: %0'
1814						.EVEN		

```

1816 .SBTTL PRMSGEXP - PRINT EXPD/RCV MESSAGE BUFFERS
1817
1818
1819 :ROUTINE TO PRINT EXPECTED AND RECEIVED MESSAGE BUFFERS
1820
1821 RO - NUMBER OF WORDS IN BUFFER
1822
1823 :IMPLICIT INPUTS:
1824
1825 EXPMSG - EXPECTED MESSAGE BUFFER
1826 RECMSG - RECEIVED MESSAGE BUFFER
1827 RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1828 RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1829
1830 PRMSGEXP::
1831 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1832 MOV RO,R5 ;SAVE NUMBER OF WORDS
1833 MOV RCVLOADD,RO ;GET RECV LOW ADDRESS
1834 MOV RO,R4 ;COPY LOW ADDRESS
1835 MOV RCVHIADD,R1 ;GET RECV HIGH ADDRESS
1836 ROL RO ;SHIFT BIT15 TO C BIT
1837 ROL R1 ;SHIFT TO HIGH ORDER FOR PRINTOUT
1838 PRINTX #PRMSGC,R1,R4 ;PRINT MESSAGE BUFFER ADDRESS
1839 MOV R4,-(SP)
1840 MOV R1,-(SP)
1841 MOV #PRMSGC,-(SP)
1842 MOV #3,-(SP)
1843 MOV SP,RO
1844 TRAP C$PNTX
1845 ADD #10,SP
1846 PRINTX #PRMSG1 ;PRINT HEADER FOR CONTENTS
1847 MOV #PRMSG1,-(SP)
1848 MOV #1,-(SP)
1849 MOV SP,RO
1850 TRAP C$PNTX
1851 ADD #4,SP
1852 CLR R4 ;NUMBER OF THE CURRENT WORD
1853 MOV #EXPMSG,R1 ;GET EXPD BUFFER ADDRESS
1854 MOV #RECMSG,R2 ;GET RECV BUFFER ADDRESS
1855 MOV (R1),RO ;GET EXPD
1856 MOV (R2),R3 ;GET RECV
1857 XOR RO,R3 ;XOR EXPD/RCV
1858 PRINTX #PRMSG2,R4,(R1)+,(R2)+,R3
1859 MOV R3,-(SP)
1860 MOV (R2)+,-(SP)
1861 MOV (R1)+,-(SP)
1862 MOV R4,-(SP)
1863 MOV #PRMSG2,-(SP)
1864 MOV #5,-(SP)
1865 MOV SP,RO
1866 TRAP C$PNTX
1867 ADD #14,SP
1868 INC R4 ;NUMBER OF THE NEXT
1869 CMP R4,R5 ;DONE ALL YET?
1870 BGE 50$ ;BR IF YES
1871 BR 20$ ;DO ANOTHER
1872 50$: RTS PC ;RETURN
    
```



```

1859          .SBTTL PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER
1860
1861          +
1862          :ROUTINE TO PRINT ERROR BYTES IN MESSAGE BUFFERS
1863          :ONLY THE FIRST 8 ERRORS ENCOUNTERED ARE PRINTED DUE TO SCREEN SPACE
1864
1865          :RO - NUMBER OF BYTES IN BUFFER
1866
1867          :IMPLICIT INPUTS:
1868
1869          :EXPMSG - EXPECTED MESSAGE BUFFER
1870          :RECMSG - RECEIVED MESSAGE BUFFER
1871
1872          PRBYTEXP::
1873          SAVREG          :SAVE R1-R5 UNTIL NEXT RETURN
1874          MOV R0,R5       :SAVE NUMBER OF BYTES
1875          CLR PRMNO       :INIT ERROR COUNT
1876          CLR R4         :NUMBER OF THE CURRENT BYTE
1877          MOV #EXPMSG,R1  :GET EXPD BUFFER ADDRESS
1878          MOV #RECMSG,R2  :GET RECV BUFFER ADDRESS
1879          MOV (R1),R0     :GET EXPD BYTE
1880          BIC #C<377>,R0  :CLEAR UPPER BYTE
1881          MOVB R0,PRBEXP  :SAVE FOR ERROR REPORT
1882          MOVB (R2),R3    :GET RECV BYTE
1883          BIC #C<377>,R3  :CLEAR UPPER BYTE
1884          MOVB R3,PRBREC  :FOR ERROR REPORT
1885          XOR R0,R3       :XOR EXPD/RECV
1886          CMPB (R1)+,(R2)+ :EXPD = RECV?
1887          BEQ 30$        :BR IF YES
1888          INC PRMNO       :UPDATE ERROR COUNT
1889          CMP PRMNO,#8.   :PRINTED 8?
1890          BHI 30$        :BR IF YES
1891          PRINTX #PRBMSG,R4,PRBEXP,PRBREC,R3
1892          MOV R3,-(SP)
1893          MOV PRBREC,-(SP)
1894          MOV PRBEXP,-(SP)
1895          MOV R4,-(SP)
1896          MOV #PRBMSG,-(SP)
1897          MOV #5,-(SP)
1898          MOV SP,R0
1899          TRAP C$PNTX
1900          ADD #14,SP
1901          FORCEEXIT 50$ :aad
1902          BR 35$        :aad
1903
1904          30$: FORCERROR 27$,NOTSSR :aad
1905          35$: :aad
1906          INC R4         :NUMBER OF THE NEXT
1907          CMP R4,R5      :DONE ALL YET?
1908          BGE 50$       :BR IF YES
1909          BR 20$        :DO ANOTHER
1910          50$: PRINTX #PRBTOT,PRMNO :PRINT TOTAL ERROR COUNT
1911          MOV PRMNO,-(SP)
1912          MOV #PRBTOT,-(SP)
1913          MOV #2,-(SP)
1914          MOV SP,R0
1915          TRAP C$PNTX
    
```



1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923 015474  
015474  
1924 015474 004737 007742  
1925 015500  
015500  
015500 104423  
1926  
1927

```
.SBTTL EXPREC - PRINT EXPD/RECV WORD DATA
:
:PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
:INPUTS:
:      R1      RECEIVED DATA
:      R2      EXPECTED DATA
:-
EXPREC:: BGNMSG EXPREC
        JSR    PC,PRIXOR      ;PRINT THE DATA
        ENDMSG
L10017: TRAP    C$MSG
```

1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942 015502  
015502  
1943 015502 004737 007612  
1944 015506  
015506  
015506 104423  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969 015510  
015510  
1970 015510 004737 013776  
1971 015514  
015514 104423  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979

.SBTTL EXPBREC - PRINT EXPD/RECV BYTE DATA

:+  
:PRINT ROUTINE TO DISPLAY BYTE EXPD/RECV DATA

:INPUTS:

R1 RECEIVED DATA BYTE  
R2 EXPECTED DATA BYTE

EXPBREC:: BGNMSG EXPBREC

JSR PC,PRIBXOR ;PRINT THE DATA

ENDMSG

L10020: TRAP C\$MSG

.SBTTL RAMERR - PRINT RAM AND PACKET DATA

:+  
:PRINT ROUTINE TO DISPLAY RAM/PACKET DATA

:INPUTS:

R4 POINTER TO COMMAND PACKET

:IMPLICIT INPUTS:

RAMDATA DATA AS READ FROM THE RAM  
RAMSIZ NUMBER OF BYTES IN PACKET  
IF RAMSIZ=0 THEN DEFAULT TO 8.

:IMPLICIT OUTPUTS:

RAMSIZ SET TO 0

RAMERR:: BGNMSG RAMERR

JSR PC,PRAMPKT ;PRINT RAM/PACKET DATA

ENDMSG

L10021: TRAP C\$MSG

.SBTTL RAMTADD - PRINT TEST ADDRESS, RAM AND PACKET DATA

:+  
:PRINT ROUTINE TO DISPLAY RAM/PACKET DATA

:INPUTS:



```

1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996 015516
      015516
1997 015516 004737 010274
1998 015522 004737 013776
1999 015526
      015526
      015526 104423
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014 015530
      015530
2015 015530 042701 177400
2016 015534 042702 177400
2017 015540 004737 010066
2018 015544 004737 007742
2019 015550
      015550
      015550 104423
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
    
```

```

      R4      POINTER TO COMMAND PACKET
:IMPLICIT INPUTS:
      RAMDATA  DATA AS READ FROM THE RAM
      RAMSIZ   NUMBER OF BYTES IN PACKET
              IF RAMSIZ=0 THEN DEFAULT TO 8.
      ERRHI    HIGH ORDER TEST ADDRESS
      ERRLO    LOW ORDER TEST ADDRESS
:IMPLICIT OUTPUTS:
      RAMSIZ   SET TO 0
:-
      BGNMSG   RAMTADD
RAMTADD::
      JSR      PC,PRITADD      ;PRINT TEST ADDRESS
      JSR      PC,PRAMPKT     ;PRINT RAM/PACKET DATA
      ENDMSG
L10022:
      TRAP    C$MSG
    
```

```

      .SBTTL   RAMEXP - PRINT RAM EXPD/RECV DATA
:PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
:INPUTS:
      R1      RECEIVED DATA
      R2      EXPECTED DATA
      R4      CONTROLLER RAM ADDRESS
:-
      BGNMSG   RAMEXP
RAMEXP::
      BIC      #^C<377>,R1    ;SAVE EXPD RAM DATA BYTE
      BIC      #^C<377>,R2    ;SAVE EXPD RAM DATA BYTE
      JSR      PC,PRIRAM     ;PRINT THE RAM ADDRESS
      JSR      PC,PRIXOR     ;PRINT THE DATA
      ENDMSG
L10023:
      TRAP    C$MSG
    
```

```

      .SBTTL   TIMEXP - PRINT TIMER A,B AND EXP/REC
:PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
:AND TIMER A,B HEADER MESSAGE
:INPUTS:
      R1      RECEIVED DATA
      R2      EXPECTED DATA
    
```

```

2031
2032
2033 015552
      015552
2034 015552
      015552 012746 015600
      015556 012746 000001
      015562 010600
      015564 104415
      015566 062706 000004
2035 015572 004737 007742
2036 015576
      015576
      015576 104423
2037
2038
2039 015600 045 116 045 TIMSGO: .ASCIZ '%N% TIMER A STATUS IS IN BIT 3%N% TIMER B STATUS IS IN BIT 2'
2040 .EVEN
    
```

```

:-
TIMEXP: BGNMSG TIMEXP
        PRINTX #TIMSGO          :PRINT HEADER
        MOV    #TIMSGO,-(SP)
        MOV    #1,-(SP)
        MOV    SP,R0
        TRAP   CSPNTX
        ADD    #4,SP
        JSR    PC,PRIXOR        :PRINT THE DATA
        ENDMSG
L10024: TRAP   CSMSG
    
```

```

2042                                     .SBTTL  BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS
2043
2044                                     :+
2045                                     :PRINT ROUTINE FOR TSSR ERRORS ON DATA TRANSFERS
2046                                     :INPUTS:
2047                                     :
2048                                     :
2049                                     :
2050                                     :
2051                                     :
2052                                     :
2053                                     :
2054                                     :-
2055 015700                               BGNMSG  BADSSR
2056 015700                               BADSSR::
2057 015700 010246                        MOV     R2,-(SP)                ;SAVE DATA TRANSFERRED
2058 015702 042702 177400                  BIC     #177400,R2           ;GET JUST ONE BYTE
2059 015706 010246                        PRINTB  #XFERASC,R2
2060 015710 012746 015740                  MOV     R2,-(SP)
2061 015714 012746 000002                  MOV     #XFERASC,-(SP)
2062 015720 010600                        MOV     #2,-(SP)
2063 015722 104414                        MOV     SP,R0
2064 015724 062706 000006                  TRAP   C$PNTB
2065 015730 012602                        ADD     #6,SP
2066 015732 004737 006024                  MOV     (SP)+,R2           ;RESTORE R2
2067 015736                                JSR     PC,PRITSSR         ;DECODE TSSR CONTENTS
2068 015736                                ENDMSG
2069 015736 104423                        L10025: TRAP   C$MSG
2070 015740 045 116 045 XFERASC:          .ASCIZ  '%X%A Data Transferred = %03'
2071
2072
2073
    
```

2065  
2066  
2067  
2068  
2069  
2070  
2071

.SBTTL    GLOBAL SUBROUTINES SECTION

:++  
: THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES  
: THAT ARE USED IN MORE THAN ONE TEST.  
:--

```

2073          .SBTTL  SOFINIT - SOFT INITIALIZE OF CONTROLLER
2074
2075          :+
2076          :ROUTINE TO DO A SOFT INITIALIZE OF THE CONTROLLER
2077          :BY WRITING INTO THE TSSR REGISTER. AFTER THE INIT,
2078          :THE TSSR REGISTER IS TESTED FOR ERRORS. ANY ERRORS
2079          :DETECTED SHOULD BE TREATED AS DEVICE FATAL ERRORS.
2080
2081          :
2082          :INPUTS:
2083
2084          R5      ADDRESS OF FIRST REGISTER
2085
2086          :OUTPUTS:
2087
2088          R0      CONTENTS OF TSSR, IF ERROR
2089          CARRY   SET IF INIT WAS OKAY
2090          CLEAR  IF FATAL ERROR
2091
2092          :CALLING SEQUENCE:
2093
2094          MOV     #ADDRESS,R5
2095          JSR     PC,SOFINIT
2096          BCS    CONTINUE
2097          ERRDF                    ;REPORT FATAL ERROR
2098
2099          :-
2100
2101          SOFINIT::
2102          SAVREG                    ; SAVE THE REGISTERS
2103          MOV     #0,TSSR(R5)        ; DO THE INIT.
2104          JSR     PC,WAITF           ; WAIT FOR SSR
2105          MOV     TSSR(R5),R0       ; GET THE TSSR REGISTER
2106          MOV     R0,R4             ; TSSR CONTENTS
2107          BIC     #^C<HIADDR!OFL>,R4
2108          BIS     #SSR!NBA,R4      ; R4 HAS EXPECTED CONTENTS
2109          CMP     R4,R0             ; ONLY EXPECTED BITS SET ?
2110          BEQ     $$                ; BRANCH IF OKAY
2111          CLC                     ; CLEAR THE CARRY FOR ERROR
2112          BR     10$                ; GO TO EXIT
2113          SEC                     ; SET THE CARRY BIT
2114          RTS     PC                ; RETURN TO CALLER
    
```

2116  
 2117  
 2118  
 2119  
 2120  
 2121  
 2122  
 2123  
 2124  
 2125  
 2126  
 2127  
 2128  
 2129  
 2130  
 2131  
 2132  
 2133  
 2134  
 2135  
 2136 016044  
 2137 016044  
 2138 016050 010004  
 2139 016052 032700 100000  
 2140 016056 001004  
 2141 016060 032700 174077  
 2142 016064 001023  
 2143 016066 000424  
 2144 016070 032700 000200  
 2145 016074 001011  
 2146 016076 032700 000040  
 2147 016102 001414  
 2148 016104 042704 177761  
 2149 016110 020427 000016  
 2150 016114 001007  
 2151 016116 000410  
 2152 016120 032700 000040  
 2153 016124 001405  
 2154 016126 032700 000006  
 2155 016132 001002  
 2156 016134 000241  
 2157 016136 000401  
 2158 016140 000261  
 2159 016142 000207  
 2160

.SBTTL CHKAMB - CHECK TSSR FOR AMBIGUITY

```

: *
: THIS ROUTINE TESTS THE CONTENTS OF THE TSSR REGISTER
: FOR AMBIGUITY

```

: INPUT:

RO CONTENTS OF TSSR

: OUTPUT:

RO CONTENTS OF TSSR

CARRY SET - NO AMBIGUITY  
 CLR - AMBIGUOUS CONTENTS

CHKAMB:

```

SAVREG ;SAVE THE GENERAL REGISTERS
MOV RO,R4 ;CONTENTS OF TSSR
BIT #SC,RO ;IS BIT 15 SET ?
BNE 5$ ;BRANCH IF YES
BIT #^C<NBA!OFL!SSR!HIADDR>,RO ;ANY OTHER BITS SET ?
BNE 40$ ;MUST BE AN ERROR
BR 45$ ;RETURN WITH SUCCESS
5$: BIT #SSR,RO ;IS READY BIT SET ?
BNE 10$ ;BRANCH IF READY BIT IS SET.
BIT #BIT5,RO ;IS FATAL ERROR BIT SET ?
BEQ 40$ ;ERROR IF NOT
BIC #^CTERCLS,R4 ;CLEAR ALL BUT TERMINATION CODE
CMP R4,#16 ;ALL THREE BITS MUST BE SET
BNE 40$ ;ERROR IF NOT SET
BR 45$ ;OK IF ALL ARE SET
10$: BIT #BIT5,RO ;IS FATAL ERROR BIT SET ?
BEQ 45$ ;ERROR IF BIT IS SET WITH SSR
BIT #BIT2!BIT1,RO ;IS THIS A FUNCTION REJECT
BNE 45$ ;BR, IF TSSR IS OK
40$: CLC ;AMBIGUOUS CONTENTS
BR 50$
45$: SEC ;SHOW SUCCESS - NO AMBIGUITY
50$: RTS PC ;RETURN TO CALLER

```

```

2162          .SBTTL ENAINT,DSBINT - ENABLE/DISABLE INTERRUPTS
2163          :
2164          : DEFAULT DISPLAY INTERRUPT HANDLERS.
2165          : IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
2166          : OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
2167          :
2168          :
2169          : BIT DEFINITIONS FOR "INTMASK" AND "INTFLAG" BYTES:
2170          :
2171          :          IOKCKIN=BIT7      ; DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
2172          :          IOKSTP=BIT0      ; EXPECT "STOP" INTERRUPT.
2173          :
2174          : INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
2175          : INTMASK: .BYTE 0
2176          : INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
2177          : INTFLAG: .BYTE 0
2178          :
2179          : SAVED INTERRUPT VECTOR:
2180          : INTVEC: .WORD 0
2181          : SAVE CPU PC
2182          : INTCPC: .WORD 0
2183          :
2184          : SUBROUTINE TO ENABLE INTERRUPTS:
2185          : ENAINT: MOV R0,-(SP) ;SAVE R0
2186          :          MOV IVEC,R0 ;GET POINTER TO VECTORS
2187          :          MOV #INTR,(R0)+ ;SET UP INTERRUPT VECTOR
2188          :          MOV #PRI07,(R0)+
2189          :          MOV (SP)+,R0 ;RESTORE R0
2190          :          MOV (SP),-(SP)
2191          :          MOV #0,2(SP) ;SET CPU TO LEVEL 0
2192          :          RTI
2193          :
2194          : SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 7)
2195          : DSBINT: MOV (SP),-(SP)
2196          :          MOV #PRI07,2(SP)
2197          :          RTI
2198

```

```
2200                    .SBTTL    INTR    - INTERRUPT HANDLERS
2201
2202 016216                    BGNSRV    INTR                    ;DEFINE INTERRUPT ENTRY
     016216                    INTR::
2203 016216    012737    000001    002224                    MOV        #1,INTRECV                    ;SET FLAG TO SHOW INTERRUPT RECEIVED
2204 016224    105037    016145                    CLRB       INTFLAG                    ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2205 016230    132737    000001    016144                    BITB       #IOKSTP,INTMASK ;EXPECTING STOP INTERRUPT?
2206 016236    001003                    BNE        1$                    ;BR IF YES
2207 016240    152737    000001    016145                    BISB       #IOKSTP,INTFLAG ;NO. SET THE ERROR FLAG.
2208
2209                    ;SAVE REGISTERS, MSG BUFFER, ETC.
2210 016246                    1$:
2211 016246                    ENDSRV
     016246                    L10026:
     016246    000002                    RTI
2212
2213
```



```

2215          .SBTTL WAITF - WAIT FOR SUBSYSTEM READY
2216          :
2217          : SUBROUTINE TO WAIT FOR THE SUBSYSTEM READY FLAG
2218          :
2219          : INPUTS:
2220          :
2221          : R5      ADDRESS OF FIRST DEVICE REGISTER
2222          :
2223          : OUTPUTS:
2224          :
2225          : R0      CONTENTS OF LAST TSSR READ
2226          : CARRY   SET - READY BIT SET
2227          :          CLR - TIMEOUT WAITING FOR READY
2228          :
2229 016250 000401 WAITF:: BR      1$      ;NOP WHEN SUPER FIXED
2230 016252          BREAK          ; DO A SUPVSR BREAK FIRST.
2231 016254 104422 TRAP          CSBRK
2232 016260 012746 003000 1$: MOV    #3000,-(SP) ;300 MSEC TIMER
2233 016264 016500 000002 2$: MOV    TSSR(R5),R0 ;READ THE TSSR REGISTER
2234          TSTB    R0      ;TEST FOR READY BIT SET
2235 016266 100420          BMI    3$      ; EXIT ON STOP FLAG.
2236 016270          DELAY   1      ; WAIT 100 USEC
2237 016270 012727 000001 MOV    #1,(PC)+
2238 016274 000000          .WORD  0
2239 016276 013727 002116 MOV    LSDLY,(PC)+
2240 016302 000000          .WORD  0
2241 016304 005367 177772 DEC    -6(PC)
2242 016310 001375          BNE    -4
2243 016312 005367 177756 DEC    -22(PC)
2244 016316 001367          BNE    -20
2245 016320 005316          DEC    (SP) ;REDUCE DELAY COUNT
2246 016322 001356          BNE    2$ ;RETRY UNTIL TIMER EXPIRES
2247 016324 000241          CLC    ; C = 0, CONTROLLER STILL RUNNING...
2248 016326 000401          BR     4$ ;...OR HUNG-UP AFTER 300 MSEC.
2249 016330 000261          3$: SEC    ; C = 1, CONTROLLER IS STOPPED.
2250 016332 005326          4$: DEC    (SP)+ ;RESTORE STACK WITHOUT CHANGING CARRY BIT
2251 016334 000207          RTS    PC
    
```

2245 .SBTTL CHKTSSR - CHECK TSSR FOR READY

2246  
 2247  
 2248  
 2249  
 2250  
 2251  
 2252  
 2253  
 2254  
 2255  
 2256  
 2257  
 2258  
 2259  
 2260  
 2261  
 2262  
 2263

```

    :+
    :THIS ROUTINE WAITS FOR READY IN THE TSSR
    :AND TESTS FOR AMBIGUOUS BIT SETTINGS IN TSSR.
    :INPUT:
    :
    :R5 ADDRESS OF CSR REGISTERS
    :OUTPUT:
    :
    :R0 CONTENTS OF TSSR
    :CARRY SET - OKAY
    :CLR - NOT READY AMBIGUOUS, OR SC SET
    :-
    
```

```

2264 016336
2265 016336 004737 016250
2266 016342 103014
2267 016344 004737 016044
2268 016350 103006
2269 016352 032700 100000
2270 016356 001405
2271 016360 032700 074000
2272 016364 001402
2273 016366 000241
2274 016370 000401
2275 016372 000261
2276 016374 000207
    
```

```

CHKTSSR:
        JSR PC, WAITF           ;WAIT FOR READY
        BCC 20$                ;BRANCH IF TIME OUT
        JSR PC, CHKAMB         ;TSSR AMBIGOUS?
        BCC 10$                ;BR IF YES
        BIT #SC, R0            ;SPECIAL CONDITION SET?
        BEQ 15$                ;BR IF NO
        BIT #<SCE!BIE!RMR!NXM>, R0 ;ANY ERROR BITS SET?
        BEQ 15$                ;BR IF NO
10$:    CLC                    ;SET FAILURE
        BR 20$
15$:    SEC                    ;SET SUCCESS
20$:    RTS PC                 ;RETURN TO CALLER
    
```

```

2278 .SBTTL XNXM - CHECK FOR NONEXISTENT MEMORY
2279
2280 :+
2281 : ROUTINE TO TEST FOR A NEXM IN THE RANGE (R1) THRU (R2).
2282 : ON RETURN, IF 'C' = 1, (R1) = NEXM ADDRESS.
2283 : 'C' = 0, ALL ADDRESSES OK.
2284 :
2285 : CALL: MOV ADR1,R1
2286 : MOV ADR2,R2
2287 : JSR PC,NXM
2288 : RETURN ;TEST 'C' AND PROCEED.
2289 016376 012737 016430 000004 XNXM: MOV #2$,a#4 ; SET BUSERR VECTOR.
2290 016404 012737 000200 000006 MOV #PR104,a#6
2291 016412 005003 CLR R3 ; FLAG.
2292 016414 005711 1$: TST (R1) ; TEST THE ADDRESS(ES).
2293 ; IF ANY TRAP, CONTINUE AT 2$.
2294 016416 020102 CMP R1,R2 ; OTHERWISE, CONTINUE HERE.
2295 016420 001407 BEQ 3$ ; BR IF FINISHED (NO NEXM'S).
2296 016422 062701 000002 ADD #2,R1 ; SET NEXT ADDRESS...
2297 016426 000772 BR 1$ ; ...AND CONTINUE.
2298
2299 016430 005103 2$: COM R3 ; GOT ONE, SET FLAG...
2300 016432 012716 016440 MOV #3$, (SP)
2301 016436 000002 RTI ; ...AND DISMISS INTERRUPT...
2302 016440 012700 000004 3$: CLRVEC #4 ; ...AND GIVE BACK THE VECTOR.
2303 016444 104436 MOV #4,R0
2304 016446 005703 TRAP C$CVEC
2305 016450 001401 TST R3 ; DID WE CATCH ONE ??
2306 016452 000261 BEQ .+4 ; NO, 'C' = 0, SKIP NEXT.
2307 016454 000207 SEC ; YES, 'C' = 1, (R1) = NEXM ADDR.
2308 RTS PC
2309
2310 .SBTTL TSTLOOP - CHECK ITERATION COUNT
2311
2312 :+
2313 : SUBROUTINE TO EXECUTE TEST ITERATIONS.
2314 : EXIT WITH 'C' SET IF LOOPS ALLOWED AND LOOP COUNT NON-ZERO.
2315 : LOOP COUNTER IS SET BY 'BEGIN.TEST' MACRO.
2316 :
2317 : CALL: LOOPTO ARG
2318 :
2319 016456 005737 002170 TSTLOOP:: TST NOITS ; ITERATIONS INHIBITED?
2320 016456 001006 BNE 1$ ; YES.
2321 016462 005737 002204 TST QVP ; NO.
2322 016470 100403 BMI 1$ ; LOOPS DISALLOWED IN QUICK PASS.
2323 016472 005337 002216 DEC LOOPCNT ; BUMP LOOP COUNTER.
2324 016476 001002 BNE 2$
2325 016500 000241 1$: CLC ; LOOP DISALLOWED, OR DONE.
2326 016502 000401 BR 3$
2327 016504 000261 2$: SEC ; LOOP ENABLED.
2328 016504 000261 3$: RTS PC
2329 016504 000207

```

2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353  
2354  
2355  
2356  
2357  
2358  
2359  
2360  
2361  
2362  
2363  
2364  
2365  
2366  
2367  
2368  
2369  
2370  
2371  
2372  
2373  
2374  
2375  
2376  
2377

```

.SBTTL TSTSETUP - PRINT TEST NAME AND INIT ERROR COUNTS
:
: *
: PRINT THE NUMBER AND NAME OF EACH TEST AS WE GO ALONG.
: INCREMENT 'TESTK' TO INDICATE THE NUMBER OF TESTS
: IN THE CURRENT RUN SEQUENCE.
: CLEAR THE ERROR COUNTER AND SIGNATURE EXTENSION FLAGS.
:
: INPUT:
:
:         R0        POINTER TO TEST ID ASCIZ STRING
:
: OUTPUT:
:
:         R5        ADDRESS OF FIRST DEVICE REGISTER
:
: IMPLICIT OUTPUTS:
:
:         TSTCNT   UPDATED TO COUNT TESTS PERFORMED SINCE START OR RESTART
:
: SIDE EFFECTS:
:
:         INTERRUPT LEVEL IS RASIED TO LEVEL OF
:         THE DEVICE UNDER TEST
:
: -

```

```

016510
016510 010046
016512 005037 003154
016516 005037 016756
016522 005037 005772
016526 105037 016144
016532 013700 002202
016536 006300
016540 005737 003114
016544 001430
016546 100010
016550 052760 160000 003176
016556
016556 104455
016560 000001
016562 003740
016564 005736
016566 000407
016570 052760 160001 003176 3$
016576
016576 104455
016600 000002
016602 004335
016604 000000
016606 012737 177777 003112 2$
016614
016614 013700 002202
016620 104451
016622

```

```

TSTSETUP::
MOV     R0,-(SP)           ;SAVE THE TEST ID MESSAGE
CLR     SIFLAG             ; CLEAR 'SOFT INIT' FLAG
CLR     ERRK               ; CLEAR LOCAL ERROR COUNTER.
CLR     EXTA               ; CLEAR ERROR EXTENSION FLAG.
CLRB    INTMASK           ; CLEAR INTERRUPT MASK (CHECK ERROR)
MOV     UNITN,R0          ; GET THE UNIT NUMBER,
ASL     R0                 ; ... AND MAKE IT A WORD OFFSET.
TST     NODEV             ; DID STARTUP FIND THE DEVICE?
BEQ     4$                 ; BR IF YES
BPL     3$                 ; BR IF NOT IDLE
BIS     #16000,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
ERRDF   1,NXR,NXRERR     ; NO DEVICE HERE -- PRINT IT
TRAP    C$ERDF
        .WORD            1
        .WORD            NXR
        .WORD            NXRERR
BR      2$
BIS     #16001,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
ERRDF   2,NOINIT         ; DEVICE NOT IDLE
TRAP    C$ERDF
        .WORD            2
        .WORD            NOINIT
        .WORD            0
MOV     #-1,DUFLG         ; DROP THE UNIT
DODU    UNITN
MOV     UNITN,R0
TRAP    C$DODU
DOCLN
        ; ABORT THE PASS

```

2378	016622	104444			TRAP	CSDCLN		
2379	016624	000423			BR	5\$		
2380	016626			4\$:	RFLAGS	R0		; GET THE OPERATOR FLAGS.
	016626	104421			TRAP	C\$RFLA		
2381	016630	032700	001000		BIT	#PNT,R0		; PRINT THE TEST NUMBERS?
2382	016634	001412			BEQ	1\$		; BR IF NO
2383	016636	011600			MOV	(SP),R0		;GET THE ID MESSAGE
2384	016640				PRINTF	#TNAM,R0		;DISPLAY THE TEST ID
	016640	010046			MOV	R0,-(SP)		
	016642	012746	016704		MOV	#TNAM,-(SP)		
	016646	012746	000002		MOV	#2,-(SP)		
	016652	010600			MOV	SP,R0		
	016654	104417			TRAP	C\$PNTF		
	016656	062706	000006		ADD	#6,SP		
2385	016662	005237	002214	1\$:	INC	TSTCNT		; BUMP TEST COUNTER.
2386	016666				SETPRI	IPRI		;PRIORITY THAT OF DEVICE
	016666	013700	002212		MOV	IPRI,R0		
	016672	104441			TRAP	C\$SPRI		
2387	016674	005726		5\$:	TST	(SP)+		;FIX UP THE STACK
2388	016676	013705	002206		MOV	C\$RADDR,R5		; ADDRESS OF TSV REGISTERS ON UNIBUS
2389	016702	000207			RTS	PC		
2390	016704	045	123	045	TNAM:	.ASCIZ	'%SXT%A Test'	
2391						.EVEN		

```

2393
2394
2395
2396
2397
2398 016720
      016720 104421
2399 016722 030027 020000
2400 016726 001412
2401 016730
      016730 013746 016756
      016734 012746 016760
      016740 012746 000002
      016744 010600
      016746 104417
      016750 062706 000006
2402 016754 000207
2403
2404 016756 000000
2405 016760 045 101 040
2406 016777 105 122 122
2407
2408
2409
2410
2411
2412
2413 017044 005237 016756
2414 017050 010046
2415 017052 013700 002202
2416 017056 006300
2417 017060 062700 003176
2418 017064 005210
2419 017066 032710 007777
2420 017072 001001
2421 017074 005310
2422 017076 012600
2423 017100 000207
2424
2425 017102 010046
2426 017104 013700 002202
2427 017110 006300
2428 017112 016000 003176
2429 017116 042700 170000
2430 017122 020037 002174
2431 017126 103004
2432 017130 023737 016756 002172
2433 017136 103417
2434 017140
      017140 104421
2435 017142 032700 000040
2436 017146 001013
2437 017150 012737 177777 003112
2438 017156
      017156 104455
      017160 000004
      017162 016777

```

```

.SBTTL TSTEND - PRINT ERRORS RECEIVED
:
: AT END OF EACH TEST, PRINT THE NUMBER OF ERRORS RECEIVED
: IF NORMAL ERROR REPORTING IS DISABLED (FLA:IER).
:
TSTEND: RFLAGS RO
        TRAP CSRFLA
        BIT RO,#IER
        BEQ 1$ ; BR IF "IER" NOT SET.
        PRINTF #ESUM,ERRK ; PRINT ERROR COUNT.
        MOV ERRK,-(SP)
        MOV #ESUM,-(SP)
        MOV #2,-(SP)
        MOV SP,RO
        TRAP C$PNTF
        ADD #6,SP
1$: RTS PC

ERRK: 0 ; LOCAL ERROR COUNT.
ESUM: .ASCIZ /%A %D%A ERRORS/
EMAXDU: .ASCIZ /ERROR LIMIT REACHED -- DROPPING UNIT/
        .EVEN

.SBTTL INCERK - INCREMENT LOCAL ERROR COUNT
:
: ROUTINES TO INCREMENT LOCAL ERROR COUNT AND CHECK FOR LIMIT:
:
INCERK: INC ERRK ; INCREMENT LOCAL ERROR COUNT
        MOV RO,-(SP) ; SAVE RO
        MOV UNITN,RO ; GET UNIT NUMBER,
        ASL RO ; ... AND MAKE IT A WORD OFFSET.
        ADD #ERTABL,RO ; RO GETS ADDRESS OF ERROR TABLE ENTRY.
        INC (RO) ; INCREMENT THE DEVICE ERROR COUNT
        BIT #7777,(RO) ; DID WE OVERFLOW THE FIELD?
        BNE 1$ ; BR IF NO.
        DEC (RO) ; YES -- BACK IT UP TO 7777.
1$: MOV (SP)+,RO ; RESTORE RO
        RTS PC ; RETURN TO CALLER.

CKEMAX: MOV RO,-(SP) ; SAVE RO
        MOV UNITN,RO ; GET UNIT NUMBER
        ASL RO ; ... AND MAKE IT A WORD OFFSET
        MOV ERTABL(RO),RO ; GET ERROR TABLE ENTRY
        BIC #170000,RO ; EXTRACT ERROR COUNT FIELD
        CMP RO,GERRMAX ; IS GLOBAL LIMIT EXCEEDED FOR THIS UNIT?
        BHIS 1$ ; BR IF YES
        CMP ERRK,LERRMAX ; IS LOCAL LIMIT EXCEEDED FOR THIS TEST?
        BLO 2$ ; BR IF NO
1$: RFLAGS RO ; GET OPERATOR FLAGS
        TRAP CSRFLA
        BIT #IDU,RO ; IS DROPPING INHIBITED?
        BNE 2$ ; BR IF YES.
        MOV #-1,DUFLG ; NO -- DROP THE UNIT
        ERRDF 4,EMAXDU
        TRAP C$ERDF
        .WORD 4
        .WORD EMAXDU

```

2439	017164	0G0000		.WORD	0	
	017166			DODU	UNITN	
	017166	013700	002202	MOV	UNITN,RO	
	017172	104451		TRAP	CSDODU	
2440	017174			DOCLN		
	017174	104444		TRAP	CSDCLN	
2441	017176	012600	2\$:	MOV	(SP)+,RO	: RESTORE RO
2442	017200	000207		RTS	PC	: RETURN TO CALLER
2443						

```
2445 .SBTTL CKDROP - CHECK IF UNIT SHOULD BE DROPPED
2446
2447 :+ CHECK IF UNIT SHOULD BE DROPPED
2448 :-
2449 CKDROP: MOV R0,-(SP)
2450 FORCERROR 1$,NOTSSR
2451 RFLAGS R0
2452 TRAP CSRFLA
2453 BIT #IDU,R0
2454 BNE 1$
2455 MOV (SP),R0
2456 MOV #-1,DUFLG
2457 DODU UNITN
2458 MOV UNITN,R0
2459 TRAP CSDODU
2460 ;ABORT THE PASS
2461
2462 1$: MOV (SP)+,R0
2463 RTS PC
2464
2465 .SBTTL CONFIG - DETERMINE CONFIGURATION OF SYSTEM
2466 : SUBROUTINE - DETERMINE CONFIGURATION OF TSV05 SYSTEM.
2467 :
2468 CONFIG:
2469 JSR PC,SOFINIT
2470 RTS PC
2471
2472
2473
```



```
2475 .SBTTL KTON,KTOFF - ENABLE/DISABLE MEMORY MANAGEMENT
2476
2477 : SUBROUTINE - ENABLE MEM MGT.
2478 :
2479 017256 005737 003132 KTON: TST KTFLG : GOT KT?
2480 017262 001403 BEQ 1$ : NO.
2481 017264 012737 000001 177572 MOV #1,SRO : YES. ENABLE KT11.
2482 017272 000207 1$: RTS PC
2483
2484
2485
2486 : SUBROUTINE - DISABLE MEM MGT.
2487 :
2488 :
2489 017274 005737 003132 KTOFF: TST KTFLG : GOT KT11?
2490 017300 001405 BEQ 1$ : NO.
2491 017302 000240 NOP
2492 017304 000240 NOP
2493 017306 012737 000000 177572 MOV #0,SRO : DISABLE KT.
2494 017314 000207 1$: RTS PC
2495
2496
```

2498  
 2499  
 2500  
 2501  
 2502  
 2503  
 2504  
 2505  
 2506  
 2507  
 2508  
 2509  
 2510  
 2511  
 2512  
 2513  
 2514  
 2515  
 2516  
 2517 017316  
 2518 017316  
 2519 017322 005737 003132  
 2520 017326 001433  
 2521 017330 010102  
 2522 000006  
 2523  
 2524  
 2525  
 2526 017362 042701 000177  
 2527 017366 020137 003132  
 2528 017372 103011  
 2529 017374 010137 172354  
 2530 017400 042702 160000  
 2531 017404 062702 140000  
 2532 017410 010200  
 2533 017412 000261  
 2534 017414 000401  
 2535 017416 000241  
 2536 017420 000207  
 2537

.SBTTL SETMAP - SETUP PAR6 MAPPING

```

: *
: THIS ROUTINE SETS UP KERNEL PAR6 TP HANDLE
: AN 18 BIT ADDRESS. THE OFFSET INTO THE PAGE
: IS RETURNED BIASED TO PAR6.
: INPUTS:
:         R0      HIGH ORDER ADDRESS BITS
:         R1      LOW ORDER ADDRESS BITS
: OUTPUTS:
:         R0      OFFSET INTO BLOCK WITH PAR6 BIAS (I.E. THE ADDRESS)
:         CARRY   SET IF SUCCESS
:                CLR IF ERROR
: SETMAP:
: SAVREG                                :SAVE R1-R4 UNTIL NEXT RETURN
: TST      KTF LG                       :SYSTEM HAVE ABOVE 28K?
: BEQ      10$                          :BR IF NO
: MOV      R1,R2                         :SAVE LOW ORDER BITS
: .REPT    6
: ASR     R0                             :CONVERT WORD ADDRESS TO 32W BLOCKS
: ROR     R1                             :MAKE IT DOUBLE PRECISION
: .ENDR
: BIC     #177,R1                        :ALINE FOR LOWER 4K BOUNDARY
: CMP     R1,KTF LG                      :HIGHER THAN EXISTING MEMORY?
: BHIS    10$                            :BR IF YES
: MOV     R1,@#KIPAR6                   :SETUP MAPPING REGISTER PAR6
: BIC     #160000,R2                    :SETUP DISPLACEMENT IN PAGE
: ADD     #140000,R2                    :ADD IN PAR6 BIAS
: MOV     R2,R0                          :RETURN IN R0
: SEC                                          :SET SUCCESS
: BR      15$
: 10$:   CLC                              :SET FAILURE
: 15$:   RTS      PC                      :RETURN
    
```

```

2539          .SBTTL FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN
2540          :+
2541          : FILL MEMORY WITH A BACKGROUND PATTERN
2542          :
2543          : INPUTS:
2544          :
2545          :     RO = BACKGROUND PATTERN
2546          :     FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
2547          :     KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
2548          :
2549          : OUTPUTS:
2550          :
2551          :     NONE
2552          :
2553          :
2554          : FILLMEM:
2555          : SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
2556          : JSR          PC,KTOFF ;DISABLE KT.
2557          : MOV          R0,R3      ;COPY TEST PATTERN
2558          : MOV          FREE,R1   ;GET FIRST FREE LOCATION
2559          : MOV          FRESIZ,R2  ;SIZE OF FREE SPACE BELOW 28K.
2560          : MOV          R3,(R1)+ ;STORE A BACKGROUND WORD
2561          : DEC          R2        ;DONE ALL MEMORY IN FREE SPACE?
2562          : BGT          10$      ;BR IF NO
2563          : TST          KTFLG    ; GOT KT?
2564          : BEQ          55$      ; NO. GET OUT.
2565          : JSR          PC,KTON   ; YES. ENABLE KT.
2566          : CLR          R0        ;HIGH ORDER ADDRESS START
2567          : MOV          PST32W,R1 ;GET >28K START ADDRESS (IN 32W BLOCKS)
2568          : .REPT          6
2569          : CLC          ;CLEAR C BIT
2570          : ROL          R1      ;CONVERT BLOCKS TO WORDS
2571          : ROL          R0      ;MAKE IT DOUBLE PRECISION
2572          : .ENDR
2573          : JSR          PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
2574          : MOV          R3,(R0)+ ;STORE TEST PATTERN IN >28K ADDRESS
2575          : CMP          R0,#160000 ;END OF PAR6 MAPPING AREA?
2576          : BLO          30$      ;BR IF NO
2577          : SUB          #20000,R0 ;BACKUP INTO PAR6 MAPPING BEGIN
2578          : ADD          #200,@#KIPAR6 ;POINT TO NEXT 4K BLOCK >28K.
2579          : CMP          @#KIPAR6,KTFLG ;END OF MEMORY?
2580          : BEQ          50$      ;BR IF YES
2581          : TST          T23A    ;11/23A?
2582          : BEQ          35$      ;NO KEEP GOING
2583          : MOV          SR0,R4   ;GET SR0 CONTENTS
2584          : BIC          #177761,R4 ;CLEAR ALL BUT PAGE NUMBER
2585          : CMP          #16,R4   ;SEE IF PAGE 7
2586          : BEQ          50$      ;EXIT IF THERE
2587          : TST          T23B    ;11/23B?
2588          : BEQ          45$      ;NO KEEP GOING
2589          : CMP          @#KIPAR6,#7600 ;REACHED 18 BITS?
2590          : BHS          40$      ;YES
2591          : BR          45$      ;NO KEEP GOING
2592          : MOV          #20,SR3  ;SET 22 BIT RELOCATION
2593          : JMP          30$      ;KEEP GOING ON ETC.
2594          : JSR          PC,KTOFF ;DISABLE KT.
2595          : RTS          PC
  
```

2596  
2597

```

2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621 017660
2622 017660
2623 017664 010003
2624 017666 004737 017274
2625 017672 013701 003124
2626 017676 013702 003126
2627 017702 020311
2628 017704 001411
2629 017706 010137 002240
2630 017712 005037 002236
2631 017716 010337 002232
2632 017722 011137 002234
2633 017726 000474
2634 017730 005721
2635 017732 005302
2636 017734 003362
2637 017736 005737 003132
2638 017742 001472
2639 017744 004737 017256
2640 017750 005000
2641 017752 013701 003152
2642 000006
2643
2644
2645
2646 020006 042701 000177
2647 020012 010046
2648 020014 010146
2649 020016 004737 017316
2650 020022 010004
2651 020024 012601
2652 020026 012600
2653 020030 020314
2654 020032 001411
2655 020034 010037 002236
  
```

```

.SBTTL CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN
+
COMPARE MEMORY WITH A BACKGROUND PATTERN
INPUTS:
    RO = BACKGROUND PATTERN
    FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
    KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
OUTPUTS:
    CARRY - SET IF NO ERROR
    CARRY - CLR IF ERROR
IMPLICIT OUTPUTS:
    ERRHI - ERROR HIGH ADDRESS
    ERRLO - ERROR LOW ADDRESS
    EXPD  - EXPECTED DATA
    RECV  - RECEIVED DATA
-
CMPMEM:
    SAVREG                ;SAVE R1-R5 UNTIL NEXT RETURN
    MOV RO,R3             ;COPY TEST PATTERN
    JSR PC,KTOFF          ;DISABLE KT.
    MOV FREE,R1           ;GET FIRST FREE LOCATION
    MOV FRESIZ,R2        ;SIZE OF FREE SPACE BELOW 28K.
10$:  CMP R3,(R1)         ;FREE SPACE LOCATION EQUAL TO EXPD?
    BEQ 15$              ;BR IF YES
    MOV R1,ERRLO         ;SAVE ADDRESS IN ERROR
    CLR ERRHI            ;NO HIGH ADDRESS
    MOV R3,EXPD          ;SAVE EXPD FOR ERROR REPORT
    MOV (R1),RECV       ;SAVE RECV FOR ERROR REPORT
15$:  BR 50$              ;
    TST (R1)+            ;POINT TO NEXT ADDRESS
    DEC R2                ;DONE ALL MEMORY IN FREE SPACE?
    BGT 10$              ;BR IF NO
    TST KTFLG            ; GOT KT?
    BEQ 55$              ; NO. GET OUT.
    JSR PC,KTON          ; YES. ENABLE KT.
    CLR R0               ;HIGH ORDER ADDRESS START
    MOV PST32W,R1        ;GET >28K START ADDRESS (IN 32W BLOCKS)
    .REPT 6
    ROL R1                ;CONVERT BLOCKS TO WORDS
    ROL R0                ;MAKE IT DOUBLE PRECISION
    .ENDR
    BIC #177,R1          ;ALINE 4K BOUNDARY
    MOV RO,-(SP)         ;SAVE HIGH ORDER
    MOV R1,-(SP)         ;SAVE LOW ORDER
    JSR PC,SETMAP        ;SETUP PAR6 MAPPING REGISTER
    MOV RO,R4            ;COPY ADDRESS BIASED TO PAR6
    MOV (SP)+,R1         ;RESTORE LOW ORDER IN NON PAR6 FORMAT
    MOV (SP)+,R0         ;RESTORE HIGH ORDER IN NON PAR6 FORMAT
30$:  CMP R3,(R4)         ;ABOVE 28K LOCATION EQUAL EXPD?
    BEQ 32$              ;BR IF YES
    MOV R0,ERRHI        ;SAVE HIGH ORDER IN ERROR
  
```

2656	020040	010137	002240		MOV	R1,ERRLO	:SAVE LOW ORDER IN ERROR
2657	020044	010337	002232		MOV	R3,EXPD	:SAVE EXPD FOR ERROR REPORT
2658	020050	011437	002234		MOV	(R4),RECV	:SAVE RECV FOR ERROR REPORT
2659	020054	000421			BR	50\$	:
2660	020056	062701	000002	32\$:	ADD	#2,R1	:UPDATE NON PAR6 ADDRESS
2661	020062	005500			ADC	R0	:MAKE IT DOUBLE PRECISION ADD
2662	020064	062704	000002		ADD	#2,R4	:UPDATE PAR FORMAT ADDRESS
2663	020070	020427	160000		CMP	R4,#160000	:END OF PAR6 MAPPING AREA?
2664	020074	103755			BLO	30\$	:BR IF NO
2665	020076	162704	020000		SUB	#20000,R4	:BACKUP INTO PAR6 MAPPING BEGIN
2666	020102	062737	000200	172354	ADD	#200,@#KIPAR6	:POINT TO NEXT 4K BLOCK >28K.
2667	020110	023737	172354	003132	CMP	@#KIPAR6,KTFLG	:END OF MEMORY?
2668	020116	101744			BLOS	30\$	:BR IF NO
2669	020120	004737	017274	50\$:	JSR	PC,KTOFF	:TURN OFF MEMORY MAPPING
2670	020124	000241			CLC		:SET FAILURE
2671	020126	000403			BR	60\$	:
2672	020130	004737	017274	55\$:	JSR	PC,KTOFF	:TURN OFF MEMORY MAPPING
2673	020134	000261			SEC		:SET SUCCESS
2674	020136	000207		60\$:	RTS	PC	
2675							

```

2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697 020140
2698 020140 010446
2699 020142 010346
2700 020144 010246
2701 020146 010146
2702 020150 010546
2703 020152 016605 000012
2704 020156 004736
2705 020160 012601
2706 020162 012602
2707 020164 012603
2708 020166 012604
2709 020170 012605
2710 020172 000207
2711
  
```

```

      .SBTTL  REGSAV - SAVE R1-R5 ON STACK
      :+
      :ROUTINE TO
      :SAVE R1 THROUGH R5 ON THE STACK
      :CALLING SEQUENCE:
      :
      :       JSR      R5,REGSAV
      :
      :THIS IS A COOROUTINE WHICH TRANSFER CONTROL BACK TO
      :THE CALLING ROUTINE. AT THE END OF THE CALLING ROUTINE,
      :THE RTS PC RETURNS CONTROL TO THIS ROUTINE TO RESTORE
      :REGISTERS.
      :
      :THIS ROUTINE SHOULD ONLY BE CALLED FROM ROUTINES WHICH ARE
      :CALLED VIA A JSR PC INSTRUCTION
      :-
      :
      REGSAV:
      MOV      R4,-(SP)
      MOV      R3,-(SP)
      MOV      R2,-(SP)
      MOV      R1,-(SP)
      MOV      R5,-(SP)
      MOV      10.(SP),R5
      JSR      PC,@(SP)+
      MOV      (SP)+,R1
      MOV      (SP)+,R2
      MOV      (SP)+,R3
      MOV      (SP)+,R4
      MOV      (SP)+,R5
      RTS      PC
  
```

```

2713 .SBTTL GETPAT - GET 8 BIT PATTERN FROM OPERATOR
2714
2715
2716 ;ROUTINE TO REQUEST AN 8 BIT DATA PATTERN FROM THE OPERATOR
2717
2718 ;INPUTS:
2719
2720 NONE.
2721
2722 ;OUTPUTS:
2723
2724 RO OCTAL NUMBER FROM THE OPERATOR
2725
2726 ;CALLING SEQUENCE:
2727
2728 JSR PC,GETPAT
2729
2730 ;-
2731
2732 GETPAT::
2733 1$: SAVREG ;SAVE THE GENERAL REGISTERS
2734 1$: GMANID DATASC,PATDAT,0,377,0,377,NO
      TRAP CS$GMAN
      BR 10000$
      .WORD PATDAT
      .WORD T$CODE
      .WORD DATASC
      .WORD 377
      .WORD T$LOLIM
      .WORD T$HILIM
2735 10000$: BNCOMPLETE 1$ ;RETRY IF ERROR
      BCC 1$
      MOV PATDAT,RO ;DATA PATTERN FROM OPERATOR
      RTS PC ;RETURN TO CALLER
2736
2737
2738 ;+
2739 ;LOCAL DATA AREA
2740
2741 ;-
2742
2743 020174 104443
2744 020174 000406
2745 020200 020230 020230
      020204 020230
      020206 U00022
      020210 020232
      020212 000377
      020214 000000
      020216 000377
      020220
      020220 103367
      020222 013700 020230
      020226 000207
      020230 000000
      020232 105 116 124
      PATDAT: .WORD 0 ;TEMPORARY STORAGE FOR DATA
      DATASC: .ASCIZ 'ENTER DATA PATTERN'
      .EVEN
  
```



```

2747          .SBTTL  GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE
2748
2749          :+
2750          :ROUTINE TO ISSUE A MENU AND GET
2751          :THE OPERATOR'S RESPONSE.
2752
2753          :INPUTS:
2754
2755          :      R0      ADDRESS OF ASCIZ STRING OF MENU
2756          :      R1      MAXIMUM ALLOWABLE OPERATOR RESPONSE
2757
2758          :OUTPUTS:
2759
2760          :      R0      NUMBER OF THE OPERATOR'S SELECTION
2761
2762          :-
2763
2764          GETSEL::
2765          SAVREG          ;SAVE GENERAL REGISTERS
2766          MOV            R0,R2          ;SAVE THE MENU ADDRESS
2767          MOV            R2,R3          ;START OF MENU STRING
2768          1$:           MOV            (R3)          ;END OF ASCII ?
2769          2$:           TST            (R3)
2770          BEQ            3$             ;BRANCH IF ALL LINES DISPLAYED
2771          PRINTF        #SELASC,(R3)+  ;DISPLAY THE MENU
2772          MOV            (R3)+,-(SP)
2773          MOV            #SELASC,-(SP)
2774          MOV            #2,-(SP)
2775          MOV            SP,R0
2776          TRAP          C$PNTF
2777          ADD            #6,SP
2778          BR            2$
2779          3$:           GMANID        MENASC,MENRES,D,-1,0,-1,NO
2780          TRAP          C$GMAN
2781          BR            10001$
2782          .WORD        MENRES
2783          .WORD        T$CODE
2784          .WORD        MENASC
2785          .WORD        -1
2786          .WORD        T$LOLIM
2787          .WORD        T$HILIM
2788          10001$:      BNCOMPLETE    1$          ;RETRY IF ERROR
2789          BCC            1$
2790          MOV            MENRES,R0      ;GET THE OPERATOR'S REPLY
2791          CMP            R0,R1          ;COMPARE TO MAXIMUM ALLOWED
2792          BLOS           5$             ;BRANCH IF OK
2793          PRINTF        #MENERR        ;DISPLAY ERROR MESSAGE
2794          MOV            #MENERR,-(SP)
2795          MOV            #1,-(SP)
2796          MOV            SP,R0
2797          TRAP          C$PNTF
2798          ADD            #4,SP
2799          BR            1$             ;RETRY
2800          5$:           RTS            PC          ;RETURN TO CALLER
2801          MENERR:      .ASCIZ        '%N% *** Menu Selection Too Large ***'
2802          SELASC:      .ASCIZ        '%N%T'
2803          MENASC:      .ASCIZ        'Enter Menu Selection: '
  
```

2783  
2784 020476 000000

MENRES: .EVEN    .WORD 0

```

2786 .SBTTL CHKMAN - CHECK MANUAL INTERVENTION LEGALITY
2787
2788
2789 :ROUTINE TO TEST FOR MANUAL INTERVENTION LEGALITY.
2790
2791 :INPUT:
2792
2793 :      NONE.
2794
2795 :OUTPUT:
2796
2797 :      CARRY  0      MANUAL INTERVENTION NOT ALLOWED
2798 :            1      MANUAL INTERVENTION IS OK
2799
2800 :SIDE EFFECTS:
2801
2802 :      A MESSAGE IS DISPLAYED WARNING THAT TEST IS
2803 :      NOT EXECUTED IF MANUAL INTERVENTION IS NOT
2804 :      ALLOWED.
2805
2806 :-
2807
2808 020500 CHKMAN::
2809 020500      SAVREG      ;SAVE THE REGISTERS
2810 020504      MANUAL    ;SEE IF MANUAL INTERVENTION OK
2811 020504      TRAP      C$MANI
2812 020506      BCOMPLETE 1$      ;BRANCH IF ALLOWED
2813 020506      BCS       1$
2814 020510      PRINTF   #NOMAN   ;PRINT THE WARNING MESSAGE
2815 020510      MOV       #NOMAN,-(SP)
2816 020514      MOV       #1,-(SP)
2817 020520      MOV       SP,R0
2818 020522      TRAP     C$PNTF
2819 020524      ADD       #4,SP
2820 020530      CLC      ;CLEAR CARRY FOR ERROR
2821 020532      RTS      PC      ;RETURN
2822
2823 020534      045      116      045  NOMAN: .ASCIZ  'XNZA *** Manual Intervention not Allowed - Test Aborted ***'
2824
2825 .even
  
```

```

2819                                     .SBTTL  ENVIRN  - SETUP FREE DIAGNOSTIC SPACE
2820                                     :
2821                                     : SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
2822                                     :
2823 020630 ENVIRN: MEMORY R0
      020630 104431 TRAP C$MEM
2824 020632 010037 003124 MOV R0,FREE ; GET 1ST FREE ADDRESS...
2825 020636 062737 000002 003124 ADD #2,FREE ;...AND WORD COUNT.
2826 020644 011037 003126 MOV (R0),FRESIZ
2827 020650 162737 000004 003126 SUB #4,FRESIZ
2828 020656 013702 002012 MOV L$UNIT,R2 ; GET NUMBER OF UNITS
2829 020662 162737 000007 003126 10$: SUB #7,FRESIZ ; TAKE AWAY 7 WORDS PER UNIT
2830 020670 005302 DEC R2
2831 020672 001373 BNE 10$
2832 020674 013700 003124 MOV FREE,R0 ;GET FIRST FREE ADDRESS
2833 020700 063700 003126 ADD FRESIZ,R0 ;POINT TO LAST FREE ADDRESS
2834 020704 162700 000002 SUB #2,R0 ;BACKUP 1 WORD
2835 020710 010037 003130 MOV R0,FREEHI ;STORE LAST FREE ADDRESS
2836 020714 000240 NOP ;*****
2837 020716 012701 177520 MOV #BDVPCR,R1 ;GET BDV11 PCR ADDRESS
2838 020722 010102 MOV R1,R2 ;COPY TO R2
2839 020724 062702 000002 ADD #2,R2 ;SET THE RANGE
2840 020730 004737 016376 JSR PC,XNXM ;SEE IF WE HAVE ONE
2841 020734 103001 BCC 15$ ;OK TO SET FLAGS
2842 020736 000445 BR 40$ ;RETURN WITH FLAGS CLEAR
2843 020740 013701 177520 15$: MOV BDVPCR,R1 ;SAVE PCR CONTENTS
2844 020744 062701 000001 ADD #1,R1 ;ADD ONE TO IT
2845 020750 012702 177520 MOV #BDVPCR,R2 ;GET BDV11 PCR ADDRESS
2846 020754 005212 INC (R2) ;TRY TO WRITE TO IT
2847 020756 013703 177520 MOV BDVPCR,R3 ;GET RESULTS
2848 020762 020103 CMP R1,R3 ;DID IT CHANGE?
2849 020764 001017 BNE 20$ ;NO, MUST BE 11/23B
2850 020766 005237 003144 INC T23A ;SET THE FLAG
2851 020772 042737 170000 002120 BIC #170000,L$HIME ;SUPERVISOR COULD BE WRONG
2852 021000 000240 NOP ;BR 40$ FOR RELEASE
2853 021002 PRINTF #M8186 ;TELL THE SYSTEM TYPE
      021002 012746 005554 MOV #M8186,-(SP)
      021006 012746 000001 MOV #1,-(SP)
      021012 010600 MOV SP,R0
      021014 104417 TRAP C$PNTF
2854 021016 062706 000004 ADD #4,SP
2855 021022 000413 BR 40$ ;RETURN
2856 021024 005237 003146 20$: INC T23B ;SET THE FLAG
2857 021032 PRINTF #M8189 ;TELL THE SYSTEM TYPE
      021032 012746 005645 MOV #M8189,-(SP)
      021036 012746 000001 MOV #1,-(SP)
      021042 010600 MOV SP,R0
      021044 104417 TRAP C$PNTF
2858 021046 062706 000004 ADD #4,SP
2859 021052 000207 40$: RTS PC ;RETURN

```

```

2861                                     .SBTTL  KTINIT  - SETUP KT11 MEMORY MANAGEMENT REGISTERS
2862
2863                                     :+
2864                                     :ROUTINE TO INIT KT-11
2865                                     :-
2866
2867
2868 021054                               KTINIT:
2869 021054 005037 003132                 CLR      KTFLG                ; INIT >28K MEMORY FLAG
2870 021060 005037 003134                 CLR      KTENABLE           ; INIT TEST >28K FLAG
2871 021064 023727 002120 001577         CMP      LSHIME,#1577        ; GOT ENOUGH MEMORY (>28K)?
2872 021072 101444 9$                     BLOS     9$                  ; NO.
2873 021074 013700 000004                 MOV      @#ERRVEC,R0        ; SAVE OLD ERR VEC PTR.
2874 021100 012737 021172 000004         MOV      #2$,@#ERRVEC      ; SET ERR VEC PTR.
2875 021106 005737 177572                 TST      @#SRO              ; GOT KT11?
2876 021112 000240 9$                     NOP                               ; (TRAP IF NO).
2877 021114 013737 002120 003132         MOV      LSHIME,KTFLG      ; YES. SET KT FLAG.
2878 021122 042737 000177 003132         BIC      #177,KTFLG
2879 021130 010037 000004                 MOV      R0,@#ERRVEC
2880 021134 005000 9$                     CLR      R0                  ; RESTORE OLD ERR VEC PTR.
2881 021136 012701 172340                 MOV      #KIPAR0,R1        ; R0 = AR DATA.
2882 021142 012761 077406 177740 1$:     MOV      #77406,-40(R1)    ; R1 = KI REGS PTR.
2883 021150 010021 9$                     MOV      R0,(R1)+          ; SET DESCRIPTOR REG.
2884 021152 062700 000200                 ADD      #200,R0           ; SET KIPAR REG.
2885 021156 020027 002000                 CMP      R0,#2000          ; BUMP AR DATA BY '4k'.
2886 021162 001367 9$                     BNE     1$                  ; AT 'I/O'?
2887 021164 012741 177600                 MOV      #177600,-(R1)    ; NO.
2888 021170 000405 9$                     BR       9$                  ; YES. SET KTPAR7 FOR I/O.
2889
2890 021172 012716 021200 2$:             MOV      #6$, (SP)        ; SET UP RETURN
2891 021176 000002 9$                     RTI                          ; RTI TO NEXT LOCATION
2892
2893 021200 010037 000004 6$:             MOV      R0,@#ERRVEC      ; RESTORE OLD ERR VEC PTR.
2894
2895 021204 000207 9$:                     RTS      PC
2896

```

```

2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911 021206
2912
2913 021206 005737 002226
2914 021212 001020
2915 021214 012737 100206 021260
2916 021222 012737 021270 021262
2917 021230 012737 000006 021266
2918 021236 012737 100010 021270
2919 021244 012704 021260
2920 021250 004737 010662
2921 021254 000207
2922
2923
2924
2925
2926 021260
2927
2928 021260 000000
2929 021262 000000
2930 021264 000000
2931 021266 000000
2932
2933
2934
2935
2936 021270 000000
2937 021272 000000
2938 021274 000000
2939
2940
    
```

```

: *
: SUBROUTINE TO SET EXTENDED FEATURES SWITCH
: Requires that SOFINIT and WRTCHR have been done previous to call.
:
: INPUTS:
: R5 CURRENT UNIT NUMBER
: OUTPUTS:
: The Extended Features Switch is set.
:
: -
:
: INVERT::
:
: TST EXTFEA ; IS SWITCH SET?
: BNE 1$ ; YES, EXIT STAGE RIGHT! (or the next one outa town!)
: MOV #100206, CMDPKT ; WRT SUB-SYS MEM CMD
: MOV #WSMBK, CMDPKT+2 ; MSG BUF ADDR
: MOV #6, CMDPKT+6 ; BYTE COUNT
: MOV #100010, WSMBK ; INVERT THE SWITCH
: MOV #CMDPKT, R4 ; SET CMDPKT INTO R4
: JSR PC, WRTCHR ; DO IT
: RTS PC ; RETURN
:
: COMMAND PACKET.
:
: = <.+3>&177774 ; MUST BE ON MOD 4 BOUNDRY.
:
: CMDPKT:: 0 ; 1ST WORD IS TS05 COMMAND.
: 0 ; 2ND WORD IS THE BUFFER LOW ADDRESS.
: 0 ; 3RD WORD IS THE BUFFER HIGH ADDRESS.
: 0 ; 4TH WORD IS THE BYTE/RECORD/FILE COUNT.
:
: WRITE SUB-SYSTEM MEMORY CHARACTERISTIC BLOCK.
:
: WSMBK:: 0 ; 1ST WORD:: SEL 0
: 0 ; 2ND WORD:: SEL 2
: 0 ; 3RD WORD:: SEL 4
: .EVEN
    
```



```
2994          000006          .REPT      6
2995          .ASL          R1          ;PUT IN PLACE FOR XFER
2996          .ENDR
2997 021504    010137    003140    MOV        R1,NXMLO          ;SAVE TEST ADDRESS LOW
2998          000012          .REPT     10
2999          .ASR          R2          ;PUT IN PLACE FOR XFER
3000          .ENDR
3001 021534    042702    177700    BIC        #177700,R2       ;DON'T WANT ILA!
3002 021540    010237    003142    MOV        R2,NXMHI       ;SAVE TEST ADDRESS HIGH
3003 021544    000207          RTS          PC            ;RETURN
3004
3005
3006
3007
3008 021546          ENDMOD
```



7  
8  
9 021546  
021546  
10  
16

.TITLE TSV4 - MISCELLANEOUS SECTIONS  
BGNMOD TSV4  
TSV4::

18  
19 021546  
    021546  
20 021546 177777 177777 177777  
21 021556  
22

.SBTTL PROTECTION TABLE  
BGNPROT  
L\$PROT::  
.WORD -1, -1, -1, -1  
ENDPROT

:NO DEVICE PROTECTION REQUIRED.

```

24
25
26
27
28
29
30
31
32
33
34
35
36
37 021556
    021556
38 021556 005037 002226
39 021562 005037 003136
40 021566 012737 006360 002200
41 021574 005037 003154
42 021600 005037 003134
43 021604 005037 002302
44 021610
    021610 012700 000036
    021614 104447
45 021616
    021616 103023
46 021620 023737 002202 002012
47 021626 103070
48 021630 005737 003112
49 021634 100472
50 021636 013701 002202
51 021642 006301
52 021644 005761 003176
53 021650 001516
54 021652 032761 040000 003176
55 021660 001060
56 021662
    021662 104432
    021664 000416
57 021666
    021666 012700 000035
    021672 104447
58 021674
    021674 103052
59 021676
    021676 012700 000040
    021702 104447
60 021704
    021704 103404
61 021706
    021706 012700 000037
    021712 104447
62 021714
    021714 103031
63 021716
64 021716
    021716 104433
    
```

```

.SBTTL INITIALIZE SECTION

:++
:THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
:AT THE BEGINNING OF EACH PASS.
:
:IF "START" OR "RESTART", SET QUICK-PASS FLAG AND BUS-INIT.
:IF "CONTINUE", NOTHING IS REQUIRED.
:
:--
:+
:INSERT TEMPORARY JUMP TO ODT
:-

BGNINIT
LSINIT::
40$: CLR EXTFEA
    CLR NXMFLG
    MOV #EPR1,EPR1SW ;SET UP PRIMARY MESSAGE FOR REPLACEMENT
    CLR SIFLAG ;CLEAR "SOFT INIT" FLAG
    CLR KTENABLE ;CLEAR TEST ABOVE 28K FLAG
    CLR RAMSIZ ;CLEAR RAM SIZE FOR RAMERR ROUTINE
    READEF #EF.CONTINUE
    MOV #EF.CONTINUE,RO
    TRAP CSREFG
    BNCOMPLETE 1$
    BCC 1$
    CMP UNITN,LSUNIT ;UNIT IN RANGE?
    BHIS 4$ ;BR IF NO.
    TST DUFLG ;DROPPED UNIT?
    BMI NXTU ;BR IF YES
    MOV UNITN,R1
    ASL R1
    TST ERTABL(R1)
    BEQ SETU
    BIT #BIT14,ERTABL(R1) ;DROPPED?
    BNE NXTU
    EXIT INIT ;DO NOTHING IF "CONTINUE".
    TRAP C$EXIT
    .WORD L10030-.
1$: READEF #EF.NEW
    MOV #EF.NEW,RO
    TRAP CSREFG
    BNCOMPLETE NXTU ;TAKE NEXT UNIT IF NOT NEW PASS.
    BCC NXTU
    READEF #EF.START
    MOV #EF.START,RO
    TRAP CSREFG
    BCOMPLETE 2$
    BCS 2$
    READEF #EF.RESTART
    MOV #EF.RESTART,RO
    TRAP CSREFG
    BNCOMPLETE 31$
    BCC 31$
2$: BRESET ;1ST PASS, BUS-INIT...
    TRAP C$RESET ;BUS RESET.
    
```

```

65 021720 005037 002214      CLR      TSTCNT      ;NUMBER OF TESTS RUN IN PASS
66 021724 005037 002222      CLR      FATFLG     ;CLEAR FATAL ERROR COUNT
67 021730 005037 003144      CLR      T23A      ;CLEAR 11/23A FLAG
68 021734 005037 003146      CLR      T23B      ;CLEAR 11/23B FLAG
69      :      MOV      #340,-(SP)
70      :      MOV      #20$,-(SP)      ;RETURN TO DEBUGGER
71      :      JMP      0.ODT      ;:@ENTER THE DEBUGGER
72 021740 005037 003400      CLR      SKIPT      ;CLEAR THE SUBTEST 'SKIPPER'
73 021744      :
74 021744 012737 177777 002204 20$:      MOV      #-1,QVP      ;...QUICK VERIFY...
75 021752 004737 020630      JSR      PC,ENVIRN   ;SET ENVIRONMENT.
76 021756 004737 021054      JSR      PC,KTINIT  ;INITIALIZE KT MEMORY MANAGEMENT
77 021762 012700 003176      MOV      #ERTABL,RO
78 021766 005020 30$:      CLR      (RO)+      ;CLEAR THE ERROR TABLE
79 021770 020027 003376      CMP      RO,#ERTABE
80 021774 103774      BLO     30$
81 021776 000404      BR      4$
82 022000 005037 002204 31$:      CLR      QVP
83 022004 000137 022054      JMP      PASRPT      ;GO REPORT THE STATUS
84
85 022010 4$:
86 022010 012737 177777 002202 NEWPAS: MOV      #-1,UNITN      ;INIT UNIT NUMBER...
87 022016 005037 002220      CLR      DEVCNT     ;CLEAR COUNT OF DEVICES RUNNING
88 022022      :
89 022022 104422      :
90 022024 005237 002202 002012 NXTU:  TRAP     C$BRK      ;...AND SET NEXT UNIT NUMBER.
91 022030 023737 002202      INC      UNITN
92 022036 103423      CMP      UNITN,LSUNIT
93 022040 012737 177777 003112      BLO     SETU
94 022046 000401      MOV      #-1,DUFLG
95 022050      BR      11$
96 022050 104444      DOCLN
97 022052 000240      TRAP     C$DCLN
98 022054 11$:
99 022054 023727 002012 000001 PASRPT: NOP
100 022062 101752      CMP      LSUNIT,#1      ;HOW MANY UNITS SELECTED?
101 022064 005737 002220      BLOS    NEWPAS         ;BR IF ONLY 1
102 022070 001747      TST      DEVCNT        ;ARE ANY STILL RUNNING?
103 022072      BEQ     NEWPAS        ;BR IF NO
104 022072 104421      RFLAGS  RO
105 022074 032700 000100      TRAP     C$RFLA
106 022100 001343      BIT      #ISR,RO
107 022102      BNE     NEWPAS        ;SHOULD WE PRINT STATISTICS
108 022102      ;BR IF NO
109 022102 104424      DORPT
110 022104 000741      TRAP     C$DRPT
111 022106      BR      NEWPAS
112 022106 10$:
113 022106 013700 002202 SETU:  GPHARD  UNITN,RO      ;GET UNIT N P-TABLE POINTER.
114 022112 104442      MOV      UNITN,RO
115 022114      TRAP     C$GPHRD
116 022114 103342      BNCOMPLETE NXTU      ;BR IF UNIT NOT AVAILABLE.
117 022116 005037 003112      BCC     NXTU
118 022122 005237 002220      CLR      DUFLG        ;CLEAR 'DROPPED' FLAG.
119 022126 012001      INC      DEVCNT
120 022130 010137 002206      MOV      (RO)+,R1
121 022130      MOV      R1,CSRADDR   ;GET 1ST REGISTER ADDRESS.
122 022130      ;ADDRESS OF REGISTERS OF UNIT UNDER TEST

```

```

115
116 022134 012001          MOV      (R0)+,R1          ;GET VECTOR ADDRESS.
117                      ;MOV      (R0),R2          ;GET INTERRUPT PRIORITY
118                      ;MOV      R2,IPRI        ;SET INTERRUPT PRIORITY.
119 022136 010137 002210    MOV      R1,IVEC          ;SET INTERRUPT VECTOR POINTER...
120 022142 012721 016216    MOV      #INTR,(R1)+      ;...VECTOR...
121 022146 013721 002212    MOV      IPRI,(R1)+      ;...AND PRIORITY.
122
123 022152                1$:
124                      ;          TST      QVP          ;1ST PASS ??
125                      ;          BEQ      5$          ;NO, SKIP THE PASS 1 STUFF.
126
127
128                      ;1ST PASS, CHECK THAT DEVICE ADDRESSES ARE VALID, AND
129                      ;THAT THE DISPLAY STATUS IS PROPERLY INITIALIZED.
130
131 022152 013701 002202    MOV      UNITN,R1
132 022156 006301          ASL      R1
133 022160 052761 100000 003176  BIS      #BIT15,ERTABL(R1) ;SAY DEVICE RUNNING
134 022166 005037 005772    CLR      EXTA            ;CLEAR ERROR EXTENSION FLAG.
135 022172 023727 002012 000001  CMP      L$UNIT,#1      ;ARE WE TESTING MULTIPLE UNITS?
136 022200 101416          BLOS    10$             ;BR IF NO.
137 022202          RFLAGS    R0      ;YES -- GET OPERATOR FLAGS.
138 022204 032700 001000    TRAP    CSRFLA
139 022210 001412          BIT      #PNT,R0        ;SHOULD WE PRINT UNIT #?
140 022212          BEQ      10$      ;BR IF NOT.
141 022212 013746 002202    PRINTF  #PUNIT,UNITN    ;PRINT THE UNIT #
142 022216 012746 022304    MOV      UNITN,-(SP)
143 022222 012746 000002    MOV      #PUNIT,-(SP)
144 022226 010600          MOV      #2,-(SP)
145 022230 104417          MOV      SP,R0
146 022232 062706 000006    TRAP    C$PNTF
147 022236          ADD      #6,SP
148 022242 013701 002206    CLR      NODEV
149 022246 010102          MOV      CSRADDR,R1     ;ADDRESS OF FIRST REGISTER
150 022250 062702 000002    MOV      R1,R2          ;START OF REGISTERS
151 022254 004737 016376    ADD      #TSSR,R2       ;ADDRESS OF TSSR REGISTER
152 022260 103005          JSR      PC,XNXM        ;TEST BOTH CONTROLLER REGISTERS...
153 022262 010137 003114    BCC     2$              ;...AND BR IF ALL OK.
154 022266 012737 177777 003112  MOV      R1,NODEV       ;FLAG DEVICE AS NON-EXISTENT
155 022274          MOV      #-1,DUFLG     ;DROP THIS UNIT.
156
157                      2$:
158                      ;FINALLY, SET CPU PRIORITY AND WE'RE DONE.
159
160                      5$:
161 022274          SETPRI  #PRI00          ;ENABLE INTERRUPTS.
162 022274 012700 000000    MOV      #PRI00,R0
163 022300 104444          TRAP    C$SPRI
164 022302          ENDINIT
165 022302          L10030:
166 022302 104411          TRAP    C$INIT
167
168 022304 045 116 045 PUNIT: .ASCIZ  /%N%N%A***** TESTING UNIT %D2%A *****/
169 .EVEN

```

```

160
161
162
163
164
165
166
167 022352
    022352
168 022352 010001
169 022354 006301
170 022356 052761 100000 003176
171 022364 042761 040000 003176
172 022372
    022372 010046
    022374 012746 022420
    022400 012746 000002
    022404 010600
    022406 104417
    022410 062706 000006
173 022414
    022414 000167
    022416 000026
174 022420 045 116 045
175
176
177 022446
    022446
    022446 104452
178
179
180
181
182
183
184
185
186
187
188
189 022450
    022450
190 022450 012737 177777 003112
191 022456 010001
192 022460 006301
193 022462 052761 140000 003176
194 022470 000240 000240 000240
195 022476
    022476 010046
    022500 012746 022524
    022504 012746 000002
    022510 010600
    022512 104417
    022514 062706 000006
196 022520
    022520 000167
    022522 000030
    
```

.SBTTL ADD AND DROP UNITS SECTIONS

```

:++
: THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
: TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
: OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
:--
      BGNAU
L$AU::
      MOV      R0,R1          ; GET UNIT TO BE ADDED (R0)
      ASL      R1            ; MAKE IT A WORD INDEX
      BIS      #100000,ERTABL(R1) ; SET THE 'ACTIVE' BIT
      BIC      #40000,ERTABL(R1) ; CLEAR THE 'DROPPED' BIT
      PRINTF   #1$,R0
      MOV      R0,-(SP)
      MOV      #1$,-(SP)
      MOV      #2,-(SP)
      MOV      SP,R0
      TRAP     C$PNTF
      ADD      #6,SP
      EXIT     AU
      .WORD    JSJMP
      .WORD    L10031-2-.
      .ASCIZ   /%N% UNIT %D% ADDED/
      .EVEN
    
```

```

      ENDAU          ; UNUSED.
L10031:
      TRAP     C$AU
    
```

```

:++
: THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
: TO BE REMOVED FROM THE TEST LIST.
:
: SUPVSR DOES THE 'DROPPING'. THIS IS JUST TO TELL THE MAN.
: 'DROPPED' UNITS ARE RE-SELECTED ON OPERATOR 'STA' OR 'ADD'
: COMMAND, OTHERWISE REMAIN INACTIVE. THE 'DISPLAY' COMMAND
: WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
: WHICH ARE STILL ACTIVE.
: UPON ENTRY, R0 CONTAINS THE UNIT TO BE DROPPED.
    
```

```

      BGN DU
L$DU::
      MOV      #-1,DUFLG
      MOV      R0,R1
      ASL      R1
      BIS      #140000,ERTABL(R1) ; SAY DROPPED
      240,240,240 ; ?????????
      PRINTF   #1$,R0
      MOV      R0,-(SP)
      MOV      #1$,-(SP)
      MOV      #2,-(SP)
      MOV      SP,R0
      TRAP     C$PNTF
      ADD      #6,SP
      EXIT     DU
      .WORD    JSJMP
      .WORD    L10032-2-.
    
```

```

197 022524    045    116    045 1$:    .ASCIZ /%NZA UNIT %DZA DROPPED/
198                                                    .EVEN
199 022554                                            ENDDU
     022554                                            L10032:
     022554    104453                                TRAP    C$DU
200                                                :++
201                                                : AUTO-DROP CODE SECTION.
202                                                :--
203 022556                                            BGNAUTO
     022556                                            L$AUTO::
204 022556    013705    002206                    MOV        CSRADDR,R5                    ;POINT TO DEVICE REGISTER
205 022562    012703    000550                    MOV        #360.,R3                    ;ENOUGH TIME FOR 2400' REEL TO REWIND
206 022566    004737    016250                    10$:    JSR        PC,WAITF                    ;WAIT FOR SSR TO SET
207 022572    103420                                BCS        20$                        ;LEAVE WHEN SSR IS SET
208 022574                                            DELAY      250.                       ;WAIT FOR .25 SECONDS
     022574    012727    000372                    MOV        #250.,(PC)+
     022600    000000                                .WORD      0
     022602    013727    002116                    MOV        L$DLY,(PC)+
     022606    000000                                .WORD      0
     022610    005367    177772                    DEC        -6(PC)
     022614    001375                                BNE        -4
     022616    005367    177756                    DEC        -22(PC)
     022622    001367                                BNE        -20
209 022624    005303                                DEC        R3                        ;BUMP COUNTER DOWN
210 022626    001357                                BNE        10$                        ;KEEP GOING
211 022630    004737    017202                    JSR        PC,CKDROP                   ;TRY AND DROP UNIT
212 022634                                            20$:
213 022634                                            ENDAUTO                                ; UNUSED.
     022634                                            L10033:
     022634    104461                                TRAP    C$AUTO
  
```

```

215                                     .SBTTL CLEAN-UP AND REPORT CODING SECTIONS
216
217
218     :++
219     : THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS
220     : EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).
221     : USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.
222     :--
222 022636                                BGNCLN
222 022636                                L$CLEAN::
223 022636 013705 002206                    MOV     CSRADDR,R5                ;POINT TO DEVICE REGISTER
224 022642 005737 003112                    TST     DUFLG                    ;'DROPPED' FLAG IS SET ON...
225 022646 100405                            BMI     1$                       ;...AND GROSS CONTROLLER FAULT...
226                                         ;...DON'T TRY TO XCT CLEANUP CODE.
227
228 022650 012765 000000 000002            MOV     #0,TSSR(R5)              ;DO SOFT INIT
229 022656 004737 016250                    JSR     PC,WAITF
230 022662                                1$:
231 022662                                2$:
231 022662                                L10034:
231 022662 104412                            TRAP   C$CLEAN
232
233     :++
234     : THE REPORT CODING SECTION CONTAINS THE
235     : 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
236     :--
236 022664                                BGNRPT
236 022664                                L$RPT::
237 022664                                PRINTS #DEVSUM
237 022664 012746 023126                    MOV     #DEVSUM,-(SP)
237 022670 012746 000001                    MOV     #1,-(SP)
237 022674 010600                            MOV     SP,R0
237 022676 104416                            TRAP   C$PNTS
237 022700 062706 000004                    ADD     #4,SP
238 022704 010246                            MOV     R2,-(SP)
239 022706 010346                            MOV     R3,-(SP)
240 022710 010446                            MOV     R4,-(SP)
241 022712 012704 003176                    MOV     #ERTABL,R4                ; GET START OF ERROR TABLE.
242 022716 005003                            CLR     R3                        ; CLEAR UNIT NUMBER
243 022720 011402                                1$:
243 022722 001467                            MOV     (R4),R2                  ; GET ERROR TABLE ENTRY & TEST IT.
244 022724 100066                            BEQ     4$                        ; ZERO IF UNIT NOT RUN
245 022726 032702 040000                    BPL     4$
246 022732 001015                            BIT     #BIT14,R2                ; WAS UNIT DROPPED?
247 022734 042702 170000                    BNE     2$                        ; BR IF YES
248 022740                                BIC     #^C7777,R2              ; GET ERROR COUNT FIELD
249 022740 010246                            PRINTS #DEVONL,R3,R2            ; PRINT
249 022742 010346                            MOV     R2,-(SP)
249 022744 012746 023163                    MOV     R3,-(SP)
249 022750 012746 000003                    MOV     #DEVONL,-(SP)
249 022754 010600                            MOV     #3,-(SP)
249 022756 104416                            MOV     SP,R0
249 022760 062706 000010                    TRAP   C$PNTS
250 022764 000446                            ADD     #10,SP
251 022766 020227 160000                    BR      4$
252 022772 001012                                2$:
252 022774                                CMP     R2,#160000                ; WAS UNIT NON-EXISTENT?
253 022776 010346                            BNE     3$                        ; BR IF NO
253 022776 012746 023233                    PRINTS #DEVNXR,R3
253 022776 012746 023233                    MOV     R3,-(SP)
253 022776 012746 023233                    MOV     #DEVNXR,-(SP)
    
```



```

023002 012746 000002      MOV      #2,-(SP)
023006 010600      MOV      SP,R0
023010 104416      TRAP     C$PNTS
023012 062706 000006      ADD      #6,SP
254 023016 000431      BR       4$
255 023020 020227 160001      3$:     CMP      R2,#160001      ; WAS UNIT NOT READY AT STARTUP?
256 023024 001012      BNE     30$           ; BR IF NO.
257 023026      PRINTS  #DEVNRD,R3
023026 010346      MOV      R3,-(SP)
023030 012746 023315      MOV      #DEVNRD,-(SP)
023034 012746 000002      MOV      #2,-(SP)
023040 010600      MOV      SP,R0
023042 104416      TRAP     C$PNTS
023044 062706 000006      ADD      #6,SP
258 023050 000414      BR       4$
259 023052 042702 170000      30$:    BIC      #^C7777,R2
260 023056      PRINTS  #DEVDR0,R3,R2
023056 010246      MOV      R2,-(SP)
023060 010346      MOV      R3,-(SP)
023062 012746 023376      MOV      #DEVDR0,-(SP)
023066 012746 000003      MOV      #3,-(SP)
023072 010600      MOV      SP,R0
023074 104416      TRAP     C$PNTS
023076 062706 000010      ADD      #10,SP
261 023102 062704 000002      4$:     ADD      #2,R4
262 023106 005203      INC      R3
263 023110 020427 003376      CMP      R4,#ERTABE
264 023114 103701      BLO     1$
265 023116 012604      MOV      (SP)+,R4
266 023120 012603      MOV      (SP)+,R3
267 023122 012602      MOV      (SP)+,R2
268 023124      ENDRPT      ; UNUSED.
023124      L10035:
023124 104425      TRAP     C$RPT
269
270
271 023126      045      116      045  DEVSUM: .ASCIZ  /%N%ADEVICE STATUS SUMMARY:%N/
272 023163      045      101      040  DEVONL: .ASCIZ  /%A UNIT %D3%A ONLINE, ERRORS = %D%N/
273 023233      045      101      040  DEVNXR: .ASCIZ  /%A UNIT %D3%A DROPPED, NON-EXISTENT REGISTER%N/
274 023315      045      101      040  DEVNRD: .ASCIZ  /%A UNIT %D3%A DROPPED, NOT READY AT STARTUP%N/
275 023376      045      101      040  DEVDR0: .ASCIZ  /%A UNIT %D3%A DROPPED, ERRORS = %D%N/
276      .EVEN
277
278 023446      ENDMOD
279
280
    
```

1  
2  
9  
10 023446  
023446  
16  
24

.TITLE TSV5 - HARDWARE TESTS

TSV5:: BGNMOD TSV5

```

26          .SBTTL TEST 1: INITIALIZE AFTER WRITE CHARACTERISTICS
27
28          TEST DESCRIPTION:
29
30          This test verifies that a Hardware Initialize command
31          invoked after a Write Characteristics command sets up
32          the Command, Message and Characteristic image blocks
33          in the controller ram correctly.
34
35          TEST STEPS:
36
37          REPEAT FOR LOOPCNT
38          BEGIN
39              Do WRITE CHARACTERISTICS command.
40              If the NBA bit in the TSSR register is NOT=0 then Print Error.
41              Write to TSSR register to soft initialize the controller
42              If controller RAM 310-377 NOT=0 then Print Error
43          END
44
45
46
47          BGNTST
48
49          T1::
50          MOV #TST13ID,R0          ;ASCII MESSAGE TO IDENTIFY TEST
51          JSR PC,TSTSETUP          ;DO INITIAL TEST SETUP
52          MOV #10.,LOOPCNT        ;PERFORM 10 ITERATIONS
53          T13LOOP:
54          JSR PC,T13REST          ;SET PACKET TO START-UP VALUES
55
56          MOV #TSTBLK+10.,R3      ;START OF TEST DATA
57          MOV #T13PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
58          MOV #8.,PKBCNT(R4)     ;START WITH MINIMUM ALLOWABLE VALUE
59          5$:
60          JSR PC,SOFINIT          ;WRITE TO TSSR TO SOFT INITIALIZE
61          BCS 10$                 ;BR IF SOFT INIT OKAY
62          MOV R0,R1              ;SAVE CONTENTS OF TSSR
63          ERDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
64
65          TRAP CSERDF            TRAP CSERDF
66          .WORD 100              .WORD 100
67          .WORD SFIERR           .WORD SFIERR
68          .WORD SFIMSG           .WORD SFIMSG
69
70          ;Do WRITE CHARACTERISTICS command.
71          10$: CLR FATFLG          ;CLEAR FATAL ERROR FLAG
72          MOV R4,TSDB(R5)        ;SET THE PACKET ADDRESS TO EXECUTE
73          JSR PC,CHKTSSR        ;WAIT FOR SSR TO SET
74          FORCERROR 12$         ;@DFORCE ERROR IF FORCER=1
75          BCS 15$               ;BR IF CARRY SET (GOOD RETURN)
76          MOV R0,R1              ;SAVE CONTENTS OF TSSR
77          NEXT.ERRNO
78          12$: ERDF ERRNO,T13SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
79
80          TRAP CSERDF            TRAP CSERDF
81          .WORD 101              .WORD 101
82          .WORD T13SSR           .WORD T13SSR
83          .WORD PKTSSR           .WORD PKTSSR
84
85          15$: INC FATFLG        ;SET FATAL ERROR FLAG
86          CKLOOP                 ;LOOP ON ERROR, IF FLAG SET
  
```

```

79 023576 104406                                TRAP    C$CLP1
80 023600 016501 000002                        MOV     TSSR(R5),R1      ;GET THE CONTENTS OF TSSR
81 023604 012702 000200                        MOV     #SSR,R2         ;EXPECTED CONTENTS OF TSSR
82 023610 032701 000100                        BIT     #OFL,R1         ;IS OFF-LINE BIT SET ?
83 023614 001402                                BEQ     25$              ;BRANCH IF NOT OFF-LINE
84 023616 052702 000100                        BIS     #OFL,R2         ;SET OFF-LINE IN EXPECTED DATA
85
86 023622                                ;If the NBA bit in the TSSR register is NOT=0 then Print Error.
87 023622 25$:                                FORCERROR 27$           ;@@@
88 023636 020201                                CMP     R2,R1           ;DOES EXPECTED MATCH RECEIVED ?
89 023640 001404                                BEQ     30$             ;OKAY IF MATCH
90 023642                                NEXT.ERRNO
91 023642 27$:                                ERRDF    ERRNO,T13NBA,PKTSSR ;NBA NOT ZERO
92 023642 104456                                TRAP    C$ERHRD
93 023644 000146                                .WORD  102
94 023646 024214                                .WORD  T13NBA
95 023650 012046                                .WORD  PKTSSR
96 023652 30$:                                CKLOOP                                ;LOOP ON ERROR ?
97 023652 104406                                TRAP    C$CLP1
98
99 023654                                ;Write to TSSR register to soft initialize the controller
100 023654 40$:                                JSR     PC,SOFINIT      ;WRITE TO TSSR TO SOFT INITIALIZE
101 023654 004737 015774                        FORCERROR 42$           ;@@@
102 023660 103405                                BCS     50$             ;BR IF SOFT INIT OKAY
103 023674 010001                                MOV     R0,R1           ;SAVE CONTENTS OF TSSR
104 023700 42$:                                NEXT.ERRNO
105 023700 104455                                ERRDF    ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
106 023702 000147                                TRAP    C$ERDF
107 023704 003652                                .WORD  103
108 023706 012034                                .WORD  SFIERR
109 023706 012034                                .WORD  SFIMSG
110
111 023710 50$:                                ;If controller RAM 310-377 NOT=0 then Print Error
112 023714 000310                                MOV     #310,R4         ;START WITH LOC 310
113 023716 005002                                CLR     R2              ;MEMORY EXPECTED SHOULD BE 000000
114 023722 0105065 000000                        CLRB   TSDB(R5)        ;SET MAINTENANCE MODE
115 023726 004737 016336                        JSR     PC,CHKTSSR      ;WAIT FOR SSR READY
116 023732 010465 000000                        60$: MOV     R4,TSDB(R5) ;SELECT RAM ADDRESS
117 023736 004737 016336                        JSR     PC,CHKTSSR      ;WAIT FOR SSR READY
118 023742 116501 000000                        MOV     TSBA(R5),R1     ;READ LOC CONTENTS
119 023746 62$:                                FORCERROR 62$,NOTSSR   ;@@@
120 023752 120102                                CMP     R1,R2           ;CHECK MEMORY FOR 000000
121 023756 001406                                BEQ     70$             ;BRANCH IF DATA OKAY
122 023756 62$:                                NEXT.ERRNO
123 023756 104455                                ERRDF    ERRNO,T13MEM,RAMEXP ;MEMORY NOT ZERO AFTER INIT.
124 023760 000150                                TRAP    C$ERDF
125 023762 024155                                .WORD  104
126 023764 015530                                .WORD  T13MEM
127 023766 005237 002222                        .WORD  RAMEXP
128 023772 70$:                                INC     FATFLG          ;SET THE FATAL ERROR FLAG
129 023772 104406                                CKLOOP
130 023774 118:                                ESCAPE  TST             ;EXIT ON FATAL ERROR
131 023774 104410                                TRAP    C$CLP1
132 023776 000426                                .WORD  C$ESCAPE
133 023776 000426                                .WORD  L10036-
    
```

```
119
120 024000 005204          82$:  INC    R4          ;LOOK AT NEXT RAM LOC.
121 024002 020427 000400    CMP    R4,#400      ;AT TOP OF RAM ADDRESS SPACE
122 024006 001347          BNE    60$          ;BRANCH TILL ALL MEMORY TESTED
123
124
125 024010 005737 002222          TST    FATFLG      ;ANY FATAL ERRORS ?
126 024014 001402          BEQ    160$        ;BRANCH IF NOT
127 024016 004737 017202          JSR    PC,CKDROP   ;TRY TO DROP THE UNIT
128 024022 004737 016456    160$: JSR    PC,ISTLOOP ;DONE ALL ITERATIONS?
129 024026 103002          BCC    165$        ;BR IF YES
130 024030 000137 023464          JMP    T13LOOP     ;LOOP UNTIL ITERATION COUNT DONE
131 024034          165$:  EXIT    TST
132 024034          TRAP   C$EXIT
      024034 104432          .WORD  L10036-.
      024036 000366
133
134
```

```

136
137
138
139
143 024040
144 024040 100004
145 024042 024050
146 024044 000000
147 024046 000010
148
149 024050
150 024050 024062
151 024052 000000
152 024054 000016
153 024056 000000 000000
154
155 024062
156
157
158
159 024102 111 156 151 TST13ID: .ASCIZ 'Initialization After WRITE CHARACTERISTICS'
160 024155 111 156 143 T13MEM: .ASCIZ 'Incorrect RAM Data After Init'
161 .EVEN
162 024214 127 122 111 T13NBA: .ASCIZ 'WRITE CHARACTERISTICS Command Not Accepted'
163 024267 103 157 156 T13SSR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
164
    
```

```

;+
;LOCAL STORAGE FOR THIS TEST
;-
    
```

```

T13PACKET:
    .WORD 100004 ;COMMAND PACKET FOR TEST
    .WORD T13DATA ;WRITE CHARACTERISTICS COMMAND, WITH ACK
    .WORD 0 ;ADDRESS OF CHARACTERISTICS BLOCK
    .WORD 8. ;STARTING VALUE OF BLOCK SIZE
    
```

```

T13DATA:
    .WORD T13BFR ;CHARACTERISTICS DATA BLOCK
    .WORD 0 ;ADDRESS OF MESSAGE BUFFER
    .WORD 14. ;LENGTH OF MESSAGE BUFFER
    .WORD 0,0
    
```

```

T13BFR: .BLKW 8. ;MESSAGE BUFFER
    
```

```

;LOCAL TEXT MESSAGES FOR TEST
;-
    
```

166  
 167  
 168  
 169  
 170  
 171  
 172  
 173  
 174  
 175 024356  
 176 024356  
 177 024362 012701 024040  
 178 024366 012721 100004  
 179 024372 012721 024050  
 180 024376 005021  
 181 024400 012721 000010  
 182 024404 012721 024062  
 183 024410 005021  
 184 024412 012721 000016  
 185 024416 005021  
 186 024420 005011  
 187 024422 000207  
 188 024424  
 024424  
 024424 104401

```

: +
: ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
: -

```

.EVEN

T13REST:

```

SAVREG ;SAVE THE REGISTERS
MOV #T13PACKET,R1 ;START OF THE PACKET
MOV #100004,(R1)+ ;WRITE CHARACTERISTICS WITH ACK
MOV #T13DATA,(R1)+ ;ADDRESS OF CHAR DATA BLOCK
CLR (R1)+ ;EXTENDED ADDRESS
MOV #8,(R1)+ ;SIZE OF DATA BLOCK IN BYTES
MOV #T13BFR,(R1)+ ;ADDRESS OF MESSAGE BUFFER
CLR (R1)+
MOV #14,(R1)+ ;LENGTH OF MESSAGE BUFFER
CLR (R1)+
CLR (R1)
RTS PC ;RETURN
ENDTST

```

L10036: TRAP C\$ETST

```

190
191             .SBTTL TEST 2: BASIC WRITE SUBSYSTEM MEMORY COMMAND
192
193
194           :+
195           :THIS TEST VERIFIES THAT THE WRITE SUBSYSTEM MEMORY COMMAND WITH A
196           :BSELO SELECT CODE OF 0 (NO-OP) EXECUTES CORRECTLY. IT ALSO
197           :VERIFIES THAT A WRITE SUBSYSTEM MEMORY COMMAND WITH A NON-ZERO
198           :MODE FIELD IS REJECTED. THE TEST FURTHER VERIFIES MICROPROGRAM
199           :COMMAND DECODING AND HANDLING SEQUENCES.
200           :-
201
202           BGNTST
203
204
205           T2::
206           024426 012700 026241          MOV     #TST14ID,R0      ;ASCII MESSAGE TO IDENTIFY TEST
207           024432 004737 016510          JSR     PC,TSTSETUP    ;DO INITIAL TEST SETUP
208           024436 012737 000024 002216  MOV     #20.,LOOPCNT   ;PERFORM 20 ITERATIONS
209           024444
210           024444          BGNSUB        ;////////////////// BEGIN SUBTEST ////////////////////
211           024444 104402          T2.1:
212           024446 004737 026306          JSR     PC,T14REST     ;SET PACKET TO INITIAL VALUES     TRAP     C$BSUB
213           024452 004737 026344          JSR     PC,T14RST     ;SET PACKET TO INITIAL VALUES
214           024456 012700 000000          SETPRI  #PRI00        ;LOWER PRIORITY TO ALLOW INTERRUPTS
215           024462 104441          MOV     #PRI00,R0     ;
216           024464          TRAP     C$SPRI
217           024466          5$:
218           024466          BGNSEG        ;>>>>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>>>>
219           024466 104404          JSR     PC,SOFINIT    ;DO SOFT INIT OF CONTROLLER     TRAP     C$BSEG
220           024472 103405          BCS     10$          ;BR IF SOFT INIT = OK
221           024474 010001          MOV     R0,R1        ;SAVE CONTENTS OF TSSR
222           024476          ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
223           024476 104455          TRAP     C$ERDF
224           024500 000311          .WORD  201
225           024502 003652          .WORD  SFIERR
226           024504 012034          .WORD  SFIMSG
227           024506          10$:
228           024506 012704 025600          MOV     #T14PK2,R4    ;SUBROUTINE NEEDS PACKET ADDRESS
229           024512 004737 010662          JSR     PC,WRTCHR     ;ISSUE WRITE CHARACTERISTICS
230           024516 103405          BCS     11$          ;BR, IF COMMAND ISSUED OK
231           024520 010001          MOV     R0,R1        ;SAVE CONTENTS OF TSSR
232           024522          ERRHRD  ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTIC FAILED
233           024522 104456          TRAP     C$ERHRD
234           024524 000312          .WORD  202
235           024526 005056          .WORD  WRTMSG
236           024530 012034          .WORD  SFIMSG
237           024532          11$:
238           024532 005037 002222          CLR     FATFLG        ;CLEAR FATAL ERROR FLAG
239           024536 005037 002224          CLR     INTRECV       ;CLEAR INTERRUPT RECEIVED FLAG
240           024542 012704 025160          MOV     #T14PACKET,R4 ;SET UP NEW WRT. SUBSYS MEM. COMMAND
241           024546 010465 000000          MOV     R4,TSDB(R5)   ;SET THE PACKET ADDRESS
242           024552 004737 016336          JSR     PC,CHKTSSR    ;WAIT FOR SSR TO SET
243           024556 103407          BCS     15$          ;BR IF CARRY SET (GOOD RETURN)
244           024560 010001          MOV     R0,R1        ;SAVE CONTENTS OF TSSR
245           024562          ERRDF  ERRNO,T14SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
246           024564 104455          TRAP     C$ERDF
247           024564 000313          .WORD  203

```









```

355 025146          EXIT   TST          ;ALL DONE THIS TEST
      025146 104432
      025150 001246          TRAP   C$EXIT
                                .WORD L10037-.
356
357
358      ;+
359      ;LOCAL STORAGE FOR THIS TEST
360      ;-
362          025160          .=<.+10>&177770
364 025160          T14PACKET:          ;COMMAND PACKET FOR TEST
365 025160 100206          .WORD   100206          ;WRITE SUBSYSTEM MEM. COMMAND, WITH IE, ACK
366 025162 025170          .WORD   T14DATA          ;ADDRESS OF CHARACTERISTICS BLOCK
367 025164 000000          .WORD   0
368 025166 000006          .WORD   6.          ;STARTING VALUE OF BLOCK SIZE
369 025170          T14DATA:          ;CHARACTERISTICS DATA BLOCK
370 025170          T14BS0: .BYTE   0          ;BSEL0 BYTE
371 025171          T14BS1: .BYTE   0          ;BSEL1 BYTE
372 025172 000000          T14BS2: .WORD   0          ;BSEL1 WORD
373 025174 000000          .WORD   0          ;DATA
374 025176          T14BFR: .BLKW  128.          ;MESSAGE BUFFER
375
376
378          025600          .=<.+10>&177770
380 025600          T14PK2:          ;COMMAND PACKET FOR TEST
381 025600 100204          .WORD   100204          ;WRITE CHARA. MEM. CMND., WITH IE, ACK
382 025602 025610          .WORD   T14DTA          ;ADDRESS OF SELECT DATA BLOCK
383 025604 000000          .WORD   0
384 025606 000010          .WORD   8.          ;STARTING VALUE OF BLOCK SIZE
385
386
387 025610          T14DTA:          ;SELECT DATA BLOCK
388 025610 025176          .WORD   T14BFR          ;ADDRESS OF MESSAGE BUFFER
389 025612 000000          .WORD   0
390 025614 000400          .WORD   256.          ;LENGTH OF MESSAGE BUFFER
391 025616 000000 000000          .WORD   0.0
392
393
394      ;+
395      ;LOCAL TEXT MESSAGES FOR TEST
396      ;-
397
398 025622          127      122      111  T14NBA: .ASCIZ  'WRITE SUBSYSTEM MEMORY Command Not Accepted'
399 025676          127      122      111  T142REJ: .ASCIZ  'WRITE SUBSYSTEM MEMORY Not Rejected With Non-Zero Mode Field'
400 025773          103      157      156  T14SSR: .ASCIZ  'Contents of TSSR Incorrect After WRITE SUBSYSTEM MEMORY'
401 026063          105      170      160  T14NINT: .ASCIZ  'Expected Interrupt Not Received On WRITE SUBSYSTEM MEMORY'
402 026155          111      156      143  T14TSBA: .ASCIZ  'Incorrect TSBA Address After WRITE SUBSYSTEM MEMORY'
403 026241          102      141      163  TST14ID: .ASCIZ  'Basic WRITE SUBSYSTEM MEMORY Command'
404
405          .EVEN
    
```

```

407
408
409
410
411
412
413
414
415 026306
416 026306
417 026312 012701 025160
418 026316 012721 100206
419 026322 012721 025170
420 026326 005021
421 026330 012721 000006
422 026334 005021
423 026336 005021
424 026340 005011
425 026342 000207
426
427
428 026344
429 026344
430 026350 012701 025600
431 026354 012721 100204
432 026360 012721 025610
433 026364 005021
434 026366 012721 000010
435 026372 012721 025176
436 026376 005021
437 026400 012721 000400
438 026404 005021
439 026406 005011
440 026410 005037 025176
441 026414 000207
442 026416
    026416
    026416 104401

: +
: ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
: WRITE SUBSYSTEM MEMORY COMMAND
: -

T14REST:
    SAVREG
    MOV #T14PACKET,R1
    MOV #100206,(R1)+
    MOV #T14DATA,(R1)+
    CLR (R1)+
    MOV #6,(R1)+
    CLR (R1)+
    CLR (R1)+
    CLR (R1)
    RTS PC
: SAVE THE REGISTERS
: START OF THE PACKET
: WRITE SUBSYSTEM MEM. WITH ACK, IE
: ADDRESS OF DATA BLOCK
: EXTENDED ADDRESS
: SIZE OF DATA BLOCK IN BYTES
: CLEAR BSEL0 AND BSEL1
: CLEAR SEL2
: CLEAR DATA AREA
: RETURN

T14RST:
    SAVREG
    MOV #T14PK2,R1
    MOV #100204,(R1)+
    MOV #T14DTA,(R1)+
    CLR (R1)+
    MOV #8,(R1)+
    MOV #T14BFR,(R1)+
    CLR (R1)+
    MOV #256,(R1)+
    CLR (R1)+
    CLR (R1)
    CLR T14BFR
    RTS PC
    ENDTST
: SAVE THE REGISTERS
: START OF THE PACKET
: WRITE CHARA. WITH ACK, IE
: ADDRESS OF CHARAISTICS DATA BLOCK
: EXTENDED ADDRESS
: SIZE OF DATA BLOCK IN BYTES
: MESSAGE BUFFER ADDRESS
: LENGTH OF MESSAGE BUFFER
: CLEAR 1ST LOC IN MESSAGE BUFFER
: RETURN

L10037:
    TRAP C$ETST
    
```

444

446  
 447  
 448  
 449  
 450  
 451  
 452  
 453  
 454  
 455  
 456  
 457  
 458  
 459  
 460  
 461  
 462  
 463  
 464  
 465  
 466  
 467  
 468  
 469  
 470  
 471  
 472  
 477  
 478  
 479  
 480  
 481  
 482  
 483  
 484

.SBTTL TEST 3: DMA MEMORY ADDRESSING

++  
 TEST 3

TEST DESCRIPTION

This test verifies that the controller can properly address and access all available CPU memory (other than that occupied by the diagnostic and diagnostic supervisor code) for both reading (DATI) and writing (DATO). Verified are the LSI-11 Bus drivers for all available address lines. Up to this point only 16 bits have been used for DMA transfers.

TEST STEPS

REPEAT FROM 1 TO LOOPCNT

BEGIN

- Do Subtest 1 - Verify GET STATUS selected locations
- Do Subtest 2 - Verify message packets selected locations
- Do Subtest 3 - Verify Characteristic data selected locations
- Do Subtest 4 - Verify NXM to selected invalid addresses

END

--

BGNTST

MOV #TST12ID,R0  
 JSR PC,TSTSETUP  
 MOV #10.,LOOPCNT  
 INC T3BFLG  
 JSR PC,MEMCK

T3::  
 ;ASCII MESSAGE TO IDENTIFY TEST  
 ;DO INITIAL TEST SETUP  
 ;PERFORM 10 ITERATIONS  
 ;SET TEST FLAG  
 ;CHECK MEMORY

T12LOOP:

;LOOP ON TEST LABEL

002216

026420  
 026420  
 C26420 012700 030450  
 026424 004737 016510  
 026430 012737 000012  
 026436 005237 003150  
 026442 004737 021276

026446





026536	000456						.WORD	302
026540	030552						.WORD	T12WRTSSR
026542	012046						.WORD	PKTSSR

  

```

536
537
538
539
540 026544 005037 002222
541 026550 005037 030310
542 026554 012702 030314
543 026560
544 026560 005037 003134
545 026564 012201
546 026566 005000
547 026570 005737 030310
548 026574 001407
549 026576 016200 177776
550 026602 042700 177774
551 026606 012737 000001 003134
552 026614 004737 031316
553 026620 103034
554 026622 013704 030304
555 026626 013703 030302
556 026632 004737 031666
557 026636 042703 177774
558 026642 050304
559 026644 004737 017274
560 026650 010465 000000
561 026654 004737 016336
562 026660
563 026674 103405
564 026676 010001
565 026700
566 026700
    026700 104455
    026702 000457
    026704 030476
    026706 012064
567 026710
    026710 104406
568 026712
569 026712
570 026722 020227 030446
571 026726 103002
572 026730 000137 026560
573 026734 005737 030310
574 026740 003012
575 026742 005737 003132
576 026746 001407
577 026750 012737 000001 030310
578 026756 012702 030314
579 026762 000137 026560
580 026766 004737 017274
581 026772
    026772
    026772 104403
582 026774 005737 002222
    
```

  

```

:Verify a Get Status can be fetched from each address
:Get a valid modulo-4 test address
:Do a GET STATUS command from the test address
20$: CLR FATFLG ;CLEAR FATAL ERROR FLAG
    CLR T12KT ;TEST ABOVE 28K SWITCH
    MOV #T12BLK,R2 ;POINT TO TEST PATTERN TABLE
T121LOOP:
    CLR KTENABLE ;TURN OFF ABOVE 28K TEST FLAG
    MOV (R2)+,R1 ;GET TEST PATTERN ADDRESS
    CLR R0 ;ASSUME NO TEST ABOVE 28K
    TST T12KT ;TEST ABOVE 28K THIS TIME?
    BEQ 25$ ;BR IF NO
    MOV -2(R2),R0 ;GET TEST PATTERN AGAIN
    BIC #^C<A1716>,R0 ;SAVE 18 BIT ADDRESS ONLY
    JSR #1,KTENABLE ;TURN ON ABOVE 28K TEST FLAG
    JSR PC,T12CONVERT ;CONVERT TEST PATTERN TO TEST ADDRESS
    BCC 65$ ;BR IF INVALID PACKET ADDRESS
    MOV T12LOAD,R4 ;COPY CURRENT PACKET LOW ADDRESS
    MOV T12HIADD,R3 ;COPY CURRENT PACKET HIGH ADDRESS
    JSR PC,T12SETGET ;SETUP CURRENT PACKET TO GET STATUS
    BIC #^C<A1716>,R3 ;SAVE ADDRESS BITS 17+16
    BIS R3,R4 ;SETUP 18 BIT PACKET ADDRESS
    JSR PC,KTOFF ;TURN OFF KT-11
    MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
    JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
    FORCERROR 32$
    BCS 40$ ;BR IF SSR SET IN CHKTSSR
    MOV R0,R1 ;SAVE CONTENTS OF TSSR
    NEXT.ERRNO
32$: ERRDF ERRNO,T12GETSSR,PKTGETS ;DEVICE FATAL SSR FAILED TO SET
    TRAP C$ERDF
    .WORD 303
    .WORD T12GETSSR
    .WORD PKTGETS
40$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
    TRAP C$CLP1
65$: FORCEEXIT 80$
    CMP R2,#T12TBE ;DONE ALL TSTBLK TEST PATTERNS?
    BHIS 70$ ;BR IF YES
    JMP T121LOOP ;DO ANOTHER MODULO-4 ADDRESS
70$: TST T12KT ;DONE ABOVE 28K TESTING TOO?
    BGT 80$ ;BR IF YES
    TST KTFLG ;ANY MEMORY ABOVE 28K ON SYSTEM?
    BEQ 80$ ;BR IF NO
    MOV #1,T12KT ;SET SWITCH
    MOV #T12BLK,R2 ;RESET TEST PATTERN TABLE
    JMP T121LOOP ;DO ABOVE 28K TESTING
80$: JSR PC,KTOFF ;TURN OFF KT11
    ENDSUB ;////////////////// END SUBTEST ////////////////////
    L10043:
    TRAP C$ESUB
    
```

SEQ 0137

583 027000 001402  
584 027002 004737 017202  
585 027006  
586

100\$: BEQ 100\$  
JSR PC,CKDROP

;BRANCH IF NOT  
;TRY TO DROP THE UNIT

588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617

```
.SBTTL TEST 3: SUBTEST 2: MESSAGE PACKETS TO SELECTED LOCATIONS
**
TEST 3: SUBTEST 2:
SUBTEST DESCRIPTION:
    This subtest verifies the controller can deposit message packets
    to all available memory locations.
    Write Characteristics commands are executed with message
    buffer addresses set to various addresses in each available
    memory location.
    The various addresses are determined by floating a 1 then a 0
    through the address bits.
TEST STEPS:
BEGIN
    Write to TSSR to soft initialize
    Do a WRITE CHARACTERISTICS to setup a message buffer to compare
REPEAT FOR SELECTED ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
BEGIN
    Get a valid modulo-4 test address
    Set the packet message buffer to the TEST ADDRESS
    Do a WRITE CHARACTERISTICS
    Restore the test message buffer to background pattern
END
END
--
```

618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638

```
027006          BGNSUB          ;////////// BEGIN SUBTEST ////////////
027006          104402          T3.2:          TRAP      C$BSUB
027006

:Write to TSSR to soft initialize
027010 004737 015774 JSR      PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
027014 103405 BCS      15$          ;BR IF SOFT INIT = OK
027016 NEXT.ERRNO
027016 010001 MOV      R0,R1          ;SAVE CONTENTS OF TSSR
027020 ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
                                TRAP      C$ERDF
                                .WORD    304
                                .WORD    SFIERR
                                .WORD    SFIMSG
027020 104455
027022 000460
027024 003652
027026 012034

:Do a WRITE CHARACTERISTICS to setup a message buffer to compare
15$:
027030 012704 030240 MOV      #T12PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
027030 004737 031620 JSR      PC,T12SWRT          ;SET PACKET TO WRITE CHARACTERISTICS
027034 004737 017274 JSR      PC,KT0FF          ;TURN OFF KT-11
027040 010465 000000 MOV      R4,TSDB(R5)          ;SET THE PACKET ADDRESS
027044 004737 016336 JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
027050 004737 016336 FORCERROR 17$
027054 103405 BCS      20$          ;BR IF SSR SET IN CHKTSSR
027070 103405 MOV      R0,R1          ;SAVE CONTENTS OF TSSR
027072 010001 NEXT.ERRNO
```

```

639 027074          17$:  ERRDF  ERRNO,T12WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      027074 104455                                     TRAP  C$ERDF
      027076 000461                                     .WORD 305
      027100 030552                                     .WORD T12WRTSSR
      027102 012046                                     .WORD PKTSSR

640
641
642          :Get a valid modulo-4 test address
643          :Set the packet message buffer to the test address
643          :Do a WRITE CHARACTERISTICS
644 027104 005037 002222
645 027110 012703 030314
646 027114
647 027114 012301
648 027116 010100
649 027120 042700 177774
650 027124 042701 000003
651 027130 004737 031316
652 027134 103402
653 027136 000137 027234
654 027142 012704 030240
655 027146 004737 031620
656 027152 013737 030304 030250
657 027160 013737 030302 030252
658 027166 004737 017274
659 027172 010465 000000
660 027176 004737 016336
661 027202
662 027216 103405
663 027220 010001
664 027222
665 027222          32$:  ERRDF  ERRNO,T12WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      027222 104455                                     TRAP  C$ERDF
      027224 000462                                     .WORD 306
      027226 030552                                     .WORD T12WRTSSR
      027230 012046                                     .WORD PKTSSR

666 027232          50$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      027232 104406                                     TRAP  C$CLP1

667 027234          150$:
668 027234          FORCEEXIT          160$
669 027244 020327 030446          CMP  R3,#T12TBE          ;DONE ALL TST12BLK TEST PATTERNS?
670 027250 103002          BHIS 160$          ;BR IF YES
671 027252 000137 027114          JMP  T122LOOP          ;DO ANOTHER MODULO- 4 ADDRESS
672 027256 004737 017274          160$: JSR  PC,KTOFF          ;TURN OFF KT11
673 027262          ENDSUB          ;//////////////////// END SUBTEST //////////////////////
      027262          L10044:
      027262 104403          TST  FATFLG          ;ANY FATAL ERRORS ?
      027264 005737 002222          BEQ  180$          ;BRANCH IF NOT
674 027264 005737 002222          JSR  PC,CKDROP          ;TRY TO DROP THE UNIT
675 027270 001402
676 027272 004737 017202
677 027276          180$:
678
679
680
    
```

682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732

.SBTTL TEST 3: SUBTEST 3: CHARACTERISTIC DATA SELECTED LOCATIONS

++  
 TEST 3: SUBTEST 3:  
 SUBTEST DESCRIPTION:

This subtest verifies the controller can fetch a Write Characteristics data block from all available memory locations. Write Characteristics commands are executed with characteristic data blocks at various memory addresses. The various memory addresses are determined by floating a 1 then a 0 through the address bits.

TEST STEPS:

```

BEGIN
  Write to TSSR to soft initialize

  REPEAT FOR SELECTED VALID ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
  BEGIN
    Get a valid test address
    Set the test packet characteristics data pointer to the test address.
    Store expected characteristic data in test address block
    Do a WRITE CHARACTERISTIC command
  END
END
    
```

```

BGNSUB                               ;////////// BEGIN SUBTEST ////////////
                                        T3.3:
                                        TRAP   CSBSUB
    
```

```

;Write to TSSR to soft initialize
JSR   PC,SOFINIT                       ;DO SOFT INIT OF CONTROLLER
BCS   20$                               ;BR IF SOFT INIT = OK
NEXT.ERRNO
MOV   R0,R1                             ;SAVE CONTENTS OF TSSR
ERRDF ERRNO,SFIERR,SFIMSG              ;DEVICE FATAL ERROR DURING INIT
                                        TRAP   CSERDF
                                        .WORD 307
                                        .WORD SFIERR
                                        .WORD SFIMSG
    
```

```

;Get a valid test address
20$: CLR   FATFLG                       ;CLEAR FATAL ERROR FLAG
      CLR   T12KT                       ;TEST ABOVE 28K SWITCH
      MOV   #T12BLK,R3                  ;POINT TO TEST PATTERN TABLE

T123LOOP:
      CLR   KTENABLE                     ;TURN OFF ABOVE 28K TEST FLAG
      MOV   (R3)+,R1                     ;GET TEST PATTERN ADDRESS
      MOV   R1,R0                         ;GET ADDRESS ALL '18 BITS'
      BIC   #177774,R0                   ;LEAVE ONLY A17 AND A16
      BIC   #3,R1                         ;GET RID OF A17 AND A16
      TST   T12KT                         ;TEST ABOVE 28K THIS TIME?
    
```

```

027276 104402
027276 004737 015774
027304 103405
027306 010001
027310 104455
027312 000463
027314 003652
027316 012034

027320 005037 002222
027324 005037 030310
027330 012703 030314
027334 005037 003134
027340 012301
027342 010100
027344 042700 177774
027350 042701 000003
027354 005737 030310
    
```

```

733 027360 001407          BEQ      25$          :BR IF NO
734 027362 016300 177776    MOV      -2(R3),R0      :GET TEST PATTERN AGAIN
735 027366 042700 177774    BIC      #^C<A1716>,R0 :SAVE 18 BIT ADDRESS ONLY
736 027372 012737 000001 003134  MOV      #1,KTENABLE    :TURN ON ABOVE 28K TEST FLAG
737 027400 004737 031316 25$:   JSR      PC,T12CONVERT  :CONVERT TEST PATTERN TO TEST ADDRESS
738 027404 103402          BCS      30$          :BR IF VALID TEST ADDRESS
739 027406 000137 027510    JMP      60$          :GET NEXT TEST PATTERN
740          :Set the test packet characteristics data pointer to the test address
741 027412 012704 030240 30$:   MOV      #T12PACKET,R4  :GET THE ADDRESS OF COMMAND PACKET
742 027416 004737 031620    JSR      PC,T12SWRT    :RESTORE PACKET TO STARTING VALUES
743 027422 013764 030304 000002  MOV      T12LOADD,PKLOW(R4) :STORE CHAR. DATA PTR LOW ADDRESS
744 027430 013764 030302 000004  MOV      T12HIADD,PKHI(R4)  :STORE CHAR. DATA PTR HIGH ADDRESS
745 027436 004737 031730    JSR      PC,T12CHAR    :STORE EXPECTED DATA IN DATA BLOCK
746          :Do a WRITE CHARACTERISTIC command
747 027442 004737 017274    JSR      PC,KTOFF      :TURN OFF KT-11
748 027446 010465 000000    MOV      R4,TSDB(R5)   :SET THE PACKET ADDRESS TO EXECUTE
749 027452 004737 016336    JSR      PC,CHKTSSR    :WAIT FOR SSR TO SET
750 027456          FORCERROR      32$
751 027472 103405          BCS      40$          :BR IF SSR SET IN CHKTSSR
752 027474 010001          MOV      R0,R1         :SAVE CONTENTS OF TSSR
753 027476          NEXT.ERRNO
754 027476          32$:   ERRDF   ERRNO,T12WRTSSR,PKTSSR :DEVICE FATAL SSR FAILED TO SET
          TRAP   C$ERDF
          .WORD  308
          .WORD  T12WRTSSR
          .WORD  PKTSSR
          TRAP   C$CLP1
          027476 104455
          027500 000464
          027502 030552
          027504 012046
755 027506          40$:   CKLOOP          :LOOP ON ERROR, IF FLAG SET
          027506 104406          TRAP   C$CLP1
756 027510          60$:
757 027510 020327 030446    CMP      R3,#T12TBE    :DONE ALL TSTBLK TEST PATTERNS?
758 027514 103002          BHS      65$          :BR IF YES
759 027516 000137 027334    JMP      T123LOOP      :DO ANOTHER MODULO- 4 ADDRESS
760 027522 005737 030310 65$:   TST      T12KT         :DONE ABOVE 28K TESTING TOO?
761 027526 003012          BGT      70$          :BR IF YES
762 027530 005737 003132    TST      KTFLG         :ANY MEMORY ABOVE 28K ON SYSTEM?
763 027534 001407          BEQ      70$          :BR IF NO
764 027536 012737 000001 030310  MOV      #1,T12KT      :SET SWITCH
765 027544 012703 030314    MOV      #T12BLK,R3    :RESET TEST PATTERN TABLE
766 027550 000137 027334    JMP      T123LOOP      :DO ABOVE 28K TESTING
767 027554 004737 017274 70$:   JSR      PC,KTOFF      :TURN OFF KT11
768 027560          ENDSUB          :////////////////// END SUBTEST ////////////////////
          L10045:
          TRAP   C$ESUB
          027560 104403
769 027562 005737 002222    TST      FATFLG        :ANY FATAL ERRORS ?
770 027566 001402          BEQ      75$          :BRANCH IF NOT
771 027570 004737 017202    JSR      PC,CKDROP     :TRY TO DROP THE UNIT
772 027574
773
    
```

775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813

```

.SBTTL TEST 3: SUBTEST 4: NXM TO SELECTED INVALID ADDRESSES
**
TEST 3: SUBTEST 4:
SUBTEST DESCRIPTION:

This subtest verifies the NXM error bit in the TSSR
register is set when attempting to fetch data (a characteristic
data block) from selected nonexistent locations.
If NXM fails to set it is likely that an LSI-11 Bus driver is
failing to assert an address line.
Addresses tested include all combinations of high-order address
bits (i.e bits 16-21).
*****
CAUTION

The LSI BUS drivers for all available address lines(16-21)
are only checked when running on a 11/23B system with more than
128K words of memory!
*****

TEST STEPS:

BEGIN
Write to TSSR to soft initialize
Do a write characteristic command
Invert the extended features switch

REPEAT FOR SELECTED NON-EXISTENT MEMORY ADDRESSES
BEGIN
Get an invalid test address
Set the test packet characteristics data pointer to the
test address.
Do a WRITE CHARACTERISTIC command
If TSSR register NXM bit not set then print error message
END
END

```

```

814 027574          BGNSUB          ;////////// BEGIN SUBTEST //////////
      027574          T3.4:          TRAP      C$SUB
      027574 104402
815
816
817 027576 005737 003136          TST      NXMFLG          ;GOT ENOUGH MEMORY?
818 027602 001002          BNE      10$          ;IF SET STAY
819 027604 000137 030164          JMP      NOEXTF        ;LEAVE IF NOT SET
820
821 ;Write to TSSR to soft initialize
822
823 027610 004737 015774 10$:     JSR      PC,SOFINIT        ;DO SOFT INIT OF CONTROLLER
824 027614 103405          BCS      11$          ;BR IF SOFT INIT = OK
825 027616          NEXT.ERRNO
826 027616 010001          MOV      R0,R1        ;SAVE CONTENTS OF TSSR
827 027620          ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      027620 104455          TRAP      C$ERDF
      027622 000465          .WORD    309

```

```

027624 003652
027626 012034
828
829
830
831 027630 11$: CKLOOP ;LOOP IF SELECTED
027630 104406 ;TRAP C$CLP1
832 027632 012704 030240 MOV #T12PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
833 027636 004737 031620 JSR PC,T12SWRT ;RESTORE PACKET TO STARTING VALUES
834 027642 005037 003134 CLR KTENABLE ;TURN OFF KT-11
835 027646 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS
836 027652 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
837 027656 FORCERROR 15$
838 027672 103405 BCS 17$ ;BR IF SSR SET IN CHKTSSR
839 027674 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
840 027676 NEXT.ERRNO
841 027676 15$: ERRDF ERRNO,T12WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
027676 104455 ;TRAP C$ERDF
027700 000466 ;.WORD 310
027702 030552 ;.WORD T12WRTSSR
027704 012046 ;.WORD PKTSSR
842 027706 17$: CKLOOP ;LOOP IF SELECTED
027706 104406 ;TRAP C$CLP1
843 027710 004737 021206 JSR PC,INVERT ;INVERT THE SWITCH
844
845 ;Get an invalid test address
846
847 027714 005037 002222 20$: CLR FATFLG ;CLEAR FATAL ERROR FLAG
848 027720 25$:
849 027720 013737 003142 030302 MOV NXMHI,T12HIADD ;SAVE TEST ADDRESS HIGH
850 027726 013737 003140 030304 MOV NXMLO,T12LOADD ;SAVE TEST ADDRESS LOW
851 027734 T124LOOP:
852
853 ;Set the test packet characteristics data pointer to the
854 ; test address.
855
856 027734 012704 030240 30$: MOV #T12PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
857 027740 004737 031620 JSR PC,T12SWRT ;RESTORE PACKET TO STARTING VALUES
858 027744 013764 030304 000002 MOV T12LOADD,PKLOW(R4) ;STORE CHAR. DATA PTR LOW ADDRESS
859 027752 013764 030302 000004 MOV T12HIADD,PKHI(R4) ;STORE CHAR. DATA PTR HIGH ADDRESS
860
861 ;Do a WRITE CHARACTERISTIC command
862 027760 004737 017274 JSR PC,KTOFF ;TURN OFF KT-11
863 027764 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
864 027770 004737 016250 JSR PC,WAITF ;WAIT FOR SSR TO SET
865 027774 FORCERROR 32$
866 030010 103407 BCS 40$ ;BR IF SSR SET IN CHKTSSR
867 030012 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
868 030014 NEXT.ERRNO
869 030014 32$: ERRDF ERRNO,T12WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
030014 104455 ;TRAP C$ERDF
030016 000467 ;.WORD 311
030020 030552 ;.WORD T12WRTSSR
030022 012046 ;.WORD PKTSSR
870 030024 005237 002222 40$: INC FATFLG ;SET FATAL ERROR FLAG
871 030030 40$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
030030 104406 ;TRAP C$CLP1
    
```



```

872 030032          FORCERROR      45$,NOTSSR
873 030042          ESCAPE SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      030042 104410                                     TRAP C$ESCAPE
      030044 000124                                     .WORD L10046-
874          ;If TSSR register NXM bit not set then print error message
875 030046 016501 000002 45$:
876 030046          MOV TSSR(R5),R1          ;GET TSSR CONTENTS
877 030052          FORCERROR      52$
878 030066 032701 004000          BIT #NXM,R1          ;NXM SET?
879 030072 001012          BNE 60$          ;BR IF YES
880 030074          NEXT.ERRNO
881 030074 013737 030304 002240 52$: MOV T12LOADD,ERRLO          ;MEMORY TEST ADDRESS LOW
882 030102 013737 030302 002236      MOV T12HIADD,ERRHI          ;MEMORY TEST ADDRESS HIGH
883 030110          ERRHRD ERRNO,T12NXM,ADDSR ;REPORT ADDRESS AND TSSR ERROR
      030110 104456                                     TRAP C$ERHRD
      030112 000470                                     .WORD 312
      030114 031207                                     .WORD T12NXM
      030116 012126                                     .WORD ADDSR
884
885 030120 030120 104406 60$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      030120 104406                                     TRAP C$CLP1
886 030122          FORCEEXIT      90$
887 030132 005737 003144          TST T23A          ;IS IT A 11/23A?
888 030136 001012          BNE 90$          ;YES WERE DONE
889 030140 013700 030302          MOV T12HIADD,RO          ;GET CURRENT HIGH ADDRESS
890 030144 005200 65$: INC RO          ;GET NEXT ADDRESS
891 030146 020027 000077          CMP RO,#77          ;DONE A21-A16?
892 030152 101004          BHI 90$          ;BR IF YES
893 030154 010037 030302 75$: MOV RO,T12HIADD          ;SETUP NEW HIGH ORDER ADDRESS
894 030160 000137 027734          JMP T124LOOP          ;DO ANOTHER NON-EXISTENT ADDRESS
895 030164          90$:
896 030164          NOEXTF:
897 030164 004737 017274          JSR PC,KTOFF          ;TURN OFF KT11
898 030170          ENDSUB          ;////////////////// END SUBTEST ////////////////////
      030170          L10046:
      030170 104403                                     TRAP C$ESUB
899 030172 005737 002222          TST FATFLG          ;ANY FATAL ERRORS ?
900 030176 001402          BEQ 100$          ;BRANCH IF NOT
901 030200 004737 017202          JSR PC,CKDROP          ;TRY TO DROP THE UNIT
902 030204 004737 016456 100$: JSR PC,TSTLOOP          ;SHOULD WE DO ITERATIONS?
903 030210 103002          BCC 105$          ;BR IF NO
904 030212 000137 026446          JMP T12LOOP          ;LOOP UNTIL ITERATION COUNT DONE
905 030216          105$:
906 030216 004737 017274          JSR PC,KTOFF          ;TURN OFF MEMORY MANAGEMENT
907 030222 005037 003150          CLR T3BFLG          ;CLEAR TEST FLAG
908 030226          EXIT TST          ;ALL DONE THIS TEST
      030226 104432                                     TRAP C$EXIT
      030230 001542                                     .WORD L10042-
909
910
911          ;+
912          ;LOCAL STORAGE FOR THIS TEST
913          ;-
914
916          ;=<.+10>&177770
918 030240          T12PACKET:
919 030240 100004          .WORD 100004          ;COMMAND PACKET FOR TEST
          ;WRITE CHARACTERISTICS COMMAND, WITH ACK

```

920	030242	030250	.WORD	T12DATA	:ADDRESS OF CHARACTERISTICS BLOCK
921	030244	000000	.WORD	0	
922	030246	000010	.WORD	8.	:STARTING VALUE OF BLOCK SIZE
923					
924	030250			T12DATA:	:CHARACTERISTICS DATA BLOCK
925	030250	030262	.WORD	T12BFR	:LOW ADDRESS OF MESSAGE BUFFER
926	030252	000000	.WORD	0	:HIGH ORDER OF MESSAGE BUFFER
927	030254	000016	.WORD	14.	:LENGTH OF MESSAGE BUFFER
928	030256	000000	.WORD	0,0	
929					
930	030262		T12BFR:	.BLKW 8.	:MESSAGE BUFFER
931					
932	030302	000000	T12HIADD:	.WORD 0	:HIGH ADDRESS
933	030304	000000	T12LOADD:	.WORD 0	:LOW ADDRESS
934	030306	000000	T12PAR6:	.WORD 0	:ADDRESS IN PAR FORMAT
935	030310	000000	T12KT:	.WORD 0	:TEST ABOVE 28K SWITCH
936	030312	000000	T124TST:	.WORD 0	:ADDRESS TEST BIT
937					
938					
939					
940					
941					
942	030314	000001	T12BLK:	.WORD 000001	
943	030316	000002		.WORD 000002	
944	030320	000003		.WORD 000003	
945	030322	000005		.WORD 000005	
946	030324	000006		.WORD 000006	
947	030326	000007		.WORD 000007	
948	030330	000011		.WORD 000011	
949	030332	000012		.WORD 000012	
950	030334	000013		.WORD 000013	
951	030336	000021		.WORD 000021	
952	030340	000022		.WORD 000022	
953	030342	000023		.WORD 000023	
954	030344	000041		.WORD 000041	
955	030346	000042		.WORD 000042	
956	030350	000043		.WORD 000043	
957	030352	000101		.WORD 000101	
958	030354	000102		.WORD 000102	
959	030356	000103		.WORD 000103	
960	030360	000201		.WORD 000201	
961	030362	000202		.WORD 000202	
962	030364	000203		.WORD 000203	
963	030366	000401		.WORD 000401	
964	030370	000402		.WORD 000402	
965	030372	000403		.WORD 000403	
966	030374	001001		.WORD 001001	
967	030376	001002		.WORD 001002	
968	030400	001003		.WORD 001003	
969	030402	002001		.WORD 002001	
970	030404	002002		.WORD 002002	
971	030406	002003		.WORD 002003	
972	030410	004001		.WORD 004001	
973	030412	004002		.WORD 004002	
974	030414	004003		.WORD 004003	
975	030416	010001		.WORD 010001	
976	030420	010002		.WORD 010002	

:TABLE OF ADDRESSES

977	030422	010003				.WORD	010003
978	030424	020001				.WORD	020001
979	030426	020002				.WORD	020002
980	030430	020003				.WORD	020003
981	030432	040001				.WORD	040001
982	030434	040002				.WORD	040002
983	030436	040003				.WORD	040003
984	030440	100001				.WORD	100001
985	030442	100002				.WORD	100002
986	030444	100003				.WORD	100003
987	030446	177777				.WORD	177777

T12TBE: .WORD 177777

988 ;+  
989 ;LOCAL TEXT MESSAGES FOR TEST  
990 ;-  
991

992	030450	104	115	101	TST12ID:	.ASCIZ	'DMA Memory Addressing'
993	030476	103	157	156	T12GETSSR:	.ASCIZ	'Contents of TSSR Incorrect After GET STATUS'
994	030552	103	157	156	T12WRTSSR:	.ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
995	030641	115	145	163	T12MSGBUF:	.ASCIZ	'Message Buffer Contents Incorrect After WRITE CHARACTERISTICS'
996	030737	102	141	143	T12BKGND:	.ASCIZ	'Background Pattern Disturbed By WRITE CHARACTERISTICS'
997	031025	105	170	160	T12NINT:	.ASCIZ	'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
998	031116	127	162	151	T12DPR:	.ASCIZ	'Write Characteristic data in ram does not match expected'
999	031207	124	123	123	T12NXM:	.ASCIZ	'TSSR NXM bit failed to set when non-existent memory address specifi
1000						.EVEN	
1001							

1003  
 1004  
 1005  
 1006  
 1007  
 1008  
 1009  
 1010  
 1011  
 1012  
 1013  
 1014  
 1015  
 1016  
 1017  
 1018  
 1019  
 1020  
 1021  
 1022  
 1023 031316  
 1024 031316  
 1025 031322 005037 030304  
 1026 031326 005037 030302  
 1027 031332 005037 030306  
 1028 031336 042701 170000  
 1029 031342 010005  
 1030 031344 004737 017274  
 1031 031350 013702 003124  
 1032 031354 062702 000020  
 1033 031360 060102  
 1034 031362 042702 000003  
 1035 031366 013703 003130  
 1036 031372 162703 000020  
 1037 031376 010237 030304  
 1038 031402 010237 030306  
 1039 031406 020203  
 1040 031410 101007  
 1041 031412 020237 003124  
 1042 031416 103007  
 1043 031420 005737 003134  
 1044 031424 001004  
 1045 031426 000424  
 1046 031430 162702 000020  
 1047 031434 000754  
 1048 031436  
 1049 031436 005737 003134  
 1050 031442 001420  
 1051 031444 005737 003132  
 1052 031450 001413  
 1053 031452 004737 017256  
 1054 031456 010500  
 1055 031460 010037 030302  
 1056 031464 010201  
 1057 031466 004737 017316  
 1058 031472 010037 030306  
 1059 031476 103403

```

:ROUTINE TO CONVERT A TEST PATTERN TO A VALID ADDRESS IN DIAGNOSTIC FREE SPACE
DIAGNOSTIC FREE SPACE IS BETWEEN THE END OF THE DIAGNOSTIC AND THE
BEGINNING OF THE SUPERVISOR. THIS IS ALWAYS BELOW 24K.
IF MEMORY ABOVE 28K SPECIFIED (VIA R1) THEN PAR 6 IS SET
TO THE RELOCATION BASE.

INPUTS:
      RO      HIGH ORDER ADDRESS BITS
      R1      LOW ORDER ADDRESS BITS

OUPUTS:
T12PAR6 = ADDRESS BIASED TO PAR6 IF >28K UNDER TEST
T12HIADD = HIGH ORDER ADDRESS IN NON PAR6 FORMAT
T12LOADD = LOW ORDER ADDRESS IN NON PAR6 FORMAT
C BIT = 1 IF GOOD ADDRESS RETURNED
C BIT = 0 IF TEST PATTERN DID NOT YIELD A VALID ADDRESS

T12CONVERT:
      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
      CLR         T12LOADD ;CLEAR LOW ADDRESS
      CLR         T12HIADD ;CLEAR HIGH ADDRESS
      CLR         T12PAR6  ;CLEAR PAR6 BIASED ADDRESS
      BIC         #'C<7777>,R1 ;FORCE TO LOWER 12 BITS OF ADDRESS
      MOV         R0,R5    ;SAVE HIGH ORDER ADDRESS BITS
      JSR         PC,KTOFF ;SHUTOFF MEMORY MANAGEMENT
      MOV         FREE,R2  ;GET FIRST FREE ADDRESS
      ADD         #16.,R2  ;IN CASE TEST PATTERN=0
      ADD         R1,R2    ;ADD IN TEST PATTERN
      BIC         #3,R2    ;MAKE IT MODULO-4
25$:  MOV         FREEHI,R3 ;GET LAST FREE ADDRESS
      SUB         #16.,R3  ;SAVE AT LEAST 8 WORDS (IN CASE MESSAGE BUFFER)
      MOV         R2,T12LOADD ;SAVE POSSIBLE LOW ADDRESS
      MOV         R2,T12PAR6 ;SAVE IT IN PAR6 BIASED TOO
      CMP         R2,R3    ;IS THIS ADDRESS ABOVE FREE SPACE?
      BHI         35$     ;BR IF YES
      CMP         R2,FREE  ;IS IT IN FREE SPACE?
      BHIS        50$     ;BR IF YES- ITS GOOD
      TST         KTENABLE ;TESTING ABOVE 28K?
      BNE         50$     ;BR IF YES
      BR          90$     ;BR IF NOT IN FREE SPACE
35$:  SUB         #16.,R2  ;FORCE FIT THE TEST PATTERN
      BR          25$     ;TRY THIS TEST PATTERN ADDRESS
50$:  TST         KTENABLE ;TESTING ABOVE 28K?
      BEQ         100$    ;BR IF NO
      TST         KTFLG   ;ANY MEMORY ABOVE 28K?
      BEQ         90$     ;BR IF NO
      JSR         PC,KTON  ;TURN ON MEMORY MANAGEMENT
      MOV         R5,R0    ;GET HIGH ORDER ADDRESS
      MOV         R0,T12HIADD ;SAVE POSSIBLE HIGH ADDRESS
      MOV         R2,R1    ;GET COMPUTED LOW ORDER ADDRESS
      JSR         PC,SETMAP ;RETURN PAR6 BIASED ADDRESS IN R0
      MOV         R0,T12PAR6 ;COPY PAR6 BIASED ADDRESS
      BCS         105$    ;BR IF VALID ADDRESS
  
```

```

1060 031500 0G0241          90$:   CLC           ;CLR C BIT FOR FAILURE
1061 031502 000401          BR           105$
1062 031504 000261          100$:  SEC           ;SET SUCCESS
1063 031506 000207          105$:  RTS            PC          ;RETURN
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092 031510
1093 031510
1094 031514 012701 002242
1095 031520 012702 000201
1096 031524 005003
1097 031526 004737 016336
1098 031532 112765 000000 000000
1099 031540 004737 016336
1100 031544 010265 000000
1101 031550 004737 016336
1102 031554 116511 000000
1103 031560 122124
1104 031562 001401
1105 031564 005203
1106 031566 005202
1107 031570 020227 000203
1108 031574 002761
1109 031576 005703
1110 031600 001402
1111 031602 000241
1112 031604 000401
1113 031606 000261
1114 031610 012737 000002 002302
1115 031616 000207
1116
    
```

```

: +
: ROUTINE TO READ THE FIRST 2 BYTES FROM RAM
: MEMORY AND COMPARE THIS DATA TO A COMMAND PACKET.
: INPUT:
: R4 ADDRESS OF THE COMMAND PACKET
: R5 FIRST DEVICE UNIBUS ADDRESS
: OUTPUT:
: CARRY SET - RAM MATCHES PACKET
: CLR - RAM DOES NOT MATCH PACKET
: IMPLICIT OUTPUT:
: THE TABLE RAMDATA IS FILLED WITH THE
: DATA HELD IN RAM.
: RAMSIZ SET TO 2 FOR PRAMPKT ROUTINE
: SIDE EFFECTS:
: THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
    
```

```

T12CKRAM::
SAVREG
MOV #RAMDATA,R1 ;SAVE THE GENERAL REGISTERS
MOV #RMPKTBEG,R2 ;ADDRESS TO SAVE THE RAM DATA
CLR R3 ;BYTE ADDRESS OF FIRST RAM DATA
JSR PC,CHKTSSR ;CLEAR THE ERROR FLAG
MOVB #0,TSDB(R5) ;WAIT FOR SSR
JSR PC,CHKTSSR ;SET MAINTENANCE MODE
MOV R2,TSDB(R5) ;WAIT FOR SSR TO SET
JSR PC,CHKTSSR ;SELECT NEXT RAM ADDRESS
MOVB TSBA(R5),(R1) ;WAIT FOR SSR TO SET
CMPB (R1)+,(R4)+ ;READ THE RAM DATA
BEQ 20$ ;COMPARE TO EXPECTED
INC R3 ;BRANCH IF OK
INC R2 ;SET ERROR FLAG
CMP R2,#RMPKTBEG+2 ;ADDRESS OF NEXT RAM LOCATION
BLT 10$ ;DONE 2 BYTES?
TST R3 ;BR IF NO
BEQ 30$ ;WAS AN ERROR FOUND ?
CLC ;BRANCH IF NOT
BR 50$ ;CLEAR CARRY TO SHOW ERROR
SEC ;AND EXIT
MOV #2,RAMSIZ ;SHOW GOOD COMPARE
RTS PC ;SETUP RAMSIZ
;RETURN
    
```

1117  
 1118  
 1119  
 1120  
 1121  
 1122 031620  
 1123 031620  
 1124 031624 012701 030240  
 1125 031630 012721 100004  
 1126 031634 012721 030250  
 1127 031640 005021  
 1128 031642 012721 000010  
 1129 031646 012721 030262  
 1130 031652 005021  
 1131 031654 012721 000016  
 1132 031660 005021  
 1133 031662 005011  
 1134 031664 000207  
 1135  
 1136  
 1137  
 1138  
 1139  
 1140  
 1141  
 1142  
 1143  
 1144  
 1145 031666  
 1146 031666  
 1147 031672 010401  
 1148 031674 005737 003134  
 1149 031700 001404  
 1150 031702 010300  
 1151 031704 004737 017316  
 1152 031710 010001  
 1153 031712 012700 000017  
 1154 031716 052700 100000  
 1155 031722 010021  
 1156 031724 005021  
 1157 031726 000207  
 1158  
 1159  
 1160  
 1161  
 1162  
 1163  
 1164  
 1165 031730  
 1166 031730  
 1167 031734 012700 030250  
 1168 031740 013701 030304  
 1169 031744 005737 003134  
 1170 031750 001402  
 1171 031752 013701 030306  
 1172 031756 012021  
 1173 031760 012021

```

: +
: ROUTINE TO SETUP PACKET TO WRITE CHARACTERISTICS
: -
T12SWRT:
    SAVREG                ;SAVE THE REGISTERS
    MOV #T12PACKET,R1    ;START OF THE PACKET
    MOV #100004,(R1)+    ;WRITE CHARACTERISTICS WITH ACK
    MOV #T12DATA,(R1)+  ;ADDRESS OF CHAR DATA BLOCK
    CLR (R1)+            ;EXTENDED ADDRESS
    MOV #8,(R1)+        ;SIZE OF DATA BLOCK IN BYTES
    MOV #T12BFR,(R1)+   ;ADDRESS OF MESSAGE BUFFER
    CLR (R1)+
    MOV #14,(R1)+       ;LENGTH OF MESSAGE BUFFER
    CLR (R1)+
    CLR (R1)
    RTS PC                ;RETURN

: +
: ROUTINE TO SETUP A GET STATUS COMMAND PACKET AT CURRENT PACKET ADDRESS
: -
R3    HIGH ORDER PACKET ADDRESS
R4    LOW ORDER PACKET ADDRESS
NOTE: R3 IS IGNORED IF KENABLE FLAG CLEAR

T12SETGET:
    SAVREG                ;SAVE THE REGISTERS
    MOV R4,R1             ;GET LOW ORDER ADDRESS
    TST KENABLE          ;TESTING ABOVE 28K?
    BEQ 10$              ;BR IF NO
    MOV R3,R0            ;GET HIGH ORDER ADDRESS
    JSR PC,SETMAP       ;RETURN ADDRESS BIASED TO PAR6 IN R0
    MOV R0,R1           ;GET ADDRESS
10$: MOV #P.GETSTATUS,R0 ;GET STATUS COMMAND CODE NO IE
    BIS #P.ACK,R0       ;SET ACK
    MOV R0,(R1)+        ;STORE GET STATUS IN PACKET
    CLR (R1)+           ;CLEAR UNUSED WORD
    RTS PC              ;RETURN

: +
: ROUTINE TO SETUP A CHARACTERISTIC DATA BLOCK AT A TEST ADDRESS
: -
T12CHAR:
    SAVREG                ;SAVE R1-R5 UNTIL NEXT RETURN
    MOV #T12DATA,R0      ;GET T12PACKET DATA POINTER
    MOV T12LOADD,R1     ;ASSUME NOT ABOVE 28K
    TST KENABLE          ;TESTING ABOVE 28K?
    BEQ 10$              ;BR IF NO
    MOV T12PAR6,R1      ;SET TEST ADDRESS ABOVE 28K
10$: MOV (R0)+,(R1)+    ;STORE DATA WORD 1
    MOV (R0)+,(R1)+    ;STORE DATA WORD 2
    
```

1174	031762	012021	MOV	(R0)+,(R1)+	:STORE DATA WORD 3
1175	031764	012021	MOV	(R0)+,(R1)+	:STORE DATA WORD 4
1176	031766	012021	MOV	(R0)+,(R1)+	:STORE DATA WORD 5
1177	031770	000207	RTS	PC	:RETURN
1178					
1179	031772		ENDTST		
	031772				
	031772	104401			

L10042: TRAP CSETST

1181



```

1183 .SBTTL TEST 4: RAM EXERCISER TEST
1184
1185
1186 :THIS TEST USES THE READ AND WRITE RAM (BOTH SINGLE AND 256
1187 :LOCATIONS) SELECT CODES OF THE WRITE SUBSYSTEM MEMORY COMMAND
1188 :TO EXERCISE THE CONTROLLER'S RAM MEMORY AND DMA LOGIC
1189
1190
1191 031774 BGNTST
1192 031774
1193
1198 031774 005737 002214 TST TSTCNT ;CHECK FOR RUN MODE
1199 032000 001402 BEQ 10$ ;BR, IF NOT ONLY PROGRAM RUN
1200 032002 005237 003400 INC SKIPT ;SET SKIP SW
1201 032006 012700 034433 10$: MOV #TST15ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
1202 032012 004737 016510 JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
1203 032016 012737 000005 002216 MOV #5,LOOPCNT ;PERFORM 5 ITERATIONS
1204 032024 T15LOOP:
1205
1206 :TEST 4, SUBTEST 1
1207
1208 THIS SUBTEST WRITES THE ADDRESS (8 BITS) INTO THE
1209 RAM MEMORY SINGLE WORD (8 BITS) MODE
1210
1211
1212
1213 032024 BGNSUB ;////////// BEGIN SUBTEST ////////////
1214 032024 104402 T4.1:
1215 032026 012700 000000 SETPRI #PRI00 ;LOWER PRIORITY TO ALLOW INTERRUPTS
1216 032032 104441 TRAP C$SUB
1217 032034 005737 003400 TST SKIPT ;SHOULD WE SKIP THIS SUBTEST
1218 032040 001402 BEQ 10$ ;BR, IF NOW SKIP REQUIRED
1219 032042 000137 032324 JMP 50$ ;SKIP SUBTEST
1220 032046 004737 034452 10$: JSR PC,T15REST ;SET COMMAND PACKET
1221 032052 004737 034524 JSR PC,T15RT2 ;SET UP OTHER COMMAND PACKET
1222 032056 004737 015774 JSR PC,SOFINIT ;DO INITIALIZE ON CONTROLLER
1223 032062 103405 BCS 20$ ;BR IF INIT WAS OK
1224 032064 010001 MOV R0,R1 ;CONTENTS OF TSSR REGISTER
1225 032066 104455 ERRDF ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
1226 032070 000621 TRAP C$ERDF
1227 032072 003652 .WORD 401
1228 032074 012034 .WORD SFIERR
1229 032076 012704 033350 20$: MOV #T15PACKET,R4 ;SUBROUTINE NEEDS PACKET ADDRESS
1230 032102 004737 010662 JSR PC,WRTCHR ;ISSUE WRITE CHARACTERISTICS
1231 032106 103405 BCS 23$ ;BR, IF COMMAND ISSUED OK
1232 032110 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
1233 032112 104456 ERRHRD ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTIC FAILED
1234 032114 000622 TRAP C$ERHRD
1235 032116 005056 .WORD 402
1236 032120 012034 .WORD WRTMSG
1236 032122 012703 000400 23$: MOV #256.,R3 ;STARTING ADDRESS FOR RAM WRITE
    
```

```

1237 032126 112737 000001 034061      MOVB  #1,T15BS1      :SIZE OF TRANSFER
1238 032134 112737 000002 034060      MOVB  #2,T15BS0      :WRITE RAM 'COMMAND'
1239 032142                                25$:
1240 032142 010337 034062      MOV   R3,T15S2        :ADDRESS FOR RAM
1241 032146 012704 034050      MOV   #T15PK2,R4      :WRITE SUBSYS MEM PACKET
1242 032152 110337 034064      MOVB  R3,T15S3        :DATA FOR WRITE (ADDRESS)
1243 032156 010465 000000      MOV   R4,TSDB(R5)     :ISSUE COMMAND
1244 032162 004737 016336      JSR   PC,CHKTSSR      :WAIT FOR SSR
1245 032166 103407      BCS   30$             :BR, IF NO ERROR
1246 032170 010001      MOV   R0,R1           :ERROR, SAVE TSSR
1250 032172      ERRHRD ERRNO,T15SSR,PKTSSR :TSSR NOT CORRECT AFTER WRITE SUB MEM
                                TRAP  C$ERHRD
                                .WORD 403
                                .WORD T15SSR
                                .WORD PKTSSR
1251 032202      ESCAPE SUB           :DON'T CONTINUE IF ERROR ON WRITE
                                TRAP  C$ESCAPE
                                .WORD L10050-.
1252 032206 104406      30$: CKLOOP          :SCOPE LOOP
                                TRAP  C$CLP1
1253
1254
1255 032210 005203      INC   R3              :NEXT ADDRESS
1256 032212 020327 010000      CMP   R3,#10000       :END OF RAM MEMORY CHECK
1257 032216 001351      BNE   25$             :LOOP TILL ALL RAM WRITTEN
1258 032220 005002      CLR   R2              :CLEAR OUT R2 HIGH BITS
1259 032222 005303      DEC   R3              :SET BACK TO 7777
1260 032224 110337 034064      MOVB  R3,T15S3        :SET DATA PATTERN BACK IN SHAPE
1261 032230 010337 034062      MOV   R3,T15S2        :ADDRESS FOR RAM READ
1262 032234 112737 000001 034060      MOVB  #1,T15BS0       :READ RAM COMMAND
1263 032242 010465 000000      MOV   R4,TSDB(R5)     :SEND OUT PACKET ADDRESS TO CONTR.
1264 032246 004737 016336      JSR   PC,CHKTSSR      :WAIT FOR READY, NON-AMBIGUOUS
1265 032252 103405      BCS   43$             :BR, IF NO PROBLEM
1266 032254 010001      MOV   R0,R1           :SAVE TSSR
1270 032256      ERRDF  ERRNO,T15SSR,PKTSSR :TSSR NOT CORRECT
                                TRAP  C$ERDF
                                .WORD 404
                                .WORD T15SSR
                                .WORD PKTSSR
1271 032266 104406      43$: CKLOOP          :SCOPE LOOP
                                TRAP  C$CLP1
1272 032270 013701 033412      MOV   T15BFR+20,R1    :GET RAM READ DATA
1273 032274 010302      MOV   R3,R2           :SET UP FOR COMPARE
1274 032276 120102      CMPB  R1,R2           :CHECK WITH DATA WRITTEN
1275 032300 001404      BEQ   45$             :BR IF OK, DATA IN = DATA OUT
1279 032302      ERRHRD ERRNO,T15AM4,EXPBREC :WRITTEN DATA NOT = TO READ
                                TRAP  C$ERHRD
                                .WORD 405
                                .WORD T15AM4
                                .WORD EXPBREC
1280 032312 104406      45$: CKLOOP          :SCOPE LOOP
                                TRAP  C$CLP1
1281 032314 005303      DEC   R3              :DROP DATA COUNTER (PATTERN)
1282 032316 020327 000377      CMP   R3,#255.        :AT BOTTOM YET
1283 032322 001340      BNE   40$             :BR, IF MORE TO CHECK
1284 032324 104406      50$: CKLOOP          :SCOPE LOOP
                                TRAP  C$CLP1
    
```

1285 032326  
032326  
032326 104403  
1286

ENDSUB

;////////// END SUBTEST //////////  
L10050:  
TRAP CSFSUB

```

1288
1289 032330          BGNSUB          ://////////////// BEGIN SUBTEST //////////////////
      032330          T4.2:
      032330 104402          TRAP      CSBSUB
1290
1291          :+
1292          :
1293          :TEST 4, SUBTEST 2
1294          :
1295          :
1296          :
1297          :
1298          :
1299 032332 004737 034452          JSR      PC,T15REST          :RESTORE PACKET FOR WRITE CHARA
1300 032336 004737 034524          JSR      PC,T15RT2          :RESTORE PACKET FOR WRT SUB SYS MEM
1301 032342 004737 015774          JSR      PC,SOFINIT        :DO INITIALIZE ON CONTROLLER
1302 032346 103405          BCS      20$              :BR IF INIT WAS OK
1306 032350 010001          MOV      R0,R1            :CONTENTS OF TSSR REGISTER
1307 032352          ERRDF  ERRNO,SFIERR,SFIMSG :FATAL ERROR TSSR WAS NOT OK
      032352 104455          TRAP      CSERDF
      032354 000626          .WORD   406
      032356 003652          .WORD   SFIERR
      032360 012034          .WORD   SFIMSG
1308 032362          20$:
1309 032362 012704 033350          MOV      #T15PACKET,R4    :SUBROUTINE NEEDS PACKET ADDRESS
1310 032366 004737 010662          JSR      PC,WRTCHR        :ISSUE WRITE CHARACTERISTICS
1311 032372 103405          BCS      25$              :BR, IF COMMAND ISSUED OK
1315 032374 010001          MOV      R0,R1            :SAVE CONTENTS OF TSSR
1316 032376          ERRHRD ERRNO,WRTMSG,SFIMSG :WRITE CHARACTERISTIC FAILED
      032376 104456          TRAP      CSERHRD
      032400 000627          .WORD   407
      032402 005056          .WORD   WRTMSG
      032404 012034          .WORD   SFIMSG
1317 032406          25$:
1318 032406 112737 000001 034061          MOV      #1,T15BS1        :SET SIZE OF TRANSFER 1 BYTE
1319 032414 012704 034050          MOV      #T15PK2,R4      :SET NEW PACKET ADDRESS
1320 032420 012703 000400          MOV      #256,R3         :STARTING ADDRESS IN RAM
1321 032424 112737 000002 034060          MOV      #2,T15BS0        :WRITE RAM COMMAND
1322 032432 105037 034064          CLRB    T15S3           :SET DATA TO 000
1323 032436 010337 034062          30$: MOV      R3,T15S2        :ADDRESS TO PACKET DATA AREA
1324 032442 010465 000000          MOV      R4,TSDB(R5)     :SEND OUT PACKET ADDRESS
1325 032446 004737 016336          JSR      PC,CHKTSSR      :WAIT FOR SSR
1326 032452 103405          BCS      33$              :BR, IF NO PROBLEM
1327 032454 010001          MOV      R0,R1            :SAVE TSSR
1331 032456          ERRHRD ERRNO,T15SSR,PKTSSR :TSSR NOT CORRECT
      032456 104456          TRAP      CSERHRD
      032460 000630          .WORD   408
      032462 034066          .WORD   T15SSR
      032464 012046          .WORD   PKTSSR
1332 032466          33$: CKLOOP          :SCOPE LOOP
      032466 104406          TRAP      CSCLP1
1333
1334
1335 032470 005203          INC      R3              :NEXT ADDRESS
1336 032472 020327 010000          CMP     R3,#10000        :END OF RAM MEMORY CHECK
1337 032476 001357          BNE     30$              :BR, MORE RAM TO GO
1338 032500 005303          35$: DEC      R3              :SET BACK TO 7777
    
```

1339	032502	005002			40\$:	CLR	R2	:	SET TO ALL ZEROS		
1340	032504	112737	000001	034060		MOVB	#1,T15BS0	:	READ RAM COMMAND		
1341	032512	010337	034062			MOV	R3,T15S2	:	ADDRESS TO BE READ TO PACKET DATA		
1342	032516	010465	000000			MOV	R4,TSDB(R5)	:	SEND OUT PACKET ADDRESS		
1343	032522	004737	016336			JSR	PC,CHKTSSR	:	WAIT FOR SSR TO SET		
1344	032526	103405				BCS	41\$	:	BR, IF ALL IS WELL		
1345	032530	010001				MOV	R0,R1	:	SAVE TSSR		
1349	032532					ERRHRD	ERRNO,T15SSR,PKTSSR	:	TSSR NOT CORRECT		
	032532	104456								TRAP	C\$ERHRD
	032534	000631								.WORD	409
	032536	034066								.WORD	T15SSR
	032540	012046								.WORD	PKTSSR
1350	032542				41\$:	CKLOOP		:	SCOPE LOOP		
	032542	104406								TRAP	C\$CLP1
1351	032544	013701	033412			MOV	T15BFR+20,R1	:	PICK UP READ DATA		
1352	032550	120102				CMPB	R1,R2	:	BOTH SHOULD BE 00000000 BINARY		
1353	032552	001404				BEQ	42\$	:	BR, IF DATA IS GOOD		
1357	032554					ERRHRD	ERRNO,T15AM3,EXPBREC	:	CHARACTERISTICS DATA NOT CORRECT		
	032554	104456								TRAP	C\$ERHRD
	032556	000632								.WORD	410
	032560	034243								.WORD	T15AM3
	032562	015502								.WORD	EXPBREC
1358	032564				42\$:	CKLOOP		:	SCOPE LOOPER		
	032564	104406								TRAP	C\$CLP1
1359	032566	012702	000377			MOV	#000377,R2	:	SET ALL ONES WORD		
1360	032572	112737	000002	034060		MOVB	#2,T15BS0	:	WRITE RAM COMMAND		
1361	032630	112737	000377	034064		MOVB	#000377,T15S3	:	ALL ONES PATTERN		
1362	032606	010465	000000			MOV	R4,TSDB(R5)	:	PASS PACKET ADDRESS TO CONTR.		
1363	032612	004737	016336			JSR	PC,CHKTSSR	:	WAIT FOR SSR		
1364	032616	103405				BCS	43\$	:	BR, IF OK (NO ERROR)		
1365	032620	010001				MOV	R0,R1	:	SAVE TSSR		
1369	032622					ERRHRD	ERRNO,T15SSR,PKTSSR	:	TSSR NOT CORRECT		
	032622	104456								TRAP	C\$ERHRD
	032624	000633								.WORD	411
	032626	034066								.WORD	T15SSR
	032630	012046								.WORD	PKTSSR
1370	032632				43\$:	CKLOOP		:	SCOPE LOOP		
	032632	104406								TRAP	C\$CLP1
1371	032634	112737	000001	034060		MOVB	#1,T15BS0	:	SET UP FOR RAM READ		
1372	032642	010465	000000			MOV	R4,TSDB(R5)	:	ISSUE RAM READ		
1373	032646	004737	016336			JSR	PC,CHKTSSR	:	WAIT FOR SSR TO SET		
1374	032652	103405				BCS	44\$	:	BR, IF OK (NO ERROR)		
1375	032654	010001				MOV	R0,R1	:	SAVE TSSR		
1379	032656					ERRDF	ERRNO,T15SSR,PKTSSR	:	TSSR NOT CORRECT		
	032656	104455								TRAP	C\$ERDF
	032660	000634								.WORD	412
	032662	034066								.WORD	T15SSR
	032664	012046								.WORD	PKTSSR
1380	032666	013701	033412		44\$:	MOV	T15BFR+20,R1	:	PICK UP REC'D DATA		
1381	032672	120102				CMPB	R1,R2	:	CHECK WITH DATA WRITTEN		
1382	032674	001404				BEQ	45\$	:	BR IF OK, DATA IN = DATA OUT		
1386	032676					ERRHRD	ERRNO,T15AM2,EXPBREC	:	WRITTEN DATA NOT = TO READ		
	032676	104456								TRAP	C\$ERHRD
	032700	000635								.WORD	413
	032702	034142								.WORD	T15AM2
	032704	015502								.WO )	EXPBREC
1387	032706				45\$:	CKLOOP		:	SCOPE LOOP		

1388 032706 104406  
1389 032710 005303  
1390 032712 020327 000377  
1391 032716 001271  
1392 032720  
032720  
1393 032720 104403

DEC R3  
CMP R3,#255.  
BNE 40\$  
  
ENDSUB

TRAP C\$CLP1  
:DROP RAM ADDRESS POINTER  
:AT START YET  
:BR, IF MORE RAM TO CHECK  
  
:////////////////// END SUBTEST ///////////////////  
L10051:  
TRAP C\$ESUB

```

1395 032722          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      032722          ;T4.3:          TRAP      CSBSUB
      032722 104402
1396
1397
1398
1399
1400
1401
1402
1403
1404 032724 005737 003400          TST      SKIPT          ;CHECK RUN MODE
1405 032730 001402          BEQ      10$          ;BR, IF NO SKIP
1406 032732 000137 033326          JMP      50$          ;SKIP SUBTEST
1407 032736 004737 034452 10$: JSR      PC,T15REST    ;RESTORE PACKET FOR WRITE CHARA
1408 032742 004737 034524          JSR      PC,T15RT2    ;RESTORE PACKET FOR WRT SUB SYS MEM
1409 032746 004737 015774          JSR      PC,SOFINIT   ;DO INITIALIZE ON CONTROLLER
1410 032752 103405          BCS      20$          ;BR IF INIT WAS OK
1414 032754 010001          MOV      R0,R1        ;CONTENTS OF TSSR REGISTER
1415 032756          ERRDF   ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
      032756 104455          ;TRAP      CSERDF
      032760 000636          ;.WORD    414
      032762 003652          ;.WORD    SFIERR
      032764 012034          ;.WORD    SFIMSG
1416 032766          20$:
1417 032766 012704 033350          MOV      #T15PACKET,R4 ;SUBROUTINE NEEDS PACKET ADDRESS
1418 032772 004737 010662          JSR      PC,WRTCHR     ;ISSUE WRITE CHARACTERISTICS
1419 032776 103405          BCS      25$          ;BR, IF COMMAND ISSUED OK
1423 033000 010001          MOV      R0,R1        ;SAVE CONTENTS OF TSSR
1424 033002          ERRHRD  ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTIC FAILED
      033002 104456          ;TRAP      CSERHRD
      033004 000637          ;.WORD    415
      033006 005056          ;.WORD    WRTMSG
      033010 012034          ;.WORD    SFIMSG
1425 033012          25$:
1426 033012 112737 000001 034061          MOVB     #1,T15BS1     ;SET SIZE TO 1 BYTE
1427 033020 012704 034050          MOV      #T15PK2,R4   ;SET NEW PACKET ADDRESS
1428 033024 012703 000400          MOV      #256,R3      ;STARTING ADDRESS IN RAM
1429 033030 112737 000002 034060          MOVB     #2,T15BS0     ;WRITE RAM COMMAND
1430 033036 112737 000377 034064          MOVB     #377,T15S3    ;SET DATA TO 377
1431 033044 010337 034062          30$: MOV      R3,T15S2     ;ADDRESS TO PACKET DATA AREA
1432 033050 010465 000000          MOV      R4,TSD8(R5)  ;SEND OUT PACKET ADDRESS
1433 033054 004737 016336          JSR      PC,CHKTSSR   ;WAIT FOR SSR
1434 033060 103405          BCS      33$          ;BR, IF NO PROBLEM
1435 033062 010001          MOV      R0,R1        ;SAVE TSSR
1439 033064          ERRHRD  ERRNO,T15SSR,PKTSSR ;TSSR NOT CORRECT
      033064 104456          ;TRAP      CSERHRD
      033066 000640          ;.WORD    416
      033070 034066          ;.WORD    T15SSR
      033072 012046          ;.WORD    PKTSSR
1440 033074          33$: CKLOOP          ;SCOPE LOOP
      033074 104406          ;TRAP      CSCLP1
1441
1442
1443 033076 005203          INC      R3            ;NEXT ADDRESS
1444 033100 020327 010000          CMP      R3,#10000    ;END OF RAM MEMORY CHECK
1445 033104 001357          BNE     30$          ;BR, MORE RAM TO GO
    
```

1446	033106	005303		35\$:	DEC	R3		:SET BACK TO 7777		
1447	033110	112702	000377	40\$:	MOVB	#377,R2		:SET TO ALL ONES		
1448	033114	112737	000001	034060	MOVB	#1,T15BS0		:READ RAM COMMAND		
1449	033122	010337	034062		MOV	R3,T15S2		:ADDRESS TO BE READ TO PACKET DATA		
1450	033126	010465	000000		MOV	R4,TSDB(R5)		:SEND OUT PACKET ADDRESS		
1451	033132	004737	016336		JSR	PC,CHKTSSR		:WAIT FOR SSR TO SET		
1452	033136	103405			BCS	41\$		:BR, IF ALL IS WELL		
1453	033140	010001			MOV	R0,R1		:SAVE TSSR		
1457	033142				ERRHRD	ERRNO,T15SSR,PKTSSR		:TSSR NOT CORRECT		
	033142	104456							TRAP	C\$ERHRD
	033144	000641							.WORD	417
	033146	034066							.WORD	T15SSR
	033150	012046							.WORD	PKTSSR
1458	033152			41\$:	CKLOOP			:SCOPE LOOP		
	033152	104406							TRAP	C\$CLP1
1459	033154	013701	033412		MOV	T15BFR+20,R1		:PICK UP READ DATA		
1460	033160	120102			CMPB	R1,R2		:BOTH SHOULD BE 11111111 BINARY		
1461	033162	001404			BEQ	42\$		:BR, IF DATA IS GOOD		
1465	033164				ERRHRD	ERRNO,T15AM3,EXPBREC		:CHARACTERISTICS DATA NOT CORRECT		
	033164	104456							TRAP	C\$ERHRD
	033166	000642							.WORD	418
	033170	034243							.WORD	T15AM3
	033172	015502							.WORD	EXPBREC
1466	033174	012702	000377	42\$:	MOV	#000377,R2		:SET ALL ONES WORD		
1467	033200	012737	000002	034060	MOV	#2,T15BS0		:WRITE RAM COMMAND		
1468	033206	112737	000377	034064	MOVB	#000377,T15S3		:ALL ONES PATTERN		
1469	033214	010465	000000		MOV	R4,TSDB(R5)		:PASS PACKET ADDRESS TO CONTR.		
1470	033220	004737	016336		JSR	PC,CHKTSSR		:WAIT FOR SSR		
1471	033224	103405			BCS	43\$		:BR, IF OK (NO ERROR)		
1472	033226	010001			MOV	R0,R1		:SAVE TSSR		
1476	033230				ERRHRD	ERRNO,T15SSR,PKTSSR		:TSSR NOT CORRECT		
	033230	104456							TRAP	C\$ERHRD
	033232	000643							.WORD	419
	033234	034066							.WORD	T15SSR
	033236	012046							.WORD	PKTSSR
1477	033240			43\$:	CKLOOP			:SCOPE LOOP		
	033240	104406							TRAP	C\$CLP1
1478	033242	112737	000001	034060	MOVB	#1,T15BS0		:SET UP FOR RAM READ		
1479	033250	010465	000000		MOV	R4,TSDB(R5)		:ISSUE RAM READ		
1480	033254	004737	016336		JSR	PC,CHKTSSR		:WAIT FOR SSR TO SET		
1481	033260	103405			BCS	44\$		:BR, IF OK (NO ERROR)		
1482	033262	010001			MOV	R0,R1		:SAVE TSSR		
1486	033264				ERRHRD	ERRNO,T15SSR,PKTSSR		:TSSR NOT CORRECT		
	033264	104456							TRAP	C\$ERHRD
	033266	000644							.WORD	420
	033270	034066							.WORD	T15SSR
	033272	012046							.WORD	PKTSSR
1487	033274	013701	033412	44\$:	MOV	T15BFR+20,R1		:PICK UP REC'D DATA		
1488	033300	120102			CMPB	R1,R2		:CHECK WITH DATA WRITTEN		
1489	033302	001404			BEQ	45\$		:BR IF OK, DATA IN = DATA OUT		
1493	033304				ERRHRD	ERRNO,T15AM2,EXPBREC		:WRITTEN DATA NOT = TO READ		
	033304	104456							TRAP	C\$ERHRD
	033306	000645							.WORD	421
	033310	034142							.WORD	T15AM2
	033312	015502							.WORD	EXPBREC
1494	033314			45\$:	CKLOOP			:SCOPE LOOP		
	033314	104406							TRAP	C\$CLP1



1495 033316 0G5303  
1496 033320 020327 000377  
1497 033324 001271  
1498  
1499 033326  
1500 033326  
033326  
033326 104403

50\$:

DEC R3  
CMP R3,#255.  
BNE 40\$  
  
ENDSUB

:DROP RAM ADDRESS POINTER  
:AT START YET  
:BR, IF MORE RAM TO CHECK

:////////// END SUBTEST //////////  
L10052:  
TRAP C\$ESUB

```

1502
1503 033330 004737 016456          JSR    PC,TSTLOOP          ;DO WE NEED TO ITERATE TEST ?
1504 033334 103002                   BCC    63$                 ;BRANCH IF NOT
1505 033336 000137 032024          JMP    T15LOOP            ;EXECUTE AGAIN
1506 033342 104432                   EXIT   TST                 ;ALL DONE THIS TEST
      033344 001216
                                     TRAP  C$EXIT
                                     .WORD L10047-.

1507
1508
1509
1510
1511
1512          033350
      T15PACKET:  .=<.+10>&177770
      .WORD      100204
      .WORD      T15DATA
      .WORD      0
      .WORD      8.
      T15DATA:   .WORD      T15BFR
      .WORD      0
      .WORD      256.
      .WORD      0,0
      T15BFR:    .BLKW     150.
      ;WRITE SUBSYSTEM MEMORY COMMAND PACKET
      .=<.+10>&177770
      T15PK2:    .WORD      100206
      .WORD      T15BF2
      .WORD      0
      .WORD      6.
      .EVEN
      T15BF2:    .BYTE     0
      T15BS0:    .BYTE     0
      T15BS1:    .BYTE     0
      T15S2:    .WORD     0
      T15S3:    .WORD     0
      ;COMMAND PACKET FOR TEST
      ;WRITE CHARACTERISTICS COMMAND, WITH IE, ACK
      ;ADDRESS OF CHARACTERISTICS BLOCK
      ;STARTING VALUE OF BLOCK SIZE
      ;CHARACTERISTICS DATA BLOCK
      ;ADDRESS OF MESSAGE BUFFER
      ;LENGTH OF MESSAGE BUFFER
      ;MESSAGE BUFFER
      ;WRITE SUB SYS MEM COMMAND, IE AND ACK
      ;ADDRESS OF SELECT BLOCK DATA
      ;SIZE OF DATA PACKET
      ;BSEL0 AREA
      ;BSEL1 AREA
      ;SEL 2 AREA
      ;DATA AREA

1513
1514 033350 100204
1515 033352 033360
1516 033354 000000
1517 033356 000010
1518 033360 033372
1519 033362 000000
1520 033364 000400
1521 033366 000000 000000
1522 033372
1523
1524
1525
1526
1527
1528
1529          034050
1530
1531 034050 100206
1532 034052 034060
1533 034054 000000
1534 034056 000006
1535
1536
1537
1538 034060
1539 034062 000
1540 034064 000
1541 034066 000000
1542 034068 000000
1543
1544
1545
    
```

```

1547
1548
1549      ;+
1550      ;LOCAL TEXT MESSAGES FOR TEST
1551      ;-
1552 034066 127 122 111 T15SSR: .ASCIZ 'WRITE SUBSYSTEM MEMORY Command Not Accepted'
1553 034142 127 122 111 T15AM2: .ASCIZ 'WRITE SUBSYSTEM MEMORY COMMAND Failed On All Ones Word Read Back'
1554 034243 127 122 111 T15AM3: .ASCIZ 'WRITE SUBSYSTEM MEMORY COMMAND Failed On All Zeros Word Read Back'
1555 034345 127 122 111 T15AM4: .ASCIZ 'WRITE SUBSYSTEM MEMORY COMMAND Failed On Address Test'
1556 034433 122 101 115 TST15ID: .ASCIZ 'RAM Exerciser'
1557      .EVEN
1558
1559      ;+
1560      ;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
1561      ;WRITE SUBSYSTEM MEMORY COMMAND
1562      ;-
1563
1564
1565 034452      T15REST:
1566 034452      SAVREG      ;SAVE THE REGISTERS
1567 034456 012701 033350  MOV      #T15PACKET,R1      ;START OF THE PACKET
1568 034462 012721 100204  MOV      #100204,(R1)+      ;WRITE SUBSYSTEM MEM. WITH ACK, IE
1569 034466 012721 033360  MOV      #T15DATA,(R1)+      ;ADDRESS OF CHARAISTICS DATA BLOCK
1570 034472 005021      CLR      (R1)+      ;EXTENDED ADDRESS
1571 034474 012721 000010  MOV      #8,(R1)+      ;SIZE OF DATA BLOCK IN BYTES
1572 034500 012721 033372  MOV      #T15BFR,(R1)+      ;ADDRESS OF MESSAGE BUFFER
1573 034504 005021      CLR      (R1)+
1574 034506 012721 000400  MOV      #256,(R1)+      ;LENGTH OF MESSAGE BUFFER
1575 034512 005021      CLR      (R1)+
1576 034514 005011      CLR      (R1)
1577 034516 005037 033372  CLR      T15BFR      ;CLEAR 1ST LOC IN MESSAGE BUFFER
1578 034522 000207      RTS      PC      ;RETURN
1579
1580
1581 034524      T15RT2:
1582 034524      SAVREG      ;SAVE THE REGISTERS
1583 034530 012701 034050  MOV      #T15PK2,R1      ;START OF THE PACKET
1584 034534 012721 100206  MOV      #100206,(R1)+      ;WRITE SUBSYSTEM MEM. WITH ACK, IE
1585 034540 012721 034060  MOV      #T15BF2,(R1)+      ;ADDRESS OF DATA BLOCK
1586 034544 005021      CLR      (R1)+      ;EXTENDED ADDRESS
1587 034546 012721 000006  MOV      #6,(R1)+      ;SIZE OF DATA BLOCK IN BYTES
1588 034552 005021      CLR      (R1)+
1589 034554 005021      CLR      (R1)+
1590 034556 005011      CLR      (R1)
1591 034560 000207      RTS      PC      ;RETURN
1592 034562      ENDTST
      034562
      034562 104401
    
```

L10047: TRAP CSETST

1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1618  
1619  
1620  
1621  
1622

.SBTTL TEST 5: EXTENDED FEATURES SWITCH AND TIMERS A,B

++  
: TEST DESCRIPTION:

: This test verifies the Invert Extended Features function  
: can logically invert the Extended features switch and  
: that the internal timers A and B operate correctly.

: TEST STEPS:

: REPEAT FOR LOOPCNT

: BEGIN

: Do Subtest 1 - Verify Extended Features Switch

: Do Subtest 2 - Verify Timers A,B

: END

:--

BGNTST

MOV #TST16ID,RO  
JSR PC,TSTSETUP  
MOV #10.,LOOPCNT

T5::  
:ASCII MESSAGE TO IDENTIFY TEST  
:DO INITIAL TEST SETUP  
:PERFORM 10 ITERATIONS

T16LOOP:

012700 036642  
004737 016510  
012737 000012 002216

034564  
034564  
034564  
034570  
034574  
034602

```
1624 .SBTTL TEST 5: SUBTEST 1: VERIFY EXTENDED FEATURES TEST
1625
1626
1627 :++
1628 : TEST 5: SUBTEST 1:
1629 :
1630 : SUBTEST DESCRIPTION:
1631 :
1632 : This subtest verifies that the Invert Sense of Extended features
1633 : Switch function (Write Subsystem Memory,Write Misc command)
1634 : operates properly.
1635 : First the state of the Extended Features switch is read in the
1636 : message packet supplied by the write characteristics command.
1637 : Then, the sense of the switch is logically inverted.
1638 : A Write characteristics command is executed and it is verified
1639 : that the Extended status register (XST4) is returned when
1640 : in Extended mode, and not returned if not in extended mode.
1641 : The subtest also verifies that specifying a Message Buffer
1642 : address with any of bits 21-19 .set will cause the command to
1643 : be rejected.
1644 :
1645 : TEST STEPS:
1646 :
1647 : BEGIN
1648 : Write to TSSR register to soft initialize the controller
1649 : Do WRITE CHARACTERISTICS to check for Extended Features Switch
1650 : IF Extended Features Hardware Switch CLEAR
1651 : THEN
1652 : (* Verify Extended Features switch can be Inverted to SET *)
1653 : Do Write Subsystem Write Miscellaneous to SET Extended Features.
1654 : DO a WRITE CHARACTERISTICS with an extended characteristic word
1655 : Compare the controller ram to the extended characteristic word
1656 : If Data word in controller ram NOT= to word sent Then Print Error
1657 : If Message Buffer Data Length NOT= 12. Then Print Error
1658 : ELSE
1659 : (* Verify Extended Features switch can be Inverted to CLEAR *)
1660 : Do Write Subsystem Write Miscellaneous to CLEAR Extended Features.
1661 : Do a WRITE CHARACTERISTICS without an extended characteristic word
1662 : If Message Buffer Data Length NOT= 10. Then Print Error
1663 : END-IF
1664 : (* Verify Function Reject when Message Buffer 21-19 are non-zero *)
1665 : Write to TSSR register to soft initialize the controller
1666 : REPEAT FOR MESSAGE BUFFER ADDRESS bits <21:19> FROM 0 TO 7
1667 : DO a WRITE CHARACTERISTICS with a message address bit<21:19> non-zero
1668 : If TSSR termination code NOT= Function Reject Then Print Error
1669 : END-REPEAT
1670 :
1671 : END
1672 :--
1672 034602 BGNSUB :////////// BEGIN SUBTEST //////////
1673 034602 104402 T5.1: TRAP C$SUB
1674
1675 034604 5$:
1676 :
1677 034604 004737 015774 : Write to TSSR register to soft initialize the controller
1678 034610 103405 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
BCS 10$ ;BR IF SOFT INIT OKAY
```

```

1679 034612 010001          MOV    R0,R1          ;SAVE CONTENTS OF TSSR
1680 034614          ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
      034614 104455          TRAP  CSERDF
      034616 000764          .WORD 500
      034620 003652          .WORD SFIERR
      034622 012034          .WORD SFIMSG
1681          ; Do WRITE CHARACTERISTICS to check for Extended Features Switch
1682 034624 004737 040010 10$: JSR    PC,T16REST      ;RESTORE PACKET DEFAULTS
1683 034630 005037 002222  CLR    FATFLG         ;CLEAR FATAL ERROR FLAG
1684 034634 012704 040170  MOV    #T16PACKET,R4  ;GET THE ADDRESS OF COMMAND PACKET
1685 034640 004737 010662  JSR    PC,WRTCHR       ;DO WRITE CHARACTERISTICS COMMAND
1686 034644          FORCERROR 12$         ;@@DFORCE ERROR IF FORCER=1
1687 034660 103407          BCS    15$            ;BR IF CARRY SET (GOOD RETURN)
1688 034662 010001          MOV    R0,R1          ;SAVE CONTENTS OF TSSR
1689 034664          NEXT.ERRNO
1690 034664 12$: ERRDF  ERRNO,T16SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      034664 104455          TRAP  CSERDF
      034666 000765          .WORD 501
      034670 036712          .WORD T16SSR
      034672 012046          .WORD PKTSSR
1691 034674 005237 002222 15$: INC    FATFLG         ;SET FATAL ERROR FLAG
1692 034700          CKLOOP              ;LOOP ON ERROR, IF FLAG SET
      034700 104406          TRAP  CSCLP1
1693          ;
1694          ; If Extended Features Hardware Switch Clear then:
1695          ; (* Verify Extended Features switch can be Inverted to SET *)
1696          ; REPEAT FOR TEST PATTERNS IN TSTBLK TABLE
1697 034702 012701 040212  MOV    #T16BFR,R1      ;MESSAGE BUFFER ADDRESS
1698 034706 032761 000200 000012 BIT    #X2.EXTF,XST2(R1) ;EXTENDED FEATURES SWITCH CLEAR?
1699 034714 001402          BEQ    20$            ;BR IF YES
1700 034716 000137 035266  JMP    200$           ;
1701 034722 012703 002764 20$: MOV    #TSTBLK+10.,R3 ;START OF TEST DATA
1702          ; Do Write Subsystem Write Miscellaneous to SET Extended Features.
1703          ;
1704 034726 004737 040150  JSR    PC,T16SEXT     ;SETUP PACKET FOR WRITE MISC INVERT
1705 034732 012704 040240  MOV    #T16PK2,R4     ;GET WRITE SUBSYSTEM COMMAND PACKET
1706 034736 010465 000000  MOV    R4,TSDB(R5)    ;SET THE PACKET ADDRESS TO EXECUTE
1707 034742 004737 016336  JSR    PC,CHKTSSR     ;WAIT FOR SSR TO SET
1708 034746          FORCERROR 32$         ;@@DFORCE ERROR IF FORCER=1
1709 034762 103407          BCS    40$            ;BR IF CARRY SET (GOOD RETURN)
1710 034764 010001          MOV    R0,R1          ;SAVE CONTENTS OF TSSR
1711 034766          NEXT.ERRNO
1712 034766 32$: ERRDF  ERRNO,T162SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      034766 104455          TRAP  CSERDF
      034770 000766          .WORD 502
      034772 036747          .WORD T162SSR
      034774 012046          .WORD PKTSSR
1713 034776 005237 002222 40$: INC    FATFLG         ;SET FATAL ERROR FLAG
1714 035002          CKLOOP              ;LOOP ON ERROR, IF FLAG SET
      035002 104406          TRAP  CSCLP1
1715          ;
1716          ; DO a WRITE CHARACTERISTICS with an extended characteristic word
1717 035004 012737 125252 002312 MOV    #125252,DATA    ;SETUP TEST DATA FOR EXTENDED WORD
1718 035012 012704 040170  MOV    #T16PACKET,R4  ;GET THE ADDRESS OF COMMAND PACKET
1719 035016 012764 000020 000006 MOV    #16.,PKBCNT(R4) ;STORE MESSAGE PACKET SIZE
1720 035024 013737 002312 040210 MOV    DATA,T16DATA+10 ;STORE TEST DATA IN EXTENDED WORD
1721 035032 004737 010662  JSR    PC,WRTCHR       ;DO WRITE CHARACTERISTICS COMMAND
    
```

```

1722 035036          FORCERROR          42$          :@@DFORCE ERROR IF FORCER=1
1723 035052 103407  BCS          50$          :BR IF CARRY SET (GOOD RETURN)
1724 035054 010001  MOV          R0,R1          :SAVE CONTENTS OF TSSR
1725 035056          NEXT.ERRNO
1726 035056          ERRDF  ERRNO,T16SSR,PKTSSR  :DEVICE FATAL SSR FAILED TO SET
      035056 104455          TRAP  C$ERDF
      035060 000767          .WORD  503
      035062 036712          .WORD  T16SSR
      035064 012046          .WORD  PKTSSR
1727 035066 005237 002222  INC          FATFLG          :SET FATAL ERROR FLAG
1728 035072          CKLOOP          :LOOP ON ERROR, IF FLAG SET
      035072 104406          TRAP  C$CLP1
1729          :      If the TSBA Address Register NOT= Expected Then Print Error
1730 035074 016501 000000  MOV          TSBA(R5),R1      :GET TSBA REGISTER CONTENTS
1731 035100 012702 040212  MOV          #T16BFR,R2      :START OF THE DATA BUFFER
1732 035104 062702 000020  ADD          #16.,R2          :EXPECTED CONTENTS OF TSBA
1733 035110          FORCERROR          72$,NOTSSR      :@@DFORCE ERROR IF FORCER=1
1734 035120 020102  CMP          R1,R2          :COMPARE EXPECTED TO RECEIVED
1735 035122 001404  BEQ          80$          :ERROR IF NOT EQUAL
1736 035124          NEXT.ERRNO
1737 035124          ERRHRD  ERRNO,T16TSBA,EXPREC  :PRINT THE ERROR & EXPD/RECV
      035124 104456          TRAP  C$ERHRD
      035126 000770          .WORD  504
      035130 037060          .WORD  T16TSBA
      035132 015474          .WORD  EXPREC
1738 035134          CKLOOP          :LOOP ON ERROR, IF FLAG SET
      035134 104406          TRAP  C$CLP1
1739          :      Compare the controller ram to the extended characteristic word
1740          :      If Data word in controller ram NOT= to word sent Then Print Error
1741 035136 012704 040200  MOV          #T16DATA,R4      :GET CHARACTERISTIC DATA ADDRESS
1742 035142 004737 011224  JSR          PC,CKRAM2        :DOES RAM DATA EQUAL DATA SENT?
1743 035146          FORCERROR          92$          :@@DFORCE ERROR IF FORCER=1
1744 035162 103404  BCS          100$          :BR IF YES
1745 035164          NEXT.ERRNO
1746 035164          ERRHRD  ERRNO,PKTRAM,RAMERR  :REPORT THE RAM ERROR(S)
      035164 104456          TRAP  C$ERHRD
      035166 000771          .WORD  505
      035170 004745          .WORD  PKTRAM
      035172 015510          .WORD  RAMERR
1747 035174          CKLOOP          :LOOP ON ERROR, IF FLAG SET
      035174 104406          TRAP  C$CLP1
1748          :      If Message Buffer Data Length NOT= 12. Then Print Error
1749 035176 012702 040212  MOV          #T16BFR,R2      :GET MESSAGE BUFFER ADDRESS
1750 035202 016201 000002  MOV          2(R2),R1        :GET RECV DATA FIELD LENGTH
1751 035206 012702 000014  MOV          #12.,R2         :GET EXPD DATA FIELD LENGTH
1752 035212          FORCERROR          112$,NOTSSR      :@@DFORCE ERROR IF FORCER=1
1753 035222 020102  CMP          R1,R2          :COMPARE EXPECTED TO RECEIVED
1754 035224 001404  BEQ          120$          :ERROR IF NOT EQUAL
1755 035226          NEXT.ERRNO
1756 035226          ERRHRD  ERRNO,T16LEN,EXPREC  :PRINT THE ERROR & EXPD/RECV
      035226 104456          TRAP  C$ERHRD
      035230 000772          .WORD  506
      035232 037162          .WORD  T16LEN
      035234 015474          .WORD  EXPREC
1757 035236          CKLOOP          :LOOP ON ERROR, IF FLAG SET
      035236 104406          TRAP  C$CLP1
1758
    
```

```

1759 035240 004737 015774      JSR    PC,SOFINIT      ;WRITE TO TSSR TO SOFT INITIALIZE
1760 035244 103405      BCS    125$            ;BR IF SOFT INIT OKAY
1761 035246 010001      MOV    R0,R1          ;SAVE CONTENTS OF TSSR
1762 035250      ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
      035250 104455      TRAP   C$ERDF
      035252 000772      .WORD 506
      035254 003652      .WORD SFIERR
      035256 012034      .WORD SFIMSG
1763 035260      125$: CKLOOP          ;LOOP IF SELECTED
      035260 104406      TRAP   C$CLP1
1764 035262 000137 035446      JMP    300$           ;
1765
1766      ; (* Verify Extended Features switch can be Inverted to CLEAR *)
1767 035266      200$:
1768      ; Do Write Subsystem Write Miscellaneous to CLEAR Extended Features.
1769 035266 004737 040150      JSR    PC,T16SEXT     ;SETUP PACKET FOR WRITE MISC INVERT
1770 035272 012704 040240      MOV    #T16PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
1771 035276 010465 000000      MOV    R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
1772 035302 004737 016336      JSR    PC,CHKTSSR    ;WAIT FOR SSR TO SET
1773 035306      FORCERROR 232$      ;@@DFORCE ERROR IF FORCER=1
1774 035322 103407      BCS    240$          ;BR IF CARRY SET (GOOD RETURN)
1775 035324 010001      MOV    R0,R1          ;SAVE CONTENTS OF TSSR
1776 035326      NEXT.ERRNO
1777 035326      232$: ERRDF  ERRNO,T162SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      035326 104455      TRAP   C$ERDF
      035330 000773      .WORD 507
      035332 036747      .WORD T162SSR
      035334 012046      .WORD PKTSSR
1778 035336 005237 002222      INC    FATFLG        ;SET FATAL ERROR FLAG
1779 035342      240$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      035342 104406      TRAP   C$CLP1
1780
1781      ; DO a WRITE CHARACTERISTICS without an extended characteristic word
1782 035344 012704 040170      MOV    #T16PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
1783 035350 012764 000016 000006      MOV    #14.,PKBCNT(R4) ;STORE MESSAGE PACKET SIZE
1784 035356 004737 010662      JSR    PC,WRTCHR     ;DO WRITE CHARACTERISTICS COMMAND
1785 035362      FORCERROR 242$      ;@@DFORCE ERROR IF FORCER=1
1786 035376 103407      BCS    250$          ;BR IF CARRY SET (GOOD RETURN)
1787 035400 010001      MOV    R0,R1          ;SAVE CONTENTS OF TSSR
1788 035402      NEXT.ERRNO
1789 035402      242$: ERRDF  ERRNO,T16SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      035402 104455      TRAP   C$ERDF
      035404 000774      .WORD 508
      035406 036712      .WORD T16SSR
      035410 012046      .WORD PKTSSR
1790 035412 005237 002222      INC    FATFLG        ;SET FATAL ERROR FLAG
1791 035416      250$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      035416 104406      TRAP   C$CLP1
1792
1793      ; If Message Buffer Data Length NOT= 10. Then Print Error
1793 035420 013701 040214      MOV    T16BFR+2,R1   ;GET RECV DATA FIELD LENGTH
1794 035424 012702 000012      MOV    #10.,R2      ;GET EXPD DATA FIELD LENGTH
1795 035430 020102      CMP    R1,R2         ;COMPARE EXPECTED TO RECEIVED
1796 035432 001404      BEQ    270$          ;ERROR IF NOT EQUAL
1797 035434      NEXT.ERRNO
1798 035434      262$: ERRHRD  ERRNO,T16LEN,EXPREC ;PRINT THE ERROR & EXPD/RECV
      035434 104456      TRAP   C$ERHRD
      035436 000775      .WORD 509
    
```



```

035440 037162
035442 015474
1799 035444 104406 270$: CKLOOP ;LOOP ON ERROR, IF FLAG SET TRAP C$CLP1
1800
1801
1802
1803 ; (* Verify Function Reject when Message Buffer 21-19 are non-zero *)
1804 035446 300$: Write to TSSR register to soft initialize the controller
1805
1806 035446 012737 000001 002312 320$: REPEAT FOR MESSAGE BUFFER ADDRESS bits <21:19> FROM 0 TO 7
1807 ; MOV #1,DATA ;START AT BITS<21:19>=001
1808 035454 325$: DO a WRITE CHARACTERISTICS with a message address bit<21:19> non-zero
1809 035454 012704 040170 MOV #T16PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
1810 035460 012764 000016 000006 MOV #14.,PKBCNT(R4) ;STORE MESSAGE PACKET SIZE
1811 035466 013700 002312 MOV DATA,R0 ;GET TEST DATA
1812 ; .REPT 3
1813 ; ASL R0 ;SHIFT INTO BITS 21:19
1814 ; .ENDR
1815 035500 010037 040202 MOV R0,T16DATA+2 ;STORE BUFFER ADDRESS BITS 21:19
1816 035504 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
1817 035510 004737 016250 JSR PC,WAITF ;WAIT FOR SSR
1818 035514 FORCERROR 342$ ;@@DFORCE ERROR IF FORCER=1
1819 035530 103407 BCS 350$ ;BR IF CARRY SET (GOOD RETURN)
1820 035532 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
1821 035534 NEXT.ERRNO
1822 035534 342$: ERRDF ERRNO,T16SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
035534 104455 TRAP C$SERDF
035536 000776 .WORD 510
035540 036712 .WORD T16SSR
035542 012046 .WORD PKTSSR
1823 035544 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
1824 035550 350$: CKLOOP ;LOOP ON ERROR, IF FLAG SET TRAP C$CLP1
035550 104406
1825
1826 ; If TSSR termination code NOT= Function Reject Then Print Error
1827 035552 016501 000002 MOV TSSR(R5),R1 ;GET RECV TSSR
1828 035556 010102 MOV R1,R2 ;COPY RECV TSSR
1829 035560 042702 000016 BIC #TERCLS,R2 ;CLEAR TC<2:0> EXPD
1830 035564 052702 000006 BIS #TSREJ,R2 ;SET EXPD TC<2:0>= FUNCTION REJECT
1831 035570 FORCERROR 352$,NOTSSR ;@@DFORCE ERROR IF FORCER=1
1832 035600 020102 CMP R1,R2 ;EXPD EQUAL RECV?
1833 035602 001404 BEQ 360$ ;BR IF YES
1834 035604 NEXT.ERRNO
1835 035604 352$: ERRHRD ERRNO,T16REJ,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
035604 104456 TRAP C$SERHRD
035606 000777 .WORD 511
035610 037274 .WORD T16REJ
035612 012046 .WORD PKTSSR
1836 035614 360$: CKLOOP ;LOOP ON ERROR, IF FLAG SET TRAP C$CLP1
035614 104406
1837 035616 FORCEEXIT 370$
1838 035626 005237 002312 INC DATA ;GET NEXT TST PATTERN
1839 035632 023727 002312 000007 CMP DATA,#7 ;DONE ALL DATA?
1840 035640 101002 BHI 370$ ;BR IF YES
1841 035642 000137 035454 JMP 325$ ;DO ANOTHER TEST PATTERN
1842 ; END-REPEAT

```

1843 035646  
1844 035646  
035646  
035646

104403

370\$:

ENDSUB

;/!!!!!!!!!!!! END SUBTEST !!!!!!!!!!!!!

L10054:

TRAP C\$ESUB

1845  
1846 035650 005737 002222  
1847 035654 001402  
1848 035656 004737 017202  
1849 035662

460\$:

TST FATFLG  
BEQ 460\$  
JSR PC,CKDROP

;ANY FATAL ERRORS ?  
;BRANCH IF NOT  
;TRY TO DROP THE UNIT

1850  
1851  
1852  
1853  
1854

1856  
 1857  
 1858  
 1859  
 1860  
 1861  
 1862  
 1863  
 1864  
 1865  
 1866  
 1867  
 1868  
 1869  
 1870  
 1871  
 1872  
 1873  
 1874  
 1875  
 1876  
 1877  
 1878  
 1879  
 1880  
 1881  
 1882  
 1883  
 1884  
 1885  
 1886  
 1887  
 1888  
 1889  
 1890  
 1891  
 1892  
 1893  
 1894  
 1895  
 1896  
 1897  
 1898  
 1899  
 1900  
 1901  
 1902  
 1903  
 1904  
 1905  
 1906

.SBTTL TEST 5: SUBTEST 2: VERIFY TIMERS A,B

++  
 TEST 5: SUBTEST 2:

SUBTEST DESCRIPTION:

This subtest verifies that timers A,B can be reset and that Timer A is twice the frequency of Timer B. Timer A has a period of 25 microseconds and Timer B has a period of 50 microseconds. The timers are checked at 1, 28, 53, and 78 microseconds.

TEST STEPS:

Write to TSSR register to soft initialize the controller  
 Do WRITE CHARACTERISTICS to setup a Message Buffer  
 (\* Verify Timers A,B after RESET TIMER with 0 microsecond delay \*)  
 Do a Write Control RESET TIMER with 1 microsecond delay  
 Do a Write Subsystem READ STATUS  
 If Timer A NOT= 0 Then Print Error  
 If Timer B NOT= 0 Then Print Error  
 (\* Verify Timers A,B after RESET TIMER with 28 microsecond delay \*)  
 Do a Write Control RESET TIMER with 28 microsecond delay  
 If Timer A NOT= 1 Then Print Error  
 If Timer B NOT= 1 Then Print Error  
 Do a Write Control RESET TIMER with 53 microsecond delay  
 If Timer A NOT= 0 Then Print Error  
 If Timer B NOT= 1 Then Print Error  
 Do a Write Control RESET TIMER with 78 microsecond delay  
 If Timer A NOT= 1 Then Print Error  
 If Timer B NOT= 0 Then Print Error

---  
 BGNSUB ;////////// BEGIN SUBTEST //////////  
 T5.2:

TRAP CSBSUB  
 Write to TSSR register to soft initialize the controller

5\$:

JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT  
 TRAP CSERDF  
 .WORD 511  
 .WORD SFIERR  
 .WORD SFIMSG

10\$:

Do WRITE CHARACTERISTICS to setup a Message Buffer  
 JSR PC,T16REST ;RESTORE PACKET DEFAULTS  
 CLR FATFLG ;CLEAR FATAL ERROR FLAG  
 MOV #T16PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET  
 MOV #8.,PKBCNT(R4) ;MESSAGE PACKET SIZE NO EXTEND  
 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND  
 FORCERROR 12\$ ;@@DFORCE ERROR IF FORCER=1  
 BCS 15\$ ;BR IF CARRY SET (GOOD RETURN)  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 NEXT.ERRNO

035662  
 035662 104402  
 035664  
 035664 004737 015774  
 035670 103405  
 035672 010001  
 035674 104455  
 035676 000777  
 035700 003652  
 035702 012034  
 035704 004737 040010  
 035710 005037 002222  
 035714 012704 040170  
 035720 012764 000010 000006  
 035726 004737 010662  
 035732  
 035746 103407  
 035750 010001  
 035752

```

1907 035752      12$:  ERRDF  ERRNO,T16SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
      035752 104455      TRAP  CSERDF
      035754 001000      .WORD 512
      035756 036712      .WORD T16SSR
      035760 012046      .WORD PKTSSR
1908 035762 005237 002222      INC  FATFLG      ;SET FATAL ERROR FLAG
1909 035766      15$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      035766 104406      TRAP  CSCLP1
1910
1911      ;      (* Verify Timers A,B after RESET TIMER with 1 microsecond delay *)
1912      ;      Do a Write Control RESET TIMER with 1 microsecond delay
1913 035770 012700 000001      MOV  #MS.RSD,R0      ;RESET TIMER COMMAND
1914 035774 013701 036632      MOV  T16D01,R1      ;1 MICROSECOND DELAY
1915 036000 004737 040122      JSR  PC,T16WMISC     ;SETUP T16PK2 COMMAND PACKET
1916 036004 012704 040240      MOV  #T16PK2,R4     ;GET WRITE SUBSYSTEM COMMAND PACKET
1917 036010 010465 000000      MOV  R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
1918 036014 004737 016336      JSR  PC,CHKTSSR    ;WAIT FOR SSR TO SET
1919 036020      FORCERROR 32$      ;@@DFORCE ERROR IF FORCER=1
1920 036034 103407      BCS  40$          ;BR IF CARRY SET (GOOD RETURN)
1921 036036 010001      MOV  R0,R1        ;SAVE CONTENTS OF TSSR
1922 036040
1923 036040      32$:  ERRDF  ERRNO,T162SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
      036040 104455      TRAP  CSERDF
      036042 001001      .WORD 513
      036044 036747      .WORD T162SSR
      036046 012046      .WORD PKTSSR
1924 036050 005237 002222      INC  FATFLG      ;SET FATAL ERROR FLAG
1925 036054      40$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      036054 104406      TRAP  CSCLP1
1926      ;      If Timer A NOT= 0 Then Print Error
1927      ;      If Timer B NOT= 0 Then Print Error
1928 036056 005002      CLR  R2          ;INIT EXPD
1929 036060 042702 000010      BIC  #S2.ATIM,R2   ;TIMER A EXPD=0
1930 036064 042702 000004      BIC  #S2.BTIM,R2   ;TIMER B EXPD=0
1931 036070 012700 040232      MOV  #T16BFSTA,R0 ;GET RECV READ STATUS
1932 036074 016001 000002      MOV  2(R0),R1     ;GET RECV BYTE 2
1933 036100 042701 177763      BIC  #*C<S2.ATIM!S2.BTIM>,R1 ;SAVE TIMER A:B RECV ONLY
1934 036104      FORCERROR 72$,NOTSSR      ;@@
1935 036114 020201      CMP  R2,R1        ;EXPD EQUAL RECV?
1936 036116 001404      BEQ  80$          ;BR IF YES
1937 036120      NEXT.ERRNO
1938 036120      72$:  ERRHRD  ERRNO,T16T01,TIMEXP      ;REPORT ERROR
      036120 104456      TRAP  CSERHRD
      036122 001002      .WORD 514
      036124 037411      .WORD T16T01
      036126 015552      .WORD TIMEXP
1939 036130      80$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      036130 104406      TRAP  CSCLP1
1940
1941      ;      Do a Write Control RESET TIMER with 28 microsecond delay
1942 036132 012700 000001      MOV  #MS.RSD,R0      ;RESET TIMER COMMAND
1943 036136 013701 036634      MOV  T16D28,R1      ;28 MICROSECOND DELAY
1944 036142 004737 040122      JSR  PC,T16WMISC     ;SETUP T16PK2 COMMAND PACKET
1945 036146 012704 040240      MOV  #T16PK2,R4     ;GET WRITE SUBSYSTEM COMMAND PACKET
1946 036152 010465 000000      MOV  R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
1947 036156 004737 016336      JSR  PC,CHKTSSR    ;WAIT FOR SSR TO SET
1948 036162      FORCERROR 112$      ;@@DFORCE ERROR IF FORCER=1
    
```

```

1949 036176 103407          BCS      120$          ;BR IF CARRY SET (GOOD RETURN)
1950 036200 010001          MOV      R0,R1        ;SAVE CONTENTS OF TSSR
1951 036202                NEXT.ERRNO
1952 036202                112$:  ERRDF  ERRNO,T162SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      036202 104455                TRAP      C$ERDF
      036204 001003                .WORD    515
      036206 036747                .WORD    T162SSR
      036210 012046                .WORD    PKTSSR
1953 036212 005237 002222          INC      FATFLG        ;SET FATAL ERROR FLAG
1954 036216                120$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      036216 104406                TRAP      C$CLP1
1955                ;      If Timer A NOT= 1 Then Print Error
1956                ;      If Timer B NOT= 1 Then Print Error
1957 036220 005002          CLR      R2            ;INIT EXPD
1958 036222 052702 000010          BIS      #S2.ATIM,R2   ;TIMER A EXPD=1
1959 036226 052702 000004          BIS      #S2.BTIM,R2   ;TIMER B EXPD=1
1960 036232 012700 040232          MOV      #T16BFSTA,R0  ;GET RECV READ STATUS
1961 036236 016001 000002          MOV      2(R0),R1      ;GET RECV BYTE 2
1962 036242 042701 177763          BIC      #^C<S2.ATIM!S2.BTIM>,R1 ;SAVE TIMER A:B RECV ONLY
1963 036246                FORCERRR 172$,NOTSSR ;@@@
1964 036256 020201          CMP      R2,R1        ;EXPD EQUAL RECV?
1965 036260 001404          BEQ     180$          ;BR IF YES
1966 036262                NEXT.ERRNO
1967 036262                172$:  ERRHRD  ERRNO,T16T28,TIMEXP ;REPORT ERROR
      036262 104456                TRAP      C$ERHRD
      036264 001004                .WORD    516
      036266 037510                .WORD    T16T28
      036270 015552                .WORD    TIMEXP
1968 036272                180$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      036272 104406                TRAP      C$CLP1
1969                ;
1970                ;      Do a Write Control RESET TIMER with 53 microsecond delay
1971 036274 012700 000001          MOV      #MS.RSD,R0    ;RESET TIMER COMMAND
1972 036300 013701 036636          MOV      T16D53,R1     ;53 MICROSECOND DELAY
1973 036304 004737 040122          JSR      PC,T16WMISC    ;SETUP T16PK2 COMMAND PACKET
1974 036310 012704 040240          MOV      #T16PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
1975 036314 010465 000000          MOV      R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
1976 036320 004737 016336          JSR      PC,CHKTSSR    ;WAIT FOR SSR TO SET
1977 036324                FORCERRR 212$          ;@@@FORCE ERROR IF FORCER=1
1978 036340 103407          BCS      220$          ;BR IF CARRY SET (GOOD RETURN)
1979 036342 010001          MOV      R0,R1        ;SAVE CONTENTS OF TSSR
1980 036344                NEXT.ERRNO
1981 036344                212$:  ERRDF  ERRNO,T162SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      036344 104455                TRAP      C$ERDF
      036346 001005                .WORD    517
      036350 036747                .WORD    T162SSR
      036352 012046                .WORD    PKTSSR
1982 036354 005237 002222          INC      FATFLG        ;SET FATAL ERROR FLAG
1983 036360                220$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      036360 104406                TRAP      C$CLP1
1984                ;      If Timer A NOT= 0 Then Print Error
1985                ;      If Timer B NOT= 1 Then Print Error
1986 036362 005002          CLR      R2            ;INIT EXPD
1987 036364 042702 000010          BIC      #S2.ATIM,R2   ;TIMER A EXPD=0
1988 036370 052702 000004          BIS      #S2.BTIM,R2   ;TIMER B EXPD=1
1989 036374 012700 040232          MOV      #T16BFSTA,R0  ;GET RECV READ STATUS
1990 036400 016001 000002          MOV      2(R0),R1      ;GET RECV BYTE 2
    
```

```

1991 036404 042701 177763      BIC      #^C<S2.ATIM!S2.BTIM>,R1  :SAVE TIMER A:B RECV ONLY
1992 036410                    FORCERROR      272$,NOTSSR      :@@D
1993 036420 020201            CMP      R2,R1                :EXPD EQUAL RECV?
1994 036422 001404            BEQ      280$                  :BR IF YES
1995 036424                    NEXT.ERRNO
1996 036424                    272$:  ERRHRD  ERRNO,T16T53,TIMEXP  :REPORT ERROR
      036424 104456                    TRAP      C$ERHRD
      036426 001006                    .WORD    518
      036430 037610                    .WORD    T16T53
      036432 015552                    .WORD    TIMEXP
1997 036434                    280$:  CKLOOP                  :LOOP ON ERROR, IF FLAG SET
      036434 104406                    TRAP      C$CLP1
1998 :
1999 036436 012700 000001      : Do a Write Control RESET TIMER with 78 microsecond delay
2000 036442 013701 036640      MOV      #MS.RSD,R0            :RESET TIMER COMMAND
2001 036446 004737 040122      MOV      T16D78,R1            :78 MICROSECOND DELAY
2002 036452 012704 040240      JSR      PC,T16WMISC          :SETUP T16PK2 COMMAND PACKET
2003 036456 010465 000000      MOV      #T16PK2,R4          :GET WRITE SUBSYSTEM COMMAND PACKET
2004 036462 004737 015336      MOV      R4,TSDB(R5)         :SET THE PACKET ADDRESS TO EXECUTE
2005 036466                    JSR      PC,CHKTSSR           :WAIT FOR SSR TO SET
2006 036502 103407            FORCERROR      312$          :@@DFORCE ERROR IF FORCER=1
2007 036504 010001            BCS      320$                  :BR IF CARRY SET (GOOD RETURN)
2008 036506                    MOV      R0,R1                :SAVE CONTENTS OF TSSR
2009 036506                    312$:  ERRDF   ERRNO,T162SSR,PKTSSR  :DEVICE FATAL SSR FAILED TO SET
      036506 104455                    TRAP      C$ERDF
      036510 001007                    .WORD    519
      036512 036747                    .WORD    T162SSR
      036514 012046                    .WORD    PKTSSR
2010 036516 005237 002222      INC      FATFLG                :SET FATAL ERROR FLAG
2011 036522                    320$:  CKLOOP                  :LOOP ON ERROR, IF FLAG SET
      036522 104406                    TRAP      C$CLP1
2012 :
2013 :
2014 036524 005002            : If Timer A NOT= 1 Then Print Error
2015 036526 052702 000010      : If Timer B NOT= 0 Then Print Error
2016 036532 042702 000004      CLR      R2                    :INIT EXPD
2017 036536 012700 040232      BIS      #S2.ATIM,R2          :TIMER A EXPD=1
2018 036542 016001 000002      BIC      #S2.BTIM,R2          :TIMER B EXPD=0
2019 036546 042701 177763      MOV      #T16BFSTA,R0        :GET RECV READ STATUS
2020 036552                    MOV      2(R0),R1            :GET RECV BYTE 2
2021 036562 020201            BIC      #^C<S2.ATIM!S2.BTIM>,R1  :SAVE TIMER A:B RECV ONLY
2022 036564 001404            FORCERROR      372$,NOTSSR      :@@D
2023 036566                    CMP      R2,R1                :EXPD EQUAL RECV?
2024 036566                    372$:  ERRHRD  ERRNO,T16T78,TIMEXP  :REPORT ERROR
      036566 104456                    TRAP      C$ERHRD
      036570 001010                    .WORD    520
      036572 037710                    .WORD    T16T78
      036574 015552                    .WORD    TIMEXP
2025 036576                    380$:  CKLOOP                  :LOOP ON ERROR, IF FLAG SET
      036576 104406                    TRAP      C$CLP1
2026 :
2027 036600                    ENDSUB                          :////////// END SUBTEST ////////////
      036600                    L10055:
      036600 104403                    TRAP      C$ESUB
2028 :
2029 036602 005737 002222      TST      FATFLG                :ANY FATAL ERRORS ?
2030 036606 001402            BEQ      460$                  :BRANCH IF NOT
    
```

2031 036610 004737 017202  
2032 036614 004737 016456  
2033 036620 103002  
2034 036622 000137 034602  
2035 036626  
2036  
2037  
2038 036626  
036626 104432  
036630 001524  
2039  
2040

460\$: JSR PC,CKDROP  
JSR PC,TSTLOOP  
BCC 465\$  
JMP T16LOOP  
465\$:

:TRY TO DROP THE UNIT  
:SHOULD WE DO ITERATIONS?  
:BR IF NO  
:LOOP UNTIL ITERATIONS DONE

EXIT TST

:////////////// EXIT TEST //////////////

TRAP C\$EXIT  
.WORD L10053-

```

2042
2043      ;+
2044      ; LOCAL STORAGE FOR THIS TEST
2045      ;-
2045 036632 000001 T16D01:      .WORD 1      ;1 MICROSECOND DELAY (ACTUALLY .8 MIC)
2046 036634 000040 T16D28:      .WORD 40     ;28 MICROSECOND DELAY (.8 MICROS PER)
2047 036636 000076 T16D53:      .WORD 76     ;53 MICROSECOND
2048 036640 000142 T16D78:      .WORD 142    ;78 MICROSECOND
2049
2050      ;+
2051      ; LOCAL TEXT MESSAGES FOR TEST
2052      ;-
2053 036642      105      170      164 TST16ID:      .ASCII 'Extended Features Switch and Timers A,B'
2054 036712      127      122      111 T16SSR: .ASCII 'WRITE CHARACTERISTICS Failed'
2055 036747      127      122      111 T162SSR: .ASCII 'WRITE SUBSYSTEM (Write Misc) Failed'
2056 037013      127      122      111 T163SSR: .ASCII 'WRITE SUBSYSTEM (Read Status) Failed'
2057 037060      102      165      163 T16TSBA: .ASCII 'Bus Address Register (TSBA) Incorrect after Write Characteristics'
2058 037162      104      141      164 T16LEN: .ASCII 'Data Field Length in Message Buffer Incorrect after Write Characteristics'
2059 037274      124      123      123 T16REJ: .ASCII 'TSSR Function Reject Not Returned When Non-Existent Buffer Address Specifie
2060 037411      124      151      155 T16T01: .ASCII 'Timer A,B Incorrect after Reset Timer with 1 microsecond Delay'
2061 037510      124      151      155 T16T28: .ASCII 'Timer A,B Incorrect after Reset Timer with 28 microsecond Delay'
2062 037610      124      151      155 T16T53: .ASCII 'Timer A,B Incorrect after Reset Timer with 53 microsecond Delay'
2063 037710      124      151      155 T16T78: .ASCII 'Timer A,B Incorrect after Reset Timer with 78 microsecond Delay'
2064      .EVEN
2065
2066      ;+
2067      ; SET DEFAULT PACKET
2068      ;-
2069 040010      T16REST:
2070 040010 012700 040170      MOV      #T16PACKET,R0      ;PACKET ADDRESS
2071 040014 012720 100004      MOV      #100004,(R0)+      ;WRITE CHARACTERISTICS WITH ACK
2072 040020 012720 040200      MOV      #T16DATA,(R0)+      ;ADDRESS OF CHAR DATA BLOCK
2073 040024 005020      CLR      (R0)+              ;EXTENDED ADDRESS
2074 040026 012720 000012      MOV      #10,(R0)+          ;SIZE OF MESSAGE PACKET
2075 040032 012720 040212      MOV      #T16BFR,(R0)+      ;MESSAGE BUFFER ADDRESS
2076 040036 005020      CLR      (R0)+              ;CLEAR EXTENDED BUFFER ADDRESS
2077 040040 012720 000024      MOV      #20,(R0)+          ;LENGTH OF MESSAGE BUFFER
2078 040044 005020      CLR      (R0)+              ;CLEAR ESS,EMB,EAI,ERI
2079 040046 005010      CLR      (R0)               ;CLEAR EXTENDED FEATURES WORD
2080 040050 005037 040212      CLR      T16BFR              ;CLEAR 1ST LOCATION IN MESSAGE BUFFER
2081 040054 000207      RTS      PC                  ;
2082
2083      ;+
2084      ; CLEAR MESSAGE BUFFER
2085      ;-
2086 040056      T16CLRBUF:
2087 040056      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
2088 040062 012701 040212      MOV      #T16BFR,R1          ;GET MESSAGE BUFFER ADDRESS
2089 040066 012702 000026      MOV      #T16BEND-T16BFR,R2 ;SIZE OF MESSAGE BUFFER IN BYTES
2090 040072 105021 10$:      CLRB      (R1)+              ;CLEAR A BYTE
2091 040074 005302      DEC      R2                  ;DONE?
2092 040076 003375      BGT      10$                 ;BR IF NO
2093 040100 000207      RTS      PC                  ;RETURN
2094
2095      ;+
2096      ; SETUP T16PK2 PACKET FOR READ STATUS
2097      ;-
2098 040102      T16SRD:
    
```



```

2099 040102 004737 040056      JSR      PC,T16CLRBUF      ;CLEAR MESSAGE BUFFER
2100 040106 012700 040250      MOV      #T16DT2,R0       ;WRITE SUBSYSTEM DATA BUFFER
2101 040112 112720 000005      MOVB    #PW.RDSTATUS,(R0)+ ;STORE READ STATUS COMMAND IN BSELO
2102 040116 105010              CLRB     (R0)             ;CLEAR BSEL1
2:03 040120 000207      RTS      PC               ;RETURN
2104
2105
2106
2107      ;+
2108      ; SETUP T16PK2 PACKET FOR WRITE MISC.
2109
2110      ; INPUT:
2111      ; R0      CONTAINS WRITE MISC FUNCTION CODE (BSEL1)
2112      ; R1      CONTAINS DELAY (TIMES 800 NS) FOR BSEL2
2113
2114      ;-
2115      T16WMISC:
2116      SAVREG                    ;SAVE R1-R5 UNTIL NEXT RETURN
2117      JSR      PC,T16CLRBUF      ;CLEAR MESSAGE BUFFER
2118      MOV      #T16DT2,R2       ;WRITE SUBSYSTEM DATA BUFFER
2119      MOVB    #PW.WMISC,(R2)+   ;STORE WRITE MISCELLANEOUS IN BSELO
2120      MOVB    R0,(R2)+         ;STORE WRITE MISC CODE IN BSEL1
2121      MOVB    R1,(R2)         ;STORE DELAY (RESET TIMER) IN BSEL2
2122      RTS      PC               ;RETURN
2123
2124      ;+
2125      ; SETUP T16PK2 PACKET FOR WRITE MISC. INVERT EXTENDED FEATURES SWITCH
2126
2127      ;-
2128      T16SEXT:
2129      MOV      #T16DT2,R0       ;WRITE SUBSYSTEM DATA BUFFER
2130      MOVB    #PW.WMISC,(R0)+   ;STORE WRITE MISCELLANEOUS IN BSELO
2131      MOVB    #MS.EXT,(R0)     ;STORE INVERT EXTENDED FEATURES IN BSEL1
2132      RTS      PC               ;RETURN
2133
2134      ;
2135      ; .=<. +10>B177770
2136
2137      ; WRITE CHARACTERISTICS COMMAND PACKET
2138
2139      ;-
2140      T16PACKET:
2141      ; .WORD 100004           ;COMMAND PACKET FOR TEST
2142      ; .WORD T16DATA         ;WRITE CHARACTERISTICS COMMAND, WITH ACK
2143      ; .WORD 0               ;ADDRESS OF CHARACTERISTICS BLOCK
2144      ; .WORD 10.             ;MESSAGE PACKET SIZE
2145
2146      ;-
2147      T16DATA:
2148      ; .WORD T16BFR         ;CHARACTERISTICS DATA BLOCK
2149      ; .WORD 0               ;ADDRESS OF MESSAGE BUFFER
2150      ; .WORD 20.            ;LENGTH OF MESSAGE BUFFER
2151      ; .WORD 0               ;ESS,EMB,EAI,ERI
2152      ; .WORD 0               ;EXTENDED FEATURES WORD
2153
2154      ; MESSAGE BUFFER
2155
2156      ;-
2157      T16BFR:
2158      ; .WORD 0               ;BEGIN MESSAGE BUFFER
2159      ; .WORD 0               ;MESSAGE TYPE
2160      ; .WORD 0               ;DATA FIELD LENGTH
    
```

```

2158 040216 000000          .WORD 0          :RBPCR
2159 040220 000000          .WORD 0          :XST0
2160 040222 000000          .WORD 0          :XST1
2161 040224 000000          .WORD 0          :XST2
2162 040226 000000          .WORD 0          :XST3
2163 040230 000000          .WORD 0          :XST4 (ALWAYS PRESENT FOR WRITE SUBSYSTEM
2164 040232          T16BFSTA: .BLKB 6.      :READ STATUS AND WRITE FIFO BUFFER
2165 040240          T16BEND:          :END OF MESSAGE BUFFER
2166
2167          :WRITE SUBSYSTEM READ STATUS COMMAND PACKET
2168
2172 040240          T16PK2:
2173 040240 100006          .WORD P.WRTSUB!P.ACK :WRITE SUBSYSTEM WITH ACK
2174 040242 040250          .WORD T16DT2         :LOW ADDRESS OF DATA BLOCK
2175 040244 000000          .WORD 0              :HIGH ADDRESS OF DATA BLOCK
2176 040246 000012          .WORD 10.           :MINIMUM MESSAGE PACKET SIZE
2177
2178 040250          T16DT2:          :DATA BLOCK
2179 040250          .BYTE 0              :BSEL0
2180 040251          .BYTE 0              :BSEL1
2181 040252 000000          .WORD 0              :SEL2
2182 040254          .BLKB 64.          :WRITE FIFO DATA OUTPUT BUFFER
2183
2184
2185 040354          ENDTST
      040354
      040354 104401          L10053:          TRAP C$ETST
  
```

2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2215  
2216  
2217  
2218  
2219  
2220  
2221  
2222  
2223

.SBTTL TEST 6: FIFO EXERCISER

:\*\*  
: TEST DESCRIPTION:

: This test uses the Write Subsystem Memory command to  
: verify the controller's FIFO and associated status and  
: control logic.

: TEST STEPS:

: REPEAT FOR LOOPCNT

: BEGIN

- : Do Subtest 1 - FIFO Initialize status test
- : Do Subtest 2 - FIFO Write Single Byte test
- : Do Subtest 3 - FIFO Write Multiple Bytes test
- : Do Subtest 4 - FIFO Verify ILW Status test
- : Do Subtest 5 - FIFO Input Ready test
- : Do Subtest 6 - FIFO Verify Reset FIFO test

: END

:--

BGNTST

```

MOV #TST17ID,R0
JSR PC,TSTSETUP
MOV #10,LOOPCNT
JSR PC,KTOFF
CLR KTENABLE

```

T17LOOP:

T6::

```

:ASCII MESSAGE TO IDENTIFY TEST
:DO INITIAL TEST SETUP
:PERFORM 10 ITERATIONS
:SHUT OFF MEMORY MANAGEMENT
:REALLY SHUT DOWN KT-11

```

```

012700 046606
004737 016510
012737 000012 002216
004737 017274
005037 003134

```

```

2225 .SBTTL TEST 6: SUBTEST 1: FIFO INITIALIZE STATUS TEST
2226
2227 ++
2228 TEST 6: SUBTEST 1:
2229
2230 SUBTEST DESCRIPTION:
2231
2232 This test verifies, by using the Read Status select code,
2233 that the FIFO status is in the correct initial state after
2234 the controller is initialized (Input Ready TRUE,
2235 Output Ready and Data In Miss FALSE). These status
2236 signals are checked by the controller's self-test
2237 sequence, so this subtest is actually more of a partial
2238 check of the Read Status function than the FIFO status.
2239
2240 TEST STEPS:
2241
2242 BEGIN
2243 Write to TSSR to soft initialize
2244 Do a WRITE CHARACTERISTICS to setup a message buffer
2245 Do a WRITE SUBSYSTEM Read Status
2246 If Input Ready NOT=1 Then Print Error
2247 If Output Ready NOT=0 Then Print Error
2248 If Data In Miss NOT=0 Then Print Error
2249
2250 -- END
2251
2252 BGNSUB ;//////////////// BEGIN SUBTEST //////////////////
2253 040404 104402 T6.1: TRAP C$BSUB
2254 040404
2255
2256 5$: Write to TSSR register to soft initialize the controller
2257
2258 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
2259 040406 004737 015774 BCS 10$ ;BR IF SOFT INIT OKAY
2260 040412 103405 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2261 040414 010001 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
2262 040416
2263 040416 104455 TRAP C$ERDF
2264 040420 001130 .WORD 600
2265 040422 003652 .WORD SFIERR
2266 040424 012034 .WORD SFIMSG
2267
2268 10$: Do a WRITE CHARACTERISTICS to setup a message buffer
2269
2270 CLR FATFLG ;CLEAR FATAL ERROR FLAG
2271 040426 005037 002222 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
2272 040432 012704 050200 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
2273 040436 004737 010662 FORCERROR 42$ ;@@DFORCE ERROR IF FORCER=1
2274 040442
2275 040456 103407 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
2276 040460 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2277 040462
2278 42$: NEXT.ERRNO
2279 040462 104455 ERRDF ERRNO,T17SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
2280 040464 001131 TRAP C$ERDF
2281 040466 046625 .WORD 601
2282 040470 012046 .WORD T17SSR
2283 040472 005237 002222 .WORD PKTSSR
2284 040476
2285 50$: INC FATFLG ;SET FATAL ERROR FLAG
2286 040476 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
2287
2288 TRAP C$CLP1
2289
2290 ; Do a Write Subsystem READ STATUS
    
```

```

2271 040500 004737 047764      JSR      PC,T17SRD      ;SETUP PACKET FOR READ STATUS
2272 040504 012704 050350      MOV      #T17PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
2273 040510 010465 000000      MOV      R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
2274 040514 004737 016336      JSR      PC,CHKTSSR   ;WAIT FOR SSR TO SET
2275 040520                      FORCERROR      62$    ;@@DFORCE ERROR IF FORCER=1
2276 040534 103407                      BCS      70$        ;BR IF CARRY SET (GOOD RETURN)
2277 040536 010001                      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
2278 040540                      NEXT.ERRNO
2279 040540                      62$: ERRDF      ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      040540 104455                      TRAP      C$ERDF
      040542 001132                      .WORD    602
      040544 046726                      .WORD    T173SSR
      040546 012046                      .WORD    PKTSSR
2280 040550 005237 002222      INC      FATFLG      ;SET FATAL ERROR FLAG
2281 040554                      70$: CKLOOP        ;LOOP ON ERROR, IF FLAG SET
      040554 104406                      TRAP      C$CLP1
2282                      ; Set WORDS 0-7 of expd message buffer = to recv since not testing
2283 040556 004737 050146      JSR      PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2284 040562 012701 046402      MOV      #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2285 040566 012702 050242      MOV      #T17BFSTA,R2 ;GET RECV READ STATUS
2286 040572 012221                      MOV      (R2)+,(R1)+ ;SET EXPD WORD #8 = RECV TEMP
2287 040574 011211                      MOV      (R2),(R1)   ;SET EXPD WORD #9 = RECV TEMP
2288 040576 052711 000020      BIS      #S2.INRDY,(R1) ;SET EXP INPUT READY= TRUE
2289 040602 042711 000040      BIC      #S2.OUTRDY,(R1) ;SET EXP OUTPUT READY= FALSE
2290 040606 042711 000200      BIC      #S2.DIM,(R1) ;SET EXP DATA IN MISS = FALSE
2291                      ; If Input Ready NOT=1 then Print Error
2292                      ; If Output Ready NOT=0 or Data in Miss NOT=0 Then Print Error
2293 040612 005000      CLR      R0          ;HIGH RECV ADDRESS FOR CKMSG2
2294 040614 012701 050222      MOV      #T17BFR,R1  ;LOW RECV ADDRESS FOR CKMSG2
2295 040620 012702 046362      MOV      #T17EXP,R2  ;EXPD ADDRESS
2296 040624 012703 000024      MOV      #20,,R3     ;NUMBER OF BYTES TO COMPARE
2297 040630 004737 011500      JSR      PC,CKMSG2   ;EXPD EQUAL RECV?
2298 040634                      FORCERROR      82$,NOTSSR ;@@
2299 040644 103404                      BCS      90$        ;BR IF YES
2300 040646                      NEXT.ERRNO
2301 040646                      82$: ERRHRD      ERRNO,T171CMP,MSGSTAT ;REPORT ERROR
      040646 104456                      TRAP      C$ERHRD
      040650 001133                      .WORD    603
      040652 047145                      .WORD    T171CMP
      040654 012350                      .WORD    MSGSTAT
2302 040656                      90$: CKLOOP        ;LOOP ON ERROR, IF FLAG SET
      040656 104406                      TRAP      C$CLP1
2303                      ;
2304 040660                      ENDSUB              ;////////// END SUBTEST //////////
      040660                      L10057:
      040660 104403                      TRAP      C$ESUB
2305                      ;
2306 040662 005737 002222      TST      FATFLG      ;ANY FATAL ERRORS ?
2307 040666 001402                      BEQ      160$       ;BRANCH IF NOT
2308 040670 004737 017202      JSR      PC,CKDROP   ;TRY TO DROP THE UNIT
2309 040674                      160$:
2310
    
```

2312  
2313  
2314  
2315  
2316  
2317  
2318  
2319  
2320  
2321  
2322  
2323  
2324  
2325  
2326  
2327  
2328  
2329  
2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353  
2354  
2355  
2356

.SBTTL TEST 6: SUBTEST 2: FIFO WRITE SINGLE BYTE TEST

:+  
: TEST 6: SUBTEST 2:

: SUBTEST DESCRIPTION:

: This subtest verifies the ability of the FIFO to correctly  
 : pass a single data byte from input to output. For each  
 : of 256 data values (0-377 octal) the following is done:  
 : 1. Initial FIFO status is checked  
 : 2. The Write FIFO function, specifying a count of  
 : one byte to be written is executed.  
 : 3. Read Status is executed and FIFO status is checked.  
 : 4. Read FIFO is executed and the data and final status  
 : is checked.

: TEST STEPS:

: BEGIN

: Write to TSSR to soft initialize  
 : Do a WRITE CHARACTERISTICS to setup a message buffer  
 : Do a Write Subsystem READ STATUS  
 : If Input Ready NOT=1 Then Print Error  
 : If Output Ready NOT=0 Then Print Error  
 : If Data In Miss NOT=0 Then Print Error

: REPEAT FOR DATA FROM 0 TO 377 OCTAL

: BEGIN

: Do a Write Subsystem WRITE NPR to set tape direction out  
 : Do a Write Subsystem WRITE FIFO with byte count equal to 1  
 : Do a Write Subsystem READ STATUS  
 : If Input Ready NOT=1 Then Print Error  
 : If Output Ready NOT=1 Then Print Error  
 : If Data In Miss NOT=0 Then Print Error  
 : Do Write Subsystem READ FIFO with byte count equal to 1  
 : If Data read from FIFO NOT= to Data sent Then Print Error  
 : Do a Write Subsystem READ STATUS  
 : If Input Ready NOT=1 Then Print Error  
 : If Output Ready NOT=0 Then Print Error  
 : If Data In Miss NOT=0 Then Print Error

: END

: END

-- BGNSUB

:////////// BEGIN SUBTEST //////////  
 T6.2:

TRAP C\$BSUB

: Write to TSSR register to soft initialize the controller

5\$:

JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT

TRAP C\$ERDF  
 .WORD 603  
 .WORD SFIFRR

040674  
040674  
040674 104402  
  
2357  
2358  
2359 040676  
2360 040676 004737 015774  
2361 040702 103405  
2362 040704 010001  
2363 040706  
040706 104455  
040710 001133  
040712 003652

```

040714 012034                                     .WORD SFIMSG
2364                                     :
2365 040716 005037 002222 10$: Do a WRITE CHARACTERISTICS to setup a message buffer
2366 040722 012704 050200 CLR FATFLG ;CLEAR FATAL ERROR FLAG
2367 040726 004737 010662 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
2368 040732 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
2369 040746 103407 FORCERROR 42$ ;@@DFORCE ERROR IF FORCER=1
2370 040750 010001 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
2371 040752 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2372 040752 42$: NEXT.ERRNO
ERRDF ERRNO,T17SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
040752 104455 TRAP C$ERDF
040754 001134 .WORD 604
040756 046625 .WORD T17SSR
040760 012046 .WORD PKTSSR
2373 040762 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
2374 040766 50$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
040766 104406 TRAP C$CLP1
2375 : Do a Write Subsystem READ STATUS
2376 040770 004737 047764 JSR PC,T17SRD ;SETUP PACKET FOR READ STATUS
2377 040774 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
2378 041000 010465 000000 MOV R4,TSD8(R5) ;SET THE PACKET ADDRESS TO EXECUTE
2379 041004 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
2380 041010 FORCERROR 62$ ;@@DFORCE ERROR IF FORCER=1
2381 041024 103407 BCS 70$ ;BR IF CARRY SET (GOOD RETURN)
2382 041026 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2383 041030 62$: NEXT.ERRNO
ERRDF ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
041030 104455 TRAP C$ERDF
041032 001135 .WORD 605
041034 046726 .WORD T173SSR
041036 012046 .WORD PKTSSR
2385 041040 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
2386 041044 70$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
041044 104406 TRAP C$CLP1
2387 : Set WORDS 0-7 of expd message buffer = to recv since not testing
2388 041046 004737 050146 JSR PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2389 041052 012701 046402 MOV #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2390 041056 012702 050242 MOV #T17BFSTA,R2 ;GET RECV READ STATUS
2391 041062 012221 MOV (R2)+,(R1)+ ;SET EXPD WORD #8 = RECV TEMP
2392 041064 011211 MOV (R2),(R1) ;SET EXPD WORD #9 = RECV TEMP
2393 041066 052711 000020 BIS #S2.INRDY,(R1) ;SET EXP INPUT READY= TRUE
2394 041072 042711 000040 BIC #S2.OUTRDY,(R1) ;SET EXP OUTPUT READY= FALSE
2395 041076 042711 000200 BIC #S2.DIM,(R1) ;SET EXP DATA IN MISS = FALSE
2396 : If Input Ready NOT=1 then Print Error
2397 : If Output Ready NOT=0 or Data in Miss NOT=0 Then Print Error
2398 041102 005000 CLR R0 ;HIGH RECV ADDRESS FOR CKMSG2
2399 041104 012701 050222 MOV #T17BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
2400 041110 012702 046362 MOV #T17EXP,R2 ;EXPD ADDRESS
2401 041114 012703 000024 MOV #20,R3 ;NUMBER OF BYTES TO COMPARE
2402 041120 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RECV?
2403 041124 FORCERROR 82$,NOTSSR ;@@
2404 041134 103404 BCS 90$ ;BR IF YES
2405 041136 82$: NEXT.ERRNO
ERRHRD ERRNO,T171CMP,MSGSTAT ;REPORT ERROR
041136 104456 TRAP C$ERHRD
041140 001136 .WORD 606
041142 047145 .WORD T171CMP
    
```

```

2407 041144 012350          90$:  CKLOOP                ;LOOP ON ERROR, IF FLAG SET      .WORD  MSGSTAT
      041146 104406                                     TRAP   C$CLP1
2408
2409          ; Repeat for DATA from 0 to 377
2410 041150 012737 000000 002312  MOV      #0,DATA                ;GET FIRST DATA
2411 041156          100$:          ;REPEAT LABEL
2412          ; Do a Write Subsystem WRITE NPR to set tape direction out
2413 041156 012700 000100  MOV      #NP.OUT,R0            ;SET TAPE DIRECTION OUT
2414 041162 004737 050026  JSR      PC,T17SNPR           ;SETUP T17PK2 FOR WRITE NPR
2415 041166 012704 050350  MOV      #T17PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
2416 041172 010465 000000  MOV      R4,TSDB(R5)         ;SET THE PACKET ADDRESS TO EXECUTE
2417 041176 004737 016336  JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
2418 041202          FORCERROR 102$          ;@DFORCE ERROR IF FORCER=1
2419 041216 103407          BCS      105$                ;BR IF CARRY SET (GOOD RETURN)
2420 041220 010001          MOV      R0,R1                ;SAVE CONTENTS OF TSSR
2421 041222          NEXT.ERRNO
2422 041222          102$:  ERRDF  ERRNO,T174SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      041222 104455                                     TRAP   C$ERDF
      041224 001137                                     .WORD  607
      041226 046773                                     .WORD  T174SSR
      041230 012046                                     .WORD  PKTSSR
2423 041232 005237 002222          INC      FATFLG              ;SET FATAL ERROR FLAG
2424 041236          105$:  CKLOOP                ;LOOP ON ERROR, IF FLAG SET
      041236 104406                                     TRAP   C$CLP1
2425          ; Do a Write Subsystem WRITE FIFO with byte count equal to 1
2426 041240 012700 000001  MOV      #1,R0                ;WRITE 1 BYTE
2427 041244 012701 002312  MOV      #DATA,R1            ;FIFO WRITE DATA ADDRESS
2428 041250 004737 050052  JSR      PC,T17WFIF         ;SETUP T17PK2 FOR WRITE FIFO
2429 041254 012704 050350  MOV      #T17PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
2430 041260 010465 000000  MOV      R4,TSDB(R5)         ;SET THE PACKET ADDRESS TO EXECUTE
2431 041264 004737 016336  JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
2432 041270          FORCERROR 107$          ;@DFORCE ERROR IF FORCER=1
2433 041304 103407          BCS      110$                ;BR IF CARRY SET (GOOD RETURN)
2434 041306 010001          MOV      R0,R1                ;SAVE CONTENTS OF TSSR
2435 041310          NEXT.ERRNO
2436 041310          107$:  ERRDF  ERRNO,T175SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      041310 104455                                     TRAP   C$ERDF
      041312 001140                                     .WORD  608
      041314 047036                                     .WORD  T175SSR
      041316 012046                                     .WORD  PKTSSR
2437 041320 005237 002222          INC      FATFLG              ;SET FATAL ERROR FLAG
2438 041324          110$:  CKLOOP                ;LOOP ON ERROR, IF FLAG SET
      041324 104406                                     TRAP   C$CLP1
2439
2440          ; Do a Write Subsystem READ STATUS
2441 041326 004737 047764  JSR      PC,T17SRD           ;SETUP PACKET FOR READ STATUS
2442 041332 012704 050350  MOV      #T17PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
2443 041336 010465 000000  MOV      R4,TSDB(R5)         ;SET THE PACKET ADDRESS TO EXECUTE
2444 041342 004737 016336  JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
2445 041346          FORCERROR 112$          ;@DFORCE ERROR IF FORCER=1
2446 041362 103407          BCS      120$                ;BR IF CARRY SET (GOOD RETURN)
2447 041364 010001          MOV      R0,R1                ;SAVE CONTENTS OF TSSR
2448 041366          NEXT.ERRNO
2449 041366          112$:  ERRDF  ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      041366 104455                                     TRAP   C$ERDF
      041370 001141                                     .WORD  609
    
```



```

041372 046726 .WORD T173SSR
041374 012046 .WORD PKTSSR
2450 041376 005237 002222 120$: INC FATFLG ;SET FATAL ERROR FLAG
2451 041402 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
041402 104406 TRAP C$CLP1
2452 : Set WORDS 0-7 of expd message buffer = to recv since not testing
2453 041404 004737 050146 JSR PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2454 041410 012701 046402 MOV #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2455 041414 012702 050242 MOV #T17BFSTA,R2 ;GET RECV READ STATUS
2456 041420 012221 MOV (R2)+,(R1)+ ;SET EXPD WORD #8 = RECV TEMP
2457 041422 011211 MOV (R2),(R1) ;SET EXPD WORD #9 = RECV TEMP
2458 041424 052711 000020 BIS #S2.INRDY,(R1) ;SET EXP INPUT READY= 1
2459 041430 052711 000040 BIS #S2.OUTRDY,(R1) ;SET EXP OUTPUT READY= 1
2460 041434 042711 000200 BIC #S2.DIM,(R1) ;SET EXP DATA IN MISS = 0
2461 : If Input Ready NOT=1 then Print Error
2462 : If Output Ready NOT=1 or Data in Miss NOT=0 Then Print Error
2463 041440 005000 CLR R0 ;HIGH RECV ADDRESS FOR CKMSG2
2464 041442 012701 050222 MOV #T17BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
2465 041446 012702 046362 MOV #T17EXP,R2 ;EXPD ADDRESS
2466 041452 012703 000024 MOV #20,R3 ;NUMBER OF BYTES TO COMPARE
2467 041456 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RECV?
2468 041462 FORCERROR 132$,NOTSSR ;@@D
2469 041472 103404 BCS 140$ ;BR IF YES
2470 041474 NEXT.ERRNO
2471 041474 132$: ERRHRD ERRNO,T173CMP,MSGSTAT ;REPORT ERROR
041474 104456 TRAP C$ERHRD
041476 001142 .WORD 610
041500 047323 .WORD T173CMP
041502 012350 .WORD MSGSTAT
2472 041504 140$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
041504 104406 TRAP C$CLP1
2473 : Do Write Subsystem READ FIFO with byte count equal to 1
2474 :
2475 041506 012700 000001 MOV #1,R0 ;SET READ BYTE COUNT
2476 041512 004737 050106 JSR PC,T17RFIF ;SETUP T17PK2 FOR READ FIFO
2477 041516 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
2478 041522 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
2479 041526 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
2480 041532 FORCERROR 142$ ;@@DFORCE ERROR IF FORCER=1
2481 041546 103407 BCS 150$ ;BR IF CARRY SET (GOOD RETURN)
2482 041550 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2483 041552 NEXT.ERRNO
2484 041552 142$: ERRDF ERRNO,T176SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
041552 104455 TRAP C$ERDF
041554 001143 .WORD 611
041556 047102 .WORD T176SSR
041560 012046 .WORD PKTSSR
2485 041562 005237 002222 150$: INC FATFLG ;SET FATAL ERROR FLAG
2486 041566 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
041566 104406 TRAP C$CLP1
2487 : Set WORDS 0-7 of expd message buffer = to recv since not testing
2488 041570 004737 050146 JSR PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2489 041574 012701 046402 MOV #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2490 041600 012702 050242 MOV #T17BFSTA,R2 ;GET RECV READ STATUS
2491 041604 013721 002312 MOV DATA,(R1)+ ;SET EXPD WORD #8 = COUNT DATA
2492 041610 011211 MOV (R2),(R1) ;SET EXPD WORD #9 = RECV (NOT TESTING)
2493 : If Data read from FIFO NOT= to Data sent Then Print Error
    
```

```

2494 : The data is in WORD #8 of the message buffer
2495 041612 005000 CLR R0 ;HIGH RECV ADDRESS FOR CKMSG2
2496 041614 012701 050222 MOV #T17BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
2497 041620 012702 046362 MOV #T17EXP,R2 ;EXPD ADDRESS
2498 041624 012703 000022 MOV #18.,R3 ;NUMBER OF BYTES TO COMPARE
2499 041630 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RECV?
2500 041634 FORCERROR 152$,NOTSSR ;@@D
2501 041644 103404 BCS 160$ ;BR IF YES
2502 041646 NEXT.ERRNO
2503 041646 152$: ERRHRD ERRNO,T172CMP,MSGSUB ;REPORT ERROR
                TRAP CSERHRD
                .WORD 612
                .WORD T172CMP
                .WORD MSGSUB
                TRAP CSCLP1
                .WORD 104456
                .WORD 001144
                .WORD 047227
                .WORD 013742
2504 041656 160$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                TRAP CSCLP1
                .WORD 104406
2505 :
2506 : Do a Write Subsystem READ STATUS
2507 041660 004737 047764 JSR PC,T17SRD ;SETUP PACKET FOR READ STATUS
2508 041664 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
2509 041670 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
2510 041674 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
2511 041700 FORCERROR 162$ ;@@DFORCE ERROR IF FORCER=1
2512 041714 103407 BCS 170$ ;BR IF CARRY SET (GOOD RETURN)
2513 041716 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2514 041720 NEXT.ERRNO
2515 041720 162$: ERRDF ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                TRAP CSERDF
                .WORD 613
                .WORD T173SSR
                .WORD PKTSSR
2516 041730 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
2517 041734 170$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                TRAP CSCLP1
                .WORD 104406
2518 : Set WORDS 0-7 of expd message buffer = to recv since not testing
2519 041736 004737 050146 JSR PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2520 041742 012701 046402 MOV #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2521 041746 012702 050242 MOV #T17BFSTA,R2 ;GET RECV READ STATUS
2522 041752 012221 MOV (R2)+,(R1)+ ;SET EXPD WORD #8 = RECV TEMP
2523 041754 011211 MOV (R2),(R1) ;SET EXPD WORD #9 = RECV TEMP
2524 041756 052711 000020 BIS #S2.INRDY,(R1) ;SET EXP INPUT READY= 1
2525 041762 042711 000040 BIC #S2.OUTRDY,(R1) ;SET EXP OUTPUT READY= 0
2526 041766 042711 000200 BIC #S2.DIM,(R1) ;SET EXP DATA IN MISS = 0
2527 : If Input Ready NOT=1 then Print Error
2528 : If Output Ready NOT=0 or Data in Miss NOT=0 Then Print Error
2529 041772 005000 CLR R0 ;HIGH RECV ADDRESS FOR CKMSG2
2530 041774 012701 050222 MOV #T17BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
2531 042000 012702 046362 MOV #T17EXP,R2 ;EXPD ADDRESS
2532 042004 012703 000024 MOV #20.,R3 ;NUMBER OF BYTES TO COMPARE
2533 042010 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RECV?
2534 042014 FORCERROR 172$,NOTSSR ;@@D
2535 042024 103404 BCS 180$ ;BR IF YES
2536 042026 NEXT.ERRNO
2537 042026 172$: ERRHRD ERRNO,T174CMP,MSGSTAT ;REPORT ERROR
                TRAP CSERHRD
                .WORD 614
                .WORD T174CMP
    
```

```
2538 042034 012350
042036 180$: CKLOOP
042036 104406
2539 042040 FORCEEXIT 205$
2540 042050 005237 002312 INC DATA
2541 042054 023727 002312 000377 CMP DATA,#377
2542 042062 101002 BHI 205$
2543 042064 000137 041156 JMP 100$
2544 042070 205$:
2545
2546 042070 ENDSUB
042070
042070 104403
2547
2548 042072 005737 002222 TST FATFLG
2549 042076 001402 BEQ 260$
2550 042100 004737 017202 JSR PC,CKDROP
2551 042104 260$:
2552
```

```

:LOOP ON ERROR, IF FLAG SET .WORD MSGSTAT
TRAP C$CLP1
: @ @
:GET NEXT TEST DATA
:DONE 0 TO 377?
:BR IF YES
:DO ANOTHER TEST PATTERN

:////////// END SUBTEST //////////
L10060:
TRAP C$ESUB

:ANY FATAL ERRORS ?
:BRANCH IF NOT
:TRY TO DROP THE UNIT
```

2554  
 2555  
 2556  
 2557  
 2558  
 2559  
 2560  
 2561  
 2562  
 2563  
 2564  
 2565  
 2566  
 2567  
 2568  
 2569  
 2570  
 2571  
 2572  
 2573  
 2574  
 2575  
 2576  
 2577  
 2578  
 2579  
 2580  
 2581  
 2582  
 2583  
 2584  
 2585  
 2586  
 2587  
 2588  
 2589  
 2590  
 2591  
 2592  
 2593  
 2594  
 2595  
 2596  
 2597  
 2598  
 2599  
 2600  
 2601  
 2602  
 2603  
 2604  
 2605  
 2606  
 2607  
 2608

.SBTTL TEST 6: SUBTEST 3: FIFO WRITE MULTIPLE BYTES TEST

:\*\*  
 : TEST 6: SUBTEST 3:

: SUBTEST DESCRIPTION:

This subtest verifies the ability of the FIFO to correctly pass a multiple data bytes from input to output. The following sequence is done with various data patterns and byte counts from 2 to 64.

1. Initial FIFO status is checked
2. The Write FIFO function.
3. Read Status is executed and FIFO status is checked.
4. Read FIFO is executed and the data and final status is checked.

: TEST STEPS:

: BEGIN

Write to TSSR to soft initialize  
 Do a WRITE CHARACTERISTICS to setup a message buffer  
 Do a Write Subsystem READ STATUS  
 If Input Ready NOT=1 Then Print Error  
 If Output Ready NOT=0 Then Print Error  
 If Data In Miss NOT=0 Then Print Error  
 If Last Word NOT=0 Then Print Error

REPEAT FOR DATA 0 TO 377, 377 TO 0, FLOATING 1'S,0'S AND ALL 1'S/0'S  
 REPEAT FOR BYTE COUNT 2 TO 64 DECIMAL

: BEGIN

Do a Write Subsystem WRITE NPR to set tape direction out  
 Do a Write Subsystem WRITE FIFO  
 Do a Write Subsystem READ STATUS  
 If Input Ready NOT=1 Then Print Error  
 If Output Ready NOT=1 Then Print Error  
 If Data In Miss NOT=0 Then Print Error  
 If Last Word NOT=0 Then Print Error  
 Do Write Subsystem READ FIFO  
 If Data read from FIFO NOT= to Data sent Then Print Error  
 Do a Write Subsystem READ STATUS  
 If Input Ready NOT=1 Then Print Error  
 If Output Ready NOT=0 Then Print Error  
 If Data In Miss NOT=0 Then Print Error  
 If Last Word NOT=0 Then Print Error

END

END

BGNSUB

:/: BEGIN SUBTEST :/  
 T6.3:

TRAP CSBSUB

042104  
 042104 104402  
 042104

:  
 :s:

Write to TSSR register to soft initialize the controller

JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT

042106  
 042106 004737 015774  
 042112 103405  
 042114 010001  
 042116

```

042116 104455
042120 001146
042122 003652
042124 012034
2609 : Do a WRITE CHARACTERISTICS to setup a message buffer
2610 042126 005037 002222 10$: CLR FATFLG ;CLEAR FATAL ERROR FLAG
2611 042132 012704 050200 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
2612 042136 004737 010662 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
2613 042142 FORCERROR 42$ ;@@DFORCE ERROR IF FORCER=1
2614 042156 103407 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
2615 042160 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2616 042162 NEXT.ERRNO
2617 042162 42$: ERRDF ERRNO,T17SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
042162 104455 TRAP C$ERDF
042164 001147 .WORD 615
042166 046625 .WORD T17SSR
042170 012046 .WORD PKTSSR
2618 042172 005237 002222 50$: INC FATFLG ;SET FATAL ERROR FLAG
2619 042176 042176 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
TRAP C$CLP1
2620 : Do a Write Subsystem READ STATUS
2621 042200 004737 047764 JSR PC,T17SRD ;SETUP PACKET FOR READ STATUS
2622 042204 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
2623 042210 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
2624 042214 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
2625 042220 FORCERROR 62$ ;@@DFORCE ERROR IF FORCER=1
2626 042234 103407 BCS 70$ ;BR IF CARRY SET (GOOD RETURN)
2627 042236 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2628 042240 NEXT.ERRNO
2629 042240 62$: ERRDF ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
042240 104455 TRAP C$ERDF
042242 001150 .WORD 616
042244 046726 .WORD T173SSR
042246 012046 .WORD PKTSSR
2630 042250 005237 002222 70$: INC FATFLG ;SET FATAL ERROR FLAG
2631 042254 042254 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
TRAP C$CLP1
2632 : Set WORDS 0-7 of expd message buffer = to recv since not testing
2633 042256 004737 050146 JSR PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2634 042262 012701 046402 MOV #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2635 042266 012702 050242 MOV #T17BFSTA,R2 ;GET RECV READ STATUS
2636 042272 012221 MOV (R2)+,(R1)+ ;SET EXPD WORD #8 = RECV TEMP
2637 042274 011211 MOV (R2),(R1) ;SET EXPD WORD #9 = RECV TEMP
2638 042276 052711 000020 BIS #S2.INRDY,(R1) ;SET EXP INPUT READY= 1
2639 042302 042711 000040 BIC #S2.OTRDY,(R1) ;SET EXP OUTPUT READY= 0
2640 042306 042711 000200 BIC #S2.DIM,(R1) ;SET EXP DATA IN MISS = 0
2641 042312 042711 000100 BIC #S2.ILW,(R1) ;SET EXP LAST WORD (ILW)=0
2642 : If Input Ready NOT=1 then Print Error
2643 : If Output Ready NOT=0 or Data in Miss NOT=0 Then Print Error
2644 : If Last Word NOT=0 Then Print Error
2645 042316 005000 CLR R0 ;HIGH RECV ADDRESS FOR CKMSG2
2646 042320 012701 050222 MOV #T17BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
2647 042324 012702 046362 MOV #T17EXP,R2 ;EXPD ADDRESS
2648 042330 012703 000024 MOV #20,R3 ;NUMBER OF BYTES TO COMPARE
2649 042334 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RECV?
2650 042340 FORCERROR 82$,NOTSSR ;@@
2651 042350 103404 BCS 90$ ;BR IF YES
    
```

```

2652 042352
2653 042352      82$:  NEXT.ERRNO
                ERRHRD  ERRNO,T171CMP,MSGSTAT  ;REPORT ERROR
                042352 104456                TRAP  C$ERHRD
                042354 001151                .WORD  617
                042356 047145                .WORD  T171CMP
                042360 012350                .WORD  MSGSTAT
2654 042362      90$:  CKLOOP                ;LOOP ON ERROR, IF FLAG SET
                042362 104406                TRAP  C$CLP1
2655
2656
2657
2658 : REPEAT FOR BYTE COUNT 2 TO 64 DECIMAL
2659 : TSTFLAG =1 FOR INCREMENT TEST PATTERN
2660 :         =2 FOR DECREMENT TEST PATTERN
2661 :         =3 FOR TSTBLK TABLE PATTERN
2662 042364 012737 000001 002314      95$:  MOV #1,TSTFLAG                ;TEST PATTERN FLAG
2663 042372      95$:  MOV #2,COUNT                ;GET FIRST BYTE COUNT
2664 042372 012737 000002 002310      100$:
2665 042400 :
2666 : Do a Write Subsystem WRITE NPR to set tape direction out
2667 042400 012700 000100      MOV #NP.OUT,R0                ;SET TAPE DIRECTION OUT
2668 042404 004737 050026      JSR PC,T17SNPR                ;SETUP T17PK2 FOR WRITE NPR
2669 042410 012704 050350      MOV #T17PK2,R4                ;GET WRITE SUBSYSTEM COMMAND PACKET
2670 042414 010465 000000      MOV R4,TSD(BR5)                ;SET THE PACKET ADDRESS TO EXECUTE
2671 042420 004737 016336      JSR PC,CHKTSSR                ;WAIT FOR SSR TO SET
2672 042424      FORCERROR 102$                ;@@DFORCE ERROR IF FORCER=1
2673 042440 103407      BCS 105$                ;BR IF CARRY SET (GOOD RETURN)
2674 042442 010001      MOV R0,R1                ;SAVE CONTENTS OF TSSR
2675 042444
2676 042444      102$:  ERRDF  ERRNO,T174SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
                042444 104455                TRAP  C$ERDF
                042446 001152                .WORD  618
                042450 046773                .WORD  T174SSR
                042452 012046                .WORD  PKTSSR
2677 042454 005237 002222
2678 042460      105$:  INC  FATFLG                ;SET FATAL ERROR FLAG
                042460 104406                ;LOOP ON ERROR, IF FLAG SET
                TRAP  C$CLP1
2679 : Do a Write Subsystem WRITE FIFO
2680 042462 004737 050126      JSR PC,T17CLEXP                ;CLEAR EXPD BUFFER
2681 042466 012701 046504      MOV #T17WFDATA,R1                ;EXPD WRITE FIFO DATA BUFFER
2682 042472 013702 002310      MOV COUNT,R2                ;TEST PATTERN SIZE
2683 042476 022737 000001 002314      CMP #1,TSTFLAG                ;INCREMENT PATTERN THIS TIME THRU?
2684 042504 001005      BNE 115$                ;BR IF NO
2685 042506 005000      CLR R0                ;INCREMENT TEST PATTERN
2686 042510 110021      110$:  MOVB R0,(R1)+                ;STORE INCREMENT TEST BYTE
2687 042512 005200      INC R0                ;SET NEXT PATTERN
2688 042514 005302      DEC R2                ;DONE?
2689 042516 003374      BGT 110$                ;BR IF NO
2690 042520 022737 000002 002314      115$:  CMP #2,TSTFLAG                ;DECREMENT PATTERN THIS TIME THRU?
2691 042526 001006      BNE 125$                ;BR IF NO
2692 042530 012700 000377      MOV #377,R0                ;DECREMENT TEST PATTERN
2693 042534 110021      120$:  MOVB R0,(R1)+                ;STORE DECREMENT TEST BYTE
2694 042536 005300      DEC R0                ;SET NEXT PATTERN
2695 042540 005302      DEC R2                ;DONE?
2696 042542 003374      BGT 120$                ;BR IF NO
2697 042544 022737 000003 002314      125$:  CMP #3,TSTFLAG                ;TSTBLK PATTERNS THIS TIME THRU?
2698 042552 001005      BNE 135$                ;BR IF NO
    
```

```

2699 042554 012700 002752      MOV      #TSTBLK,R0      ;FLOAT 1'S/0'S ETC. TEST TABLE
2700 042560 112021      130$:  MOVB     (R0)+,(R1)+  ;STORE A TSTBLK BYTE
2701 042562 005302      DEC      R2              ;DONE?
2702 042564 003375      BGT      130$           ;BR IF NO
2703 042566      135$:
2704 042566 013700 002310      MOV      COUNT,R0      ;FIFO BYTE COUNT
2705 042572 012701 046504      MOV      #T17WFDATA,R1 ;FIFO WRITE DATA ADDRESS
2706 042576 004737 050052      JSR      PC,T17WFIF    ;SETUP T17PK2 FOR WRITE FIFO
2707 042602 012704 050350      MOV      #T17PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
2708 042606 010465 000000      MOV      R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
2709 042612 004737 016336      JSR      PC,CHKTSSR    ;WAIT FOR SSR TO SET
2710 042616      FORCERROR 142$      ;@@DFORCE ERROR IF FORCER=1
2711 042632 103407      BCS     150$           ;BR IF CARRY SET (GOOD RETURN)
2712 042634 010001      MOV      R0,R1         ;SAVE CONTENTS OF TSSR
2713 042636      NEXT.ERRNO
2714 042636 142$:  ERRDF  ERRNO,T175SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 619
                                .WORD T175SSR
                                .WORD  PKTSSR
2715 042646 005237 002222      INC      FATFLG        ;SET FATAL ERROR FLAG
2716 042652 104406      150$:  CKLOOP           ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
2717
2718      : Do a Write Subsystem READ STATUS
2719 042654 004737 047764      JSR      PC,T17SRD    ;SETUP PACKET FOR READ STATUS
2720 042660 012704 050350      MOV      #T17PK2,R4  ;GET WRITE SUBSYSTEM COMMAND PACKET
2721 042664 010465 000000      MOV      R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
2722 042670 004737 016336      JSR      PC,CHKTSSR  ;WAIT FOR SSR TO SET
2723 042674      FORCERROR 157$      ;@@DFORCE ERROR IF FORCER=1
2724 042710 103407      BCS     160$           ;BR IF CARRY SET (GOOD RETURN)
2725 042712 010001      MOV      R0,R1         ;SAVE CONTENTS OF TSSR
2726 042714      NEXT.ERRNO
2727 042714 157$:  ERRDF  ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 620
                                .WORD T173SSR
                                .WORD  PKTSSR
2728 042724 005237 002222      INC      FATFLG        ;SET FATAL ERROR FLAG
2729 042730 104406      160$:  CKLOOP           ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
2730
2731      : Set WORDS 0-7 of expd message buffer = to recv since not testing
2732 042732 004737 050146      JSR      PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2733 042736 012701 046402      MOV      #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2734 042742 012702 050242      MOV      #T17BFSTA,R2 ;GET RECV READ STATUS
2735 042746 012221      MOV      (R2)+,(R1)+  ;SET EXPD WORD #8 = RECV TEMP
2736 042750 011211      MOV      (R2),(R1)    ;SET EXPD WORD #9 = RECV TEMP
2737 042752 052711 000020      BIS      #S2.INRDY,(R1);SET EXP INPUT READY= 1
2738 042756 052711 000040      BIS      #S2.OUTRDY,(R1);SET EXP OUTPUT READY= 1
2739 042762 042711 000200      BIC      #S2.DIM,(R1) ;SET EXP DATA IN MISS = 0
2740 042766 042711 000100      BIC      #S2.ILW,(R1) ;SET EXP LAST WORD (ILW)=0
2741
2742      : If Input Ready NOT=1 then Print Error
2743 042772 005000      CLR      R0            ;HIGH RECV ADDRESS FOR CKMSG2
2744 042774 012701 050222      MOV      #T17BFR,R1   ;LOW RECV ADDRESS FOR CKMSG2
2745 043000 012702 046362      MOV      #T17EXP,R2   ;EXPD ADDRESS
    
```

```

2746 043004 012703 000024      MOV      #20,R3          ;NUMBER OF BYTES TO COMPARE
2747 043010 004737 011500      JSR      PC,CKMSG2      ;EXPD EQUAL RECV?
2748 043014                    FORCERROR 162$,NOTSSR    ;@@
2749 043024 103404                    BCS      170$          ;BR IF YES
2750 043026                    NEXT.ERRNO
2751 043026 162$:  ERRHRD  ERRNO,T173CMP,MSGSTAT ;:REPORT ERROR
                                TRAP      C$ERHRD
                                .WORD    621
                                .WORD    T173CMP
                                .WORD    MSGSTAT
                                TRAP      C$CLP1
                                043026 104456
                                043030 001155
                                043032 047323
                                043034 012350
2752 043036 170$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
                                043036 104406
2753
2754 :      Do Write Subsystem READ FIFO
2755 043040 013700 002310      MOV      COUNT,R0      ;SET READ BYTE COUNT
2756 043044 004737 050106      JSR      PC,T17RFIF    ;SETUP T17PK2 FOR READ FIFO
2757 043050 012704 050350      MOV      #T17PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
2758 043054 010465 000000      MOV      R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
2759 043060 004737 016336      JSR      PC,CHKTSSR    ;WAIT FOR SSR TO SET
2760 043064                    FORCERROR 172$          ;@@DFORCE ERROR IF FORCER=1
2761 043100 103407                    BCS      180$          ;BR IF CARRY SET (GOOD RETURN)
2762 043102 010001                    MOV      R0,R1         ;SAVE CONTENTS OF TSSR
2763 043104                    NEXT.ERRNO
2764 043104 172$:  ERRDF  ERRNO,T176SSR,PKTSSR ;:DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD    622
                                .WORD    T176SSR
                                .WORD    PKTSSR
                                043104 104455
                                043106 001156
                                043110 047102
                                043112 012046
2765 043114 005237 002222      INC      FATFLG        ;SET FATAL ERROR FLAG
2766 043120 180$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
                                043120 104406
2767
2768 :      If Data read from FIFO NOT= to Data sent Then Print Error
2769 043122 005000                    CLR      R0            ;HIGH RECV ADDRESS FOR CKMSG2
2770 043124 012702 046504      MOV      #T17WFDATA,R2 ;GET EXPECTED ADDRESS FOR CKMSG2
2771 043130 012701 050242      MOV      #T17BFSTA,R1  ;GET RECEIVED ADDRESS FOR CKMSG2
2772 043134 013703 002310      MOV      COUNT,R3     ;NUMBER OF BYTES TO COMPARE
2773 043140 004737 011500      JSR      PC,CKMSG2     ;EXPD EQUAL RECV?
2774 043144                    FORCERROR 192$,NOTSSR    ;@@
2775 043154 103406                    BCS      200$          ;BR IF YES
2776 043156                    NEXT.ERRNO
2777 043156 013701 002310      MOV      COUNT,R1     ;GET BYTE COUNT
2778 043162 192$:  ERRHRD  ERRNO,T175CMP,FIFEXP ;:REPORT ERROR
                                TRAP      C$ERHRD
                                .WORD    623
                                .WORD    T175CMP
                                .WORD    FIFEXP
                                043162 104456
                                043164 001157
                                043166 047472
                                043170 012170
2779 043172 200$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
                                043172 104406
2780
2781 :      Do a Write Subsystem READ STATUS
2782 043174 004737 047764      JSR      PC,T17SRD     ;SETUP PACKET FOR READ STATUS
2783 043200 012704 050350      MOV      #T17PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
2784 043204 010465 000000      MOV      R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
2785 043210 004737 016336      JSR      PC,CHKTSSR    ;WAIT FOR SSR TO SET
2786 043214                    FORCERROR 212$          ;@@DFORCE ERROR IF FORCER=1
2787 043230 103407                    BCS      220$          ;BR IF CARRY SET (GOOD RETURN)
    
```



```

2788 043232 010001      MOV      RO,R1      ;SAVE CONTENTS OF TSSR
2789 043234             NEXT.ERRNO
2790 043234             ERRDF  ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      043234 104455             TRAP  C$ERDF
      043236 001160             .WORD 624
      043240 046726             .WORD T173SSR
      043242 012046             .WORD PKTSSR
2791 043244 005237 002222  INC      FATFLG      ;SET FATAL ERROR FLAG
2792 043250             CKLOOP              ;LOOP ON ERROR, IF FLAG SET
      043250 104406             TRAP  C$CLP1
2793 :                   Set WORDS 0-7 of expd message buffer = to recv since not testing
2794 043252 004737 050146  JSR      PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
2795 043256 012701 046402  MOV      #T17EXSTA,R1 ;GET EXPECTED READ STATUS
2796 043262 012702 050242  MOV      #T17BFSTA,R2 ;GET RECV READ STATUS
2797 043266 012221             MOV      (R2)+,(R1)+ ;SET EXPD WORD #8 = RECV TEMP
2798 043270 011211             MOV      (R2),(R1)    ;SET EXPD WORD #9 = RECV TEMP
2799 043272 052711 000020  BIS      #S2.INRDY,(R1) ;SET EXP INPUT READY= 1
2800 043276 042711 000040  BIC      #S2.OTRDY,(R1) ;SET EXP OUTPUT READY= 0
2801 043302 042711 000200  BIC      #S2.DIM,(R1)  ;SET EXP DATA IN MISS = 0
2802 043306 042711 000100  BIC      #S2.ILW,(R1)  ;SET EXP LAST WORD (ILW)=0
2803 :                   If Input Ready NOT=1 then Print Error
2804 :                   If Output Ready NOT=0 or Data in Miss NOT=0 Then Print Error
2805 043312 005000             CLR      RO          ;HIGH RECV ADDRESS FOR CKMSG2
2806 043314 012701 050222  MOV      #T17BFR,R1   ;LOW RECV ADDRESS FOR CKMSG2
2807 043320 012702 046362  MOV      #T17EXP,R2   ;EXPD ADDRESS
2808 043324 012703 000024  MOV      #20.,R3      ;NUMBER OF BYTES TO COMPARE
2809 043330 004737 011500  JSR      PC,CKMSG2    ;EXPD EQUAL RECV?
2810 043334             FORCERROR 232$,NOTSSR ;@@D
2811 043344 103404             BCS     240$         ;BR IF YES
2812 043346             NEXT.ERRNO
2813 043346             ERRHRD  ERRNO,T174CMP,MSGSTAT ;REPORT ERROR
      043346 104456             TRAP  C$ERHRD
      043350 001161             .WORD 625
      043352 047407             .WORD T174CMP
      043354 012350             .WORD MSGSTAT
2814 043356             CKLOOP              ;LOOP ON ERROR, IF FLAG SET
      043356 104406             TRAP  C$CLP1
2815 043360             FORCEEXIT 250$         ;@@D
2816 043370 005237 002310  INC      COUNT        ;GET NEXT BYTE COUNT
2817 043374 023727 002310 000077  CMP      COUNT,#77    ;DONE 0 TO 77
2818 043402 101002             BHI     250$         ;BR IF YES
2819 043404 000137 042400  JMP      100$         ;DO ANOTHER BYTE COUNT
2820 043410 005237 002314 250$:  INC      TSTFLAG      ;GET NEXT TEST PATTERN CODE
2821 043414 023727 002314 000003  CMP      TSTFLAG,#3   ;DONE INC,DEC,TSTBLK PATTERNS?
2822 043422 101002             BHI     255$         ;BR IF YES
2823 043424 000137 042372  JMP      95$          ;DO ANOTHER TEST PATTERN
2824 043430             255$:
2825 043430             ENDSUB              ;////////// END SUBTEST ////////////
      043430             L10061:
      043430 104403             TRAP  C$ESUB
2826
2827 043432 005737 002222  TST     FATFLG        ;ANY FATAL ERRORS ?
2828 043436 001402             BEQ    260$         ;BRANCH IF NOT
2829 043440 004737 017202  JSR     PC,CKDROP     ;TRY TO DROP THE UNIT
2830 043444             260$:
2831
2832
    
```

2833

```

2835          .SBTTL TEST 6: SUBTEST 4: FIFO Verify ILW Status
2836
2837          :++
2838          : TEST 6: SUBTEST 4:
2839          :
2840          : SUBTEST DESCRIPTION:
2841          :
2842          :     This subtest verifies that reading the FIFO when it is
2843          :     empty causes the Last Word (ILW) status to assert.
2844          :
2845          : TEST STEPS:
2846          :
2847          : BEGIN
2848          :     Write to TSSR to soft initialize
2849          :     Do Write Subsystem READ FIFO with byte count equal to 1
2850          :     Do a Write Subsystem READ STATUS
2851          :     If Input Ready NOT=1 Then Print Error
2852          :     If Output Ready NOT=0 Then Print Error
2853          :     If Data In Miss NOT=0 Then Print Error
2854          :     If Last Word (ILW) NOT=1 Then Print Error
2855          : END
2856          :--
2857          043444          BGNSUB          ;////////// BEGIN SUBTEST //////////
                043444          T6.4:          TRAP          C$BSUB
                043444          104402
2858
2859          :
2860          043446          5$:          Write to TSSR register to soft initialize the controller
2861          043446          004737          015774          JSR          PC,SOFINIT          ;WRITE TO TSSR TO SOFT INITIALIZE
2862          043452          103405          BCS          10$          ;BR IF SOFT INIT OKAY
2863          043454          010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
2864          043456          104455          ERRDF          ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL DURING INIT
                043460          001161          TRAP          C$ERDF
                043462          003652          .WORD          625
                043464          012034          .WORD          SFIERR
                .WORD          SFIMSG
2865          :
2866          043466          005037          002222          10$:          Do a WRITE CHARACTERISTICS to setup a message buffer
2867          043472          012704          050200          CLR          FATFLG          ;CLEAR FATAL ERROR FLAG
2868          043476          004737          010662          MOV          #T17PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
2869          043502          FORCERROR          42$          JSR          PC,WRTCHR          ;DO WRITE CHARACTERISTICS COMMAND
2870          043516          103407          BCS          50$          ;@ADFORCE ERROR IF FORCER=1
2871          043520          010001          MOV          R0,R1          ;BR IF CARRY SET (GOOD RETURN)
2872          043522          NEXT.ERRNO          ;SAVE CONTENTS OF TSSR
2873          043522          104455          42$:          ERRDF          ERRNO,T17SSR,PKTSSR          ;DEVICE FATAL SSR FAILED TO SET
                043524          001162          TRAP          C$ERDF
                043526          046625          .WORD          626
                043530          012046          .WORD          T17SSR
                .WORD          PKTSSR
2874          043532          005237          002222          INC          FATFLG          ;SET FATAL ERROR FLAG
2875          043536          104406          50$:          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                TRAP          C$CLP1
2876          :
2877          :          Do Write Subsystem READ FIFO with byte count equal to 1
2878          043540          012700          000001          MOV          #1,R0          ;SET READ BYTE COUNT
2879          043544          004737          050106          JSR          PC,T17RFIF          ;SETUP T17PK2 FOR READ FIFO
2880          043550          012704          050350          MOV          #T17PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
    
```

```

2881 043554 010465 000000      MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
2882 043560 004737 016336      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2883 043564          FORCERROR 142$      ;@@DFORCE ERROR IF FORCER=1
2884 043600 103407      BCS      150$      ;BR IF CARRY SET (GOOD RETURN)
2885 043602 010001      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
2886 043604          NEXT.ERRNO
2887 043604          142$:      ERRDF  ERRNO,T176SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD      627
                                .WORD      T176SSR
                                .WORD      PKTSSR
2888 043614 005237 002222      INC      FATFLG      ;SET FATAL ERROR FLAG
2889 043620          150$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
2890
2891          :      Do a Write Subsystem READ STATUS
2892 043622 004737 047764      JSR      PC,T17SRD      ;SETUP PACKET FOR READ STATUS
2893 043626 012704 050350      MOV      #T17PK2,R4      ;GET WRITE SUBSYSTEM COMMAND PACKET
2894 043632 010465 000000      MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
2895 043636 004737 016336      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2896 043642          FORCERROR 162$      ;@@DFORCE ERROR IF FORCER=1
2897 043656 103407      BCS      170$      ;BR IF CARRY SET (GOOD RETURN)
2898 043660 010001      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
2899 043662          162$:      NEXT.ERRNO
2900 043662          162$:      ERRDF  ERRNO,T173SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD      628
                                .WORD      T173SSR
                                .WORD      PKTSSR
2901 043672 005237 002222      INC      FATFLG      ;SET FATAL ERROR FLAG
2902 043676          170$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
2903          :      Set WORDS 0-7 of expd message buffer = to recv since not testing
2904 043700 004737 050146      JSR      PC,T17SETEXP      ;SET WORDS 0-7 EXPD=RECV
2905 043704 012701 046402      MOV      #T17EXSTA,R1      ;GET EXPECTED READ STATUS
2906 043710 012702 050242      MOV      #T17BFSTA,R2      ;GET RECV READ STATUS
2907 043714 012221      MOV      (R2)+,(R1)+      ;SET EXPD WORD #8 = RECV TEMP
2908 043716 011211      MOV      (R2),(R1)      ;SET EXPD WORD #9 = RECV TEMP
2909 043720 052711 000020      BIS      #S2.INRDY,(R1)      ;SET EXP INPUT READY= 1
2910 043724 042711 000040      BIC      #S2.OURDY,(R1)      ;SET EXP OUTPUT READY= 0
2911 043730 042711 000200      BIC      #S2.DIM,(R1)      ;SET EXP DATA IN MISS = 0
2912 043734 052711 000100      BIS      #S2.ILW,(R1)      ;SET EXP LAST WORD (ILW)=1
2913          :      If Input Ready NOT=1 then Print Error
2914          :      If Output Ready NOT=0 or Data in Miss NOT=0 Then Print Error
2915          :      If Last Word (ILW) NOT=1 Then Print Error
2916 043740 005000          CLR      R0      ;HIGH RECV ADDRESS FOR CKMSG2
2917 043742 012701 050222      MOV      #T17BFR,R1      ;LOW RECV ADDRESS FOR CKMSG2
2918 043746 012702 046362      MOV      #T17EXP,R2      ;EXPD ADDRESS
2919 043752 012703 000024      MOV      #20.,R3      ;NUMBER OF BYTES TO COMPARE
2920 043756 004737 011500      JSR      PC,CKMSG2      ;EXPD EQUAL RECV?
2921 043762          FORCERROR 172$,NOTSSR      ;@@D
2922 043772 103404      BCS      180$      ;BR IF YES
2923 043774          172$:      NEXT.ERRNO
2924 043774          172$:      ERRHRD  ERRNO,T176CMP,MSGSTAT      ;REPORT ERROR
                                TRAP      C$ERHRD
                                .WORD      629
                                .WORD      T176CMP
043774 104456
043776 001165
044000 047546

```

2925	044002	012350							
	044004		180\$:	CKLOOP				.WORD	MSGSTAT
	044004	104406						SET	
2926								TRAP	C\$CLP1
2927	044006			ENDSUB					
	044006								
	044006	104403							
2928									
2929	044010	C05737		TST	FATFLG				
2930	044014	001402		BEQ	260\$				
2931	044016	004737		JSR	PC,CKDROP				
2932	044022		260\$:						
2933									
2934									

-y

```
;LOOP ON ERROR, IF FLAG SET
;TRAP C$CLP1
;////////// END SUBTEST //////////
;L10062:
;TRAP C$ESUB
;ANY FATAL ERRORS ?
;BRANCH IF NOT
;TRY TO DROP THE UNIT
```

2936  
 2937  
 2938  
 2939  
 2940  
 2941  
 2942  
 2943  
 2944  
 2945  
 2946  
 2947  
 2948  
 2949  
 2950  
 2951  
 2952  
 2953  
 2954  
 2955  
 2956  
 2957  
 2958  
 2959  
 2960  
 2961  
 2962  
 2963  
 2964  
 2965  
 2966  
 2967  
 2968  
 2969  
 2970  
 2971  
 2972  
 2973  
 2974  
 2975  
 2976  
 2977  
 2978  
 2979  
 2980  
 2981  
 2982  
 2983  
 2984  
 2985  
 2986

.SBTTL TEST 6: SUBTEST 5: FIFO Verify Input Ready

TEST 6: SUBTEST 5:

SUBTEST DESCRIPTION:

This subtest verifies that writing 64. bytes into the FIFO without reading any out causes the Input Ready status to negate. The Subtest then verifies that writing a 65th byte into the FIFO causes the Data In Miss status to assert. Next it is verified that the original 64 bytes can be read out correctly and that the data has not been corrupted.

TEST STEPS:

BEGIN

Write to TSSR to soft initialize  
 Do a WRITE CHARACTERISTICS to setup a message buffer  
 Do a Write Subsystem WRITE NPR to set tape direction out  
 Do a Write Subsystem WRITE FIFO 64. bytes incrementing pattern  
 Do a Write Subsystem READ STATUS  
 If Input Ready NOT=0 Then Print Error  
 If Output Ready NOT=1 Then Print Error  
 If Data In Miss NOT=0 Then Print Error  
 Do a Write Subsystem WRITE FIFO 1 byte for a total of 65. written  
 Do a Write Subsystem READ STATUS  
 If Input Ready NOT=0 Then Print Error  
 If Output Ready NOT=1 Then Print Error  
 If Data In Miss NOT=1 Then Print Error  
 Do Write Subsystem READ FIFO  
 If Data read from FIFO NOT= to Data sent Then Print Error  
 Do a Write Subsystem READ STATUS  
 If Input Ready NOT=1 Then Print Error  
 If Output Ready NOT=0 Then Print Error  
 If Data In Miss NOT=1 Then Print Error

END

BGNSUB

////////// BEGIN SUBTEST //////////  
 T6.5:

TRAP CSBSUB

Write to TSSR register to soft initialize the controller

5s:

JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT  
 TRAP CSERDF  
 .WORD 629  
 .WORD SFIERR  
 .WORD SFIMSG

10s:

Do a WRITE CHARACTERISTICS to setup a message buffer  
 CLR FATFLG ;CLEAR FATAL ERROR FLAG  
 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET  
 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND  
 FORCERROR 42\$ ;@DFORCE ERROR IF FORCER=1

044022  
 044022 104402  
 044024 004737 015774  
 044030 103405  
 044032 010001  
 044034 104455  
 044036 001165  
 044040 003652  
 044042 012034  
 044044 005037 002222  
 044050 012704 050200  
 044054 004737 010662  
 044060

```

2987 044074 103407      BCS      50$      :BR IF CARRY SET (GOOD RETURN)
2988 044076 010001      MOV      R0,R1   :SAVE CONTENTS OF TSSR
2989 044100              NEXT.ERRNO
2990 044100 42$:      ERRDF  ERRNO,T17SSR,PKTSSR :DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD     630
                                .WORD     T17SSR
                                .WORD     PKTSSR
2991 044110 005237 002222      INC      FATFLG  :SET FATAL ERROR FLAG
2992 044114 50$:      CKLOOP                :LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
2993
2994
2995 044116 012700 000100      100$:      Do a Write Subsystem WRITE NPR to set tape direction out
2996 044122 004737 050026      MOV      #NPR.OUT,R0 :SET TAPE DIRECTION OUT
2997 044126 012704 050350      JSR      PC,T17SNPR  :SETUP T17PK2 FOR WRITE NPR
2998 044132 010465 000000      MOV      #T17PK2,R4  :GET WRITE SUBSYSTEM COMMAND PACKET
2999 044136 004737 016336      MOV      R4,TSDB(R5) :SET THE PACKET ADDRESS TO EXECUTE
3000 044142              JSR      PC,CHKTSSR  :WAIT FOR SSR TO SET
3001 044156 103407              FORCERROR 102$      :@DFORCE ERROR IF FORCER=1
3002 044160 010001      BCS      105$      :BR IF CARRY SET (GOOD RETURN)
3003 044162              MOV      R0,R1   :SAVE CONTENTS OF TSSR
3004 044162 102$:      ERRDF  ERRNO,T174SSR,PKTSSR :DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD     631
                                .WORD     T174SSR
                                .WORD     PKTSSR
3005 044172 005237 002222      INC      FATFLG  :SET FATAL ERROR FLAG
3006 044176 105$:      CKLOOP                :LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
3007
3008
3009 044200 012737 000100 002310      :      Do a Write Subsystem WRITE FIFO 64. bytes incrementing pattern
3010 044206 012701 046504      MOV      #64.,COUNT :WRITE 64 BYTES
3011 044212 012702 000100      MOV      #T17WFDATA,R1 :EXPD WRITE FIFO DATA BUFFER
3012 044216 005000      MOV      #64.,R2     :TEST PATTERN SIZE
3013 044220 110021 110$:      CLR      R0          :INCREMENT TEST PATTERN
3014 044222 005200      MOVVB   R0,(R1)+     :STORE INCREMENT TEST BYTE
3015 044224 005302      INC      R0          :SET NEXT PATTERN
3016 044226 003374      DEC      R2          :DONE?
3017 044230 013700 002310      BGT      110$        :BR IF NO
3018 044234 012701 046504      MOV      COUNT,R0   :FIFO BYTE COUNT
3019 044240 004737 050052      MOV      #T17WFDATA,R1 :FIFO WRITE DATA ADDRESS
3020 044244 012704 050350      JSR      PC,T17WFIF  :SETUP T17PK2 FOR WRITE FIFO
3021 044250 010465 000000      MOV      #T17PK2,R4  :GET WRITE SUBSYSTEM COMMAND PACKET
3022 044254 004737 016336      MOV      R4,TSDB(R5) :SET THE PACKET ADDRESS TO EXECUTE
3023 044260              JSR      PC,CHKTSSR  :WAIT FOR SSR TO SET
3024 044274 103407              FORCERROR 142$      :@DFORCE ERROR IF FORCER=1
3025 044276 010001      BCS      150$      :BR IF CARRY SET (GOOD RETURN)
3026 044300              MOV      R0,R1   :SAVE CONTENTS OF TSSR
3027 044300 142$:      ERRDF  ERRNO,T175SSR,PKTSSR :DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD     632
                                .WORD     T175SSR
                                .WORD     PKTSSR
3028 044310 005237 002222      INC      FATFLG  :SET FATAL ERROR FLAG
3029 044314 150$:      CKLOOP                :LOOP ON ERROR, IF FLAG SET
    
```

```

044314 104406 TRAP C$CLP1
3030
3031 : Do a Write Subsystem READ STATUS
3032 : If Input Ready NOT=0 Then Print Error
3033 : If Output Ready NOT=1 Then Print Error
3034 : If Data In Miss NOT=0 Then Print Error
3035 044316 004737 047764 JSR PC,T17SRD ;SETUP PACKET FOR READ STATUS
3036 044322 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3037 044326 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3038 044332 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3039 044336 FORCERROR 157$ ;@@DFORCE ERROR IF FORCER=1
3040 044352 103407 BCS 160$ ;BR IF CARRY SET (GOOD RETURN)
3041 044354 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3042 044356 NEXT.ERRNO
3043 044356 157$: ERRDF ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP C$ERDF
                                .WORD 633
                                .WORD T173SSR
                                .WORD PKTSSR
                                TRAP C$CLP1
044356 104455
044360 001171
044362 046726
044364 012046
3044 044366 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
3045 044372 160$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP C$CLP1
044372 104406
3046 : Set WORDS 0-7 of expd message buffer = to rcv since not testing
3047 044374 004737 050146 JSR PC,T17SETEXP ;SET WORDS 0-7 EXPD=RCV
3048 044400 012701 046402 MOV #T17EXSTA,R1 ;GET EXPECTED READ STATUS
3049 044404 012702 050242 MOV #T17BFSTA,R2 ;GET RCV READ STATUS
3050 044410 012221 MOV (R2)+,(R1)+ ;SET EXPD WORD #8 = RCV TEMP
3051 044412 011211 MOV (R2),(R1) ;SET EXPD WORD #9 = RCV TEMP
3052 044414 042711 000020 BIC #S2.INRDY,(R1) ;SET EXP INPUT READY= 0
3053 044420 052711 000040 BIS #S2.OURDY,(R1) ;SET EXP OUTPUT READY= 1
3054 044424 042711 000200 BIC #S2.DIM,(R1) ;SET EXP DATA IN MISS = 0
3055 044430 005000 CLR R0 ;HIGH RCV ADDRESS FOR CKMSG2
3056 044432 012701 050222 MOV #T17BFR,R1 ;LOW RCV ADDRESS FOR CKMSG2
3057 044436 012702 046362 MOV #T17EXP,R2 ;EXPD ADDRESS
3058 044442 012703 000024 MOV #20.,R3 ;NUMBER OF BYTES TO COMPARE
3059 044446 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RCV?
3060 044452 FORCERROR 162$,NOTSSR ;@@
3061 044462 103404 BCS 170$ ;BR IF YES
3062 044464 NEXT.ERRNO
3063 044464 162$: ERRHRD ERRNO,T173CMP,MSGSTAT ;REPORT ERROR
                                TRAP C$ERHRD
                                .WORD 634
                                .WORD T173CMP
                                .WORD MSGSTAT
                                TRAP C$CLP1
044464 104456
044466 001172
044470 047323
044472 012350
3064 044474 170$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP C$CLP1
044474 104406
3065
3066
3067 : Do a Write Subsystem WRITE FIFO 1 byte for a total of 65. written
3068 044476 012700 000001 MOV #1,R0 ;FIFO BYTE COUNT
3069 044502 012701 046504 MOV #T17WFDATA,R1 ;FIFO WRITE DATA ADDRESS
3070 044506 004737 050052 JSR PC,T17WFIF ;SETUP T17PK2 FOR WRITE FIFO
3071 044512 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3072 044516 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3073 044522 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3074 044526 FORCERROR 172$ ;@@DFORCE ERROR IF FORCER=1
3075 044542 103407 BCS 180$ ;BR IF CARRY SET (GOOD RETURN)
    
```



```

3076 044544 010001      MOV     R0,R1      ;SAVE CONTENTS OF TSSR
3077 044546             NEXT.ERRNO
3078 044546 172$:      ERRDF  ERRNO,T175SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 635
                                .WORD T175SSR
                                .WORD  PKTSSR
3079 044556 005237 002222  INC     FATFLG      ;SET FATAL ERROR FLAG
3080 044562 180$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
3081
3082      :      Do a Write Subsystem READ STATUS
3083      :      If Input Ready NOT=0 Then Print Error
3084      :      If Output Ready NOT=1 Then Print Error
3085      :      If Data In Miss NOT=1 Then Print Error
3086 044564 004737 047764  JSR     PC,T17SRD   ;SETUP PACKET FOR READ STATUS
3087 044570 012704 050350  MOV     #T17PK2,R4  ;GET WRITE SUBSYSTEM COMMAND PACKET
3088 044574 010465 000000  MOV     R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3089 044600 004737 016336  JSR     PC,CHKTSSR  ;WAIT FOR SSR TO SET
3090 044604             FORCERROR 187$      ;@@DFORCE ERROR , FORCER=1
3091 044620 103407             BCS     190$      ;BR IF CARRY SET (GOOD RETURN)
3092 044622 010001             MOV     R0,R1      ;SAVE CONTENTS OF TSSR
3093 044624             NEXT.ERRNO
3094 044624 187$:      ERRDF  ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 636
                                .WORD T173SSR
                                .WORD  PKTSSR
3095 044634 005237 002222  INC     FATFLG      ;SET FATAL ERROR FLAG
3096 044640 190$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
3097      :      Set WORDS 0-7 of expd message buffer = to recv since not testing
3098 044642 004737 050146  JSR     PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
3099 044646 012701 046402  MOV     #T17EXSTA,R1 ;GET EXPECTED READ STATUS
3100 044652 012702 050242  MOV     #T17BFSTA,R2 ;GET RECV READ STATUS
3101 044656 012221             MOV     (R2)+,(R1)+ ;SET EXPD WORD #8 = RECV TEMP
3102 044660 011211             MOV     (R2),(R1)    ;SET EXPD WORD #9 = RECV TEMP
3103 044662 042711 000020  BIC     #S2.INRDY,(R1) ;SET EXP INPUT READY= 0
3104 044666 052711 000040  BIS     #S2.OTRDY,(R1) ;SET EXP OUTPUT READY= 1
3105 044672 052711 000200  BIS     #S2.DIM,(R1)  ;SET EXP DATA IN MISS = 1
3106 044676 005000             CLR     R0          ;HIGH RECV ADDRESS FOR CKMSG2
3107 044700 012701 050222  MOV     #T17BFR,R1   ;LOW RECV ADDRESS FOR CKMSG2
3108 044704 012702 046362  MOV     #T17EXP,R2   ;EXPD ADDRESS
3109 044710 012703 000024  MOV     #20.,R3      ;NUMBER OF BYTES TO COMPARE
3110 044714 004737 011500  JSR     PC,CKMSG2    ;EXPD EQUAL RECV?
3111 044720             FORCERROR 192$,NOTSSR ;@@
3112 044730 103404             BCS     200$      ;BR IF YES
3113 044732             NEXT.ERRNO
3114 044732 192$:      ERRHRD  ERRNO,T173CMP,MSGSTAT ;REPORT ERROR
                                TRAP  C$ERHRD
                                .WORD 637
                                .WORD T173CMP
                                .WORD  MSGSTAT
3115 044742 200$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
3116      :      Do Write Subsystem READ FIFO
3117 044744 013700 002310  MOV     COUNT,R0     ;SET READ BYTE COUNT
    
```

```

3118 044750 004737 050106 JSR PC,T17RFIF ;SETUP T17PK2 FOR READ FIFO
3119 044754 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3120 044760 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3121 044764 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3122 044770 FORCERROR 212$ ;@DFORCE ERROR IF FORCER=1
3123 045004 103407 BCS 220$ ;BR IF CARRY SET (GOOD RETURN)
3124 045006 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3125 045010 NEXT.ERRNO
3126 045010 212$: ERRDF ERRNO,T176SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP C$ERDF
                                .WORD 638
                                .WORD T176SSR
                                .WORD PKTSSR
3127 045020 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
3128 045024 220$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP C$CLP1
3129
3130 : If Data read from FIFO NOT= to Data sent Then Print Error
3131 045026 005000 CLR R0 ;HIGH RECV ADDRESS FOR CKMSG2
3132 045030 012702 046504 MOV #T17WFDATA,R2 ;GET EXPECTED ADDRESS FOR CKMSG2
3133 045034 012701 050242 MOV #T17BFSTA,R1 ;GET RECEIVED ADDRESS FOR CKMSG2
3134 045040 013703 002310 MOV COUNT,R3 ;NUMBER OF BYTES TO COMPARE
3135 045044 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RECV?
3136 045050 FORCERROR 232$,NOTSSR ;@D
3137 045060 103406 BCS 240$ ;BR IF YES
3138 045062 NEXT.ERRNO
3139 045062 013701 002310 232$: MOV COUNT,R1 ;GET BYTE COUNT
3140 045066 ERRHRD ERRNO,T175CMP,FIFEXP ;REPORT ERROR
                                TRAP C$ERHRD
                                .WORD 639
                                .WORD T175CMP
                                .WORD FIFEXP
3141 045076 240$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP C$CLP1
3142
3143 : Do a Write Subsystem READ STATUS
3144 : If Input Ready NOT=1 Then Print Error
3145 : If Output Ready NOT=0 Then Print Error
3146 : If Data In Miss NOT=1 Then Print Error
3147 045100 004737 047764 JSR PC,T17SRD ;SETUP PACKET FOR READ STATUS
3148 045104 012704 050350 MOV #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3149 045110 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3150 045114 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3151 045120 FORCERROR 252$ ;@DFORCE ERROR IF FORCER=1
3152 045134 103407 BCS 260$ ;BR IF CARRY SET (GOOD RETURN)
3153 045136 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3154 045140 NEXT.ERRNO
3155 045140 252$: ERRDF ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP C$ERDF
                                .WORD 640
                                .WORD T173SSR
                                .WORD PKTSSR
3156 045150 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
3157 045154 260$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP C$CLP1
3158
3159 045156 004737 050146 : Set WORDS 0-7 of expd message buffer = to recv since not testing
                                JSR PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV
    
```



3188  
 3189  
 3190  
 3191  
 3192  
 3193  
 3194  
 3195  
 3196  
 3197  
 3198  
 3199  
 3200  
 3201  
 3202  
 3203  
 3204  
 3205  
 3206  
 3207  
 3208  
 3209  
 3210  
 3211  
 3212  
 3213  
 3214  
 3215  
 3216  
 3217  
 3218  
 3219  
 3220  
 3221  
 3222  
 3223  
 3224  
 3225  
 3226  
 3227  
 3228  
 3229  
 3230  
 3231  
 3232  
 3233  
 3234  
 3235  
 3236  
 3237  
 3238

.SBTTL TEST 6: SUBTEST 6: FIFO Verify Reset FIFO Test

++  
 : TEST 6: SUBTEST 6:

: SUBTEST DESCRIPTION:

: This subtest verifies that the Reset FIFO function within  
 : the Write Miscellaneous Control 1 function initializes  
 : the FIFO to correct initial status. The following steps  
 : are performed:

1. Reset an already initialized FIFO and check for proper status.
2. Write a varying number of bytes (1-65.) into the FIFO and verify that after each block of bytes is written the FIFO can be reset to it's initial state.

: TEST STEPS:

: BEGIN

: Write to TSSR to soft initialize  
 : Do a WRITE CHARACTERISTICS to setup a message buifer  
 : Do a Write Subsystem Write Misc to Reset FIFO  
 : Do a Write Subsystem READ STATUS  
 : If all Tape Status 2 (ICER,IFMK,IHER) flip-flop  
 : signals NOT=0 Then Print Error  
 : Do a Write Subsystem WRITE NPR to set tape direction out

: REPEAT FOR BYTE COUNT 1 TO 65.

: BEGIN

: Do a Write Subsystem WRITE FIFC with the current byte count  
 : Do a Write Subsystem Write Misc to Reset FIFO  
 : Do a Write Subsystem READ STATUS  
 : If all Tape Status 2 (ICER,IFMK,IHER) flip-flop  
 : signals NOT=0 Then Print Error

: END

--

BGNSUB ;//////////////// BEGIN SUBTEST //////////////////  
 T6.6: TRAP C8SUB

: Write to TSSR register to soft initialize the controller

5\$:

JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT  
 TRAP C8ERDF  
 .WORD 641  
 .WORD SFIERR  
 .WORD SFIMSG

: Do a WRITE CHARACTERISTICS to setup a message buffer

10\$:

CLR FATFLG ;CLEAR FATAL ERROR FLAG  
 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET  
 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND  
 FORCERROR 42\$ ;aADFORCE ERROR IF FORCER=1

045274  
 045274 104402  
 045276  
 045276 004737 015774  
 045302 103405  
 045304 010001  
 045306 104455  
 045306 001201  
 045310 003652  
 045312 012034  
 045316 005037 002222  
 045322 012704 050200  
 045326 004737 010662  
 045332

```

3239 045346 103407      BCS      5.  ;BR IF CARRY SET (GOOD RETURN)
3240 045350 010001      MOV      RL.  ;SAVE CONTENTS OF TSSR
3241 045352              NEXT.ERRNO
3242 045352      42$:  ERRDF  ERRNO,T17SSH,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 642
                                .WORD T17SSR
                                .WORD  PKTSSR
                                045352 104455
                                045354 001202
                                045356 046625
3243 045362 005237 002222  INC      FATFLG ;SET FATAL ERROR FLAG
3244 045366      50$:  CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
3245      ; Do a Write Subsystem Write Misc to Reset FIFO
3246 045370 004737 050004  JSR      PC,T17RSFIF ;SETUP PKT FOR WRITE MISC RESET FIFO
3247 045374 012704 050350  MOV      #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3248 045400 010465 000000  MOV      R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3249 045404 004737 016336  JSR      PC,CHKTSSR ;WAIT FOR SSR TO SET
3250 045410      FORCERROR 62$ ;@DFORCE ERROR IF FORCER=1
3251 045424 103407      BCS      70$ ;BR IF CARRY SET (GOOD RETURN)
3252 045426 010001      MOV      R0,R1 ;SAVE CONTENTS OF TSSR
3253 045430      NEXT.ERRNO
3254 045430      62$:  ERRDF  ERRNO,T172SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 643
                                .WORD T172SSR
                                .WORD  PKTSSR
                                045430 104455
                                045432 001203
                                045434 046662
3255 045440 005237 002222  INC      FATFLG ;SET FATAL ERROR FLAG
3256 045444      70$:  CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
3257      ; Do a Write Subsystem READ STATUS
3258      ; If all Tape Status 2 (ICER,IFMK,IHER) flip-flop
3259      ; signals NOT=0 Then Print Error
3260      ;
3261 045446 004737 047764  JSR      PC,T17SRD ;SETUP PACKET FOR READ STATUS
3262 045452 012704 050350  MOV      #T17PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3263 045456 010465 000000  MOV      R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3264 045462 004737 016336  JSR      PC,CHKTSSR ;WAIT FOR SSR TO SET
3265 045466      FORCERROR 77$ ;@DFORCE ERROR IF FORCER=1
3266 045502 103407      BCS      80$ ;BR IF CARRY SET (GOOD RETURN)
3267 045504 010001      MOV      R0,R1 ;SAVE CONTENTS OF TSSR
3268 045506      NEXT.ERRNO
3269 045506      77$:  ERRDF  ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 644
                                .WORD T173SSR
                                .WORD  PKTSSR
                                045506 104455
                                045510 001204
                                045512 046726
                                045514 012046
3270 045516 005237 002222  INC      FATFLG ;SET FATAL ERROR FLAG
3271 045522      80$:  CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
3272 045524 004737 050146  JSR      PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
3273 045530 012701 046402  MOV      #T17EXSTA,R1 ;GET EXPECTED READ STATUS
3274 045534 012702 050242  MOV      #T17BFSTA,R2 ;GET RECV READ STATUS
3275 045540 011211      MOV      (R2),(R1) ;SET EXPD WORD #8 = RECV TEMP
3276 045542 042711 002000  BIC      #S1.ICER,(R1) ;SET EXPD ICER =0
3277 045546 042711 001000  BIC      #S1.IFMK,(R1) ;SET EXPD IFMK =0
3278 045552 042711 000400  BIC      #S1.IHER,(R1) ;SET EXPD IHER =0
3279 045556 016261 000002 000002  MOV      2(R2),2(R1) ;SET EXPD WORD #9 = RECV (NOT TESTING)
3280 045564 005000      CLR      R0 ;HIGH RECV ADDRESS FOR CKMSG2
    
```

```

3281 045566 012701 050222      MOV      #T17BFR,R1      ;LOW RECV ADDRESS FOR CKMSG2
3282 045572 012702 046362      MOV      #T17EXP,R2     ;EXPD ADDRESS
3283 045576 012703 000024      MOV      #20.,R3       ;NUMBER OF BYTES TO COMPARE
3284 045602 004737 011500      JSR      PC,CKMSG2      ;EXPD EQUAL RECV?
3285 045606                    FORCERROR 92$,NOTSFR    ;@D
3286 045616 103404                    BCS      100$          ;BR IF YES
3287 045620                    NEXT.ERRNO
3288 045620 92$:      ERRHRD  ERRNO,T177CMP,MSGSTAT ;REPORT ERROR
                                TRAP      C$ERHRD
                                .WORD     645
                                .WORD     T177CMP
                                .WORD     MSGSTAT
                                TRAP      C$CLP1
3289 045630 100$:      CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
3290
3291 :      Do a Write Subsystem WRITE NPR to set tape direction out
3292 045632 012700 000100      MOV      .NPR,OUT,R0    ;SET TAPE DIRECTION OUT
3293 045636 004737 050026      JSR      PC,T17SNPR     ;SETUP T17PK2 FOR WRITE NPR
3294 045642 012704 050350      MOV      #T17PK2,R1    ;GET WRITE SUBSYSTEM COMMAND PACKET
3295 045646 010465 000000      MOV      R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
3296 045652 004737 016336      JSR      PC,CHKTSSR    ;WAIT FOR SSR TO SET
3297 045656                    FORCERROR 112$          ;@DFORCE ERROR IF FORCER=1
3298 045672 103407                    BCS      120$          ;BR IF CARRY SET (GOOD RETURN)
3299 045674 010001                    MOV      R0,R1         ;SAVE CONTENTS OF TSSR
3300 045676                    NEXT.ERRNO
3301 045676 112$:      ERRDF  ERRNO,T174SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD     646
                                .WORD     T174SSR
                                .WORD     PKTSSR
3302 045706 005237 002222      INC      FATFLG        ;SET FATAL ERROR FLAG
3303 045712 120$:      CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
3304
3305 :      Setup incrementing pattern in FIFO data buffer
3306 045714 012701 046402      MOV      #T17EXSTA,R1  ;EXPD WRITE FIFO DATA BUFFER
3307 045720 012702 000100      MOV      #64.,R2       ;TEST PATTERN SIZE
3308 045724 005000      CLR      R0            ;INCREMENT TEST PATTERN
3309 045726 110021 130$:      MOV      R0,(R1)+      ;STORE INCREMENT TEST BYTE
3310 045730 005200      INC      R0           ;SET NEXT PATTERN
3311 045732 005302      DEC      R2           ;DONE?
3312 045734 003374      BGT      130$         ;BR IF NO
3313
3314 :      REPEAT FOR BYTE COUNT 1 TO 65.
3315 045736 012737 000001 002310  MOV      #1,CGUNT      ;GET FIRST BYTE COUNT
3316 :      Do a Write Subsystem WRITE FIFO with the current byte count
3317 045744 150$:      MOV      COUNT,R0    ;REPEAT LOOP LABEL
3318 045744 013700 002310      MOV      #T17EXSTA,R1 ;FIFO BYTE COUNT
3319 045750 012701 046402      JSR      PC,T17WFIF    ;FIFO WRITE DATA ADDRESS
3320 045754 004737 050052      MOV      #T17PK2,R4    ;SETUP T17PK2 FOR WRITE FIFO
3321 045760 012704 050350      MOV      R4,TSDB(R5)   ;GET WRITE SUBSYSTEM COMMAND PACKET
3322 045764 010465 000000      JSR      PC,CHKTSSR    ;SET THE PACKET ADDRESS TO EXECUTE
3323 045770 004737 016336      FORCERROR 152$        ;WAIT FOR SSR TO SET
3324 045774                    ;@DFORCE ERROR IF FORCER=1
3325 046010 103407                    BCS      160$          ;BR IF CARRY SET (GOOD RETURN)
3326 046012 010001                    MOV      R0,R1         ;SAVE CONTENTS OF TSSR
3327 046014                    NEXT.ERRNO

```

```

3328 046014          152$:  ERRDF  ERRNO,T175SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      046014 104455                                TRAP  C$ERDF
      046016 001207                                .WORD 647
      046020 047036                                .WORD T175SSR
      046022 012046                                .WORD PKTSSR
3329 046024 005237 002222          INC  FATFLG  ;SET FATAL ERROR FLAG
3330 046030          160$:  CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      046030 104406                                TRAP  C$CLP1
3331
3332  :  Do a Write Subsystem Write Misc to Reset FIFO
3333 046032 004737 050004          JSR  PC,T17RSFIF ;SETUP PKT FOR WRITE MISC RESET FIFO
3334 046036 012704 050350          MOV  #T17PK2,R4  ;GET WRITE SUBSYSTEM COMMAND PACKET
3335 046042 010465 000000          MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3336 046046 004737 016336          JSR  PC,CHKTSSR  ;WAIT FOR SSR TO SET
3337 046052          FORCERROR 162$  ;@DFORCE ERROR IF FORCER=1
3338 046066 103407          BCS  170$  ;BR IF CARRY SET (GOOD RETURN)
3339 046070 010001          MOV  R0,R1  ;SAVE CONTENTS OF TSSR
3340 046072          NEXT.ERRNO
3341 046072          162$:  ERRDF  ERRNO,T172SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      046072 104455                                TRAP  C$ERDF
      046074 001210                                .WORD 648
      046076 046662                                .WORD T172SSR
      046100 012046                                .WORD PKTSSR
3342 046102 005237 002222          INC  FATFLG  ;SET FATAL ERROR FLAG
3343 046106          170$:  CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      046106 104406                                TRAP  C$CLP1
3344
3345  :  Do a Write Subsystem READ STATUS
3346  :  If all Tape Status 2 (ICER,IFMK,IHER) flip-flop
3347  :  signals NOT=0 Then Print Error
3348 046110 004737 047764          JSR  PC,T17SRD  ;SETUP PACKET FOR READ STATUS
3349 046114 012704 050350          MOV  #T17PK2,R4  ;GET WRITE SUBSYSTEM COMMAND PACKET
3350 046120 010465 000000          MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3351 046124 004737 016336          JSR  PC,CHKTSSR  ;WAIT FOR SSR TO SET
3352 046130          FORCERROR 177$  ;@DFORCE ERROR IF FORCER=1
3353 046144 103407          BCS  180$  ;BR IF CARRY SET (GOOD RETURN)
3354 046146 010001          MOV  R0,R1  ;SAVE CONTENTS OF TSSR
3355 046150          NEXT.ERRNO
3356 046150          177$:  ERRDF  ERRNO,T173SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      046150 104455                                TRAP  C$ERDF
      046152 001211                                .WORD 649
      046154 046726                                .WORD T173SSR
      046156 012046                                .WORD PKTSSR
3357 046160 005237 002222          INC  FATFLG  ;SET FATAL ERROR FLAG
3358 046164          180$:  CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      046164 104406                                TRAP  C$CLP1
3359 046166 004737 050146          JSR  PC,T17SETEXP ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
3360 046172 012701 046402          MOV  #T17EXSTA,R1 ;GET EXPECTED READ STATUS
3361 046176 012702 050242          MOV  #T17BFSTA,R2 ;GET RECV READ STATUS
3362 046202 011211          MOV  (R2),(R1)  ;SET EXPD WORD #8 = RECV TEMP
3363 046204 042711 002000          BIC  #S1.ICER,(R1) ;SET EXPD ICER =0
3364 046210 042711 001000          BIC  #S1.IFMK,(R1) ;SET EXPD IFMK =0
3365 046214 042711 000400          BIC  #S1.IHER,(R1) ;SET EXPD IHER =0
3366 046220 016261 000002 000002  MOV  2(R2),2(R1) ;SET EXPD WORD #9 = RECV (NOT TESTING)
3367 046226 005000          CLR  R0  ;HIGH RECV ADDRESS FOR CKMSG2
3368 046230 012701 050222          MOV  #T17BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
3369 046234 012702 046362          MOV  #T17EXP,R2 ;EXPD ADDRESS
    
```

```

3370 046240 012703 000024      MOV    #20.,R3      ;NUMBER OF BYTES TO COMPARE
3371 046244 004737 011500      JSR    PC,CKMSG2   ;EXPD EQUAL F.ECV?
3372 046250                    FORCERROR 192$,NOTSSR ;@@D
3373 046260 103404            BCS    200$        ;BR IF YES
3374 046262                    NEXT.ERRNO
3375 046262 192$:            ERRHRD  ERRNO,T177CMP,MSGSTAT ;REPORT ERROR
                                TRAP    C$ERHRD
                                .WORD   650
                                .WORD   T177CMP
                                .WORD   MSGSTAT
                                TRAP    C$CLP1
                                TRAP
046262 104456
046264 001212
046266 047654
046270 012350
3376 046272 200$:            CKLOOP      ;LOOP ON ERROR, IF FLAG SET
046272 104406                    TRAP    C$CLP1
3377
3378
3379 046274 250$:            FORCEEXIT 260$
3380 046274                    INC    COUNT
3381 046304 005237 002310      CMP    COUNT,#65. ;GET NEXT BYTE COUNT
3382 046310 023727 002310 000101 BHI    260$        ;DONE ALL BYTES?
3383 046316 101002                    BHI    260$        ;BR IF YES
3384 046320 000137 045744      JMP    150$        ;DO ANOTHER BYTE COUNT
3385 046324 260$:
3386
3387 046324                    ENDSUB      ;////////// END SUBTEST ////////////
046324                    L10064:
046324 104403                    TRAP    C$ESUB
3388
3389 046326 005737 002222      TST    FATFLG     ;ANY FATAL ERRORS ?
3390 046332 001402                    BEQ    300$        ;BRANCH IF NOT
3391 046334 004737 017202      JSR    PC,CKDROP  ;TRY TO DROP THE UNIT
3392 046340 004737 016456      JSR    PC,TSTLOOP ;DO ITERATIONS?
3393 046344 103002                    BCC    305$        ;BR IF NO
3394 046346 000137 040404      JMP    T17LOOP    ;LOOP UNTIL ITERATIONS DONE
3395 046352 305$:
3396
3397 046352                    EXIT    TST      ;////////// EXIT TEST ////////////
046352 104432                    TRAP    C$EXIT
046354 002112                    .WORD   L10056-.
3398
3399

```



```

3401
3402      ;+
3403      ;LOCAL STORAGE FOR THIS TEST
3404      ;-
3405
3406 046356      T17MSK:      ;MASK OF UNTESTED BITS IN READ STATUS BYTES
3407      ;UNTESTED BITS ARE SET TO 1
3408 046356      377      .BYTE      ^C<000>      ;BYTE 0 MASK
3409 046357      037      .BYTE      ^C<340>      ;BYTE 1 MASK (PAKERR,IRESV2,IRESV1)
3410 046360      360      .BYTE      ^C<017>      ;BYTE 2 (TIMER A,TIMER B,UNDEFINED<1:0>)
3411 046361      000      .BYTE      0      ;MAKE IT EVEN
3412
3413 046362      T17EXP:      ;BEGIN EXPECTED DATA BUFFER
3414 046362      000000      .WORD      0      ;MESSAGE TYPE
3415 046364      000000      .WORD      0      ;DATA FIELD LENGTH
3416 046366      000000      .WORD      0      ;RBPCR
3417 046370      000000      .WORD      0      ;XST0
3418 046372      000000      .WORD      0      ;XST1
3419 046374      000000      .WORD      0      ;XST2
3420 046376      000000      .WORD      0      ;XST3
3421 046400      000000      .WORD      0      ;XST4 (ALWAYS PRESENT FOR WRITE SUB.)
3422 046402      T17EXSTA: .BLKB 66.      ;EXPECTED READ STATUS AND WRITE FIFO DATA
3423 046504      T17EXEND:      ;END EXPECTED DATA BUFFER
3424
3425 046504      T17WFDATA: .BLKB 66.      ;WRITE FIFO EXPECTED DATA BUFFER
3426
3427      ;+
3428      ;LOCAL TEXT MESSAGES FOR TEST
3429      ;-
3430
3431 046606      106      111      106      TST17ID:      .ASCIZ      'FIFO Exerciser'
3432 046625      127      122      111      T17SSR: .ASCIZ      'WRITE CHARACTERISTICS Failed'
3433 046662      127      122      111      T172SSR: .ASCIZ      'WRITE SUBSYSTEM (Write Misc) Failed'
3434 046726      127      122      111      T173SSR: .ASCIZ      'WRITE SUBSYSTEM (Read Status) Failed'
3435 046773      127      122      111      T174SSR: .ASCIZ      'WRITE SUBSYSTEM (Write Npr) Failed'
3436 047036      127      122      111      T175SSR: .ASCIZ      'WRITE SUBSYSTEM (Write FIFO) Failed'
3437 047102      127      122      111      T176SSR: .ASCIZ      'WRITE SUBSYSTEM (Read FIFO) Failed'
3438 047145      106      111      106      T171CMP: .ASCIZ      'FIFO Status in WORD #9 Incorrect after Initialize'
3439 047227      122      145      141      T172CMP: .ASCIZ      'Read FIFO Data not equal to Write FIFO , Data is in WORD #8'
3440 047323      106      111      106      T173CMP: .ASCIZ      'FIFO Status (In WORD #9) Incorrect after WRITE FIFO'
3441 047407      106      111      106      T174CMP: .ASCIZ      'FIFO Status (In WORD #9) Incorrect after READ FIFO'
3442 047472      122      145      141      T175CMP: .ASCIZ      'Read FIFO Data not equal to Write FIFO Data'
3443 047546      106      111      106      T176CMP: .ASCIZ      'FIFO Status (In WORD #9) Incorrect after READ FIFO from an Empty FIFO'
3444 047654      106      111      106      T177CMP: .ASCIZ      'FIFO Status (In WORD #9) Incorrect after RESET FIFO'
3445      .EVEN
3446
3447      ;+
3448      ; CLEAR MESSAGE BUFFER
3449      ;-
3450 047740      T17CLRBUF:
3451 047740      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
3452 047744      012701      050222      MOV      #T17BFR,R1      ;GET MESSAGE BUFFER ADDRESS
3453 047750      012702      000120      MOV      #T17BEND-T17BFR,R2      ;SIZE OF MESSAGE BUFFER IN BYTES
3454 047754      105021      10$:      CLRB      (R1)+      ;CLEAR A BYTE
3455 047756      005302      DEC      R2      ;DONE?
3456 047760      003375      BGT      10$      ;BR IF NO
3457 047762      000207      RTS      PC      ;RETURN
    
```

```

3458
3459
3460      :+
3461      : SETUP T17PK2 PACKET FOR READ STATUS
3462      :-
3462 047764 T17SRD:
3463 047764 004737 047740      JSR      PC,T17CLRBUF      :CLEAR MESSAGE BUFFER
3464 047770 012700 050360      MOV      #T17DT2,R0      :WRITE SUBSYSTEM DATA BUFFER
3465 047774 112720 000005      MOVB     #PW.RDSTATUS,(R0)+ :STORE READ STATUS COMMAND IN BSELO
3466 050000 105010      CLR      (R0)            :CLEAR BSEL1
3467 050002 000207      RTS      PC              :RETURN
3468
3469
3470      :+
3471      : SETUP T17PK2 PACKET FOR WRITE MISC RESET FIFO
3472      :-
3472 050004 T17RSFIF:
3473 050004 004737 047740      JSR      PC,T17CLRBUF      :CLEAR MESSAGE BUFFER
3474 050010 012700 050360      MOV      #T17DT2,R0      :WRITE SUBSYSTEM DATA BUFFER
3475 050014 112720 000010      MOVB     #PW.WMISC,(R0)+  :STORE WRITE MISCELLANEOUS IN BSELO
3476 050020 112710 000030      MOVB     #MS.RSFIF!MS.RSTAP,(R0) :STORE BSEL1 CLEAR FIFO CODES
3477 050024 000207      RTS      PC              :RETURN
3478
3479
3480      :+
3481      : SETUP T17PK2 PACKET FOR WRITE NPR
3482      : INPUT:
3483      : RO CONTAINS BSEL1 NPR DATA
3484      :
3485      : SETS NP.WRP SINCE IF 0 IT WRITES WRONG PARITY.
3486      :-
3487 050026 T17SNPR:
3488 050026 004737 047740      JSR      PC,T17CLRBUF      :CLEAR MESSAGE BUFFER
3489 050032 012701 050360      MOV      #T17DT2,R1      :WRITE SUBSYSTEM DATA BUFFER
3490 050036 112721 000011      MOVB     #PW.WNPR,(R1)+  :STORE WRITE NPR IN BSELO
3491 050042 052700 000020      BIS      #NP.WRP,R0      :DON'T WRITE WRONG PARITY
3492 050046 110011      MOVB     R0,(R1)          :STORE NPR DATA IN BSEL1
3493 050050 000207      RTS      PC              :RETURN
3494
3495
3496      :+
3497      : SETUP T17PK2 PACKET FOR WRITE FIFO
3498      : INPUT:
3499      : RO CONTAINS BYTE COUNT
3500      : R1 CONTAINS DATA PATTERN BLOCK ADDRESS
3501      :-
3502 050052 T17WFIF:
3503 050052      SAVREG      :SAVE R1-R5 UNTIL NEXT RETURN
3504 050056 004737 047740      JSR      PC,T17CLRBUF      :CLEAR MESSAGE BUFFER
3505 050062 012702 050360      MOV      #T17DT2,R2      :WRITE SUBSYSTEM DATA BUFFER
3506 050066 112722 000004      MOVB     #PW.WFIFO,(R2)+  :STORE WRITE FIFO IN BSELO
3507 050072 110022      MOVB     R0,(R2)+        :STORE BYTE COUNT IN BSEL1
3508 050074 005022      CLR      (R2)+          :CLEAR SEL2 (UNUSED)
3509 050076 112122 10$:      MOVB     (R1)+,(R2)+      :STORE DATA PATTERN BYTE
3510 050100 005300      DEC      R0              :DONE ALL BYTES?
3511 050102 003375      BGT      10$             :BR IF NO
3512 050104 000207      RTS      PC              :RETURN
3513
3514      :+

```

```

3515 ; SETUP T17PK2 PACKET FOR READ FIFO
3516 ;
3517 ; INPUT:
3518 ; RO CONTAINS SEL2 BYTE COUNT
3519 ;
3520 050106 T17RFIF:
3521 050106 004737 047740 JSR PC,T17CLRBUF ;CLEAR MESSAGE BUFFER
3522 050112 012701 050360 MOV #T17DT2,R1 ;WRITE SUBSYSTEM DATA BUFFER
3523 050116 112721 000003 MOVB #PW.RFIFO,(R1)+ ;STORE READ FIFO IN BSELO
3524 050122 110021 MOVB RO,(R1)+ ;STORE BYTE COUNT IN BSEL1
3525 050124 000207 RTS PC ;RETURN
3526 ;+
3527 ; CLEAR EXPECTED DATA MESSAGE BUFFER
3528 ;
3529 050126 T17CLEXP:
3530 050126 012701 046362 MOV #T17EXP,R1 ;GET EXPD ADDRESS
3531 050132 012700 000122 MOV #T17EXEND-T17EXP,RO ;GET EXPD SIZE
3532 050136 105021 10$: CLRB (R1)+ ;CLEAR A BYTE
3533 050140 005300 DEC RO ;DONE?
3534 050142 003375 BGT 10$ ;BR IF NO
3535 050144 000207 RTS PC ;RETURN
3536 ;+
3537 ;Set WORDS 0-7 of expd message buffer = to recv since not testing
3538 ;
3539 ;
3540 050146 T17SETEXP:
3541 050146 012702 046362 MOV #T17EXP,R2 ;GET EXPD
3542 050152 012703 050222 MOV #T17BFR,R3 ;GET READ STATUS RECV BUFFER
3543 050156 012700 000010 MOV #8,RO ;SET WORDS 0-7 EXP=RECV
3544 050162 012322 5$: MOV (R3)+,(R2)+ ;SET EXPD=RECV
3545 050164 005300 DEC RO ;DONE WORDS 0-7 WORDS?
3546 050166 003375 BGT 5$ ;BR IF NO
3547 050170 000207 RTS PC ;RETURN
3548 ;
3550 050200 .=<. +10>&177770
3552 ;
3553 ;WRITE CHARACTERISTICS COMMAND PACKET
3554 ;
3555 050200 T17PACKET:
3556 050200 100004 .WORD 100004 ;COMMAND PACKET FOR TEST
3557 050202 050210 .WORD T17DATA ;WRITE CHARACTERISTICS COMMAND, WITH ACK
3558 050204 000000 .WORD 0 ;ADDRESS OF CHARACTERISTICS BLOCK
3559 050206 000012 .WORD 10. ;MINIMUM MESSAGE PACKET SIZE
3560 ;
3561 050210 T17DATA:
3562 050210 050222 .WORD T17BFR ;CHARACTERISTICS DATA BLOCK
3563 050212 000000 .WORD 0 ;ADDRESS OF MESSAGE BUFFER
3564 050214 000024 .WORD 20. ;LENGTH OF MESSAGE BUFFER
3565 050216 000000 .WORD 0 ;ESS,ENB,EAI,ERI
3566 050220 000000 .WORD 0 ;EXTENDED FEATURES UNIT NO. E1..
3567 ;
3568 ;MESSAGE BUFFER FOR ALL TEST 6 COMMANDS
3569 ;
3570 ;
3571 050222 T17BFR:
3572 050222 000000 .WORD 0 ;BEGIN MESSAGE BUFFER
3573 050224 000000 .WORD 0 ;MESSAGE TYPE
;DATA FIELD LENGTH
    
```

```

3574 050226 000000          .WORD 0          ;RPCR
3575 050230 000000          .WORD 0          ;XST0
3576 050232 000000          .WORD 0          ;XST1
3577 050234 000000          .WORD 0          ;XST2
3578 050236 000000          .WORD 0          ;XST3
3579 050240 000000          .WORD 0          ;XST4 (ALWAYS PRESENT FOR WRITE SUBSYSTEM
3580 050242          T17BFSTA: .BLKB 64. ;READ STATUS AND WRITE FIFO BUFFER
3581 050342          T17BEND:        ;END OF MESSAGE BUFFER
3582          ;
3583          ;WRITE SUBSYSTEM READ STATUS COMMAND PACKET
3584          ;
3586          050350          .=<.+10>&177770
3588 050350          T17Pr2:
3589 050350 100006          .WORD P.WRTSUB!P.ACK ;WRITE SUBSYSTEM WITH ACK
3590 050352 050360          .WORD T17DT2        ;LOW ADDRESS OF DATA BLOCK
3591 050354 000000          .WORD 0            ;HIGH ADDRESS OF DATA BLOCK
3592 050356 000012          .WORD 10.         ;MINIMUM MESSAGE PACKET SIZE
3593          ;
3594 050360          T17DT2:        ;DATA BLOCK
3595 050360          .BYTE 0            ;BSEL0
3596 050361          .BYTE 0            ;BSEL1
3597 050362 000000          .WORD 0            ;SEL2
3598 050364          .BLKB 66.         ;WRITE FIFO DATA OUTPUT BUFFER
3599          ;
3600 050466          ENDTST
3601          050466          L10056:
          050466 104401          TRAP CSETST
  
```

```

3603          .SBTTL TEST 7: STATIC TRANSPORT BUS INTERFACE TEST
3604
3605          :+ TEST DESCRIPTION:
3606
3607          : TEST STEPS:
3608
3609          : REPEAT FOR LOOPCNT
3610          : BEGIN
3611          : Write to TSSR register to soft initialize the controller
3612          : Do WRITE CHARACTERISTICS to check for Extended Features Switch
3613          : If Extended Features Hardware Switch Clear then:
3614          :   Do Write Subsystem Write Miscellaneous to Set Extended Features.
3615          : Do WRITE CHARACTERISTICS to select reserved unit 7
3616          : Do a Write Subsystem READ STATUS
3617          : If any transport interface signals are asserted then Print Error
3618          : END
3619          :--
3620
3621
3622
3623          BGNTST
3624
3625          T7::
3626          MOV      #TST18ID,R0          ;ASCII MESSAGE TO IDENTIFY TEST
3627          JSR      PC,TSTSETUP         ;DO INITIAL TEST SETUP
3628          MOV      #10.,LOOPCNT       ;PERFORM 10 ITERATIONS
3629
3630          T18LOOP:
3631          : Write to TSSR register to soft initialize the controller
3632          : 5$:
3633          JSR      PC,SOFINIT          ;WRITE TO TSSR TO SOFT INITIALIZE
3634          BCS      10$                 ;BR IF SOFT INIT OKAY
3635          MOV      R0,R1               ;SAVE CONTENTS OF TSSR
3636          ERDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
3637
3638          TRAP    CSERDF
3639          .WORD  700
3640          .WORD  SFIERR
3641          .WORD  SFIMSG
3642
3643          : Do WRITE CHARACTERISTICS to check for Extended Features Switch
3644          : 10$:
3645          CLR      FATFLG              ;CLEAR FATAL ERROR FLAG
3646          MOV      #T18PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
3647          JSR      PC,WRTCHR          ;DO WRITE CHARACTERISTICS COMMAND
3648          FORCERROR 12$                ;@@DFORCE ERROR IF FORCER=1
3649          BCS      15$                 ;BR IF CARRY SET (GOOD RETURN)
3650          MOV      R0,R1               ;SAVE CONTENTS OF TSSR
3651          NEXT.ERRNO
3652          : 12$:
3653          ERDF   ERRNO,T18SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
3654
3655          TRAP    CSERDF
3656          .WORD  701
3657          .WORD  T18SSR
3658          .WORD  PKTSSR
3659
3660          : 15$:
3661          INC      FATFLG              ;SET FATAL ERROR FLAG
3662          CKLOOP ;LOOP ON ERROR, IF FLAG SET
3663
3664          TRAP    C$CLP1
3665
3666          : If Extended Features Hardware Switch Clear then:
3667          : Do Write Subsystem Write Miscellaneous to Set Extended Features.
3668          : 365:
3669          MOV      #T18BFR,R1         ;MESSAGE BUFFER ADDRESS
3670
3671          050470
3672          050470
3673          050470 012700 051176
3674          050474 004737 016510
3675          050500 012737 000012 002216
3676          050506
3677          050506
3678          050506 004737 015774
3679          050512 103405
3680          050514 010001
3681          050516
3682          050516 104455
3683          050520 001274
3684          050522 003652
3685          050524 012034
3686
3687          050526 005037 002222
3688          050532 012704 051660
3689          050536 004737 010662
3690          050542
3691          050556 103407
3692          050560 010001
3693          050562
3694          050562 104455
3695          050564 001275
3696          050566 051235
3697          050570 012046
3698          050572 005237 002222
3699          050576
3700          050576 104406
3701
3702          050600 012701 051702
    
```

```

3654 050604 032761 000200 000012 BIT #X2.EXTF,XST2(R1) ;EXTENDED FEATURES SWITCH SET?
3655 050612 001026 BNE 30$ ;BR IF YES
3656 050614 004737 051526 JSR PC,T18SMISC ;SETUP PACKET FOR WRITE MISCELLANEOUS
3657 050620 012704 051730 MOV #T18PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3658 050624 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3659 050630 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3660 050634 FORCERROR 22$ ;@DFORCE ERROR IF FORCER=1
3661 050650 103407 BCS 30$ ;BR IF CARRY SET (GOOD RETURN)
3662 050652 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3663 050654 NEXT.ERRNO
3664 050654 22$: ERRDF ERRNO,T182SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP C$ERDF
                                .WORD 702
                                .WORD T182SSR
                                .WORD PKTSSR
3665 050664 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
3666 050670 30$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP C$CLP1
3667
3668
3669 ; Do WRITE CHARACTERISTICS to select reserved unit 7
3670 050672 005037 002222 CLR FATFLG ;CLEAR FATAL ERROR FLAG
3671 050676 012704 051660 MOV #T18PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
3672 050702 004737 010662 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
3673 050706 FORCERROR 42$ ;@DFORCE ERROR IF FORCER=1
3674 050722 103407 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
3675 050724 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3676 050726 NEXT.ERRNO
3677 050726 42$: ERRDF ERRNO,T18SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP C$ERDF
                                .WORD 703
                                .WORD T18SSR
                                .WORD PKTSSR
3678 050736 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
3679 050742 50$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP C$CLP1
3680
3681 ; Clear message buffer
3682 050744 012701 051702 MOV #T18BFR,R1 ;GET MESSAGE BUFFER ADDRESS
3683 050750 013700 051674 MOV T18DATA+4,R0 ;SIZE OF MESSAGE BUFFER IN BYTES
3684 050754 105021 60$: CLRB (R1)+ ;CLEAR A BYTE
3685 050756 005300 DEC R0 ;DONE?
3686 050760 003375 BGT 60$ ;BR IF NO
3687 ; Do a Write Subsystem READ STATUS
3688 050762 004737 051506 JSR PC,T18SRD ;SETUP PACKET FOR READ STATUS
3689 050766 012704 051730 MOV #T18PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3690 050772 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3691 050776 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3692 051002 FORCERROR 62$ ;@DFORCE ERROR IF FORCER=1
3693 051016 103407 BCS 70$ ;BR IF CARRY SET (GOOD RETURN)
3694 051020 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3695 051022 NEXT.ERRNO
3696 051022 62$: ERRDF ERRNO,T183SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP C$ERDF
                                .WORD 704
                                .WORD T183SSR
                                .WORD PKTSSR
051022 104455
051024 001300
051026 051336
051030 012046
    
```

```

3697 051032 005237 002222          INC    FATFLG          ;SET FATAL ERROR FLAG
3698 051036          70$:    CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                051036 104406          TRAP    C$CLP1
3699
3700
3701          :          Set first 8 words of expd message buffer = to recv since not testing
3702          :          Set unused bits in Read Status expd equal rcvd
3703 051040 004737 051550          JSR    PC,T18SETEXP    ;SET SOME EXPD TO RECV
3704          :          If any transport interface signals are asserted then Print Error
3705 051044 005000          CLR    R0              ;HIGH RECV ADDRESS FOR CKMSG2
3706 051046 012701 051702          MOV    #T18BFR,R1     ;LOW RECV ADDRESS FOR CKMSG2
3707 051052 012702 051146          MOV    #T18EXP,R2    ;EXPD ADDRESS
3708 051056 012703 000012          MOV    #10,R3        ;NUMBER OF WORDS TO COMPARE
3709 051062 004737 011500          JSR    PC,CKMSG2     ;EXPD EQUAL RECV?
3710 051066          FORCERROR    82$,NOTSSR    ;@@D
3711 051076 103404          BCS    90$           ;BR IF YES
3712 051100          NEXT.ERRNO
3713 051100          82$:    ERRHRD  ERRNO,T18CMP,MSGSTAT ;REPORT ERROR
                                051100 104456          TRAP    C$ERHRD
                                051102 001301          .WORD  705
                                051104 051403          .WORD  T18CMP
                                051106 012350          .WORD  MSGSTAT
3714 051110          90$:    CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                051110 104406          TRAP    C$CLP1
3715
3716 051112 005737 002222          TST    FATFLG        ;ANY FATAL ERRORS ?
3717 051116 001402          BEQ    160$         ;BRANCH IF NOT
3718 051120 004737 017202          JSR    PC,CKDROP    ;TRY TO DROP THE UNIT
3719 051124 004737 016456          160$: JSR    PC,TSTLOOP ;DO ITERATIONS?
3720 051130 103002          BCC    165$         ;BR IF NO
3721 051132 000137 050506          JMP    T18LOOP      ;LOOP UNTIL ITERATIONS DONE
3722 051136          165$:
3723 051136          EXIT    TST
                                051136 104432          TRAP    C$EXIT
                                051140 000606          .WORD  L10065-
3724
3725
    
```

```

3727
3728      ;+
3729      ;LOCAL STORAGE FOR THIS TEST
3730      ;-
3731 051142      T18MSK:
3732 051142      377      .BYTE ^C<000>      ;MASK OF UNUSED BITS IN READ STATUS BYTES
3733 051143      037      .BYTE ^C<340>      ;BYTE 0 MASK
3734 051144      100      .BYTE ^C<277>      ;BYTE 1
3735 051145      000      .BYTE 0      ;BYTE 2
3736      ;MAKE IT EVEN
3737 051146      T18EXP:
3738 051146      000000      .WORD 0      ;EXPECTED DATA BUFFER
3739 051150      000000      .WORD 0      ;MESSAGE TYPE
3740 051152      000000      .WORD 0      ;DATA FIELD LENGTH
3741 051154      000000      .WORD 0      ;RBPCR
3742 051156      000000      .WORD 0      ;XST0
3743 051160      000000      .WORD 0      ;XST1
3744 051162      000000      .WORD 0      ;XST2
3745 051164      000000      .WORD 0      ;XST3
3746 051166      000000      .WORD 0      ;XST4 (ALWAYS PRESENT FOR WRITE SUB)
3747 051170      000000      .WORD 0      ;READ STATUS BYTE 1/0
3748      ;READ STATUS BYTE 2
3749 051172      377      020      T18XS: .BYTE 377,020      ;READ STATUS BYTE 0/1 EXPECTED BASE
3750 051174      000000      .WORD 0      ;READ STATUS BYTE 2 EXPECTED BASE
3751
3752      ;+
3753      ;LOCAL TEXT MESSAGES FOR TEST
3754      ;-
3755
3756 051176      123      164      141      TST18ID: .ASCIZ 'Static Transport Bus Interface'
3757 051235      127      122      111      T18SSR: .ASCIZ 'WRITE CHARACTERISTICS Failed'
3758 051272      127      122      111      T182SSR: .ASCIZ 'WRITE SUBSYSTEM (Write Misc) Failed'
3759 051336      127      122      111      T183SSR: .ASCIZ 'WRITE SUBSYSTEM (Read Status) Failed'
3760 051403      124      162      141      T18CMP: .ASCIZ 'Transport Bus Interface Signals NOT Negated After Unit 7 Selected'
3761      .EVEN
3762
3763      ;+
3764      ; SETUP T18PK2 PACKET FOR READ STATUS
3765      ;-
3766 051506      T18SRD:
3767 051506      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
3768 051512      012700      051740      MOV #T18DT2,R0      ;WRITE SUBSYSTEM DATA BUFFER
3769 051516      112720      000005      MOVB #PW.RDSTATUS,(R0)+      ;STORE READ STATUS COMMAND IN BSEL0
3770 051522      105010      CLRB (R0)      ;CLEAR BSEL1
3771 051524      000207      RTS PC      ;RETURN
3772
3773      ;+
3774      ; SETUP T18PK2 PACKET FOR WRITE MISC.
3775      ;-
3776 051526      T18SMISC:
3777 051526      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
3778 051532      012700      051740      MOV #T18DT2,R0      ;WRITE SUBSYSTEM DATA BUFFER
3779 051536      112720      000010      MOVB #PW.WMISC,(R0)+      ;STORE WRITE MISCELLANEOUS IN BSEL0
3780 051542      112710      000200      MOVB #MS.EXT,(R0)      ;STORE INVERT EXTENDED FEATURES IN BSEL1
3781 051546      000207      RTS PC      ;RETURN
3782
3783      ;+
    
```



```

3784 ;Set first 8 words of expd message buffer = to recv since not testing
3785 ; Set unused bits in Read Status expd equal rcvd
3786 ;
3787 051550 T18SETEXP:
3788 051550 012702 051146 MOV #T18EXP,R2 ;GET EXPD
3789 051554 012703 051702 MOV #T18BFR,R3 ;GET READ STATUS RECV BUFFER
3790 051560 012700 000010 MOV #8,R0 ;SET FIRST 8 WORDS EXP=RECV
3791 051564 012322 5$: MOV (R3)+,(R2)+ ;SET EXPD=RECV
3792 051566 005300 DEC R0 ;DONE FIRST 8 WORDS?
3793 051570 003375 BGT 5$ ;BR IF NO
3794 051572 012701 051142 MOV #T18MSK,R1 ;GET UNUSED BIT MASK
3795 051576 013712 051172 MOV T18XS,(R2) ;SETUP BASE EXPECTED BYTE 1/0
3796 051602 013762 051174 000002 MOV T18XS+2,(R2) ;SETUP BASE EXPECTED BYTE 2
3797 051610 011300 MOV (R3),R0 ;GET RECV BYTE 1 AND BYTE 0
3798 051612 041100 BIC (R1),R0 ;CLEAR ALL BUT UNUSED
3799 051614 040012 BIC R0,(R2) ;CLEAR UNUSED IN EXP
3800 051616 050012 BIS R0,(R2) ;SET UNUSED EXPD=RECV FOR COMPARE
3801 051620 016300 000002 MOV 2(R3),R0 ;GET RECV BYTE 2
3802 051624 046100 000002 BIC 2(R1),R0 ;CLEAR ALL BUT UNUSED
3803 051630 040062 000002 BIC R0,2(R2) ;CLEAR UNUSED IN EXPD
3804 051634 050062 000002 BIS R0,2(R2) ;SET UNUSED EXPD=RECV FOR COMPARE
3805 051640 105062 000003 CLRB 3(R2) ;CLEAR EXPD BYTE 3 (UNUSED)
3806 051644 105063 000003 CLRB 3(R3) ;CLEAR RECV BYTE 3 (UNUSED)
3807 051650 000207 RTS PC ;RETURN
3808
3810 051660 .=<. +10>&177770
3812
3813 ;WRITE CHARACTERISTICS COMMAND PACKET
3814
3815 051660 T18PACKET: ;COMMAND PACKET FOR TEST
3816 051660 100004 .WORD 100004 ;WRITE CHARACTERISTICS COMMAND, WITH ACK
3817 051662 051670 .WORD T18DATA ;ADDRESS OF CHARACTERISTICS BLOCK
3818 051664 000000 .WORD 0
3819 051666 000012 .WORD 10. ;MESSAGE PACKET MINIMUM SIZE
3820
3821 051670 T18DATA: ;CHARACTERISTICS DATA BLOCK
3822 051670 051702 .WORD T18BFR ;ADDRESS OF MESSAGE BUFFER
3823 051672 000000 .WORD 0
3824 051674 000024 .WORD 20. ;LENGTH OF MESSAGE BUFFER
3825 051676 000000 .WORD 0 ;ESS,ENB,EAI,ERI
3826 051700 000007 .WORD 7 ;SELECT RESERVED UNIT 7
3827
3828
3829 051702 T18BFR: ;MESSAGE BUFFER
3830 051702 000000 .WORD 0 ;MESSAGE TYPE
3831 051704 000000 .WORD 0 ;DATA FIELD LENGTH
3832 051706 000000 .WORD 0 ;RBPCR
3833 051710 000000 .WORD 0 ;XST0
3834 051712 000000 .WORD 0 ;XST1
3835 051714 000000 .WORD 0 ;XST2
3836 051716 000000 .WORD 0 ;XST3
3837 051720 000000 .WORD 0 ;XST4 (ALWAYS PRESENT FOR WRITE SUBSYSTEM)
3838 051722 000000 .WORD 0 ;READ STATUS BYTE 1/0 RETURNED
3839 051724 000000 .WORD 0 ;READ STATUS BYTE 2
3840
3841 ;WRITE SUBSYSTEM READ STATUS COMMAND PACKET
3842
    
```

```
3844          051730
3846 051730
3847 051730 100006
3848 051732 051740
3849 051734 000000
3850 051736 000010
3851
3852 051740
3853 051740      000
3854 051741      000
3855 051742 000000
3856 051744 000000
3857
3858
3859 051746          ENDTST
      051746
      051746 104401

T18PK2:  .=<.+10>&177770
          .WORD  P.WRTSUB!P.ACK      ;WRITE SUBSYSTEM WITH ACK
          .WORD  T18DT2              ;LOW ADDRESS OF DATA BLOCK
          .WORD  0                    ;HIGH ADDRESS OF DATA BLOCK
          .WORD  8.                   ;BUFFER EXTENT

T18DT2:   .BYTE  0                    ;DATA BLOCK
          .BYTE  0                    ;BSELO
          .WORD  0                    ;BSEL1
          .WORD  0                    ;SEL2
          .WORD  0                    ;DATA

L10065:  TRAP      C$ETST
```

3861  
3862  
3863  
3864  
3865  
3866  
3867  
3868  
3869  
3870  
3871  
3872  
3873  
3874  
3875  
3876  
3877  
3878  
3879  
3880  
3881  
3882  
3887  
3888  
3889  
3890  
3891

.SBTTL TEST 8: TRANSPORT BUS INTERFACE LOOPBACK TEST

++  
: TEST DESCRIPTION:

: This test verifies the controller's Transport Bus  
: drivers, receivers, and signal loopback logic. Note  
: that the Static Transport Bus test must have run  
: correctly for this test to provide meaningful results.

: TEST STEPS:

: REPEAT FOR LOOPCNT

: BEGIN

: Do Subtest 1 - Loopback Control signals test  
: Do Subtest 2 - Loopback Read/Write signals test  
: Do Subtest 3 - Loopback Write Strobe test  
: Do Subtest 4 - Loopback Read Strobe test

: END

:--

BGNTST

MOV #TST19ID,R0  
JSR PC,TSTSETUP  
MOV #10.,LOOPCNT

T19LOOP:

T8::

:ASCII MESSAGE TO IDENTIFY TEST  
:DO INITIAL TEST SETUP  
:PERFORM 10 ITERATIONS

012700 060162  
004737 016510  
012737 000012 002216

051750  
051750

3893  
3894  
3895  
3896  
3897  
3898  
3899  
3900  
3901  
3902  
3903  
3904  
3905  
3906  
3907  
3908  
3909  
3910  
3911  
3912  
3913  
3914  
3915  
3916  
3917  
3918  
3919  
3920  
3921  
3922  
3923  
3924  
3925  
3926  
3927  
3928  
3929  
3930  
3931  
3932  
3933  
3934  
3935  
3936  
3937  
3938  
3939  
3940  
3941  
3942  
3943  
3944  
3945  
3946  
3947

.SBTTL TEST 8: SUBTEST 1: LOOPBACK CONTROL SIGNAL TEST

TEST 8: SUBTEST 1:

SUBTEST DESCRIPTION:

This subtest verifies the Transport Control loopback path can transmit and receive correctly. The control signals are all loopback signals other than the read/write data (IW<7:0> and IR<7:0>).

TEST STEPS:

The loopback signals IFAD,ITAD0,ITAD1 are the tape unit select lines. Since reserved unit 7 must remain selected these signals are always set low. This further means the signals they drive (ISPEED,IRDY,IONL) are only tested in the low state.

BEGIN

Write to TSSR register to soft initialize the controller  
 Do WRITE CHARACTERISTICS to check for Extended Features Switch  
 If Extended Features Hardware Switch Clear then:  
 Do Write Subsystem Write Miscellaneous to Set Extended Features.  
 Do WRITE CHARACTERISTICS to select reserved unit 7 and setup BUFFER  
 Do a Write Subsystem WRITE NPR to set tape direction out and Loopback  
 Do Write Subsystem Write Control to CLEAR loopback signals group 1.  
 Do Write Subsystem Write Format to CLEAR loopback signals group 2.  
 (the loopback signals have to be cleared here due to the flip-flops that are set on a 1 to 0 transition (IHER,IFMK,ICER))  
 Do a Write Subsystem Write Misc to Reset Tape Status F-FLOPS  
 Do a Write Subsystem READ STATUS  
 If all Tape Status 2 (ICER,IFMK,IHER) flip-flop signals NOT=0 Then Print Error

REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE

BEGIN

Do Write Subsystem Write Control to Drive loopback signals group 1.  
 Do Write Subsystem Write Format to Drive loopback signals group 2.  
 Do a Write Subsystem READ STATUS  
 If loopback data NOT= data sent Then Print Error  
 Do a Write Subsystem Write Misc to Reset Tape Status F-FLOPS  
 Do a Write Subsystem READ STATUS  
 If all Tape Status 2 (ICER,IFMK,IHER) flip-flop signals NOT=0 Then Print Error

END

BGNSUB

////////// BEGIN SUBTEST //////////  
 T8.1:

TRAP CSBSUB

§§:

Write to TSSR register to soft initialize the controller

JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT

051766  
051766  
051766 104402  
  
051770  
051770 004737 015774  
051774 103405  
051776 010001  
052000

```

052000 104455
052002 001440
052004 003652
052006 012034
3948
3949 052010 005037 002222 10$: Do WRITE CHARACTERISTICS to check for Extended Features Switch
3950 052014 012704 062310 CLR FATFLG ;CLEAR FATAL ERROR FLAG
3951 052020 004737 010662 MOV #T19PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
3952 052024 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
3953 052040 103407 FORCERROR 12$ ;@DFORCE ERROR IF FORCER=1
3954 052042 010001 BCS 15$ ;BR IF CARRY SET (GOOD RETURN)
3955 052044 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3956 052044 12$: NEXT.ERRNO ERRDF ERRNO,T19SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
052044 104455 TRAP C$ERDF
052046 001441 .WORD 801
052050 060223 .WORD T19SSR
052052 012046 .WORD PKTSSR
3957 052054 005237 002222 15$: INC FATFLG ;SET FATAL ERROR FLAG
3958 052060 CKLOOP ;LOOP ON ERROR, IF FLAG SET
052060 104406 TRAP C$CLP1
3959 ; If Extended Features Hardware Switch Clear then:
3960 ; Do Write Subsystem Write Miscellaneous to Set Extended Features.
3961 052062 012701 062332 000012 MOV #T19BFR,R1 ;MESSAGE BUFFER ADDRESS
3962 052066 032761 000200 BIF #X2.EXTF,XST2(R1) ;EXTENDED FEATURES SWITCH SET?
3963 052074 001076 BNE 30$ ;BR IF YES
3964 052076 004737 062162 JSR PC,T19SEXT ;SETUP PACKET FOR WRITE MISC INVERT
3965 052102 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3966 052106 010465 000000 MOV R4,TSD(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3967 052112 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3968 052116 FORCERROR 22$ ;@DFORCE ERROR IF FORCER=1
3969 052132 103407 BCS 30$ ;BR IF CARRY SET (GOOD RETURN)
3970 052134 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3971 052136 NEXT.ERRNO
3972 052136 22$: ERRDF ERRNO,T192SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
052136 104455 TRAP C$ERDF
052140 001442 .WORD 802
052142 060260 .WORD T192SSR
052144 012046 .WORD PKTSSR
3973 052146 005237 002222 30$: INC FATFLG ;SET FATAL ERROR FLAG
3974 052152 CKLOOP ;LOOP ON ERROR, IF FLAG SET
052152 104406 TRAP C$CLP1
3975 ; Do WRITE CHARACTERISTICS to select reserved unit 7
3976 052154 005037 002222 CLR FATFLG ;CLEAR FATAL ERROR FLAG
3977 052160 012704 062310 MOV #T19PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
3978 052164 004737 010662 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
3979 052170 FORCERROR 42$ ;@DFORCE ERROR IF FORCER=1
3980 052204 103407 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
3981 052206 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3982 052210 NEXT.ERRNO
3983 052210 42$: ERRDF ERRNO,T19SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
052210 104455 TRAP C$ERDF
052212 001443 .WORD 803
052214 060223 .WORD T19SSR
052216 012046 .WORD PKTSSR
3984 052220 005237 002222 50$: INC FATFLG ;SET FATAL ERROR FLAG
3985 052224 CKLOOP ;LOOP ON ERROR, IF FLAG SET
052224 104406 TRAP C$CLP1

```

```

3986 ; Do a Write Subsystem WRITE NPR to set tape direction out and Loopback
3987 052226 012700 000100 MOV #NP.OUT,R0 ;SET TAPE DIRECTION OUT
3988 052232 052700 000040 BIS #NP.LOOP,R0 ;SET LOOPBACK ENABLE
3989 052236 004737 062022 JSR PC,T19SNPR ;SETUP T19PK2 FOR WRITE NPR
3990 052242 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
3991 052246 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
3992 052252 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
3993 052256 FORCERROR 62$ ;@@DFORCE ERROR IF FORCER=1
3994 052272 103407 BCS 70$ ;BR IF CARRY SET (GOOD RETURN)
3995 052274 010001 MOV RO,R1 ;SAVE CONTENTS OF TSSR
3996 052276 NEXT.ERRNO
3997 052276 62$: ERRDF ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
; TRAP C$ERDF
; .WORD 804
; .WORD T194SSR
; .WORD PKTSSR
3998 052306 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
3999 052312 70$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
; TRAP C$CLP1
4000 ; Do Write Subsystem Write Control to CLEAR loopback signals group 1.
4001 ; Do Write Subsystem Write Format to CLEAR loopback signals group 2.
4002 ; (the loopback signals have to be cleared here due to the flip-flops
4003 ; that are set on a 1 to 0 transition (IHER,IFMK,ICER))
4004 052314 005000 CLR R0 ;WRITE 0'S
4005 052316 042700 000200 BIC #WC.IFAD,R0 ;IFAD MUST ALWAYS =0
4006 052322 042700 000100 BIC #WC.IOTAD,R0 ;ITADO MUST ALWAYS =0
4007 052326 042700 000040 BIC #WC.IITAD,R0 ;ITAD1 MUST ALWAYS =0
4008 052332 004737 062122 JSR PC,T19WCTL ;SETUP PACKET FOR WRITE CONTROL
4009 052336 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4010 052342 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4011 052346 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4012 052352 FORCERROR 82$ ;@@DFORCE ERROR IF FORCER=1
4013 052366 103407 BCS 90$ ;BR IF CARRY SET (GOOD RETURN)
4014 052370 010001 MOV RO,R1 ;SAVE CONTENTS OF TSSR
4015 052372 NEXT.ERRNO
4016 052372 82$: ERRDF ERRNO,T197SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
; TRAP C$ERDF
; .WORD 805
; .WORD T197SSR
; .WORD PKTSSR
4017 052402 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4018 052406 90$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
; TRAP C$CLP1
4019 052410 005000 CLR R0 ;SET FORMAT DRIVE DATA=0
4020 052412 004737 062142 JSR PC,T19WFMT ;SETUP PACKET FOR WRITE FORMAT
4021 052416 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4022 052422 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4023 052426 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4024 052432 FORCERROR 102$ ;@@DFORCE ERROR IF FORCER=1
4025 052446 103407 BCS 110$ ;BR IF CARRY SET (GOOD RETURN)
4026 052450 010001 MOV RO,R1 ;SAVE CONTENTS OF TSSR
4027 052452 NEXT.ERRNO
4028 052452 102$: ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
; TRAP C$ERDF
; .WORD 806
; .WORD T198SSR
; .WORD PKTSSR
    
```

```

4029 052462 005237 002222          INC    FATFLG          ;SET FATAL ERROR FLAG
4030 052466          110$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      052466 104406          TRAP    C$CLP1
4031          :          Do a Write Subsystem Write Misc to Reset Tape Status F-FLOPS
4032 052470 004737 062000          JSR    PC,T19RSFIF    ;SETUP PKT FOR WRITE MISC Reset Tape Status F-FLOPS
4033 052474 012704 062460          MOV    #T19PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
4034 052500 010465 000000          MOV    R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
4035 052504 004737 016336          JSR    PC,CHKTSSR    ;WAIT FOR SSR TO SET
4036 052510          FORCERROR 122$          ;@@DFORCE ERROR IF FORCER=1
4037 052524 103407          BCS   130$          ;BR IF CARRY SET (GOOD RETURN)
4038 052526 010001          MOV    R0,R1        ;SAVE CONTENTS OF TSSR
4039 052530          NEXT.ERRNO
4040 052530          122$:  ERRDF  ERRNO,T192SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      052530 104455          TRAP    C$ERDF
      052532 001447          .WORD  807
      052534 060260          .WORD  T192SSR
      052536 012046          .WORD  PKTSSR
4041 052540 005237 002222          INC    FATFLG          ;SET FATAL ERROR FLAG
4042 052544          130$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      052544 104406          TRAP    C$CLP1
4043          :          Do a Write Subsystem READ STATUS
4044          :          If all Tape Status 2 (ICER,IFMK,IHER) flip-flop
4045          :          signals NOT=0 Then Print Error
4046 052546 004737 061760          JSR    PC,T19SRD     ;SETUP PACKET FOR READ STATUS
4047 052552 012704 062460          MOV    #T19PK2,R4    ;GET WRITE SUBSYSTEM COMMAND PACKET
4048 052556 010465 000000          MOV    R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
4049 052562 004737 016336          JSR    PC,CHKTSSR    ;WAIT FOR SSR TO SET
4050 052566          FORCERROR 132$          ;@@DFORCE ERROR IF FORCER=1
4051 052602 103407          BCS   140$          ;BR IF CARRY SET (GOOD RETURN)
4052 052604 010001          MOV    R0,R1        ;SAVE CONTENTS OF TSSR
4053 052606          NEXT.ERRNO
4054 052606          132$:  ERRDF  ERRNO,T193SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      052606 104455          TRAP    C$ERDF
      052610 001450          .WORD  808
      052612 060324          .WORD  T193SSR
      052614 012046          .WORD  PKTSSR
4055 052616 005237 002222          INC    FATFLG          ;SET FATAL ERROR FLAG
4056 052622          140$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      052622 104406          TRAP    C$CLP1
4057 052624 004737 062220          JSR    PC,T19SETEXP  ;SET WORDS 0-7 EXPD=RCV (NOT TESTING)
4058 052630 012701 060062          MOV    #T19EXSTA,R1  ;GET EXPECTED READ STATUS
4059 052634 012702 062352          MOV    #T19BFSTA,R2  ;GET RCV READ STATUS
4060 052640 011211          MOV    (R2),(R1)     ;SET EXPD WORD #8 = RCV TEMP
4061 052642 042711 002000          BIC    #S1.ICER,(R1) ;SET EXPD ICER =0
4062 052646 042711 001000          BIC    #S1.IFMK,(R1) ;SET EXPD IFMK =0
4063 052652 042711 000400          BIC    #S1.IHER,(R1) ;SET EXPD IHER =0
4064 052656 016261 000002 000002  MOV    2(R2),2(R1)   ;SET EXPD WORD #9 = RCV (NOT TESTING)
4065 052664 005000          CLR    R0           ;HIGH RCV ADDRESS FOR CKMSG2
4066 052666 012701 062332          MOV    #T19BFR,R1    ;LOW RCV ADDRESS FOR CKMSG2
4067 052672 012702 060042          MOV    #T19EXP,R2    ;EXPD ADDRESS
4068 052676 012703 000024          MOV    #20.,R3       ;NUMBER OF BYTES TO COMPARE
4069 052702 004737 011500          JSR    PC,CKMSG2     ;EXPD EQUAL RCV?
4070 052706          FORCERROR 152$,NOTSSR ;@@
4071 052716 103404          BCS   160$          ;BR IF YES
4072 052720          NEXT.ERRNO
4073 052720          152$:  ERRHRD ERRNO,T197CMP,MSGLOOP ;REPORT ERROR
      052720 104456          TRAP    C$ERHRD
  
```

```

052722 001451
052724 061263
052726 013064
4074 052730 160$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
052730 104406 ; TRAP C$CLP1
4075 ; REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
4076 052732 005037 057774 CLR T19PREV ;INIT 1-0 TRANSITION FLAG
4077 052736 012703 002752 MOV #TSTBLK,R3 ;GET FIRST PATTERN ADDRESS
4078 052742 012300 200$: MOV (R3)+,R0 ;GET A TEST PATTERN
4079 052744 010337 002316 MOV R3,TSTPTR ;SAVE POINTER INTO TSTBLK
4080 052750 042700 000200 BIC #WC.IFAD,R0 ;IFAD MUST ALWAYS =0
4081 052754 042700 000100 BIC #WC.IOTAD,R0 ;ITADO MUST ALWAYS =0
4082 052760 042700 000040 BIC #WC.IITAD,R0 ;ITAD1 MUST ALWAYS =0
4083 052764 010037 002312 MOV R0,DATA ;SET DATA PATTERN
4084 ; Do Write Subsystem Write Control to Drive loopback signals group 1.
4085 ;@@@ CALL T19CNVT TO SETUP WRITE CONTROL PATTERN
4086 052770 013700 002312 MOV DATA,R0 ;GET TEST PATTERN
4087 052774 004737 062244 JSR PC,T19CNVT ;CONVERT PATTERN TO CONTROL DRIVE MASK
4088 ;R0 CONTAINS WRITE CONTROL DATA HERE
4089 053000 004737 062122 JSR PC,T19WCTL ;SETUP PACKET FOR WRITE CONTROL
4090 053004 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4091 053010 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4092 053014 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4093 053020 FORCERROR 212$ ;@@@FORCE ERROR IF FORCER=1
4094 053034 103407 BCS 220$ ;BR IF CARRY SET (GOOD RETURN)
4095 053036 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4096 053040 NEXT.ERRNO
4097 053040 212$: ERRDF ERRNO,T197SSR,PKTSSR ;DEVICE FATAL SSR FAILED 'O SET
053040 104455 TRAP C$ERDF
053042 001452 .WORD 810
053044 060543 .WORD T197SSR
053046 012046 .WORD PKTSSR
4098 053050 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4099 053054 220$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
053054 104406 ; TRAP C$CLP1
4100 ; Do Write Subsystem Write Format to Drive loopback signals group 2.
4101 ;@@@ CALL T19CNVT TO SETUP WRITE CONTROL PATTERN
4102 MOV DATA,R0 ;GET TEST PATTERN
4103 053056 013700 002312 JSR PC,T19CNVT ;CONVERT PATTERN TO FORMAT DRIVE MASK
4104 053062 004737 062244 SWAB R0 ;WRITE FORMAT DATA RETURNED IN HIGH BYTE
4105 053066 000300 JSR PC,T19WFMT ;SETUP PACKET FOR WRITE FORMAT
4106 053070 004737 062142 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4107 053074 012704 062460 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4108 053100 010465 000000 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4109 053104 004737 016336 FORCERROR 232$ ;@@@FORCE ERROR IF FORCER=1
4110 053110 BCS 240$ ;BR IF CARRY SET (GOOD RETURN)
4111 053124 103407 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4112 053126 010001 NEXT.ERRNO
4113 053130 232$: ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
053130 104455 TRAP C$ERDF
053132 001453 .WORD 811
053134 060612 .WORD T198SSR
053136 012046 .WORD PKTSSR
4115 053140 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4116 053144 240$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
053144 104406 ; TRAP C$CLP1
    
```



```

4117          : Do a Write Subsystem READ STATUS
4118 053146   004737 061760   JSR    PC,T19SRD      ;SETUP PACKET FOR READ STATUS
4119 053152   012704 062460   MOV    #T19PK2,R4     ;GET WRITE SUBSYSTEM COMMAND PACKET
4120 053156   010465 000000   MOV    R4 `SDB(R5)    ;SET THE PACKET ADDRESS TO EXECUTE
4121 053162   004737 016336   JSR    PC `KTSSR      ;WAIT FOR SSR TO SET
4122 053166   FORCERROR 252$   ;@@DFORCE ERROR IF FORCER=1
4123 053202   103407   BCS    260$           ;BR IF CARRY SET (GOOD RETURN)
4124 053204   010001   MOV    R0,R1          ;SAVE CONTENTS OF TSSR
4125 053206   NEXT.ERRNO
4126 053206   252$:  ERRDF  ERRNO,T193SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD 812
                                .WORD T193SSR
                                .WORD PKTSSR
4127 053216   005237 002222   INC    FATFLG         ;SET FATAL ERROR FLAG
4128 053222   260$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
4129          : If loopback data NOT= data sent Then Print Error
4130 053224   004737 062220   JSR    PC,T19SETEXP   ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
4131 053230   012701 060062   MOV    #T19EXSTA,R1   ;GET EXPECTED READ STATUS
4132 053234   012702 062352   MOV    #T19BFSTA,R2   ;GET RECV READ STATUS
4133 053240   013711 002312   MOV    DATA,(R1)     ;SET EXPD WORD #8 TO TEST DATA FIRST
4134 053244   013700 057774   MOV    T19PREV,R0     ;GET PREVIOUS DATA PATTERN
4135 053250   013703 002312   MOV    DATA,R3       ;GET CURRENT PATTERN
4136 053254   012704 000400   MOV    #S1.IHER,R4    ;SETUP IHER EXPECTED
4137 053260   040411   BIC    R4,(R1)        ;SET EXPD IHER =0
4138 053262   030400   BIT    R4,R0          ;PREVIOUS =1?
4139 053264   001403   BEQ    275$           ;BR IF NO
4140 053266   030403   BIT    R4,R3          ;CURRENT =0?
4141 053270   001001   BNE    275$           ;BR IF NO
4142 053272   050411   BIS    R4,(R1)        ;SET EXPD IHER =1
4143 053274   012704 001000   275$:  MOV    #S1.IFMK,R4 ;SETUP IFMK EXPECTED
4144 053300   040411   BIC    R4,(R1)        ;SET EXPD IFMK =0
4145 053302   030400   BIT    R4,R0          ;PREVIOUS =1?
4146 053304   001403   BEQ    280$           ;BR IF NO
4147 053306   030403   BIT    R4,R3          ;CURRENT =0?
4148 053310   001001   BNE    280$           ;BR IF NO
4149 053312   050411   BIS    R4,(R1)        ;SET EXPD IFMK =1
4150 053314   012704 002000   280$:  MOV    #S1.ICER,R4  ;SETUP ICER EXPECTED
4151 053320   040411   BIC    R4,(R1)        ;SET EXPD ICER =0
4152 053322   030400   BIT    R4,R0          ;PREVIOUS =1?
4153 053324   001403   BEQ    285$           ;BR IF NO
4154 053326   030403   BIT    R4,R3          ;CURRENT =0?
4155 053330   001001   BNE    285$           ;BR IF NO
4156 053332   050411   BIS    R4,(R1)        ;SET EXPD ICER =1
4157 053334   011100   285$:  MOV    (R1),R0      ;GET EXPD WORD
4158          : If previous IIDENT=1 and current is IIDENT=1 then EXPD= 0 else 1
4159 053336   012704 004000   MOV    #S1.IIDENT,R4 ;IIDENT
4160 053342   050400   BIS    R4,R0          ;ASSUME EXPD=1
4161 053344   030437 057774   BIT    R4,T19PREV     ;PREVIOUS IIDENT=1?
4162 053350   001403   BEQ    288$           ;BR IF NO
4163 053352   030403   BIT    R4,R3          ;IS CURRENT IIDENT=1?
4164 053354   001401   BEQ    288$           ;BR IF NO
4165 053356   040400   BIC    R4,R0          ;SET EXPD=0
4166 053360   052700 040000   288$:  BIS    #S1.I2RES,R0   ;IRESV2 EXPD ALWAYS=1
4167 053364   052700 020000   BIS    #S1.I1RES,R0   ;IRESV1 EXPD ALWAYS=1
4168 053370   042700 100000   BIC    #S1.PARERR,R0  ;IGNORE PARERR
    
```

```

4169 053374 032712 100000          BIT      #S1.PARERR,(R2)          ;IS PARERR SET IN RECV?
4170 053400 001402                BEQ      290$                    ;BR IF NO
4171 053402 052700 100000          BIS      #S1.PARERR,R0          ;SET IN EXPD
4172 053406 010011                MOV      R0,(R1)                ;SETUP FINAL EXPD IN WORD #8
4173 053410 016261 000002 000002 290$: MOV      2(R2),2(R1)            ;SET EXPD WORD #9 = RECV (NOT TESTING)
4174 053416 005000                CLR      R0                      ;HIGH RECV ADDRESS FOR CKMSG2
4175 053420 012701 062332          MOV      #T19BFR,R1            ;LOW RECV ADDRESS FOR CKMSG2
4176 053424 012702 060042          MOV      #T19EXP,R2           ;EXPD ADDRESS
4177 053430 012703 000024          MOV      #20.,R3              ;NUMBER OF BYTES TO COMPARE
4178 053434 004737 011500          JSR      PC,CKMSG2            ;EXPD EQUAL RECV?
4179 053440                FORCERROR 302$,NOTSSR          ;@D
4180 053450 103404                BCS      310$                    ;BR IF YES
4181 053452                NEXT.ERRNO
4182 053452 302$: ERRHRD ERRNO,T198CMP,MSGLOOP ;REPORT ERROR
    053452 104456                TRAP      C$ERHRD
    053454 001455                .WORD    813
    053456 061351                .WORD    T198CMP
    053460 013064                .WORD    MSGLOOP
4183 053462 310$: CKLOOP                ;LOOP ON ERROR, IF FLAG SET
    053462 104406                TRAP      C$CLP1
4184                ; Do a Write Subsystem Write Misc to Reset Tape Status F-FLOPS
4185 053464 004737 062000          JSR      PC,T19RSFIF          ;SETUP PKT FOR WRITE MISC Reset STATUS
4186 053470 012704 062460          MOV      #T19PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
4187 053474 010465 000000          MOV      R4,TSDB(R5)         ;SET THE PACKET ADDRESS TO EXECUTE
4188 053500 004737 016336          JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
4189 053504                FORCERROR 322$                ;@DFORCE ERROR IF FORCER=1
4190 053520 103407                BCS      330$                    ;BR IF CARRY SET (GOOD RETURN)
4191 053522 010001                MOV      R0,R1                ;SAVE CONTENTS OF TSSR
4192 053524                NEXT.ERRNO
4193 053524 322$: ERRDF ERRNO,T192SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
    053524 104455                TRAP      C$ERDF
    053526 001456                .WORD    814
    053530 060260                .WORD    T192SSR
    053532 012046                .WORD    PKTSSR
4194 053534 005237 002222          INC      FATFLG                ;SET FATAL ERROR FLAG
4195 053540 330$: CKLOOP                ;LOOP ON ERROR, IF FLAG SET
    053540 104406                TRAP      C$CLP1
4196                ; Do a Write Subsystem READ STATUS
4197 053542 004737 061760          JSR      PC,T19SRD            ;SETUP PACKET FOR READ STATUS
4198 053546 012704 062460          MOV      #T19PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
4199 053552 010465 000000          MOV      R4,TSDB(R5)         ;SET THE PACKET ADDRESS TO EXECUTE
4200 053556 004737 016336          JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
4201 053562                FORCERROR 342$                ;@DFORCE ERROR IF FORCER=1
4202 053576 103407                BCS      350$                    ;BR IF CARRY SET (GOOD RETURN)
4203 053600 010001                MOV      R0,R1                ;SAVE CONTENTS OF TSSR
4204 053602                NEXT.ERRNO
4205 053602 342$: ERRDF ERRNO,T193SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
    053602 104455                TRAP      C$ERDF
    053604 001457                .WORD    815
    053606 060324                .WORD    T193SSR
    053610 012046                .WORD    PKTSSR
4206 053612 005237 002222          INC      FATFLG                ;SET FATAL ERRJR FLAG
4207 053616 350$: CKLOOP                ;LOOP ON ERROR, IF FLAG SET
    053616 104406                TRAP      C$CLP1
4208 053620 004737 062220          JSR      PC,T19SETEXP        ;SET WORDS (-7 EXPD=RECV (NOT TESTING)
4209 053624 012701 060062          MOV      #T19EXSTA,R1        ;GET EXPECTED READ STATUS
4210 053630 012702 062352          MOV      #T19BFSTA,R2        ;GET RECV READ STATUS
    
```

```

4211 053634 011211          MOV      (R2),(R1)          ;SET EXPD WORD #8 = RECV TEMP
4212 053636 042711 002000  BIC      #S1.ICER,(R1)     ;SET EXPD ICER =0
4213 053642 042711 001000  BIC      #S1.IFMK,(R1)    ;SET EXPD IFMK =0
4214 053646 042711 000400  BIC      #S1.IHER,(R1)    ;SET EXPD IHER =0
4215 053652 016261 000002 000002  MOV      2(R2),2(R1)      ;SET EXPD WORD #9 = RECV (NOT TESTING)
4216 053660 005000          CLR      R0                ;HIGH RECV ADDRESS FOR CKMSG2
4217 053662 012701 062332  MOV      #T19BFR,R1       ;LOW RECV ADDRESS FOR CKMSG2
4218 053666 012702 060042  MOV      #T19EXP,R2       ;EXPD ADDRESS
4219 053672 012703 000024  MOV      #20.,R3          ;NUMBER OF BYTES TO COMPARE
4220 053676 004737 011500  JSR      PC,CKMSG2        ;EXPD EQUAL RECV?
4221 053702          FORCERROR 362$,NOTSSR      ;@@D
4222 053712 103404          BCS     370$              ;BR IF YES
4223 053714          NEXT.ERRNO
4224 053714          ERRHRD  ERRNO,T197CMP,MSGSTAT ;REPORT ERROR
      053714 104456          TRAP   C$ERHRD
      053716 001460          .WORD  816
      053720 061263          .WORD  T197CMP
      053722 012350          .WORD  MSGSTAT
4225 053724          370$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      053724 104406          TRAP   C$CLP1
4226          4227 053726 013737 002312 057774  MOV      DATA,T19PREV    ;SETUP PREVIOUS DATA FOR EXPD CALC.
4228 053734 013703 002316          MOV      TSTPTR,R3        ;RESTORE CURRENT TSTBLK POINTER
4229 053740 020327 003062          CMP      R3,#TBLEND      ;END OF TSTBLK?
4230 053744 103002          BHIS   400$              ;BR IF YES
4231 053746 000137 052742          JMP      200$              ;DO NEXT TSTBLK PATTERN
4232 053752          400$:
4233          ENDSUB          ;////////////////// END SUBTEST ////////////////////
4234 053752          L10067:
      053752 104403          TRAP   C$ESUB
4235          4236 053754 005737 002222          TST     FATFLG            ;ANY FATAL ERRORS ?
4237 053760 001402          BEQ    460$              ;BRANCH IF NOT
4238 053762 004737 017202          JSR    PC,CKDROP         ;TRY TO DROP THE UNIT
4239 053766          460$:
4240
4241
4242
4243
4244
    
```

```

4246 .SBTTL TEST 8: SUBTEST 2: LOOPBACK READ/WRITE SIGNALS TEST
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282 053766
      053766
      053766 104402
4283
4284 053770
4285 053770 004737 015774
4286 053774 103405
4287 053776 010001
4288 054000
      054000 104455
      054002 001460
      054004 003652
      054006 012034
4289
4290 054010 005037 002222
4291 054014 012704 062310
4292 054020 004737 010662
4293 054024
4294 054040 103407
4295 054042 010001
4296 054044
    
```

```

**
TEST 8: SUBTEST 2:
SUBTEST DESCRIPTION:
    This subtest verifies the Read/Write data loopback path.
    The Read/Write data signals are IR<7:0> and IW<7:0>
    respectively.
TEST STEPS:
REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
BEGIN
    Write to TSSR register to soft initialize the controller
    Do WRITE CHARACTERISTICS to check for Extended Features Switch
    If Extended Features Hardware Switch Clear then:
        Do Write Subsystem Write Miscellaneous to Set Extended Features.
    Do WRITE CHARACTERISTICS to select reserved unit 7 and setup BUFFER
    Do a Write Subsystem WRITE NPR to set tape direction out and Loopback
    Do a WRITE NPR to set loopback and tape direction OUT
    Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT
    Do a READ FIFO with tape direction OUT to load tape out write latch
    Do a WRITE NPR to set loopback and tape direction IN
    Do a WRITE FIFO with byte count equal to 1 and Tape direction IN
    to strobe loopback data into FIFO.
    Do a READ FIFO with tape direction IN to read data
    If Data read from FIFO NOT= to Data sent Then Print Error
    Do a Write Subsystem READ STATUS
    If Input Ready NOT=1 Then Print Error
    If Output Ready NOT=0 Then Print Error
    If Data In Miss NOT=0 Then Print Error
END
--
BGNSUB                :////////// BEGIN SUBTEST //////////
                        T8.2:
                        TRAP    C$BSUB
:
5$: Write to TSSR register to soft initialize the controller
    JSR    PC,SOFINIT          ;WRITE TO TSSR TO SOFT INITIALIZE
    BCS    10$                 ;BR IF SOFT INIT OKAY
    MOV    R0,R1               ;SAVE CONTENTS OF TSSR
    ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
                                TRAP    C$ERDF
                                .WORD   816
                                .WORD   SFIERR
                                .WORD   SFIMSG
:
10$: Do WRITE CHARACTERISTICS to check for Extended Features Switch
    CLR    FATFLG              ;CLEAR FATAL ERROR FLAG
    MOV    #T19PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
    JSR    PC,WRTCHR          ;DO WRITE CHARACTERISTICS COMMAND
    FORCERROR 12$             ;@@DFORCE ERROR IF FORCER=1
    BCS    15$                 ;BR IF CARRY SET (GOOD RETURN)
    MOV    R0,R1               ;SAVE CONTENTS OF TSSR
    NEXT.ERRNO
    
```

```

4297 054044          12$:  ERRDF  ERRNO,T19SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
      054044 104455                                     TRAP  C$ERDF
      054046 001461                                     .WORD 817
      054050 060223                                     .WORD T19SSR
      054052 012046                                     .WORD PKTSSR
4298 054054 005237 002222          INC  FATFLG      ;SET FATAL ERROR FLAG
4299 054060          15$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      054060 104406                                     TRAP  C$CLP1
4300          ;      If Extended Features Hardware Switch Clear then:
4301          ;      Do Write Subsystem Write Miscellaneous to Set Extended Features.
4302 054062 012701 062332          MOV  #T19BFR,R1      ;MESSAGE BUFFER ADDRESS
4303 054066 032761 000200 000012  BIT  #X2.EXTF,XST2(R1) ;EXTENDED FEATURES SWITCH SET?
4304 054074 001026          BNE  30$          ;BR IF YES
4305 054076 004737 062162          JSR  PC,T19SEXT     ;SETUP PACKET FOR WRITE MISC INVERT
4306 054102 012704 062460          MOV  #T19PK2,R4     ;GET WRITE SUBSYSTEM COMMAND PACKET
4307 054106 010465 000000          MOV  R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
4308 054112 004737 016336          JSR  PC,CHKTSSR    ;WAIT FOR SSR TO SET
4309 054116          FORCERROR 22$          ;@@DFORCE ENROR IF FORCER=1
4310 054132 103407          BCS  30$          ;BR IF CARRY SET (GOOD RETURN)
4311 054134 010001          MOV  R0,R1        ;SAVE CONTENTS OF TSSR
4312 054136          NEXT.ERRNO
4313 054136          22$:  ERRDF  ERRNO,T192SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      054136 104455                                     TRAP  C$ERDF
      054140 001462                                     .WORD 818
      054142 060260                                     .WORD T192SSR
      054144 012046                                     .WORD PKTSSR
4314 054146 005237 002222          INC  FATFLG      ;SET FATAL ERROR FLAG
4315 054152          30$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      054152 104406                                     TRAP  C$CLP1
4316          ;      Do WRITE CHARACTERISTICS to select reserved unit 7
4317 054154 012704 062310          MOV  #T19PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
4318 054160 004737 010662          JSR  PC,WRTCHR     ;DO WRITE CHARACTERISTICS COMMAND
4319 054164          FORCERROR 42$          ;@@DFORCE ERROR IF FORCER=1
4320 054200 103407          BCS  50$          ;BR IF CARRY SET (GOOD RETURN)
4321 054202 010001          MOV  R0,R1        ;SAVE CONTENTS OF TSSR
4322 054204          NEXT.ERRNO
4323 054204          42$:  ERRDF  ERRNO,T19SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      054204 104455                                     TRAP  C$ERDF
      054206 001463                                     .WORD 819
      054210 060223                                     .WORD T19SSR
      054212 012046                                     .WORD PKTSSR
4324 054214 005237 002222          INC  FATFLG      ;SET FATAL ERROR FLAG
4325 054220          50$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      054220 104406                                     TRAP  C$CLP1
4326
4327
4328          ; REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
4329 054222 012703 002752          MOV  #TSTBLK,R3    ;GET FIRST PATTERN ADDRESS
4330 054226 012337 002312 100$:  MOV  (R3)+,DATA      ;GET A TEST PATTERN
4331 054232 042737 177400 002312  BIC  #^C<377>,DATA ;DATA IS BYTE
4332 054240 010337 002316          MOV  R3,TSTPTR    ;SETUP CURRENT TSTBLK POINTER
4333          ;      Do a WRITE NPR to set loopback and tape direction OUT
4334 054244 012700 000100          MOV  #NP.OUT,R0   ;SET TAPE DIRECTION OUT
4335 054250 052700 000040          BIS  #NP.LOOP,R0  ;SET LOOPBACK
4336 054254 004737 062022          JSR  PC,T19SNPR   ;SETUP T19PK2 FOR WRITE NPR
4337 054260 012704 062460          MOV  #T19PK2,R4   ;GET WRITE SUBSYSTEM COMMAND PACKET
4338 054264 010465 000000          MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
    
```

```

4339 054270 004737 016336      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
4340 054274                      FORCERROR      102$      ;@DFORCE ERROR IF FORCER=1
4341 054310 103407                      BCS      105$      ;BR IF CARRY SET (GOOD RETURN)
4342 054312 010001                      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
4343 054314                      NEXT.ERRNO
4344 054314 102$:      ERRDF      ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD      820
                                .WORD      T194SSR
                                .WORD      PKTSSR
                                054314 104455
                                054316 001464
                                054320 060371
                                054322 012046
4345 054324 005237 002222      INC      FATFLG      ;SET FATAL ERROR FLAG
4346 054330 105$:      CKLOOP                      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
4347 :      Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT
4348 054332 012700 000001      MOV      #1,R0      ;WRITE 1 BYTE
4349 054336 012701 002312      MOV      #DATA,R1   ;FIFO WRITE DATA ADDRESS
4350 054342 004737 062066      JSR      PC,T19WFIF  ;SETUP T19PK2 FOR WRITE FIFO
4351 054346 012704 062460      MOV      #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4352 054352 010465 000000      MOV      R4,TSDB(R5);SET THE PACKET ADDRESS TO EXECUTE
4353 054356 004737 016336      JSR      PC,CHKTSSR ;WAIT FOR SSR TO SET
4354 054362                      FORCERROR      107$      ;@DFORCE ERROR IF FORCER=1
4355 054376 103407                      BCS      110$      ;BR IF CARRY SET (GOOD RETURN)
4356 054400 010001                      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
4357 054402                      NEXT.ERRNO
4358 054402 107$:      ERRDF      ERRNO,T195SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD      821
                                .WORD      T195SSR
                                .WORD      PKTSSR
                                054402 104455
                                054404 001465
                                054406 060434
                                054410 012046
4359 054412 005237 002222      INC      FATFLG      ;SET FATAL ERROR FLAG
4360 054416 110$:      CKLOOP                      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
4361 :      Do a READ FIFO with tape direction OUT to load tape out write latch
4362 054420 012700 000001      MOV      #1,R0      ;SET READ BYTE COUNT
4363 054424 004737 062046      JSR      PC,T19RFIF  ;SETUP T19PK2 FOR READ FIFO
4364 054430 012704 062460      MOV      #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4365 054434 010465 000000      MOV      R4,TSDB(R5);SET THE PACKET ADDRESS TO EXECUTE
4366 054440 004737 016336      JSR      PC,CHKTSSR ;WAIT FOR SSR TO SET
4367 054444                      FORCERROR      122$      ;@DFORCE ERROR IF FORCER=1
4368 054460 103407                      BCS      130$      ;BR IF CARRY SET (GOOD RETURN)
4369 054462 010001                      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
4370 054464                      NEXT.ERRNO
4371 054464 122$:      ERRDF      ERRNO,T196SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD      822
                                .WORD      T196SSR
                                .WORD      PKTSSR
                                054464 104455
                                054466 001466
                                054470 060500
                                054472 012046
4372 054474 005237 002222      INC      FATFLG      ;SET FATAL ERROR FLAG
4373 054500 130$:      CKLOOP                      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
4374 :      Do a WRITE NPR to set loopback and tape direction IN
4375 054502 005000      CLR      R0      ;CLR NP.OUT TO SET TAPE DIRECTION IN
4376 054504 052700 000040      BIS      #NP.LOOP,R0 ;SET LOOPBACK
4377 054510 004737 062022      JSR      PC,T19SNPR  ;SETUP T19PK2 FOR WRITE NPR
4378 054514 012704 062460      MOV      #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4379 054520 010465 000000      MOV      R4,TSDB(R5);SET THE PACKET ADDRESS TO EXECUTE
4380 054524 004737 016336      JSR      PC,CHKTSSR ;WAIT FOR SSR TO SET
    
```

```

4381 054530          FORCERROR          142$          ;@DFORCE ERROR IF FORCER=1
4382 054544 103407  BCS          150$          ;BR IF CARRY SET (GOOD RETURN)
4383 054546 010001  MOV          R0,R1          ;SAVE CONTENTS OF TSSR
4384 054550          NEXT.ERRNO
4385 054550 142$:  ERRDF  ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
          054550 104455          TRAP  C$ERDF
          054552 001467          .WORD  823
          054554 060371          .WORD  T194SSR
          054556 012046          .WORD  PKTSSR
4386 054560 005237 002222  INC          FATFLG          ;SET FATAL ERROR FLAG
4387 054564 150$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
          054564 104406          TRAP  C$CLP1
4388          ; Do a WRITE FIFO with byte count equal to 1 and Tape direction IN
4389 054566 012700 000001  MOV          #1,R0          ;WRITE 1 BYTE
4390 054572 012701 002312  MOV          #DATA,R1       ;FIFO WRITE DATA ADDRESS
4391 054576 004737 062066  JSR          PC,T19WFIF     ;SETUP T19PK2 FOR WRITE FIFO
4392 054602 012704 062460  MOV          #T19PK2,R4     ;GET WRITE SUBSYSTEM COMMAND PACKET
4393 054606 010465 000000  MOV          R4,TSDB(R5)    ;SET THE PACKET ADDRESS TO EXECUTE
4394 054612 004737 016336  JSR          PC,CHKTSSR     ;WAIT FOR SSR TO SET
4395 054616          FORCERROR          162$          ;@DFORCE ERROR IF FORCER=1
4396 054632 103407  BCS          170$          ;BR IF CARRY SET (GOOD RETURN)
4397 054634 010001  MOV          R0,R1          ;SAVE CONTENTS OF TSSR
4398 054636          NEXT.ERRNO
4399 054636 162$:  ERRDF  ERRNO,T195SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
          054636 104455          TRAP  C$ERDF
          054640 001470          .WORD  824
          054642 060434          .WORD  T195SSR
          054644 012046          .WORD  PKTSSR
4400 054646 005237 002222  INC          FATFLG          ;SET FATAL ERROR FLAG
4401 054652 170$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
          054652 104406          TRAP  C$CLP1
4402          ; Do a READ FIFO with tape direction IN to read data
4403          ; If Data read from FIFO NOT= to Data sent Then Print Error
4404 054654 012700 000001  MOV          #1,R0          ;SET READ BYTE COUNT
4405 054660 004737 062046  JSR          PC,T19RFIF     ;SETUP T19PK2 FOR READ FIFO
4406 054664 012704 062460  MOV          #T19PK2,R4     ;GET WRITE SUBSYSTEM COMMAND PACKET
4407 054670 010465 000000  MOV          R4,TSDB(R5)    ;SET THE PACKET ADDRESS TO EXECUTE
4408 054674 004737 016336  JSR          PC,CHKTSSR     ;WAIT FOR SSR TO SET
4409 054700          FORCERROR          182$          ;@DFORCE ERROR IF FORCER=1
4410 054714 103407  BCS          190$          ;BR IF CARRY SET (GOOD RETURN)
4411 054716 010001  MOV          R0,R1          ;SAVE CONTENTS OF TSSR
4412 054720          NEXT.ERRNO
4413 054720 182$:  ERRDF  ERRNO,T196SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
          054720 104455          TRAP  C$ERDF
          054722 001471          .WORD  825
          054724 060500          .WORD  T196SSR
          054726 012046          .WORD  PKTSSR
4414 054730 005237 002222  INC          FATFLG          ;SET FATAL ERROR FLAG
4415 054734 190$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
          054734 104406          TRAP  C$CLP1
4416 054736 004737 062220  JSR          PC,T19SETEXP   ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
4417 054742 012701 060062  MOV          #T19EXSTA,R1   ;GET EXPECTED READ STATUS
4418 054746 012702 062352  MOV          #T19BFSTA,R2   ;GET RECV READ STATUS
4419 054752 013711 002312  MOV          DATA,(R1)     ;SET EXPD WORD #8 = DATA
4420 054756 016261 000002 000002  MOV          2(R2),2(R1)    ;SET EXPD WORD #9 = RECV (NOT TESTING)
4421 054764 005000  CLR          R0          ;HIGH RECV ADDRESS FOR CKMSG2
4422 054766 012701 062332  MOV          #T19BFR,R1     ;LOW RECV ADDRESS FOR CKMSG2
    
```





```
4465
4466
4467 055212          FORCEEXIT          255$
4468 055222 013703 002316  MOV      TSTPTR,R3
4469 055226 020327 003062  CMP      R3,#TBLEND
4470 055232 103002          BHS      255$
4471 055234 000137 054226  JMP      100$
4472 055240          255$:
4473
4474 055240          ENDSUB
      055240
      055240 104403
4475
4476 055242 005737 002222  TST     FATFLG
4477 055246 001402          BEQ     260$
4478 055250 004737 017202  JSR     PC,CKDROP
4479 055254          260$:
4480
4481
```

```
:@D
:RESTORE CURRENT TSTBLK POINTER
:END OF TSTBLK?
:BR IF YES
:DO ANOTHER TSTBLK PATTERN

:////////// END SUBTEST //////////
L10070:
TRAP C$ESUB

:ANY FATAL ERRORS ?
:BRANCH IF NOT
:TRY TO DROP THE UNIT
```

4483  
4484  
4485  
4486  
4487  
4488  
4489  
4490  
4491  
4492  
4493  
4494  
4495  
4496  
4497  
4498  
4499  
4500  
4501  
4502  
4503  
4504  
4505  
4506  
4507  
4508  
4509  
4510  
4511  
4512  
4513  
4514  
4515  
4516  
4517  
4518  
4519  
4520  
4521  
4522  
4523  
4524  
4525  
4526  
4527  
4528  
4529  
4530  
4531  
4532  
4533

.SBTTL TEST 8: SUBTEST 3: LOOPBACK WRITE STROBE TEST

++  
: TEST 8: SUBTEST 3:

: SUBTEST DESCRIPTION:

: This subtest verifies the Write Strobe loopback path  
 : can strobe data from the FIFO to the Data lines.  
 : The signal IRESV3 drives IWSTR (write strobe) to write  
 : data from the FIFO to the tape data out latch.

: TEST STEPS:

: REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE

: BEGIN

: Write to TSSR register to soft initialize the controller  
 : Do WRITE CHARACTERISTICS to check for Extended Features Switch  
 : If Extended Features Hardware Switch Clear then:  
 : Do Write Subsystem Write Miscellaneous to Set Extended Features.  
 : Do WRITE CHARACTERISTICS to select reserved unit 7 and setup BUFFER  
 : Do a Write Subsystem WRITE NPR to set tape direction out and Loopback  
 : Do a WRITE NPR to set loopback and tape direction OUT  
 : Do a WRITE FORMAT to set IRESV3=>IWSTR = 1  
 : Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT  
 : Do a WRITE FORMAT to set IRESV3=>IWSTR = 0 to load write data latch  
 : Do a WRITE FORMAT to set IRESV3=>IWSTR = 1  
 : Do a WRITE NPR to set loopback and tape direction IN  
 : Do a WRITE FIFO with byte count equal to 1 and Tape direction IN  
 : to strobe loopback data into FIFO.  
 : Do a READ FIFO with tape direction IN to read data  
 : If Data read from FIFO NOT= to Data sent Then Print Error

: END

--- BGNSUB ;//////////////// BEGIN SUBTEST //////////////////  
 T8.3:

Write to TSSR register to soft initialize the controller

TRAP CSBSUB  
 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT  
 TRAP CSERDF  
 .WORD 828  
 .WORD SFIERR  
 .WORD SFIMSG

Do WRITE CHARACTERISTICS to check for Extended Features Switch

10\$: CLR FATFLG ;CLEAR FATAL ERROR FLAG  
 MOV #T19PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET  
 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND  
 FORCERROR 12\$ ;@@DFORCE ERROR IF FORCER=1  
 BCS 15\$ ;BR IF CARRY SET (GOOD RETURN)  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR

12\$: NEXT.ERRNO  
 ERRDF ERRNO,T19SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET

```

055332 104455                                TRAP  C$ERDF
055334 001475                                .WORD 829
055336 060223                                .WORD T19SSR
055340 012046                                .WORD PKTSSR
4534 055342 005237 002222                INC  FATFLG                ;SET FATAL ERROR FLAG
4535 055346                                CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
055346 104406                                TRAP  C$CLP1

4536 ; If Extended Features Hardware Switch Clear then:
4537 ; Do Write Subsystem Write Miscellaneous to Set Extended Features.
4538 055350 012701 062332                MOV  #T19BFR,R1           ;MESSAGE BUFFER ADDRESS
4539 055354 032761 000200 000012        BIT  #X2.EXTF,XST2(R1)    ;EXTENDED FEATURES SWITCH SET?
4540 055362 001026                                BNE  30$                 ;BR IF YES
4541 055364 004737 062162                JSR  PC,T19SEXT           ;SETUP PACKET FOR WRITE MISC INVERT
4542 055370 012704 062460                MOV  #T19PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
4543 055374 010465 000000                MOV  R4,TSDB(R5)         ;SET THE PACKET ADDRESS TO EXECUTE
4544 055400 004737 016336                JSR  PC,CHKTSSR          ;WAIT FOR SSR TO SET
4545 055404                                FORCERROR 22$            ;@@DFORCE ERROR IF FORCER=1
4546 055420 103407                                BCS  30$                 ;BR IF CARRY SET (GOOD RETURN)
4547 055422 010001                                MOV  R0,R1               ;SAVE CONTENTS OF TSSR
4548 055424                                NEXT.ERRNO
4549 055424 22$: ERRDF  ERRNO,T192SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
055424 104455                                TRAP  C$ERDF
055426 001476                                .WORD 830
055430 060260                                .WORD T192SSR
055432 012046                                .WORD PKTSSR
4550 055434 005237 002222                INC  FATFLG                ;SET FATAL ERROR FLAG
4551 055440                                CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
055440 104406                                TRAP  C$CLP1

4552 ; Do WRITE CHARACTERISTICS to select reserved unit 7
4553 055442 012704 062310                MOV  #T19PACKET,R4       ;GET THE ADDRESS OF COMMAND PACKET
4554 055446 004737 010662                JSR  PC,WRTCHR           ;DO WRITE CHARACTERISTICS COMMAND
4555 055452                                FORCERROR 42$            ;@@DFORCE ERROR IF FORCER=1
4556 055466 103407                                BCS  50$                 ;BR IF CARRY SET (GOOD RETURN)
4557 055470 010001                                MOV  R0,R1               ;SAVE CONTENTS OF TSSR
4558 055472                                NEXT.ERRNO
4559 055472 42$: ERRDF  ERRNO,T19SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
055472 104455                                TRAP  C$ERDF
055474 001477                                .WORD 831
055476 060223                                .WORD T19SSR
055500 012046                                .WORD PKTSSR
4560 055502 005237 002222                INC  FATFLG                ;SET FATAL ERROR FLAG
4561 055506                                CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
055506 104406                                TRAP  C$CLP1

4562 ; REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
4563 ;
4564 055510 012703 002752                MOV  #TSTBLK,R3          ;GET FIRST PATTERN ADDRESS
4565 055514 012337 002312 100$: MOV  (R3)+,DATA          ;GET A TEST PATTERN
4566 055520 042737 177400 002312        BIC  #^C<377>,DATA       ;DATA IS BYTE
4567 055526 010337 002316                MOV  R3,TSTPTR          ;SETUP CURRENT TSTBLK POINTER
4568 ; Do a WRITE NPR to set loopback and tape direction OUT
4569 055532 012700 000100                MOV  #NP.OUT,R0         ;SET TAPE DIRECTION OUT
4570 055536 052700 000040                BIS  #NP.LOOP,R0        ;SET LOOPBACK
4571 055542 004737 062022                JSR  PC,T19SNPR         ;SETUP T19PK2 FOR WRITE NPR
4572 055546 012704 062460                MOV  #T19PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
4573 055552 010465 000000                MOV  R4,TSDB(R5)         ;SET THE PACKET ADDRESS TO EXECUTE
4574 055556 004737 016336                JSR  PC,CHKTSSR          ;WAIT FOR SSR TO SET
4575 055562                                FORCERROR 102$          ;@@DFORCE ERROR IF FORCER=1
    
```

```

4576 055576 103407          BCS      105$          :BR IF CARRY SET (GOOD RETURN)
4577 055600 010001          MOV      R0,R1        :SAVE CONTENTS OF TSSR
4578 055602          NEXT.ERRNO
4579 055602          102$:  ERRDF  ERRNO,T194SSR,PKTSSR :DEVICE FATAL SSR FAILED TO SET
      055602 104455          TRAP      C$ERDF
      055604 001500          .WORD     832
      055606 060371          .WORD     T194SSR
      055610 012046          .WORD     PKTSSR
4580 055612 005237 002222          INC      FATFLG      :SET FATAL ERROR FLAG
4581 055616          105$:  CKLOOP          :LOOP ON ERROR, IF FLAG SET
      055616 104406          TRAP      C$CLP1
4582          :          Do a WRITE FORMAT to set IRESV3==>IWSTR = 1
4583 055620 012700 000002          MOV      #WF.I3RES,R0 :IRESV3==>IWSTR=1
4584 055624 004737 062142          JSR      PC,T19WFMT   :SETUP T9PK2 FOR WRITE FORMAT
4585 055630 012704 062460          MOV      #T19PK2,R4  :GET WRITE SUBSYSTEM COMMAND PACKET
4586 055634 010465 000000          MOV      R4,TSDB(R5) :SET THE PACKET ADDRESS TO EXECUTE
4587 055640 004737 016336          JSR      PC,CHKTSSR   :WAIT FOR SSR TO SET
4588 055644          FORCERROR 112$      :@@DFORCE ERROR IF FORCER=1
4589 055660 103407          BCS      120$          :BR IF CARRY SET (GOOD RETURN)
4590 055662 010001          MOV      R0,R1        :SAVE CONTENTS OF TSSR
4591 055664          NEXT.ERRNO
4592 055664          112$:  ERRDF  ERRNO,T198SSR,PKTSSR :DEVICE FATAL SSR FAILED TO SET
      055664 104455          TRAP      C$ERDF
      055666 001501          .WORD     833
      055670 060612          .WORD     T198SSR
      055672 012046          .WORD     PKTSSR
4593 055674 005237 002222          INC      FATFLG      :SET FATAL ERROR FLAG
4594 055700          120$:  CKLOOP          :LOOP ON ERROR, IF FLAG SET
      055700 104406          TRAP      C$CLP1
4595          :          Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT
4596 055702 012700 000001          MOV      #1,R0        :WRITE 1 BYTE
4597 055706 012701 002312          MOV      #DATA,R1     :FIFO WRITE DATA ADDRESS
4598 055712 004737 062066          JSR      PC,T19WFIF   :SETUP T19PK2 FOR WRITE FIFO
4599 055716 012704 062460          MOV      #T19PK2,R4  :GET WRITE SUBSYSTEM COMMAND PACKET
4600 055722 010465 000000          MOV      R4,TSDB(R5) :SET THE PACKET ADDRESS TO EXECUTE
4601 055726 004737 016336          JSR      PC,CHKTSSR   :WAIT FOR SSR TO SET
4602 055732          FORCERROR 132$      :@@DFORCE ERROR IF FORCER=1
4603 055746 103407          BCS      140$          :BR IF CARRY SET (GOOD RETURN)
4604 055750 010001          MOV      R0,R1        :SAVE CONTENTS OF TSSR
4605 055752          NEXT.ERRNO
4606 055752          132$:  ERRDF  ERRNO,T195SSR,PKTSSR :DEVICE FATAL SSR FAILED TO SET
      055752 104455          TRAP      C$ERDF
      055754 001502          .WORD     834
      055756 060434          .WORD     T195SSR
      055760 012046          .WORD     PKTSSR
4607 055762 005237 002222          INC      FATFLG      :SET FATAL ERROR FLAG
4608 055766          140$:  CKLOOP          :LOOP ON ERROR, IF FLAG SET
      055766 104406          TRAP      C$CLP1
4609          :          Do a WRITE FORMAT to set IRESV3==>IWSTR = 0
4610 055770 005000          CLR      R0           :SET IRESV3==>IWSTR=0
4611 055772 004737 062142          JSR      PC,T19WFMT   :SETUP T9PK2 FOR WRITE FORMAT
4612 055776 012704 062460          MOV      #T19PK2,R4  :GET WRITE SUBSYSTEM COMMAND PACKET
4613 056002 010465 000000          MOV      R4,TSDB(R5) :SET THE PACKET ADDRESS TO EXECUTE
4614 056006 004737 016336          JSR      PC,CHKTSSR   :WAIT FOR SSR TO SET
4615 056012          FORCERROR 152$      :@@DFORCE ERROR IF FORCER=1
4616 056026 103407          BCS      160$          :BR IF CARRY SET (GOOD RETURN)
4617 056030 010001          MOV      R0,R1        :SAVE CONTENTS OF TSSR
    
```

```

4618 056032
4619 056032 152$: NEXT.ERRNO
      056032 104455 ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      056034 001503 TRAP C$ERDF
      056036 060612 .WORD 835
      056040 012046 .WORD T198SSR
      056042 005237 002222 .WORD PKTSSR
4620 056042 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4621 056046 104406 160$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056046 104406 TRAP C$CLP1
4622 : Do a WRITE FORMAT to set IRESV3==>IWSTR = 1
4623 056050 012700 000002 MOV #WF.I3RES,R0 ;IRESV3==>IWSTR=1
4624 056054 004737 062142 JSR PC,T19WFMT ;SETUP T9PK2 FOR WRITE FORMAT
4625 056060 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4626 056064 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4627 056070 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4628 056074 FORCERROR 172$ ;@@DFORCE ERROR IF FORCER=1
4629 056110 103407 BCS 180$ ;BR IF CARRY SET (GOOD RETURN)
4630 056112 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4631 056114
4632 056114 172$: ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      056114 104455 TRAP C$ERDF
      056116 001504 .WORD 836
      056120 060612 .WORD T198SSR
      056122 012046 .WORD PKTSSR
4633 056124 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4634 056130 104406 180$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056130 104406 TRAP C$CLP1
4635 : Do a WRITE NPR to set loopback and tape direction IN
4636 : CLR R0 ;CLR NP.OUT TO SET TAPE DIRECTION IN
4637 056132 005000 BIS #NP.LOOP,R0 ;SET LOOPBACK
4638 056134 052700 000040 JSR PC,T19SNPR ;SETUP T19PK2 FOR WRITE NPR
4639 056140 004737 062022 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4640 056144 012704 062460 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4641 056150 010465 000000 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4642 056154 004737 016336 FORCERROR 182$ ;@@DFORCE ERROR IF FORCER=1
4643 056160 BCS 190$ ;BR IF CARRY SET (GOOD RETURN)
4644 056174 103407 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4645 056176 010001
4646 056200
4647 056200 182$: ERRDF ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      056200 104455 TRAP C$ERDF
      056202 001505 .WORD 837
      056204 060371 .WORD T194SSR
      056206 012046 .WORD PKTSSR
4648 056210 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4649 056214 104406 190$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056214 104406 TRAP C$CLP1
4650 : Do a WRITE FIFO with byte count equal to 1 and Tape direction IN
4651 056216 012700 000001 MOV #1,R0 ;WRITE 1 BYTE
4652 056222 012701 002312 MOV #DATA,R1 ;FIFO WRITE DATA ADDRESS
4653 056226 004737 062066 JSR PC,T19WFIF ;SETUP T19PK2 FOR WRITE FIFO
4654 056232 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4655 056236 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4656 056242 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4657 056246 FORCERROR 202$ ;@@DFORCE ERROR IF FORCER=1
4658 056262 103407 BCS 210$ ;BR IF CARRY SET (GOOD RETURN)
4659 056264 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
    
```

```

4660 056266
4661 056266      202$:  NEXT.ERRNO
                        ERRDF  ERRNO,T195SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
                        TRAP  C$ERDF
                        .WORD  838
                        .WORD  T195SSR
                        .WORD  PKTSSR
    056266 104455
    056270 001506
    056272 060434
    056274 012046
4662 056276 005237 002222      210$:  INC  FATFLG  ;SET FATAL ERROR FLAG
4663 056302      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
    056302 104406      TRAP  C$CLP1
4664      ; Do a READ FIFO with tape direction IN to read data
4665 056304 012700 000001      MOV  #1,R0  ;SET READ BYTE COUNT
4666 056310 004737 062046      JSR  PC,T19RFIF  ;SETUP T19PK2 FOR READ FIFO
4667 056314 012704 062460      MOV  #T19PK2,R4  ;GET WRITE SUBSYSTEM COMMAND PACKET
4668 056320 010465 000000      MOV  R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
4669 056324 004737 016336      JSR  PC,CHKTSSR  ;WAIT FOR SSR TO SET
4670 056330      FORCERROR 222$  ;@@DFORCE ERROR IF FORCER=1
4671 056344 103407      BCS  230$  ;BR IF CARRY SET (GOOD RETURN)
4672 056346 010001      MOV  R0,R1  ;SAVE CONTENTS OF TSSR
4673 056350
4674 056350      222$:  NEXT.ERRNO
                        ERRDF  ERRNO,T196SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
                        TRAP  C$ERDF
                        .WORD  839
                        .WORD  T196SSR
                        .WORD  PKTSSR
    056350 104455
    056352 001507
    056354 060500
    056356 012046
4675 056360 005237 002222      230$:  INC  FATFLG  ;SET FATAL ERROR FLAG
4676 056364      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
    056364 104406      TRAP  C$CLP1
4677      ; If Data read from FIFO NOT= to Data sent Then Print Error
4678 056366 004737 062220      JSR  PC,T19SETEXP  ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
4679 056372 012701 060062      MOV  #T19EXSTA,R1  ;GET EXPECTED READ STATUS
4680 056376 012702 062352      MOV  #T19BFSTA,R2  ;GET RECV READ STATUS
4681 056402 013711 002312      MOV  DATA,(R1)  ;SET EXPD WORD #8 = DATA
4682 056406 016261 000002 000002      MOV  2(R2),2(R1)  ;SET EXPD WORD #9 = RECV (NOT TESTING)
4683 056414 005000      CLR  R0  ;HIGH RECV ADDRESS FOR CKMSG2
4684 056416 012701 062332      MOV  #T19BFR,R1  ;LOW RECV ADDRESS FOR CKMSG2
4685 056422 012702 060042      MOV  #T19EXP,R2  ;EXPD ADDRESS
4686 056426 012703 000022      MOV  #18,,R3  ;NUMBER OF BYTES TO COMPARE
4687 056432 004737 011500      JSR  PC,CKMSG2  ;EXPD EQUAL RECV?
4688 056436      FORCERROR 242$,NOTSSR  ;@@
4689 056446 103404      BCS  250$  ;BR IF YES
4690 056450
4691 056450      242$:  NEXT.ERRNO
                        ERRHRD  ERRNO,T19WSTR,MSGSUB  ;REPORT ERROR
    056450 104456      TRAP  C$ERHRD
    056452 001510      .WORD  840
    056454 061523      .WORD  T19WSTR
    056456 013742      .WORD  MSGSUB
4692 056460      250$:  CKLOOP  ;LOOP ON ERROR, IF FLAG SET
    056460 104406      TRAP  C$CLP1
4693
4694
4695 056462      FORCEEXIT 255$  ;@@
4696 056472 013703 002316      MOV  TSTPTR,R3  ;RESTORE CURRENT TSTBLK POINTER
4697 056476 020327 003062      CMP  R3,#TBLEND  ;END OF TSTBLK?
4698 056502 103002      BHIS 255$  ;BR IF YES
4699 056504 000137 055514      JMP  100$  ;DO ANOTHER TSTBLK PATTERN
4700 056510      255$:
4701
    
```

4702 056510  
056510  
056510 104403  
4703  
4704 056512 005737 002222  
4705 056516 001402  
4706 056520 004737 017202  
4707 056524

ENDSUB

260\$:

TST FATFLG  
BEQ 260\$  
JSR PC,CKDROP

:/:/:/:/:/ END SUBTEST /:/:/:/  
L10071: TRAP C\$ESUB

:ANY FATAL ERRORS ?  
:BRANCH IF NOT  
:TRY TO DROP THE UNIT

4709  
 4710  
 4711  
 4712  
 4713  
 4714  
 4715  
 4716  
 4717  
 4718  
 4719  
 4720  
 4721  
 4722  
 4723  
 4724  
 4725  
 4726  
 4727  
 4728  
 4729  
 4730  
 4731  
 4732  
 4733  
 4734  
 4735  
 4736  
 4737  
 4738  
 4739  
 4740  
 4741  
 4742  
 4743

.SBTTL TEST 8: SUBTEST 4 LOOPBACK READ STROBE TEST

++  
 TEST 8: SUBTEST 4:

SUBTEST DESCRIPTION:

This subtest verifies the Read Strobe loopback path can strobe the data from the Data lines to the FIFO. The signal IRESV4 drives IRSTR (read strobe) to write from the data lines to the FIFO.

TEST STEPS:

REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE

BEGIN

Write to TSSR register to soft initialize the controller  
 Do WRITE CHARACTERISTICS to check for Extended Features Switch  
 If Extended Features Hardware Switch Clear then:  
 Do Write Subsystem Write Miscellaneous to Set Extended Features.  
 Do WRITE CHARACTERISTICS to select reserved unit 7 and setup BUFFER  
 Do a Write Subsystem WRITE NPR to set tape direction out and Loopback  
 Do a WRITE NPR to set loopback and tape direction OUT  
 Do a WRITE FORMAT to set IRESV4=>IRSTR = 1  
 Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT  
 Do a READ FIFO with tape direction OUT to load tape out write latch  
 Do a WRITE NPR to set loopback and tape direction IN  
 Do a WRITE FORMAT to set IRESV4=>IRSTR = 0 to write loop data to FIFO  
 Do a WRITE FORMAT to set IRESV4=>IRSTR = 1  
 (to strobe loopback data into FIFO.)  
 Do a READ FIFO with tape direction IN to read data  
 If Data read from FIFO NOT= to Data sent Then Print Error

END

4744 056524  
 056524  
 056524 104402

BGNSUB ;////////// BEGIN SUBTEST //////////  
 TB.4: TRAP CSBSUB

4745  
 4746 056526  
 4747 056526 004737 015774  
 4748 056532 103405  
 4749 056534 010001  
 4750 056536  
 056536 104455  
 056540 001510  
 056542 003652  
 056544 012034

5\$: Write to TSSR register to soft initialize the controller  
 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE  
 BCS 10\$ ;BR IF SOFT INIT OKAY  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT  
 TRAP CSERDF  
 .WORD 840  
 .WORD SFIERR  
 .WORD SFIMSG

4751  
 4752 056546 005037 002222  
 4753 056552 012704 062310  
 4754 056556 004737 010662  
 4755 056562  
 4756 056576 103407  
 4757 056600 010001  
 4758 056602  
 4759 056602

10\$: Do WRITE CHARACTERISTICS to check for Extended Features Switch  
 CLR FATFLG ;CLEAR FATAL ERROR FLAG  
 MOV #T19PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET  
 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND  
 FORCERROR 12\$ ;@@DFORCE ERROR IF FORCER=1  
 BCS 15\$ ;BR IF CARRY SET (GOOD RETURN)  
 MOV R0,R1 ;SAVE CONTENTS OF TSSR  
 NEXT.ERRNO  
 12\$: ERRDF ERRNO,T19SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET



```

056602 104455 TRAP CSERDF
056604 001511 .WORD 841
056606 060223 .WORD T19SSR
056610 012046 .WORD PKTSSR
4760 056612 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4761 056616 104406 15$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP CSCLP1
4762 : If Extended Features Hardware Switch Clear then:
4763 : Do Write Subsystem Write Miscellaneous to Set Extended Features.
4764 056620 012701 062332 MOV #T19BFR,R1 ;MESSAGE BUFFER ADDRESS
4765 056624 032761 000200 000012 BIT #X2.EXTF,XST2(R1) ;EXTENDED FEATURES SWITCH SET?
4766 056632 001026 BNE 30$ ;BR IF YES
4767 056634 004737 062162 JSR PC,T19SEXT ;SETUP PACKET FOR WRITE MISC INVERT
4768 056640 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4769 056644 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4770 056650 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4771 056654 FORCERROR 22$ ;@@DFORCE ERROR IF FORCER=1
4772 056670 103407 BCS 30$ ;BR IF CARRY SET (GOOD RETURN)
4773 056672 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4774 056674 NEXT.ERRNO
4775 056674 22$: ERRDF ERRNO,T192SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP CSERDF
                                .WORD 842
                                .WORD T192SSR
                                .WORD PKTSSR
4776 056704 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4777 056710 104406 30$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP CSCLP1
4778 : Do WRITE CHARACTERISTICS to select reserved unit 7
4779 056712 012704 062310 MOV #T19PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
4780 056716 004737 010662 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
4781 056722 FORCERROR 42$ ;@@DFORCE ERROR IF FORCER=1
4782 056736 103407 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
4783 056740 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4784 056742 NEXT.ERRNO
4785 056742 42$: ERRDF ERRNO,T19SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP CSERDF
                                .WORD 843
                                .WORD T19SSR
                                .WORD PKTSSR
4786 056752 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4787 056756 104406 50$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP CSCLP1
4788 : REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
4789 :
4790 056760 012703 002752 MOV #TSTBLK,R3 ;GET FIRST PATTERN ADDRESS
4791 056764 012337 002312 100$: MOV (R3)+,DATA ;GET A TEST PATTERN
4792 056770 042737 177400 002312 BIC #^C<377>,DATA ;DATA IS BYTE
4793 056776 010337 002316 MOV R3,TSTPTR ;SETUP CURRENT TSTBLK POINTER
4794 : Do a WRITE NPR to set loopback and tape direction OUT
4795 057002 012700 000100 MOV #NP.OUT,R0 ;SET TAPE DIRECTION OUT
4796 057006 052700 000040 BIS #NP.LOOP,R0 ;SET LOOPBACK
4797 057012 004737 062022 JSR PC,T19SNPR ;SETUP T19PK2 FOR WRITE NPR
4798 057016 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4799 057022 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4800 057026 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4801 057032 FORCERROR 102$ ;@@DFORCE ERROR IF FORCER=1
    
```

```

4802 057046 103407          BCS      105$          ;BR IF CARRY SET (GOOD RETURN)
4803 057050 010001          MOV      RO,R1         ;SAVE CONTENTS OF TSSR
4804 057052                NEXT.ERRNO
4805 057052                102$:  ERRDF   ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD    844
                                .WORD    T194SSR
                                .WORD    PKTSSR
                                057052 104455
                                057054 001514
                                057056 060371
                                057060 012046
4806 057062 005237 002222          105$:  INC      FATFLG          ;SET FATAL ERROR FLAG
4807 057066                CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
4808 057066 104406                :      Do a WRITE FORMAT to set IRESV4==>IRSTR = 1
4809 057070 012700 000001          MOV      #WF.I4RES,RO  ;IRESV4==>IRSTR=1
4810 057074 004737 062142          JSR      PC,T19WFMT    ;SETUP T19PK2 FOR WRITE FORMAT
4811 057100 012704 062460          MOV      #T19PK2,R4   ;GET WRITE SUBSYSTEM COMMAND PACKET
4812 057104 010465 000000          MOV      R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
4813 057110 004737 016336          JSR      PC,CHKTSSR   ;WAIT FOR SSR TO SET
4814 057114                FORCERROR 112$          ;@DFORCE ERROR IF FORCER=1
4815 057130 103407          BCS      120$          ;BR IF CARRY SET (GOOD RETURN)
4816 057132 010001          MOV      RO,R1         ;SAVE CONTENTS OF TSSR
4817 057134                NEXT.ERRNO
4818 057134                112$:  ERRDF   ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD    845
                                .WORD    T198SSR
                                .WORD    PKTSSR
                                057134 104455
                                057136 001515
                                057140 060612
                                057142 012046
4819 057144 005237 002222          120$:  INC      FATFLG          ;SET FATAL ERROR FLAG
4820 057150                CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
4821 057150 104406                :      Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT
4822 057152 012700 000001          MOV      #1,RO        ;WRITE 1 BYTE
4823 057156 012701 002312          MOV      #DATA,R1     ;FIFO WRITE DATA ADDRESS
4824 057162 004737 062066          JSR      PC,T19WFIF   ;SETUP T19PK2 FOR WRITE FIFO
4825 057166 012704 062460          MOV      #T19PK2,R4   ;GET WRITE SUBSYSTEM COMMAND PACKET
4826 057172 010465 000000          MOV      R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
4827 057176 004737 016336          JSR      PC,CHKTSSR   ;WAIT FOR SSR TO SET
4828 057202                FORCERROR 132$          ;@DFORCE ERROR IF FORCER=1
4829 057216 103407          BCS      140$          ;BR IF CARRY SET (GOOD RETURN)
4830 057220 010001          MOV      RO,R1         ;SAVE CONTENTS OF TSSR
4831 057222                NEXT.ERRNO
4832 057222                132$:  ERRDF   ERRNO,T195SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD    846
                                .WORD    T195SSR
                                .WORD    PKTSSR
                                057222 104455
                                057224 001516
                                057226 060434
                                057230 012046
4833 057232 005237 002222          140$:  INC      FATFLG          ;SET FATAL ERROR FLAG
4834 057236                CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
4835 057236 104406                :      Do a READ FIFO with tape direction OUT to load tape out write latch
4836 057240 012700 000001          MOV      #1,RO        ;SET READ BYTE COUNT
4837 057244 004737 062046          JSR      PC,T19RFIF   ;SETUP T19PK2 FOR READ FIFO
4838 057250 012704 062460          MOV      #T19PK2,R4   ;GET WRITE SUBSYSTEM COMMAND PACKET
4839 057254 010465 000000          MOV      R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
4840 057260 004737 016336          JSR      PC,CHKTSSR   ;WAIT FOR SSR TO SET
4841 057264                FORCERROR 152$          ;@DFORCE ERROR IF FORCER=1
4842 057300 103407          BCS      160$          ;BR IF CARRY SET (GOOD RETURN)
4843 057302 010001          MOV      RO,R1         ;SAVE CONTENTS OF TSSR
    
```

```

4844 057304
4845 057304 152$: NEXT.ERRNO
      057304 104455 ERRDF ERRNO,T196SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      057306 001517 TRAP C$ERDF
      057310 060500 .WORD 847
      057312 012046 .WORD T196SSR
      057314 005237 002222 .WORD PKTSSR
4846 057314 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4847 057320 160$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      057320 104406 TRAP C$CLP1
4848 ; Do a WRITE NPR to set loopback and tape direction IN
4849 057322 005000 CLR R0 ;CLR NP.OUT TO SET TAPE DIRECTION IN
4850 057324 052700 000040 BIS #NP.LOOP,R0 ;SET LOOPBACK
4851 057330 004737 062022 JSR PC,T19SNPR ;SETUP T19PK2 FOR WRITE NPR
4852 057334 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4853 057340 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4854 057344 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4855 057350 FORCERROR 182$ ;@DFORCE ERROR IF FORCER=1
4856 057364 103407 BCS 190$ ;BR IF CARRY SET (GOOD RETURN)
4857 057366 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4858 057370
4859 057370 182$: NEXT.ERRNO
      057370 104455 ERRDF ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      057372 001520 TRAP C$ERDF
      057374 060371 .WORD 848
      057376 012046 .WORD T194SSR
      057376 012046 .WORD PKTSSR
4860 057400 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4861 057404 190$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      057404 104406 TRAP C$CLP1
4862 ; Do a WRITE FORMAT to set IRESV4==>IRSTR = 0
4863 057406 005000 CLR R0 ;SET IRESV4==>IRSTR=0
4864 057410 004737 062142 JSR PC,T19WFMT ;SETUP T9PK2 FOR WRITE FORMAT
4865 057414 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4866 057420 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4867 057424 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4868 057430 FORCERROR 202$ ;@DFORCE ERROR IF FORCER=1
4869 057444 103407 BCS 210$ ;BR IF CARRY SET (GOOD RETURN)
4870 057446 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4871 057450
4872 057450 202$: NEXT.ERRNO
      057450 104455 ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      057452 001521 TRAP C$ERDF
      057454 060612 .WORD 849
      057456 012046 .WORD T198SSR
      057456 012046 .WORD PKTSSR
4873 057460 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
4874 057464 210$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      057464 104406 TRAP C$CLP1
4875 ; Do a WRITE FORMAT to set IRESV4==>IRSTR = 1
4876 057466 012700 000001 MOV #WF.I4RES,R0 ;IRESV4==>IRSTR=1
4877 057472 004737 062142 JSR PC,T19WFMT ;SETUP T9PK2 FOR WRITE FORMAT
4878 057476 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4879 057502 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4880 057506 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4881 057512 FORCERROR 222$ ;@DFORCE ERROR IF FORCER=1
4882 057526 103407 BCS 230$ ;BR IF CARRY SET (GOOD RETURN)
4883 057530 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4884 057532
4885 057532 222$: NEXT.ERRNO
      057532 104455 ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
    
```

```

057532 104455
057534 001522
057536 060612
057540 012046
4886 057542 005237 002222
4887 057546
057546 104406
4888
4889 057550 012700 000001
4890 057554 004737 062046
4891 057560 012704 062460
4892 057564 010465 000000
4893 057570 004737 016336
4894 057574
4895 057610 103407
4896 057612 010001
4897 057614
4898 057614
057614 104455
057616 001523
057620 060500
057622 012046
4899 057624 005237 002222
4900 057630
057630 104406
4901
4902 057632 004737 062220
4903 057636 012701 060062
4904 057642 012702 062352
4905 057646 013711 002312
4906 057652 016261 000002 000002
4907 057660 005000
4908 057662 012701 062332
4909 057666 012702 060042
4910 057672 012703 000022
4911 057676 004737 011500
4912 057702
4913 057712 103404
4914 057714
4915 057714
057714 104456
057716 001524
057720 061630
057722 013742
4916 057724
057724 104406
4917
4918
4919 057726
4920 057736 013703 002316
4921 057742 020327 003062
4922 057746 103002
4923 057750 000137 056764
4924 057754
4925
4926 057754
057754
    
```

```

                TRAP  C$ERDF
                .WORD  850
                .WORD  T198SSR
                .WORD  PKTSSR
230$:  INC  FATFLG      ;SET FATAL ERROR FLAG
        CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
:      Do a READ FIFO with tape direction IN to read data
        MOV  #1,R0      ;SET READ BYTE COUNT
        JSR  PC,T19RFIF ;SETUP T19PK2 FOR READ FIFO
        MOV  #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
        MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
        JSR  PC,CHKTSSR ;WAIT FOR SSR TO SET
        FORCERROR 282$ ;@@DFORCE ERROR IF FORCER=1
        BCS  290$      ;BR IF CARRY SET (GOOD RETURN)
        MOV  R0,R1      ;SAVE CONTENTS OF TSSR
        NEXT.ERRNO
282$:  ERRDF  ERRNO,T196SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C$ERDF
                                .WORD  851
                                .WORD  T196SSR
                                .WORD  PKTSSR
290$:  INC  FATFLG      ;SET FATAL ERROR FLAG
        CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
:      If Data read from FIFO NOT= to Data sent Then Print Error
        JSR  PC,T19SETEXP ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
        MOV  #T19EXSTA,R1 ;GET EXPECTED READ STATUS
        MOV  #T19BFSTA,R2 ;GET RECV READ STATUS
        MOV  DATA,(R1)   ;SET EXPD WORD #8 = DATA
        MOV  2(R2),2(R1)  ;SET EXPD WORD #9 = RECV (NOT TESTING)
        CLR  R0           ;HIGH RECV ADDRESS FOR CKMSG2
        MOV  #T19BFR,R1   ;LOW RECV ADDRESS FOR CKMSG2
        MOV  #T19EXP,R2   ;EXPD ADDRESS
        MOV  #18.,R3      ;NUMBER OF BYTES TO COMPARE
        JSR  PC,CKMSG2    ;EXPD EQUAL RECV?
        FORCERROR 302$,NOTSSR ;@@
        BCS  310$        ;BR IF YES
        NEXT.ERRNO
302$:  ERRHRD ERRNO,T19RSTR,MSGSUB ;REPORT ERROR
                                TRAP  C$ERHRD
                                .WORD  852
                                .WORD  T19RSTR
                                .WORD  MSGSUB
310$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$CLP1
:      FORCEEXIT 355$
        MOV  TSTPTR,R3    ;RESTORE CURRENT TSTBLK POINTER
        CMP  R3,#TBLEND  ;END OF TSTBLK?
        BHS  355$        ;BR IF YES
        JMP  100$        ;DO ANOTHER TSTBLK PATTERN
355$:
        ENDSUB
                ;////////// END SUBTEST ////////////
                L10072:
    
```

4927 057754 104403  
4928 057756 005737 002222  
4929 057762 001402  
4930 057764 004737 017202  
4931 057770  
4932  
4933 057770  
057770 104432  
057772 002602  
4934  
4935

360\$:

TST FATFLG  
BEQ 360\$  
JSR PC,CKDROP  
  
EXIT TST

:ANY FATAL ERRORS ?  
:BRANCH IF NOT  
:TRY TO DROP THE UNIT

:////////// EXIT TEST //////////

TRAP C\$ESUB

TRAP C\$EXIT  
.WORD L10066-

```

4937
4938
4939
4940
4941 057774 000000
4942
4943
4944
4945
4946
4947
4948
4949
4950 057776
4951 057776 000001
4952 060000 000002
4953 060002 000004
4954 060004 000010
4955 060006 002000
4956 060010 000040
4957 060012 000100
4958 060014 000200
4959 060016 004000
4960 060020 010000
4961 060022 020000
4962 060024 040000
4963 060026 100000
4964 060030 000000
4965 060032 000000
4966 060034 000000
4967
4968 060036
4969
4970 060036 377
4971 060037 037
4972 060040 360
4973 060041 000
4974
4975 060042
4976 060042 000000
4977 060044 000000
4978 060046 000000
4979 060050 000000
4980 060052 000000
4981 060054 000000
4982 060056 000000
4983 060060 000000
4984 060062
4985 060162
4986
4987
4988
4989
4990 060162 124 162 141 TST19ID: .ASCIZ 'Transport Bus Interface Loopback'
4991 060223 127 122 111 T19SSR: .ASCIZ 'WRITE CHARACTERISTICS Failed'
4992 060260 127 122 111 T192SSR: .ASCIZ 'WRITE SUBSYSTEM (Write Misc) Failed'
4993 060324 127 122 111 T193SSR: .ASCIZ 'WRITE SUBSYSTEM (Read Status) Failed'

:
+
: LOCAL STORAGE FOR THIS TEST
:
-
T19PREV: .WORD 0 ;DRIVE SIGNAL 1-0 TRANSITION FLAG
:
+
: LOOPBACK DRIVE SIGNAL TABLE
: THIS TABLE IS USED BY T19CNVT TO SETUP
: A DRIVE PATTERN FROM THE TEST DATA INPUT PATTERN.
:
: WRITE CONTROL SIGNALS ARE OF FORM WC.XXX
: WRITE FORMAT SIGNALS ARE OF FORM WF.XXXX
:
-
T19PCTL: ;WRITE CONTROL DRIVE SIGNALS
WC.IGO ;IGO==>IFPT DATA<0>
WC.IFEN ;IFEN==>IFBY DATA<1>
WC.IRWU ;IRWU==>IRWD DATA<2>
WC.IREW ;IREW==>IDBY DATA<3>
WF.IERASE*256. ;IFAD==>ILDPA DATA<4>
WC.IITAD ;ITAD1==>ICNL DATA<5>
WC.IOTAD ;ITAD0==>IRDY DATA<6>
WC.IFAD ;IERASE==>ISPEED DATA<7>
WF.IEDIT*256. ;IEDIT==>IHER DATA<8>
WF.IWFM*256. ;IWFM==>IFMK DATA<9>
WF.IREV*256. ;IREV==>ICER DATA<10>
WF.IWRT*256. ;IWRT==>IIDENT DATA<11>
WF.IHISP*256. ;IHISP==>IEOT DATA<12>
.WORD 0 ;IRESV2 (UNUSED)DATA<13>
.WORD 0 ;IRESV1 (UNUSED)DATA<14>
.WORD 0 ;PARERR (UNTESTED)DATA<15>

T19MSK: ;MASK OF UNTESTED BITS IN READ STATUS BYTES
;UNTESTED BITS ARE SET TO 1
;BYTE 0 MASK
;BYTE 1 MASK (PARERR,IRESV2,IRESV1)
;BYTE 2 (TIMER A,TIMER B,UNDEFINED<1:0>)
;MAKE IT EVEN
.BYTE ^C<000>
.BYTE ^C<340>
.BYTE ^C<017>
.BYTE 0

T19EXP: ;BEGIN EXPECTED DATA BUFFER
;MESSAGE TYPE
;DATA FIELD LENGTH
;RBPCL
;XST0
;XST1
;XST2
;XST3
;XST4 (ALWAYS PRESENT FOR WRITE SUB.)
;EXPECTED READ STATUS AND WRITE FIFO DATA
;END EXPECTED DATA BUFFER
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0

T19EXSTA: .BLKB 64.
T19EXEND:
:
+
: LOCAL TEXT MESSAGES FOR TEST
:
-

```

4994	060371	127	122	111	T194SSR:.ASCIZ	'WRITE SUBSYSTEM (Write Npr) Failed'
4995	060434	127	122	111	T195SSR:.ASCIZ	'WRITE SUBSYSTEM (Write FIFO) Failed'
4996	060500	127	122	111	T196SSR:.ASCIZ	'WRITE SUBSYSTEM (Read FIFO) Failed'
4997	060543	127	122	111	T197SSR:.ASCIZ	'WRITE SUBSYSTEM (Write Control) Failed'
4998	060612	127	122	111	T198SSR:.ASCIZ	'WRITE SUBSYSTEM (Write Format) Failed'
4999	060660	106	111	106	T191CMP:.ASCIZ	'FIFO Status in WORD #9 Incorrect after Initialize'
5000	060742	122	145	141	T192CMP:.ASCIZ	'Read FIFO Data not equal to Write FIFO , Data is in WORD #8'
5001	061036	124	141	160	T193CMP:.ASCIZ	'Tape Status 2 (in WORD #8) Incorrect after RESET TAPE'
5002	061124	122	145	141	T195CMP:.ASCIZ	'Read FIFO Data not equal to Write FIFO Data'
5003	061200	106	111	106	T196CMP:.ASCIZ	'FIFO Status (in WORD #9) Incorrect after READ FIFO'
5004	061263	124	141	160	T197CMP:.ASCIZ	'Tape Status 2 (in WORD #8) Incorrect after RESET TAPE'
5005	061351	103	157	156	T198CMP:.ASCIZ	'Control Signal Loopback Data Error, Data is in WORD #8'
5006	061440	122	145	141	T199CMP:.ASCIZ	'Read/Write Loopback Data Error, Data is in WORD #8'
5007	061523	114	157	157	T19WSTR:.ASCIZ	'Loopback Data Error when strobed by Write strobe, Data is in WORD #8'
5008	061630	114	157	157	T19RSTR:.ASCIZ	'Loopback Data Error when strobed by Read Strobe, Data is in WORD #8'

5009  
5010 .EVEN

5011  
5012  
5013 :+  
: CLEAR MESSAGE BUFFER  
:-

5014					T19CLRBUF:	
5015	061734				SAVREG	:SAVE R1-R5 UNTIL NEXT RETURN
5016	061734				MOV #T19BFR,R1	:GET MESSAGE BUFFER ADDRESS
5017	061740	012701	062332		MOV #T19BEND-T19BFR,R2	:SIZE OF MESSAGE BUFFER IN BYTES
5018	061744	012702	000120		10\$: CLRB (R1)+	:CLEAR A BYTE
5019	061750	105021			DEC R2	:DONE?
5020	061752	005302			BGT 10\$	:BR IF NO
5021	061754	003375			RTS PC	:RETURN
5022	061756	000207				

5023  
5024  
5025 :+  
: SETUP T19PK2 PACKET FOR READ STATUS  
:-

5026					T19SRD:	
5027	061760				JSR PC,T19CLRBUF	:CLEAR MESSAGE BUFFER
5028	061760	004737	061734		MOV #T19DT2,R0	:WRITE SUBSYSTEM DATA BUFFER
5029	061764	012700	062470		MOVB #PW.RDSTATUS,(R0)+	:STORE READ STATUS COMMAND IN BSELO
5030	061770	112720	000005		CLRB (R0)	:CLEAR BSEL1
5031	061774	105010			RTS PC	:RETURN
5032	061776	000207				

5033  
5034  
5035 :+  
: SETUP T19PK2 PACKET FOR WRITE MISC Reset Tape Status F-FLOPS  
:-

5036					T19RSFIF:	
5037	062000				JSR PC,T19CLRBUF	:CLEAR MESSAGE BUFFER
5038	062000	004737	061734		MOV #T19DT2,R0	:WRITE SUBSYSTEM DATA BUFFER
5039	062004	012700	062470		MOVB #PW.WMISC,(R0)+	:STORE WRITE MISCELLANEOUS IN BSELO
5040	062010	112720	000010		MOVB #MS.RSFIF!MS.RSTAP,(R0)	:STORE BSEL1 CLEAR FIFO CODES
5041	062014	112710	000030		RTS PC	:RETURN
5042	062020	000207				

5043  
5044  
5045 :+  
: SETUP T19PK2 PACKET FOR WRITE NPR  
:-

5046  
5047 : INPUT:  
5048 : R0 CONTAINS BSEL1 NPR DATA  
5049 :  
5050 : SETS NP.WRP SINCE IF 0 IT WRITES WRONG PARITY.

```

5051
5052 062022
5053 062022 004737 061734
5054 062026 012701 062470
5055 062032 112721 000011
5056 062036 052700 000020
5057 062042 110011
5058 062044 000207
5059
5060
5061
5062
5063
5064
5065
5066 062046
5067 062046 004737 061734
5068 062052 012701 062470
5069 062056 112721 000003
5070 062062 110021
5071 062064 000207
5072
5073
5074
5075
5076
5077
5078
5079 062066
5080 062066
5081 062072 004737 061734
5082 062076 012702 062470
5083 062102 112722 000004
5084 062106 110022
5085 062110 005022
5086 062112 112122
5087 062114 005300
5088 062116 003375
5089 062120 000207
5090
5091
5092
5093
5094
5095
5096 062122
5097 062122 004737 061734
5098 062126 012701 062470
5099 062132 112721 000006
5100 062136 110021
5101 062140 000207
5102
5103
5104
5105
5106
5107

;--
T19SNPR:
      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
      MOV      #T19DT2,R1       ;WRITE SUBSYSTEM DATA BUFFER
      MOVB     #PW.WNPR,(R1)+    ;STORE WRITE NPR IN BSELO
      BIS      #NP.WRP,R0       ;DON'T WRITE WRONG PARITY
      MOVB     R0,(R1)          ;STORE NPR DATA IN BSEL1
      RTS      PC               ;RETURN

;+
: SETUP T19PK2 PACKET FOR READ FIFO
: INPUT:
: R0 CONTAINS SEL2 BYTE COUNT
;--
T19RFIF:
      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
      MOV      #T19DT2,R1       ;WRITE SUBSYSTEM DATA BUFFER
      MOVB     #PW.RFIFO,(R1)+  ;STORE READ FIFO IN BSELO
      MOVB     R0,(R1)+        ;STORE BYTE COUNT IN BSEL1
      RTS      PC               ;RETURN

;+
: SETUP T19PK2 PACKET FOR WRITE FIFO
: INPUT:
: R0 CONTAINS BYTE COUNT
: R1 CONTAINS DATA PATTERN BLOCK ADDRESS
;--
T19WFIF:
      SAVREG                                ;SAVE R1-R5 UNTIL NEXT RETURN
      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
      MOV      #T19DT2,R2       ;WRITE SUBSYSTEM DATA BUFFER
      MOVB     #PW.WFIFO,(R2)+  ;STORE WRITE FIFO IN BSELO
      MOVB     R0,(R2)+        ;STORE BYTE COUNT IN BSEL1
      CLR      (R2)+            ;CLEAR SEL2 (UNUSED)
10$:  MOVB     (R1)+,(R2)+      ;STORE DATA PATTERN BYTE
      DEC      R0               ;DONE ALL BYTES?
      BGT     10$              ;BR IF NO
      RTS      PC               ;RETURN

;+
: SETUP T19PK2 FOR WRITE CONTROL
: INPUT:
: R0 CONTAINS DRIVING DATA PATTERN
;--
T19WCTL:
      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
      MOV      #T19DT2,R1       ;WRITE SUBSYSTEM DATA BUFFER
      MOVB     #PW.WCTL,(R1)+   ;STORE WRITE CONTROL IN BSELO
      MOVB     R0,(R1)+        ;STORE DATA WORD IN BSEL1
      RTS      PC               ;RETURN

;+
: SETUP T19PK2 FOR WRITE FORMAT TRANSPORT REGISTER
: INPUT:
: R0 CONTAINS DRIVING DATA PATTERN
;--

```



```
5108 062142
5109 062142 004737 061734
5110 062146 012701 062470
5111 062152 112721 000007
5112 062156 110021
5113 062160 000207
5114
5115
5116
5117 062162
5118 062162 012700 062470
5119 062166 112720 000010
5120 062172 112710 000200
5121 062176 000207
5122
5123
5124
5125 062200
5126 062200 012701 060042
5127 062204 012700 000120
5128 062210 105021
5129 062212 005300
5130 062214 003375
5131 062216 000207
5132
5133
5134
5135
5136 062220
5137 062220 012702 060042
5138 062224 012703 062332
5139 062230 012700 000010
5140 062234 012322
5141 062236 005300
5142 062240 003375
5143 062242 000207
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161 062244
5162 062244
5163 062250 012701 057776
5164 062254 005002

T19WFMT:
      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
      MOV      #T19DT2,R1      ;WRITE SUBSYSTEM DATA BUFFER
      MOVVB    #PW.WFMT,(R1)+   ;STORE WRITE FORMAT IN BSELO
      MOVVB    R0,(R1)+        ;STORE DATA WORD IN BSEL1
      RTS      PC              ;RETURN

      ;+
      ;SETUP T19PK2 PACKET FOR WRITE MISC. INVERT EXTENDED FEATURES SWITCH
      ;-
T19SEXT:
      MOV      #T19DT2,R0      ;WRITE SUBSYSTEM DATA BUFFER
      MOVVB    #PW.WMISC,(R0)+  ;STORE WRITE MISCELLANEOUS IN BSELO
      MOVVB    #MS.EXT,(R0)    ;STORE INVERT EXTENDED FEATURES IN BSEL1
      RTS      PC              ;RETURN

      ;+
      ;CLEAR EXPECTED DATA MESSAGE BUFFER
      ;-
T19CLEXP:
      MOV      #T19EXP,R1      ;GET EXPD ADDRESS
      MOV      #T19EXEND-T19EXP,R0 ;GET EXPD SIZE
10$:   CLRB    (R1)+          ;CLEAR A BYTE
      DEC      R0              ;DONE?
      BGT     10$             ;BR IF NO
      RTS      PC              ;RETURN

      ;+
      ;Set WORDS 0-7 of expd message BUFFER = to recv since not testing
      ;-
T19SETEXP:
      MOV      #T19EXP,R2      ;GET EXPD
      MOV      #T19BFR,R3     ;GET READ STATUS RECV BUFFER
      MOV      #8,R0          ;SET WORDS 0-7 EXP=RECV
5$:   MOV      (R3)+,(R2)+    ;SET EXPD=RECV
      DEC      R0              ;DONE WORDS 0-7 WORDS?
      BGT     5$             ;BR IF NO
      RTS      PC              ;RETURN

      ;+
      ;CONVERT A TEST PATTERN DATA WORD TO LOOPBACK DRIVE SIGNALS
      ;-
      ;INPUTS:
      ;
      ;R0 TEST PATTERN
      ;
      ;IMPLICIT INPUTS:
      ;
      ;T19BFCTL - CONTAINS WRITE CONTROL / WRITE FORMAT CONVERSION BITS
      ;
      ;OUTPUTS:
      ;
      ;R0 - LOW BYTE CONTAINS WRITE CONTROL DATA
      ;    - HIGH BYTE CONTAINS WRITE FORMAT DATA
      ;-
T19CNVT:
      SAVREG                    ;SAVE R1-R5 UNTIL NEXT RETURN
      MOV      #T19BFCTL,R1    ;CONVERSION TABLE ADDRESS
      CLR      R2              ;INIT RESULT OF CONVERSION
```

```

5165 062256 012703 000020          MOV    #16.,R3          ;BIT COUNT
5166 062262 006000          10$:  ROR    R0          ;IS THIS BIT EQUAL TO 1?
5167 062264 103001          BCC    20$             ;BR IF NO
5168 062266 051102          BIS    (R1),R2        ;SET CONVERTED BIT
5169 062270 005721          20$:  TST    (R1)+      ;POINT TO NEXT BIT IN CONVERSION TABLE
5170 062272 005303          DEC    R3             ;DONE?
5171 062274 003372          BGT    10$           ;BR IF NO
5172 062276 010200          MOV    R2,R0          ;COPY RESULT
5173 062300 000207          RTS    PC             ;RETURN
5174
5175
5176
5178          062310          .=<.+10>&177770
5180
5181          ;WRITE CHARACTERISTICS COMMAND PACKET
5182
5183 062310          T19PACKET:          ;COMMAND PACKET FOR TEST
5184 062310 100004          .WORD 100004        ;WRITE CHARACTERISTICS COMMAND, WITH ACK
5185 062312 062320          .WORD T19DATA      ;ADDRESS OF CHARACTERISTICS BLOCK
5186 062314 000000          .WORD 0
5187 062316 000012          .WORD 10.          ;MINIMUM MESSAGE PACKET SIZE
5188
5189 062320          T19DATA:           ;CHARACTERISTICS DATA BLOCK
5190 062320 062332          .WORD T19BFR       ;ADDRESS OF MESSAGE BUFFER
5191 062322 000000          .WORD 0
5192 062324 000024          .WORD 20.          ;LENGTH OF MESSAGE BUFFER
5193 062326 000000          .WORD 0            ;ESS,ENB,EAI,ERI
5194 062330 000007          .WORD 7            ;EXTENDED FEATURES UNIT NO.
5195
5196
5197          ;MESSAGE BUFFER FOR ALL TEST 8 COMMANDS
5198
5199 062332          T19BFR:           ;BEGIN MESSAGE BUFFER
5200 062332 000000          .WORD 0            ;MESSAGE TYPE
5201 062334 000000          .WORD 0            ;DATA FIELD LENGTH
5202 062336 000000          .WORD 0            ;RBPCR
5203 062340 000000          .WORD 0            ;XST0
5204 062342 000000          .WORD 0            ;XST1
5205 062344 000000          .WORD 0            ;XST2
5206 062346 000000          .WORD 0            ;XST3
5207 062350 000000          .WORD 0            ;XST4 (ALWAYS PRESENT FOR WRITE SUBSYSTEM
5208 062352          T19BFSTA: .BLKB 64. ;READ STATUS AND WRITE FIFO BUFFER
5209 062452          T19BEND:          ;END OF MESSAGE BUFFER
5210
5211          ;WRITE SUBSYSTEM READ STATUS COMMAND PACKET
5212
5214          062460          .=<.+10>&177770
5216 062460          T19PK2:           ;WRITE SUBSYSTEM WITH ACK
5217 062460 100006          .WORD P.WRTSUB!P.ACK ;LOW ADDRESS OF DATA BLOCK
5218 062462 062470          .WORD T19DT2      ;HIGH ADDRESS OF DATA BLOCK
5219 062464 000000          .WORD 0
5220 062466 000012          .WORD 10.          ;MINIMUM MESSAGE PACKET SIZE
5221
5222 062470          T19DT2:           ;DATA BLOCK
5223 062470          .BYTE 0            ;BSEL0
5224 062471          .BYTE 0            ;BSEL1
5225 062472 000000          .WORD 0            ;SEL2
  
```

5226 062474

.BLKB 64.

:WRITE FIFO DATA OUTPUT BUFFER

5227

5228

5229 062574

ENDTST

L10066:

TRAP

CSETST

062574 104401

5231  
5232  
5233  
5234  
5235  
5236  
5237  
5238  
5239  
5240  
5241  
5242  
5243  
5244  
5245  
5246  
5247  
5248  
5249  
5250  
5251  
5252  
5253  
5254  
5255  
5256  
5257  
5258  
5259  
5260  
5261  
5262  
5263  
5264  
5265  
5266  
5267  
5268  
5269  
5270  
5271  
5272  
5273  
5274  
5275  
5276  
5277  
5278  
5279  
5280  
5281  
5282  
5283  
5284  
5285  
5286  
5287

.SBTTL TEST 9: READ/WRITE DATA PARITY TEST

++  
TEST DESCRIPTION:

This test verifies that the Write Data Parity generator and the Read Data Parity checker operate properly. The Transport Bus signal loopback mode is enabled and a Set Wrong parity function is executed. Then various Write Subsystem Memory functions are performed to write data to and from the FIFO in loopback mode. The program then checks to insure a Read Data parity error occurred. A Reset FIFO is done and the Read Data parity error bit is again tested to insure it cleared. Finally a Clear wrong parity function is done and it is verified the data word can pass in loopback mode without setting Read Data parity error.

TEST STEPS:

REPEAT FOR LOOPCNT

BEGIN

Write to TSSR register to soft initialize the controller  
Do WRITE CHARACTERISTICS to check for Extended Features Switch  
If Extended Features Hardware Switch Clear then:  
Do Write Subsystem Write Miscellaneous to Set Extended Features.  
Do WRITE CHARACTERISTICS to select reserved unit 7 and setup BUFFER

REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE

BEGIN

(\* Verify Write Wrong Parity Sets Parity Error \*)  
Do a WRITE NPR to set loopback and tape direction OUT  
and SET Write Wrong Parity.  
Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)  
Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT  
Do a READ FIFO with tape direction OUT to load tape out write latch  
(this is when wrong parity (IWP) is set)  
Do a WRITE -ORMAT to set IRESV4==>IRSTR = 0 (sets read strobe low)  
(Read Strobe sets PAR IN H [Parity Error])  
Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)  
Do a Write Subsystem READ STATUS  
If Read Data parity error NOT=1 Then Print Error  
Do a Write Misc to RESET FIFO  
Do a Write Subsystem READ STATUS  
If Read Data parity error NOT=0 Then Print Error

(\* Verify Data can be transferred without a Parity Error \*)  
Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)  
Do a WRITE NPR to set loopback and tape direction OUT  
and CLEAR Write Wrong Parity.  
Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT  
Do a READ FIFO with tape direction OUT to load tape out write latch  
Do a WRITE FORMAT to set IRESV4==>IRSTR = 0 (sets read strobe low)  
(Read Strobe should NOT set PAR IN H [Parity Error] here)  
Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)  
Do a Write Subsystem READ STATUS  
If Read Data parity error NOT=0 Then Print Error

```

5288          : END
5289          :--
5290
5291
5292 062576          BGNTST
5293 062576
5297 062576 012700 065162          MOV #TST20ID,R0          ;ASCII MESSAGE TO IDENTIFY TEST
5298 062602 004737 016510          JSR PC,TSTSETUP          ;DO INITIAL TEST SETUP
5299 062606 012737 000012 002216  T20LOOP:  MOV #10.,LOOPCNT          ;PERFORM 10 ITERATIONS
5300 062614
5301
5302 062614          BGNSUB          ;////////// BEGIN SUBTEST //////////
5303 062614 104402          T9.1: TRAP CSBSUB
5304 062616          : Write to TSSR register to soft initialize the controller
5305 062616 004737 015774          5$: JSR PC,SOFINIT          ;WRITE TO TSSR TO SOFT INITIALIZE
5306 062622 103405          BCS 10$          ;BR IF SOFT INIT OKAY
5307 062624 010001          MOV R0,R1          ;SAVE CONTENTS OF TSSR
5308 062626          ERRDF ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL DURING INIT
5309 062626 104455          TRAP CSERDF
5310 062630 001604          .WORD 900
5311 062632 003652          .WORD SFIERR
5312 062634 012034          .WORD SFIMSG
5313
5314          : Do WRITE CHARACTERISTICS to check for Extended Features Switch
5315 062636 005037 002222          10$: CLR FATFLG          ;CLEAR FATAL ERROR FLAG
5316 062642 012704 066360          MOV #T20PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
5317 062646 004737 010662          JSR PC,WRTCHR          ;DO WRITE CHARACTERISTICS COMMAND
5318 062652          FORCERROR 12$          ;@DFORCE ERROR IF FORCER=1
5319 062666 103407          BCS 15$          ;BR IF CARRY SET (GOOD RETURN)
5320 062670 010001          MOV R0,R1          ;SAVE CONTENTS OF TSSR
5321 062672          NEXT.ERRNO
5322 062672          12$: ERRDF ERRNO,T20SSR,PKTSSR          ;DEVICE FATAL SSR FAILED TO SET
5323 062672 104455          TRAP CSERDF
5324 062674 001605          .WORD 901
5325 062676 065211          .WORD T20SSR
5326 062700 012046          .WORD PKTSSR
5327 062702 005237 002222          15$: INC FATFLG          ;SET FATAL ERROR FLAG
5328 062706          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
5329 062706 104406          TRAP C$CLP1
5330
5331          : If Extended Features Hardware Switch Clear then:
5332          : Do Write Subsystem Write Miscellaneous to Set Extended Features.
5333 062710 012701 066402          MOV #T20BFR,R1          ;MESSAGE BUFFER ADDRESS
5334 062714 032761 000200 000012          BIT #X2.EXTF,XST2(R1)          ;EXTENDED FEATURES SWITCH SET?
5335 062722 001026          BNE 30$          ;BR IF YES
5336 062724 004737 066276          JSR PC,T20SEXT          ;SETUP PACKET FOR WRITE MISC INVERT
5337 062730 012704 066530          MOV #T20PK2,R4          ;GET WRITE SUBSYSTEM COMMAND PACKET
5338 062734 010465 000000          MOV R4,TSDB(R5)          ;SET THE PACKET ADDRESS TO EXECUTE
5339 062740 004737 016336          JSR PC,CHKTSSR          ;WAIT FOR SSR TO SET
5340 062744          FORCERROR 22$          ;@DFORCE ERROR IF FORCER=1
5341 062760 103407          BCS 30$          ;BR IF CARRY SET (GOOD RETURN)
5342 062762 010001          MOV R0,R1          ;SAVE CONTENTS OF TSSR
5343 062764          NEXT.ERRNO
5344 062764          22$: ERRDF ERRNO,T202SSR,PKTSSR          ;DEVICE FATAL SSR FAILED TO SET
5345 062764 104455          TRAP CSERDF
5346 062766 001606          .WORD 902
5347 062770 065246          .WORD T202SSR
    
```

```

5334 062772 012046
5335 062774 005237 002222
5336 063000 104406
5337 063002 012704 066360
5338 063006 004737 010662
5339 063012
5340 063026 103407
5341 063030 010001
5342 063032
5343 063032 42$:
    063032 104455
    063034 001607
    063036 065211
    063040 012046
5344 063042 005237 002222
5345 063046 50$:
    063046 104406
5346
5347
5348
5349 063050 012703 002752
5350 063054 012337 002312
5351 063060 042737 177400 002312
5352 063066 010337 002316
5353
5354
5355 063072 012700 000100
5356 063076 052700 000040
5357 063102 042700 000020
5358 063106 004737 066146
5359 063112 012704 066530
5360 063116 010465 000000
5361 063122 004737 016336
5362 063126
5363 063142 103407
5364 063144 010001
5365 063146
5366 063146 102$:
    063146 104455
    063150 001610
    063152 065357
    063154 012046
5367 063156 005237 002222
5368 063162 105$:
    063162 104406
5369
5370 063164 012700 000001
5371 063170 004737 066242
5372 063174 012704 066530
5373 063200 010465 000000
5374 063204 004737 016336
5375 063210
5376 063224 103407
5377 063226 010001
5378 063230
    
```

```

        INC      FATFLG          ;SET FATAL ERROR FLAG      .WORD  PKTSSR
30$:    CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
                                           TRAP  C$CLP1
;      Do WRITE CHARACTERISTICS to select reserved unit 7
        MOV      #T20PACKET,R4    ;GET THE ADDRESS OF COMMAND PACKET
        JSR      PC,WRTCHR        ;DO WRITE CHARACTERISTICS COMMAND
        FORCERROR 42$             ;@@DFORCE ERROR IF FORCER=1
        BCS      50$              ;BR IF CARRY SET (GOOD RETURN)
        MOV      R0,R1            ;SAVE CONTENTS OF TSSR
        NEXT.ERRNO
42$:    ERRDF   ERRNO,T20SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                           TRAP  C$ERDF
                                           .WORD  903
                                           .WORD  T20SSR
                                           .WORD  PKTSSR
        INC      FATFLG          ;SET FATAL ERROR FLAG
50$:    CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
                                           TRAP  C$CLP1
;      REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
100$:   MOV      #TSTBLK,R3        ;GET FIRST PATTERN ADDRESS
        MOV      (R3)+,DATA        ;GET A TEST PATTERN
        BIC      #^C<377>,DATA    ;DATA IS BYTE
        MOV      R3,TSTPTR        ;SETUP CURRENT TSTBLK POINTER
        Do a WRITE NPR to set loopback and tape direction OUT and
        and SET Write Wrong Parity.
        MOV      #NP.OUT,R0        ;SET TAPE DIRECTION OUT
        BIS      #NP.LOOP,R0       ;SET LOOPBACK
        BIC      #NP.WRP,R0        ;SET WRITE WRONG PARITY (INVERTED)
        JSR      PC,T20WNPR        ;SETUP T20PK2 FOR WRITE NPR
        MOV      #T20PK2,R4        ;GET WRITE SUBSYSTEM COMMAND PACKET
        MOV      R4,TSDB(R5)       ;SET THE PACKET ADDRESS TO EXECUTE
        JSR      PC,CHKTSSR        ;WAIT FOR SSR TO SET
        FORCERROR 102$            ;@@DFORCE ERROR IF FORCER=1
        BCS      105$              ;BR IF CARRY SET (GOOD RETURN)
        MOV      R0,R1            ;SAVE CONTENTS OF TSSR
        NEXT.ERRNO
102$:   ERRDF   ERRNO,T204SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                           TRAP  C$ERDF
                                           .WORD  904
                                           .WORD  T204SSR
                                           .WORD  PKTSSR
        INC      FATFLG          ;SET FATAL ERROR FLAG
105$:   CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
                                           TRAP  C$CLP1
;      Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)
        MOV      #WF.I4RES,R0      ;IRESV4==>IRSTR = 1
        JSR      PC,T20WFMT        ;SETUP T20PK2 FOR WRITE FORMAT
        MOV      #T20PK2,R4        ;GET WRITE SUBSYSTEM COMMAND PACKET
        MOV      R4,TSDB(R5)       ;SET THE PACKET ADDRESS TO EXECUTE
        JSR      PC,CHKTSSR        ;WAIT FOR SSR TO SET
        FORCERROR 112$            ;@@DFORCE ERROR IF FORCER=1
        BCS      120$              ;BR IF CARRY SET (GOOD RETURN)
        MOV      R0,R1            ;SAVE CONTENTS OF TSSR
        NEXT.ERRNO
    
```

```

5379 063230      112$:  ERRD:  ERRNO,T208SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      063230 104455                                TRAP  C$ERDF
      063232 001611                                .WORD 905
      063234 065531                                .WORD T208SSR
      063236 012046                                .WORD PKTSSR
5380 063240 005237 002222      120$:  INC  FATFLG  ;SET FATAL ERROR FLAG
5381 063244      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063244 104406                                TRAP  C$CLP1
5382  :  Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT
5383 063246 012700 000001      MOV  #1,R0  ;WRITE 1 BYTE
5384 063252 012701 002312      MOV  #DATA,R1 ;FIFO WRITE DATA ADDRESS
5385 063256 004737 066206      JSR  PC,T20WFIF ;SETUP T20PK2 FOR WRITE FIFO
5386 063262 012704 066530      MOV  #T20PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
5387 063266 010465 000000      MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
5388 063272 004737 016336      JSR  PC,CHKTSSR ;WAIT FOR SSR TO SET
5389 063276      FORCERROR 152$ ;@DFORCE ERROR IF FORCER=1
5390 063312 103407      BCS 160$ ;BR IF CARRY SET (GOOD RETURN)
5391 063314 010001      MOV  R0,R1 ;SAVE CONTENTS OF TSSR
5392 063316      NEXT.ERRNO
5393 063316      152$:  ERRDF  ERRNO,T205SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      063316 104455                                TRAP  C$ERDF
      063320 001612                                .WORD 906
      063322 065422                                .WORD T205SSR
      063324 012046                                .WORD PKTSSR
5394 063326 005237 002222      160$:  INC  FATFLG  ;SET FATAL ERROR FLAG
5395 063332      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063332 104406                                TRAP  C$CLP1
5396  :  Do a READ FIFO with tape direction OUT to load tape out write latch
5397  :  (this is when wrong parity (IWP) is set)
5398 063334 012700 000001      MOV  #1,R0  ;SET READ BYTE COUNT
5399 063340 004737 066166      JSR  PC,T20RFIF ;SETUP T20PK2 FOR READ FIFO
5400 063344 012704 066530      MOV  #T20PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
5401 063350 010465 000000      MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
5402 063354 004737 016336      JSR  PC,CHKTSSR ;WAIT FOR SSR TO SET
5403 063360      FORCERROR 172$ ;@DFORCE ERROR IF FORCER=1
5404 063374 103407      BCS 180$ ;BR IF CARRY SET (GOOD RETURN)
5405 063376 010001      MOV  R0,R1 ;SAVE CONTENTS OF TSSR
5406 063400      NEXT.ERRNO
5407 063400      172$:  ERRDF  ERRNO,T206SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      063400 104455                                TRAP  C$ERDF
      063402 001613                                .WORD 907
      063404 065466                                .WORD T206SSR
      063406 012046                                .WORD PKTSSR
5408 063410 005237 002222      180$:  INC  FATFLG  ;SET FATAL ERROR FLAG
5409 063414      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063414 104406                                TRAP  C$CLP1
5410  :  Do a WRITE FORMAT to set IRESV4==>IRSTR = 0 (sets read strobe low)
5411  :  (Read Strobe sets PAR IN H [Parity Error])
5412 063416 005000      CLR  R0  ;IRESV4==>IRSTR = 0
5413 063420 004737 066242      JSR  PC,T20WFMT ;SETUP T20PK2 FOR WRITE FORMAT
5414 063424 012704 066530      MOV  #T20PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
5415 063430 010465 000000      MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
5416 063434 004737 016336      JSR  PC,CHKTSSR ;WAIT FOR SSR TO SET
5417 063440      FORCERROR 192$ ;@DFORCE ERROR IF FORCER=1
5418 063454 103407      BCS 200$ ;BR IF CARRY SET (GOOD RETURN)
5419 063456 010001      MOV  R0,R1 ;SAVE CONTENTS OF TSSR
5420 063460      NEXT.ERRNO
    
```

```

5421 063460      192$:  ERRDF  ERRNO,T208SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      063460 104455                                TRAP  C$ERDF
      063462 001614                                .WORD 908
      063464 065531                                .WORD T208SSR
      063466 012046                                .WORD PKTSSR
5422 063470 005237 002222      INC  FATFLG  ;SET FATAL ERROR FLAG
5423 063474      200$:  CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063474 104406                                TRAP  C$CLP1
5424      :  Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)
5425 063476 012700 000001      MOV  #WF.I4RES,R0  ;IRESV4==>IRSTR = 1
5426 063502 004737 066242      JSR  PC,T20WFMT  ;SETUP T20PK2 FOR WRITE FORMAT
5427 063506 012704 066530      MOV  #T20PK2,R4  ;GET WRITE SUBSYSTEM COMMAND PACKET
5428 063512 010465 000000      MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
5429 063516 004737 016336      JSR  PC,CHKTSSR  ;WAIT FOR SSR TO SET
5430 063522      FORCERROR 212$  ;@@DFORCE ERROR IF FORCER=1
5431 063536 103407      BCS  220$  ;BR IF CARRY SET (GOOD RETURN)
5432 063540 010001      MOV  R0,R1  ;SAVE CONTENTS OF TSSR
5433 063542      NEXT.ERRNO
5434 063542      212$:  ERRDF  ERRNO,T208SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      063542 104455                                TRAP  C$ERDF
      063544 001615                                .WORD 909
      063546 065531                                .WORD T208SSR
      063550 012046                                .WORD PKTSSR
5435 063552 005237 002222      INC  FATFLG  ;SET FATAL ERROR FLAG
5436 063556      220$:  CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063556 104406                                TRAP  C$CLP1
5437      :  Do a Write Subsystem READ STATUS
5438 063560 004737 066126      JSR  PC,T20SRD  ;SETUP PACKET FOR READ STATUS
5439 063564 012704 066530      MOV  #T20PK2,R4  ;GET WRITE SUBSYSTEM COMMAND PACKET
5440 063570 010465 000000      MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
5441 063574 004737 016336      JSR  PC,CHKTSSR  ;WAIT FOR SSR TO SET
5442 063600      FORCERROR 232$  ;@@DFORCE ERROR IF FORCER=1
5443 063614 103407      BCS  240$  ;BR IF CARRY SET (GOOD RETURN)
5444 063616 010001      MOV  R0,R1  ;SAVE CONTENTS OF TSSR
5445 063620      NEXT.ERRNO
5446 063620      232$:  ERRDF  ERRNO,T203SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      063620 104455                                TRAP  C$ERDF
      063622 001616                                .WORD 910
      063624 065312                                .WORD T203SSR
      063626 012046                                .WORD PKTSSR
5447 063630 005237 002222      INC  FATFLG  ;SET FATAL ERROR FLAG
5448 063634      240$:  CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063634 104406                                TRAP  C$CLP1
5449      :  If Read Data parity error NOT=1 Then Print Error
5450 063636 004737 066334      JSR  PC,T20SETEXP ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
5451 063642 012701 065062      MOV  #T20EXSTA,R1 ;GET EXPECTED READ STATUS
5452 063646 012702 066422      MOV  #T20BFSTA,R2 ;GET RECV READ STATUS
5453 063652 011211      MOV  (R2),(R1)  ;SET EXPD WORD #8 = RECV TEMP
5454 063654 016261 000002 000002      MOV  2(R2),2(R1) ;SET EXPD WORD #9 = RECV (NOT TESTED)
5455 063662 052711 100000      BIS  #S1.PARERR,(R1) ;SET EXP PAR ERR =1
5456 063666 005000      CLR  R0  ;HIGH RECV ADDRESS FOR CKMSG2
5457 063670 012701 066402      MOV  #T20BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
5458 063674 012702 065042      MOV  #T20EXP,R2 ;EXPD ADDRESS
5459 063700 012703 000024      MOV  #20.,R3  ;NUMBER OF BYTES TO COMPARE
5460 063704 004737 011500      JSR  PC,CKMSG2  ;EXPD EQUAL RECV?
5461 063710      FORCERROR 252$,NOTSSR  ;@@
5462 063720 103404      BCS  260$  ;BR IF YES
    
```



```

5463 063722
5464 063722 252$: NEXT.ERRNO
      063722 104456 ERRHRD ERRNO,T2OSWP,MSGSTAT ;REPORT ERROR
      063724 001617 TRAP C$ERHRD
      063726 065577 .WORD 911
      063730 012350 .WORD T2OSWP
      5465 063732 260$: CKLOOP ;LOOP ON ERROR, IF FLAG SET .WORD MSGSTAT
      063732 104406 TRAP C$CLP1
5466 : Do a Write Misc to RESET FIFO
5467 063734 012700 000020 MOV #MS.RSFIF,R0 ;SET RESET FIFO COMMAND
5468 063740 004737 066262 JSR PC,T2OWMISC ;SETUP T2OPK2 FOR WRITE MISC
5469 063744 012704 066530 MOV #T2OPK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
5470 063750 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
5471 063754 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
5472 063760 FORCERROR 282$ ;@DFORCE ERROR IF FORCER=1
5473 063774 103407 BCS 290$ ;BR IF CARRY SET (GOOD RETURN)
5474 063776 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
5475 064000 NEXT.ERRNO
5476 064000 282$: ERRDF ERRNO,T2O2SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      064000 104455 TRAP C$ERDF
      064002 001620 .WORD 912
      064004 065246 .WORD T2O2SSR
      064006 012046 .WORD PKTSSR
5477 064010 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
5478 064014 290$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      064014 104406 TRAP C$CLP1
5479 : Do a Write Subsystem READ STATUS
5480 : If Read Data parity error NOT=0 Then Print Error
5481 064016 004737 066334 JSR PC,T2OSETEXP ;SET WORDS 0-7 EXPD=RCV (NOT TESTING)
5482 064022 012701 065062 MOV #T2OEXSTA,R1 ;GET EXPECTED READ STATUS
5483 064026 012702 066422 MOV #T2OBFSTA,R2 ;GET RCV READ STATUS
5484 064032 011211 MOV (R2),(R1) ;SET EXPD WORD #8 = RCV TEMP
5485 064034 016261 000002 000002 MOV 2(R2),2(R1) ;SET EXPD WORD #9 = RCV (NOT TESTED)
5486 064042 042711 100000 BIC #S1.PARERR,(R1) ;SET EXP PAR ERR =0
5487 064046 005000 CLR R0 ;HIGH RCV ADDRESS FOR CKMSG2
5488 064050 012701 066402 MOV #T2OBFRA,R1 ;LOW RCV ADDRESS FOR CKMSG2
5489 064054 012702 065042 MOV #T2OEXPA,R2 ;EXPD ADDRESS
5490 064060 012703 000024 MOV #20.,R3 ;NUMBER OF BYTES TO COMPARE
5491 064064 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RCV?
5492 064070 FORCERROR 302$,NOTSSR ;@D
5493 064100 103404 BCS 320$ ;BR IF YES
5494 064102 NEXT.ERRNO
5495 064102 302$: ERRHRD ERRNO,T2ORSF,MSGSTAT ;REPORT ERROR
      064102 104456 TRAP C$ERHRD
      064104 001621 .WORD 913
      064106 065706 .WORD T2ORSF
      064110 012350 .WORD MSGSTAT
5496 064112 320$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      064112 104406 TRAP C$CLP1
5497 : (* Verify Data can be transferred without a Parity Error *)
5498 : Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)
5499 064114 012700 000001 MOV #WF.I4RES,R0 ;IRESV4==>IRSTR = 1
5500 064120 004737 066242 JSR PC,T2OWFMT ;SETUP T2OPK2 FOR WRITE FORMAT
5501 064124 012704 066530 MOV #T2OPK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
5502 064130 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
5503 064134 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
5504 064140 FORCERROR 332$ ;@DFORCE ERROR IF FORCER=1
  
```

```

5505 064154 103407          BCS      340$          ;BR IF CARRY SET (GOOD RETURN)
5506 064156 010001          MOV      RO,R1         ;SAVE CONTENTS OF TSSR
5507 064160                NEXT.ERRNO
5508 064160                332$:  ERRDF  ERRNO,T208SSR,PKiSSR ;DEVICE FATAL SSR FAILED TO SET
      064160 104455                                     TRAP  C$ERDF
      064162 001622                                     .WORD 914
      064164 065531                                     .WORD T208SSR
      064166 012046                                     .WORD PKTSSR
5509 064170 005237 002222          INC      FATFLG        ;SET FATAL ERROR FLAG
5510 064174                340$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      064174 104406                                     TRAP  C$CLP1
5511 :
5512 : Do a WRITE NPR to set loopback and tape direction OUT and
      : and CLEAR Write Wrong Parity.
5513 064176 012700 000100          MOV      #NP.OUT,RO    ;SET TAPE DIRECTION OUT
5514 064202 052700 000040          BIS      #NP.LOOP,RO   ;SET LOOPBACK
5515 064206 052700 000020          BIS      #NP.WRP,RO   ;CLEAR WRITE WRONG PARITY (INVERTED)
5516 064212 004737 066146          JSR      PC,T20WNPR    ;SETUP T20PK2 FOR WRITE NPR
5517 064216 012704 066530          MOV      #T20PK2,R4   ;GET WRITE SUBSYSTEM COMMAND PACKET
5518 064222 010465 000000          MOV      R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
5519 064226 004737 016336          JSR      PC,CHKTSSR   ;WAIT FOR SSR TO SET
5520 064232                FORCERROR 352$          ;@DFORCE ERROR IF FORCER=1
5521 064246 103407          BCS      360$          ;BR IF CARRY SET (GOOD RETURN)
5522 064250 010001          MOV      RO,R1         ;SAVE CONTENTS OF TSSR
5523 064252                NEXT.ERRNO
5524 064252                552$:  ERRDF  ERRNO,T204SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      064252 104455                                     TRAP  C$ERDF
      064254 001623                                     .WORD 915
      064256 065357                                     .WORD T204SSR
      064260 012046                                     .WORD PKTSSR
5525 064262 005237 002222          INC      FATFLG        ;SET FATAL ERROR FLAG
5526 064266                360$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      064266 104406                                     TRAP  C$CLP1
5527 : Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT
5528 064270 012700 000001          MOV      #1,RO         ;WRITE 1 BYTE
5529 064274 012701 002312          MOV      #DATA,R1     ;FIFO WRITE DATA ADDRESS
5530 064300 004737 066206          JSR      PC,T20WFIF   ;SETUP T20PK2 FOR WRITE FIFO
5531 064304 012704 066530          MOV      #T20PK2,R4   ;GET WRITE SUBSYSTEM COMMAND PACKET
5532 064310 010465 000000          MOV      R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
5533 064314 004737 016336          JSR      PC,CHKTSSR   ;WAIT FOR SSR TO SET
5534 064320                FORCERROR 372$          ;@DFORCE ERROR IF FORCER=1
5535 064334 103407          BCS      380$          ;BR IF CARRY SET (GOOD RETURN)
5536 064336 010001          MOV      RO,R1         ;SAVE CONTENTS OF TSSR
5537 064340                NEXT.ERRNO
5538 064340                372$:  ERRDF  ERRNO,T205SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      064340 104455                                     TRAP  C$ERDF
      064342 001624                                     .WORD 916
      064344 065422                                     .WORD T205SSR
      064346 012046                                     .WORD PKTSSR
5539 064350 005237 002222          INC      FATFLG        ;SET FATAL ERROR FLAG
5540 064354                380$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      064354 104406                                     TRAP  C$CLP1
5541 : Do a READ FIFO with tape direction OUT to load tape out write latch
5542 064356 012700 000001          MOV      #1,RO         ;SET READ BYTE COUNT
5543 064362 004737 066166          JSR      PC,T20RFIF   ;SETUP T20PK2 FOR READ FIFO
5544 064366 012704 066530          MOV      #T20PK2,R4   ;GET WRITE SUBSYSTEM COMMAND PACKET
5545 064372 010465 000000          MOV      R4,TSDB(R5)  ;SET THE PACKET ADDRESS TO EXECUTE
5546 064376 004737 016336          JSR      PC,CHKTSSR   ;WAIT FOR SSR TO SET
    
```

```

5547 064402          FORCERROR      392$          ;@@DFORCE ERROR IF FORCER=1
5548 064416 103407   BCS          400$          ;BR IF CARRY SET (GOOD RETURN)
5549 064420 010001   MOV          RO,R1          ;SAVE CONTENTS OF TSSR
5550 064422          NEXT.ERRNO
5551 064422          ERRDF      ERRNO,T206SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      064422 104455          TRAP      C$ERDF
      064424 001625          .WORD      917
      064426 065466          .WORD      T206SSR
      064430 012046          .WORD      PKTSSR
5552 064432 005237 002222   INC          FATFLG          ;SET FATAL ERROR FLAG
5553 064436          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      064436 104406          TRAP      C$CLP1
5554          ; Do a WRITE FORMAT to set IRESV4==>IRSTR = 0 (sets read strobe low)
5555          ; (Read Strobe sets PAR IN H [Parity Error])
5556 064440 005000          CLR          RO          ;IRESV4==>IRSTR = 0
5557 064442 004737 066242   JSR          PC,T20WFMT      ;SETUP T20PK2 FOR WRITE FORMAT
5558 064446 012704 066530   MOV          #T20PK2,R4      ;GET WRITE SUBSYSTEM COMMAND PACKET
5559 064452 010465 000000   MOV          R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
5560 064456 004737 016336   JSR          PC,CHKTSSR      ;WAIT FOR SSR TO SET
5561 064462          FORCERROR      412$          ;@@DFORCE ERROR IF FORCER=1
5562 064476 103407   BCS          420$          ;BR IF CARRY SET (GOOD RETURN)
5563 064500 010001   MOV          RO,R1          ;SAVE CONTENTS OF TSSR
5564 064502          NEXT.ERRNO
5565 064502          ERRDF      ERRNO,T208SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      064502 104455          TRAP      C$ERDF
      064504 001626          .WORD      918
      064506 065531          .WORD      T208SSR
      064510 012046          .WORD      PKTSSR
5566 064512 005237 002222   INC          FATFLG          ;SET FATAL ERROR FLAG
5567 064516          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      064516 104406          TRAP      C$CLP1
5568          ; Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)
5569 064520 012700 000001   MOV          #WF.I4RES,RO      ;IRESV4==>IRSTR = 1
5570 064524 004737 066242   JSR          PC,T20WFMT      ;SETUP T20PK2 FOR WRITE FORMAT
5571 064530 012704 066530   MOV          #T20PK2,R4      ;GET WRITE SUBSYSTEM COMMAND PACKET
5572 064534 010465 000000   MOV          R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
5573 064540 004737 016336   JSR          PC,CHKTSSR      ;WAIT FOR SSR TO SET
5574 064544          FORCERROR      432$          ;@@DFORCE ERROR IF FORCER=1
5575 064560 103407   BCS          440$          ;BR IF CARRY SET (GOOD RETURN)
5576 064562 010001   MOV          RO,R1          ;SAVE CONTENTS OF TSSR
5577 064564          NEXT.ERRNO
5578 064564          ERRDF      ERRNO,T208SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      064564 104455          TRAP      C$ERDF
      064566 001627          .WORD      919
      064570 065531          .WORD      T208SSR
      064572 012046          .WORD      PKTSSR
5579 064574 005237 002222   INC          FATFLG          ;SET FATAL ERROR FLAG
5580 064600          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      064600 104406          TRAP      C$CLP1
5581          ;
5582          ; Do a Write Subsystem READ STATUS
5583 064602 004737 066126   JSR          PC,T20SRD      ;SETUP PACKET FOR READ STATUS
5584 064606 012704 066530   MOV          #T20PK2,R4      ;GET WRITE SUBSYSTEM COMMAND PACKET
5585 064612 010465 000000   MOV          R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
5586 064616 004737 016336   JSR          PC,CHKTSSR      ;WAIT FOR SSR TO SET
5587 064622          FORCERROR      452$          ;@@DFORCE ERROR IF FORCER=1
5588 064636 103407   BCS          460$          ;BR IF CARRY SET (GOOD RETURN)
    
```



```

5633
5634
5635
5636
5637
5638 065036
5639
5640 065036      377
5641 065037      037
5642 065040      360
5643 065041      000
5644
5645 065042
5646 065042      000000
5647 065044      000000
5648 065046      000000
5649 065050      000000
5650 065052      000000
5651 065054      000000
5652 065056      000000
5653 065060      000000
5654 065062
5655 065162
5656
5657
5658
5659
5660 065162      122      145      141
5661 065211      127      122      111
5662 065246      127      122      111
5663 065312      127      122      111
5664 065357      127      122      111
5665 065422      127      122      111
5666 065466      127      122      111
5667 065531      127      122      111
5668 065577      122      145      141
5669 065706      122      145      141
5670 066007      122      145      141
5671
5672
5673
5674
5675
5676 066102
5677 066102
5678 066106      012701      066402
5679 066112      012702      000120
5680 066116      105021
5681 066120      005302
5682 066122      003375
5683 066124      000207
5684
5685
5686
5687
5688 066126
5689 066126      004737      066102

;+
; LOCAL STORAGE FOR THIS TEST
;-

T20MSK:
;MASK OF UNTESTED BITS IN READ STATUS
;UNTESTED BITS ARE SET TO 1
;BYTE 0 MASK
;BYTE 1 MASK (PARERR,IRESV2,IRESV1)
;BYTE 2 (TIMER A,TIMER B,UNDEFINED<1:0>)
;MAKE IT EVEN

T20EXP:
;BEGIN EXPECTED DATA BUFFER
;MESSAGE TYPE
;DATA FIELD LENGTH
;RBPCR
;XST0
;XST1
;XST2
;XST3
;XST4 (ALWAYS PRESENT FOR WRITE SUB.)
;EXPECTED READ STATUS AND WRITE FIFO DATA
;END EXPECTED DATA BUFFER

T20EXSTA: .BLKB 64.
T20EXEND:

;+
; LOCAL TEXT MESSAGES FOR TEST
;-

TST20ID: .ASCIZ 'Read/Write Data Parity'
T20SSR: .ASCIZ 'WRITE CHARACTERISTICS Failed'
T202SSR: .ASCIZ 'WRITE SUBSYSTEM (Write Misc) Failed'
T203SSR: .ASCIZ 'WRITE SUBSYSTEM (Read Status) Failed'
T204SSR: .ASCIZ 'WRITE SUBSYSTEM (Write Npr) Failed'
T205SSR: .ASCIZ 'WRITE SUBSYSTEM (Write FIFO) Failed'
T206SSR: .ASCIZ 'WRITE SUBSYSTEM (Read FIFO) Failed'
T208SSR: .ASCIZ 'WRITE SUBSYSTEM (Write Format) Failed'
T20SWP: .ASCIZ 'Read Data Parity Error (PARERR) Failed to Set after Write Wrong Parity'
T20RSF: .ASCIZ 'Read Data Parity Error (PARERR) Failed to Clear after RESET FIFO'
T20CWP: .ASCIZ 'Read Data Parity Error (PARERR) occurred in Data Loopback'
.EVEN

;+
; CLEAR MESSAGE BUFFER
;-

T20CLRBUF:
;SAVE R1-R5 UNTIL NEXT RETURN
;GET MESSAGE BUFFER ADDRESS
;SIZE OF MESSAGE BUFFER IN BYTES
;CLEAR A BYTE
;DONE?
;BR IF NO
;RETURN

SAVREG
MOV #T20BFR,R1
MOV #T20BEND-T20BFR,R2
10$: CLRB (R1)+
DEC R2
BGT 10$
RTS PC

;+
; SETUP T20PK2 PACKET FOR READ STATUS
;-

T20SRD: JSR PC,T20CLRBUF
;CLEAR MESSAGE BUFFER
    
```

```

5690 066132 012700 066540      MOV      #T20DT2,R0      ;WRITE SUBSYSTEM DATA BUFFER
5691 066136 112720 000005      MOVB     #PW.RDSTATUS,(R0)+ ;STORE READ STATUS COMMAND IN BSELO
5692 066142 105010                CLRB     (R0)           ;CLEAR BSEL1
5693 066144 000207      RTS      PC           ;RETURN
5694
5695
5696
5697      :+
5698      : SETUP T20PK2 PACKET FOR WRITE NPR
5699      : INPUT:
5700      :      RO CONTAINS BSEL1 NPR DATA
5701
5702      :-
5703 066146      T20WNPR:
5704 066146 004737 066102      JSR      PC,T20CLRBUF    ;CLEAR MESSAGE BUFFER
5705 066152 012701 066540      MOV      #T20DT2,R1    ;WRITE SUBSYSTEM DATA BUFFER
5706 066156 112721 000011      MOVB     #PW.WNPR,(R1)+ ;STORE WRITE NPR IN BSELO
5707 066162 110011                MOVB     RO,(R1)        ;STORE NPR DATA IN BSEL1
5708 066164 000207      RTS      PC           ;RETURN
5709
5710      :+
5711      : SETUP T20PK2 PACKET FOR READ FIFO
5712      : INPUT:
5713      :      RO CONTAINS SEL2 BYTE COUNT
5714
5715      :-
5716 066166      T20RFIF:
5717 066166 004737 066102      JSR      PC,T20CLRBUF    ;CLEAR MESSAGE BUFFER
5718 066172 012701 066540      MOV      #T20DT2,R1    ;WRITE SUBSYSTEM DATA BUFFER
5719 066176 112721 000003      MOVB     #PW.RFIFO,(R1)+ ;STORE READ FIFO IN BSELO
5720 066202 110021                MOVB     RO,(R1)+        ;STORE BYTE COUNT IN BSEL1
5721 066204 000207      RTS      PC           ;RETURN
5722
5723      :+
5724      : SETUP T20PK2 PACKET FOR WRITE FIFO
5725      : INPUT:
5726      :      RO CONTAINS BYTE COUNT
5727      :      R1 CONTAINS DATA PATTERN BLOCK ADDRESS
5728
5729 066206      T20WFIF:
5730 066206                SAVREG                ;SAVE R1-R5 UNTIL NEXT RETURN
5731 066212 004737 066102      JSR      PC,T20CLRBUF    ;CLEAR MESSAGE BUFFER
5732 066216 012702 066540      MOV      #T20DT2,R2    ;WRITE SUBSYSTEM DATA BUFFER
5733 066222 112722 000004      MOVB     #PW.WFIFO,(R2)+ ;STORE WRITE FIFO IN BSELO
5734 066226 110022                MOVB     RO,(R2)+        ;STORE BYTE COUNT IN BSEL1
5735 066230 005022                CLR      (R2)+          ;CLEAR SEL2 (UNUSED)
5736 066232 112122      10$: MOVB     (R1)+,(R2)+    ;STORE DATA PATTERN BYTE
5737 066234 005300                DEC      RO             ;DONE ALL BYTES?
5738 066236 003375                BGT     10$            ;BR IF NO
5739 066240 000207      RTS      PC           ;RETURN
5740
5741
5742      :+
5743      : SETUP T20PK2 FOR WRITE FORMAT TRANSPORT REGISTER
5744      : INPUT:
5745      :      RO CONTAINS DRIVING DATA PATTERN
5746

```

5747	066242			T20WFMT:		
5748	066242	004737	066102	JSR	PC,T20CLRBUF	:CLEAR MESSAGE BUFFER
5749	066246	012701	066540	MOV	#T20DT2,R1	:WRITE SUBSYSTEM DATA BUFFER
5750	066252	112721	000007	MOVB	#PW.WFMT,(R1)+	:STORE WRITE FORMAT IN BSELO
5751	066256	110021		MOVB	RO,(R1)+	:STORE DATA WORD IN BSEL1
5752	066260	000207		RTS	PC	:RETURN
5753						
5754				:+	SETUP T20PK2 PACKET FOR WRITE MISC.	
5755						
5756					RO CONTAINS WRITE MISC DATA	
5757						
5758	066262			T20WMISC:		
5759	066262	012701	066540	MOV	#T20DT2,R1	:WRITE SUBSYSTEM DATA BUFFER
5760	066266	112721	000010	MOVB	#PW.WMISC,(R1)+	:STORE WRITE MISCELLANEOUS IN BSELO
5761	066272	110011		MOVB	RO,(R1)	:STORE INVERT EXTENDED FEATURES IN BSEL1
5762	066274	000207		RTS	PC	:RETURN
5763						
5764				:+	SETUP T20PK2 PACKET FOR WRITE MISC. INVERT EXTENDED FEATURES SWITCH	
5765						
5766	066276			T20SEXT:		
5767	066276	012700	066540	MOV	#T20DT2,RO	:WRITE SUBSYSTEM DATA BUFFER
5768	066302	112720	000010	MOVB	#PW.WMISC,(RO)+	:STORE WRITE MISCELLANEOUS IN BSELO
5769	066306	112710	000200	MOVB	#MS.EXT,(RO)	:STORE INVERT EXTENDED FEATURES IN BSEL1
5770	066312	000207		RTS	PC	:RETURN
5771						
5772				:+	CLEAR EXPECTED DATA MESSAGE BUFFER	
5773						
5774	066314			T20CLEXP:		
5775	066314	012701	065042	MOV	#T20EXP,R1	:GET EXPD ADDRESS
5776	066320	012700	000120	MOV	#T20EXEND-T20EXP,RO	:GET EXPD SIZE
5777	066324	105021		10\$: CLRB	(R1)+	:CLEAR A BYTE
5778	066326	005300		DEC	RO	:DONE?
5779	066330	003375		BGT	10\$	:BR IF NO
5780	066332	000207		RTS	PC	:RETURN
5781						
5782				:+	Set WORDS 0-7 of expd message BUFFER = to recv since not testing	
5783						
5784						
5785	066334			T20SETEXP:		
5786	066334	012702	065042	MOV	#T20EXP,R2	:GET EXPD
5787	066340	012703	066402	MOV	#T20BFR,R3	:GET READ STATUS RECV BUFFER
5788	066344	012700	000010	MOV	#8,RO	:SET WORDS 0-7 EXP=RECV
5789	066350	012322		5\$: MOV	(R3)+,(R2)+	:SET EXPD=RECV
5790	066352	005300		DEC	RO	:DONE WORDS 0-7 WORDS?
5791	066354	003375		BGT	5\$	:BR IF NO
5792	066356	000207		RTS	PC	:RETURN
5793						
5794						
5795						
5799						
5800				;	WRITE CHARACTERISTICS COMMAND PACKET	
5801						
5802	066360			T20PACKET:		
5803	066360	100004		.WORD	100004	:COMMAND PACKET FOR TEST
5804	066362	066370		.WORD	T20DATA	:WRITE CHARACTERISTICS COMMAND, WITH ACK
5805	066364	000000		.WORD	0	:ADDRESS OF CHARACTERISTICS BLOCK
5806	066366	000012		.WORD	10.	:MINIMUM MESSAGE PACKET SIZE

```

5807
5808 066370          T20DATA:          ;CHARACTERISTICS DATA BLOCK
5809 066370 066402   .WORD T20BFR      ;ADDRESS OF MESSAGE BUFFER
5810 066372 000000   .WORD 0          ;
5811 066374 000024   .WORD 20.        ;LENGTH OF MESSAGE BUFFER
5812 066376 000000   .WORD 0          ;ESS,ENB,EAI,ERI
5813 066400 000007   .WORD 7          ;EXTENDED FEATURES UNIT NO.
5814
5815
5816                ;MESSAGE BUFFER FOR ALL TEST 17 COMMANDS
5817
5818 066402          T20BFR:          ;BEGIN MESSAGE BUFFER
5819 066402 000000   .WORD 0          ;MESSAGE TYPE
5820 066404 000000   .WORD 0          ;DATA FIELD LENGTH
5821 066406 000000   .WORD 0          ;RBPCR
5822 066410 000000   .WORD 0          ;XST0
5823 066412 000000   .WORD 0          ;XST1
5824 066414 000000   .WORD 0          ;XST2
5825 066416 000000   .WORD 0          ;XST3
5826 066420 000000   .WORD 0          ;XST4 (ALWAYS PRESENT FOR WRITE SUBSYSTEM
5827 066422          T20BFSTA: .BLKB 64. ;READ STATUS AND WRITE FIFO BUFFER
5828 066522          T20BEND:          ;END OF MESSAGE BUFFER
5829
5830                ;WRITE SUBSYSTEM READ STATUS COMMAND PACKET
5831
5833                ;
5835 066530          .=<. +10>&177770
5836 066530 100006   T20PK2:          ;WRITE SUBSYSTEM WITH ACK
5837 066532 066540   .WORD P.WRTSUB!P.ACK ;LOW ADDRESS OF DATA BLOCK
5838 066534 000000   .WORD T20DT2      ;HIGH ADDRESS OF DATA BLOCK
5839 066536 000012   .WORD 0          ;MINIMUM MESSAGE PACKET SIZE
5840
5841 066540          T20DT2:          ;DATA BLOCK
5842 066540          .BYTE 0          ;BSELO
5843 066541          .BYTE 0          ;BSEL1
5844 066542 000000   .WORD 0          ;SEL2
5845 066544          .BLKB 64.        ;WRITE FIFO DATA OUTPUT BUFFER
5846
5847
5848 066644          ENDTST
066644
066644 104401

```

L10073: TRAP CSETST



5850  
5851  
5852  
5853  
5854  
5855  
5856  
5857  
5858  
5859  
5860  
5861  
5862  
5863  
5864  
5865  
5866  
5867  
5868  
5869  
5370  
5871  
5872  
5873  
5874  
5875  
5876  
5877  
5878  
5879  
5880  
5881  
5882  
5883  
5884  
5885  
5886  
5887  
5888  
5889  
5890  
5891  
5892  
5893  
5894  
5895  
5896  
5897  
5898  
5899  
5900  
5901  
5902  
5903  
5904  
5905  
5906

.SBTTL TEST 10: MANUAL INTERVENTION

:THE MANUAL INTERVENTION TEST IS A STANDALONE ROUTINE (NOT REALLY A 'TEST')  
:THAT ALLOWS THE OPERATOR TO CHECK OUT VARIOUS ELEMENTS AND FUNCTIONS OF  
:THE SUBSYSTEM THAT CANNOT BE MANIPULATED BY THE PROGRAM ALONE. WHEN  
:THIS ROUTINE IS STARTED, IT FIRST PRINTS OUT A MENU OF SELECTABLE  
:SUBTESTS AND THEN WAITS FOR THE OPERATOR TO TYPE IN A SELECTION CODE.  
:THE ONLY WAYS TO EXIT THIS ROUTINE AND RETURN TO THE DIAGNOSTIC SUPERVISOR  
:ARE BY TYPING <CTRL-C> OR SELECTING CODE 7.  
:SELECTION CODES AND SUBROUTINES ARE:

CODE	ROUTINE
0	HELP. PRINTS THIS MENU.
1	TURN ON ALL M7196 LED INDICATORS
2	TURN OFF ALL M7196 LED INDICATORS
3	OFFLINE/ONLINE ATTENTION TEST
4	WRITE-PROTECT TEST
5	INITIATE TRANSPORT SERVO EXERCISER
6	PRINT EXTENDED TRANSPORT STATUS
7	EXIT (RETURN TO SUPERVISOR)

:EACH MENU ITEM CORRESPONDS TO A SUBTEST, AS FOLLOWS:

:PRINTS OUT THE MENU ON THE CONSOLE TERMINAL.

:CAUSES ALL THREE LED INDICATORS ON THE M7196 MODULE  
:TO BE ILLUMINATED. AFTER INITIATING THIS ROUTINE, THE OPERATOR  
:SHOULD OBSERVE THE LED'S AND VERIFY THAT THEY ARE INDEED ALL LIT.  
:THIS ROUTINE FIRST USES THE WRITE SUBSYSTEM MEMORY COMMAND TO  
:SET THE FORCE WRONG PARITY FLIP-FLOP, WHICH SERVES TO DRIVE THE  
:PROCESSOR NOT OK LED. THEN IT ENTERS A LOOP THAT CONTINUALLY  
:WRITES THE LOW BYTE OF TSDB AND READS THE TSSR. THESE LATTER TWO  
:OPERATIONS WILL CAUSE THE 'NOT SSR' AND 'DRIVING BUS' LED'S TO  
:GLOW -- THEY ARE NOT REALLY LIT AT ALL TIMES BUT SHOULD APPEAR  
:REASONABLY VISIBLE.

:INITIALIZES THE CONTROLLER TO CAUSE ALL LED'S TO  
:EXTINGUISH.

:THIS ROUTINE INITIALIZES THE CONTROLLER, ISSUES A  
:WRITE CHARACTERISTICS COMMAND TO ENABLE ATTENTION INTERRUPTS,  
:ISSUES A MESSAGE BUFFER RELEASE COMMAND, PRINTS A MESSAGE ON THE  
:CONSOLE TERMINAL INSTRUCTING THE OPERATOR TO TOGGLE THE ON-LINE  
:SWITCH ON THE TRANSPORT, THEN WAITS FOR AN ATTENTION INTERRUPT.  
:EACH TIME THE TRANSPORT TRANSITIONS FROM ON-LINE TO OFF-LINE OR  
:VICE-VERSA, AN ATTENTION INTERRUPT SHOULD BE GENERATED. THE PROGRAM  
:WILL REPORT THE INTERRUPT AND THE CURRENT STATE OF THE TRANSPORT.

```

5907 :THE OPERATOR SHOULD VERIFY THAT THE REPORTED STATE MATCHES THE
5908 :STATE INDICATED BY THE LED ON THE FRONT PANEL OF THE TRANSPORT.
5909 :IN ADDITION, WHEN THE TRANSPORT IS PLACED OFF-LINE, THE PROGRAM
5910 :ISSUES A SEQUENCE OF TAPE-MOTION COMMANDS (READ, WRITE, POSITION, ETC.
5911 :AND VERIFIES THAT, FOR EACH COMMAND, FUNCTION REJECT TERMINATION
5912 :RESULTS, ALONG WITH THE NON-EXECUTABLE FUNCTION (NEF) ERROR BIT BEING
5913 :SET.
5914 :
5915 :THIS ROUTINE INSTRUCTS THE OPERATOR TO MOUNT A SCRATCH
5916 :TAPE REEL THAT DOES NOT HAVE A WRITE-ENABLE RING INSTALLED, THEN
5917 :WAITS FOR THE OPERATOR TO RESPOND THAT THIS HAS BEEN ACCOMPLISHED.
5918 :UPON THE RESPONSE, THE PROGRAM VERIFIES THAT THE TRANSPORT SHOWS
5919 :A WRITE-PROTECTED STATUS, THEN ATTEMPTS TO WRITE DATA ON THE
5920 :TAPE AND EXPECTS THE APPROPRIATE ERROR TERMINATION INDICATING THAT
5921 :THE WRITE FUNCTION COULD NOT BE PERFORMED BECAUSE THE REEL IS
5922 :WRITE-PROTECTED. IF THE APPROPRIATE TERMINATION IS NOT RECEIVED,
5923 :AN ERROR IS REPORTED.
5924 :
5925 :
5926 :
5927 :INSTRUCTS THE OPERATOR TO PLACE THE TAPE TRANSPORT(S)
5928 :ON-LINE (IF ANY ARE OFF-LINE) THEN ATTEMPTS TO PERFORM AN EXTENDED
5929 :STATUS READOUT. FOR EACH TRANSPORT EQUIPPED WITH THIS FEATURE,
5930 :THE PROGRAM FORMATS AND PRINTS OUT THE RESULTING STATUS. IF THE
5931 :TRANSPORT IS NOT EQUIPPED WITH THIS FEATURE, A MESSAGE INDICATING
5932 :SUCH IS ISSUED.
5933 :
5934 :
5935 :
5936 :
5937 :
    
```

```

5937 066646          BGNTST
      066646
5942 066646          RFLAGS R0          T10::
      066646 104421          :GET OPERATOR FLAGS          TRAP CSRFLA
5943 066650 001403          BEQ 21$          :BR, IF OK TO RUN
5944 066652 012700 072230          MOV #T38NE,R0          :''TEST NOT EXECUTED''
5945 066656 000402          BR 3$          :JUMP IF NOT FIRST TEST
5946 066660          21$:
5947 066660 012700 073345          MOV #T38ID,R0          :TEST ID MESSAGE
5948 066664 004737 016510          JSR PC,TSTSETUP          :DO THE COMMON SETUP
5949 066670 004737 020500          JSR PC,CHKMAN          :IS MANUAL INTERVENTION ALLOWED?
5950 066674 103402          BCS 22$          :BR, IF MANUAL INTER ALLOWED
5951 066676 000137 071430          JMP 64$          :JUMP IF NOT ALLOWED
5952 066702          22$:
5956 066702 005037 002222          CLR FATFLG          :CLEAR THE FATAL ERROR FLAG
5957 066706 012737 176750 071442          MOV #65000.,T38DLY          :SET UP DELAY COUNTER
5958 066714 004737 015774          JSR PC,SOFINIT          :DO A SOFT INIT
5959 066720 103427          BCS 23$          :BRANCH IF OK
5960 066722 010001          MOV R0,R1          :CONTENTS OF TSSR REGISTER
5961 066724 032701 000200          BIT #SSR,R1          :CHECK FOR TSSR SET
5962 066730 001023          BNE 23$          :KEEP GOING IF NOT SET
5963 066732          DELAY 250          :CALL DELAY ROUTINE
      066732 012727 000250          MOV #250,(PC)+
      066736 000000          .WORD 0
      066740 013727 002116          MOV LSDLY,(PC)+
      066744 000000          .WORD 0
      066746 005367 177772          DEC -6(PC)
    
```

```

066752 001375
066754 005367 177756
066760 001367
5964 066762 005337 071442
5965 066766 001352
5966 066770
066770 104455
066772 001751
066774 003652
066776 012034
5967 067000 012700 073372
5968 067004 012701 000006
5969 067010 004737 020256
5970 067014 010004
5971 067016 006304
5972 067020 000174 067024
5973 067024 066702
5974 067026 067042
5975 067030 067324
5976 067032 067556
5977 067034 070212
5978 067036 071146
5979 067040 071424
5980 067042
067042 012746 073241
067046 012746 000001
067052 010600
067054 104417
067056 062706 000004
5981 067062 004737 073776
5982 067066 004737 015774
5983 067072 103405
5987 067074 010001
5988 067076
067076 104455
067100 001752
067102 003652
067104 012034
5989 067106 013737 002202 072170 100$:
5990
5991 067114 012704 072150
5992 067120 004737 010662
5993 067124 103405
5997 067126 010001
5998 067130
067130 104456
067132 001753
067134 005056
067136 012034
5999 067140
6000 067140 112737 000000 071461
6001 067146 112737 000011 071460
6002 067154 012704 071450
6003
6004
6005
6006 067160 010465 000000
    
```

```

DEC T38DLY :BUMP COUNTER DOWN
BNE 5$ :BR, IF MORE TIME LEFT
ERRDF ERRNO,SFIERR,SFIMSG :REPORT FATAL ERROR
TRAP C$ERDF
.WORD 1001
.WORD SFIERR
.WORD SFIMSG
23$: MOV #MIMENU,R0 :MENU OF MANUAL INTERVENTIONS
MOV #6,R1 :MAXIMUM ALLOWED SELECTION
JSR PC,GETSEL :GO GET THE OPERATORS SELECTION
MOV R0,R4 :GET NUMBER FROM ROUTINE
ASL R4 :CONVERT TO WORD OFFSET
JMP @6$(R4) :JUMP TO PROPER LOOP
6$: .WORD 2$ :RETYPE THE MENU
.WORD 10$ : 1 TURN ON LED'S
.WORD 15$ : 2 TURN OFF LED'S
.WORD 20$ : 3 ONLINE ATTENTION
.WORD 25$ : 4 WRITE PROTECT
.WORD 35$ : 5 EXTENDED TRANSPORT STATUS
.WORD 63$ : 6 LEAVE THE TEST
10$: PRINTF #T38MS2 :TELL OPERATOR TO CNTRL-C FOR EXIT
MOV #T38MS2,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #4,SP
JSR PC,T38REST :SET PACKET TO INITIAL VALUES
JSR PC,SOFINIT :DO SOFT INIT OF CONTROLLER
BCS 100$ :BR IF SOFT INIT = OK
MOV R0,R1 :SAVE CONTENTS OF TSSR
ERRDF ERRNO,SFIERR,SFIMSG :DEVICE FATAL ERROR DURING INIT
TRAP C$ERDF
.WORD 1002
.WORD SFIERR
.WORD SFIMSG
100$: MOV UNITN,T38DSW :SET UNIT NUMBER
5991 067114 012704 072150
5992 067120 004737 010662
5993 067124 103405
5997 067126 010001
5998 067130
067130 104456
067132 001753
067134 005056
067136 012034
110$: MOV #0,T38BS1 :CLEAR BIT #4
MOV #11,T38BS0 :WRITE MISC COMMAND
MOV #T38PACKET,R4 :SET UP NEW WRT. SUBSYS MEM. COMMAND
:NOTE: THIS COMMAND TURNS ON THE PROCESSOR FAIL LED
MOV R4,TSDB(R5) :SET THE PACKET ADDRESS
    
```

6007	067164	004737	016336		JSR	PC,CHKTSSR		:WAIT FOR SSR TO SET	
6008	067170	103405			BCS	150\$		:BR IF CARRY SET (GOOD RETURN)	
6009	067172	010001			MOV	R0,R1		:SAVE CONTENTS OF TSSR	
6013	067174				ERRDF	ERRNO,T38SSR,PKTSSR		:DEVICE FATAL SSR FAILED TO SET	
	067174	104455						TRAP	C\$ERDF
	067176	001754						.WORD	1004
	067200	072646						.WORD	T38SSR
	067202	012046						.WORD	PKTSSR
6014	067204			150\$:	CKLOOP			SET	
	067204	104406						TRAP	C\$CLP1
6015	067206				SETPRI	#PRI07			
	067206	012700	000340					MOV	#PRI07,R0
	067212	104441						TRAP	C\$SPRI
6016	067214	005037	071434		CLR	TTION2			
6017	067220	032737	000100	177560	BIT	#100,@#TTICSR		:ASSUME INTERRUPTS ARE ENABLED	
6018	067226	001005			BNE	701\$		:ARE TTI INTERRUPTS ON ?	
6019	067230	005237	071434		INC	TTION2		:BRANCH IF YES	
6020	067234	052737	000100	177560	BIS	#100,@#TTICSR		:FLAG SET IF INTERRUPTS OFF	
6021	067242	012701	000060		MOV	#TTIVEC,R1		:ENABLE INTERRUPTS	
6022	067246	011137	071436		MOV	(R1),TVSAV2		:START OF TTI VECTORS	
6023	067252	012721	070730		MOV	#590\$, (R1)+		:SAVE THE CURRENT TTI VECTOR	
6024	067256	011137	071440		MOV	(R1),TPSAV2		:SET NEW INTERRUPT ROUTINE	
6025	067262	012711	000340		MOV	#PRI07,(R1)		:SAVE THE VECTOR PRIORITY	
6026	067266				SETPRI	#PRI00		:USE PRIORITY SEVEN	
	067266	012700	000000					:LOWER INTERRUPT BR LEVEL	
	067272	104441						MOV	#PRI00,R0
6027	067274	012701	177777		MOV	#-1,R1		TRAP	C\$SPRI
6028	067300	000240			NOP				
6029	067302	012702	001750		MOV	#1000.,R2		:DATA TO WRITE TO TSDB	
6030	067306	110165	000000		MOV	R1,TSDB(R5)		:ALLOW OPERATOR TO TYPE ^C	
6031	067312	016500	000002		MOV	TSSR(R5),R0		:SET-UP INNER LOOP	
6032	067316	005302			MOV	R2		:WRITE DATA TO TSDB	
6033	067320	001372			DEC	R2		:READ TSSR	
6034	067322	000766			BNE	14\$		:REDUCE INNER COUNT	
6035					BR	12\$		:LOOP TILL EXPIRES	
6036	067324							:LOOP UNTIL HALTED	
	067324	012746	073241		15\$:	PRINTF	#T38MS2		
	067330	012746	000001					:TYPE CNTL C TO EXIT	
	067334	010600						MOV	#T38MS2,-(SP)
	067336	104417						MOV	#1,-(SP)
	067340	062706	000004					MOV	SP,R0
6037	067344	004737	015774					TRAP	C\$PNTF
6038	067350	103405			JSR	PC,SOFINIT		ADD	#4,SP
6042	067352	010001			BCS	200\$			
6043	067354				MOV	R0,R1		:DO SOFT INIT OF CONTROLLER	
	067354	104455			ERRDF	ERRNO,SFIERR,SFIMSG		:BR IF SOFT INIT = OK	
	067356	001755						:SAVE CONTENTS OF TSSR	
	067360	003652						:DEVICE FATAL ERROR DURING INIT	
	067362	012034						TRAP	C\$ERDF
6044	067364							.WORD	1005
6045	067364	013737	002202	072170	200\$:	MOV	UNITN,T38DSW		
6046	067372	012704	072150		MOV	#T38PK2,R4		:SET UNIT NUMBER	
6047	067376	004737	010662		JSR	PC,WRTCHR		:SUBROUTINE NEEDS PACKET ADDRESS	
6048	067402	103405			BCS	210\$		:ISSUE WRITE CHARACTERISTICS	
6052	067404	010001			MC	R0,R1		:BR, IF COMMAND ISSUED OK	
6053	067406				ERRHRD	ERRNO,WRTMSG,SFIMSG		:SAVE CONTENTS OF TSSR	
	067406	104456						:WRITE CHARACTERISTIC FAILED	
								TRAP	C\$ERHRD

067410	001756								.WORD	1006
067412	005056								.WORD	WRTMSG
067414	012034								.WORD	SFIMSG
6054										
6055										
6056										
6057										
6058										
6059	067416									
6060	067416	112737	000000	071461						
6061	067424	112737	000025	071460						
6062	067432	012704	071450							
6063	067436	010465	000000							
6064	067442	004737	016336							
6065	067446	103405								
6066	067450	010001								
6070	067452									
	067452	104455								
	067454	001757							TRAP	C\$ERDF
	067456	072646							.WORD	1007
	067460	012046							.WORD	T38SSR
6071	067462								.WORD	PKTSSR
	067462	104406							TRAP	C\$CLP1
6072	067464									
	067464	012700	000340						MOV	#PRI07,RO
	067470	104441							TRAP	C\$SPRI
6073	067472	005037	071434							
6074	067476	032737	000100	177560						
6075	067504	001005								
6076	067506	005237	071434							
6077	067512	052737	000100	177560						
6078	067520	012701	000060							
6079	067524	011137	071436							
6080	067530	012721	070730							
6081	067534	011137	071440							
6082	067540	012711	000340							
6083	067544									
	067544	012700	000000							
	067550	104441								
6084	067552	000240								
6085	067554	000776								
6086										
6087										
6088	067556									
	067556	012746	073241							
	067562	012746	000001							
	067566	010600								
	067570	104417								
	067572	062706	000004							
6089	067576									
	067576	012700	000000							
	067602	104441								
6090	067604	005037	002224							
6091	067610	004737	015774							
6092	067614	103405								
6096	067616	010001								
6097	067620									

```

*****
THIS WRITE SUB-SYSTEM MEMORY COMMAND JUST HOLDS THE LEDS OFF
*****

```

```

210$:
MOV #0,T38BS1      ;CLEAR BIT #4
MOV #25,T38BS0     ;STOP DRIVE TEST 22
MOV #T38PACKET,R4 ;SET UP NEW WRT. SUBSYS MEM. COMMAND
MOV R4,TSDB(R5)    ;SET THE PACKET ADDRESS
JSR PC,CHKTSSR     ;WAIT FOR SSR TO SET
BCS 250$           ;BR IF CARRY SET (GOOD RETURN)
MOV R0,R1          ;SAVE CONTENTS OF TSSR
ERRDF ERRNO,T38SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
TRAP C$ERDF
WORD 1007
WORD T38SSR
WORD PKTSSR

250$: CKLOOP      ;LOOP ON ERROR, IF FLAG SET
TRAP C$CLP1
SETPRI #PRI07     ;RAISE THE PRIORITY
MOV #PRI07,RO
TRAP C$SPRI

CLR TTION2        ;ASSUME INTERRUPTS ARE ENABLED
BIT #100,@#TTICSR ;ARE TTI INTERRUPTS ON ?
BNE 710$          ;BRANCH IF YES
INC TTION2        ;FLAG SET IF INTERRUPTS OFF
BIS #100,@#TTICSR ;ENABLE INTERRUPTS
MOV #TTIVEC,R1    ;START OF TTI VECTORS
MOV (R1),TVSAV2   ;SAVE THE CURRENT TTI VECTOR
MOV #590$, (R1)+ ;SET NEW INTERRUPT ROUTINE
MOV (R1),TPSAV2   ;SAVE THE VECTOR PRIORITY
MOV #PRI07,(R1)   ;USE PRIORITY SEVEN
SETPRI #PRI00     ;LOWER INTERRUPT BR LEVEL
MOV #PRI00,RO
TRAP C$SPRI

260$: NOP         ;ALLOW CNTL C
BR 260$           ;LOOP UNTIL STOPPED

20$: PRINTF #T38MS2 ;TELL'EM WHAT TO TYPE
MOV #T38MS2,-(SP)
MOV #1,-(SP)
MOV SP,RO
TRAP C$PNTF
ADD #4,SP
SETPRI #PRI00     ;LOWER PRIORITY TO ALLOW INTERRUPTS
MOV #PRI00,RO
TRAP C$SPRI

CLR INTRECV       ;CLEAR INTERRUPT RECEIVED FLAG
JSR PC,SOFINIT    ;DO SOFT INIT OF CONTROLLER
BCS 300$          ;BR IF SOFT INIT = OK
MOV R0,R1         ;SAVE CONTENTS OF TSSR
ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT

```

067620	104455								TRAP	C\$ERDF
067622	001760								.WORD	1008
067624	003652								.WORD	SFIERR
067626	012034								.WORD	SFIMSG
6098	067630			300\$:						
6099	067630	013737	002202	072170	MOV	UNITN,T38DSW				;SET UNIT NUMBER IN PACKET
6100	067636	012737	000040	072166	MOV	#BITS,T38EAI				;ENABLE ATTENTION INTERRUPTS
6101	067644	012704	072150		MOV	#T38PK2,R4				;SUBROUTINE NEEDS PACKET ADDRESS
6102	067650	004737	010662		JSR	PC,WRTCHR				;ISSUE WRITE CHARACTERISTICS
6103	067654	103405			BCS	310\$				;BR, IF COMMAND ISSUED OK
6107	067656	010001			MOV	R0,R1				;SAVE CONTENTS OF TSSR
6108	067660				ERRHRD	ERRNO,WRTMSG,SFIMSG				;WRITE CHARACTERISTIC FAILED
	067660	104456							TRAP	C\$ERHRD
	067662	001761							.WORD	1009
	067664	005056							.WORD	WRTMSG
	067666	012034							.WORD	SFIMSG
6109	067670			310\$:						
6110	067670	012704	072200		MOV	#T38PK3,R4				;SET UP NEW PACKET FOR MESS BUF REL
6111	067674	010465	000000		MOV	R4,TSDB(R5)				;MESSAGE BUFFER RELEASE,ACK,CVC=1 CMD
6112	067700	004737	016250		JSR	PC,WAITF				;WAIT FOR SSR TO SET
6113	067704	005002			CLR	R2				;MAKE SURE ALL IS CLEAR
6114	067706	016501	000002		MOV	TSSR(R5),R1				;GET TSSR STATUS
6115	067712	032701	000100		BIT	#OFL,R1				;IS OFL SET
6116	067716	001402			BEQ	320\$				;BR, IF OFL IS NOT SET
6117	067720	052702	000100		BIS	#OFL,R2				;SET OFL IN EXPECTED
6118	067724	052702	000200	320\$:	BIS	#SSR,R2				;SET UP EXPECTED
6119	067730	020201			CMP	R2,R1				;IS EVERYTHING OK
6120	067732	001404			BEQ	350\$				;BR, IF ALL IS WELL
6124	067734				ERRHRD	ERRNO,T38SST,PKTSSR				;DEVICE FATAL SSR FAILED TO SET
	067734	104456							TRAP	C\$ERHRD
	067736	001762							.WORD	1010
	067740	073056							.WORD	T38SST
	067742	012046							.WORD	PKTSSR
6125	067744			350\$:	CKLOOP					;LOOP ON ERROR, IF FLAG SET
6126	067744	104406			PRINTF	#T38MS1			TRAP	C\$CLP1
	067746	012746	073146						MOV	#T38MS1,-(SP)
	067752	012746	000001						MOV	#1,-(SP)
	067756	010600							MOV	SP,R0
	067760	104417							TRAP	C\$PNTF
	067762	062706	000004						ADD	#4,SP
6127	067766				PRINTF	#T38MS2				;TELL OPERATOR TO DO ^C TO EXIT
	067766	012746	073241						MOV	#T38MS2,-(SP)
	067772	012746	000001						MOV	#1,-(SP)
	067776	010600							MOV	SP,R0
	070000	104417							TRAP	C\$PNTF
	070002	062706	000004						ADD	#4,SP
6128	070006				SETPRI	#PRI07				;RAISE THE PRIORITY
	070006	012700	000340						MOV	#PRI07,R0
	070012	104441							TRAP	C\$SPRI
6129	070014	005037	071434		CLR	TTION2				;ASSUME INTERRUPTS ARE ENABLED
6130	070020	032737	000100	177560	BIT	#100,@#TTICSR				;ARE TTI INTERRUPTS ON ?
6131	070026	001005			BNE	720\$				;BRANCH IF YES
6132	070030	005237	071434		INC	TTION2				;FLAG SET IF INTERRUPTS OFF
6133	070034	052737	000100	177560	BIS	#100,@#TTICSR				;ENABLE INTERRUPTS
6134	070042	012701	000060		MOV	#TTIVEC,R1				;START OF TTI VECTORS
6135	070046	011137	071436	720\$:	MOV	(R1),TVSAV2				;SAVE THE CURRENT TTI VECTOR

6136	070052	012721	070730	MOV	#590\$, (R1)+		:SET NEW INTERRUPT ROUTINE	
6137	070056	011137	071440	MOV	(R1), TPSAV2		:SAVE THE VECTOR PRIORITY	
6138	070062	012711	000340	MOV	#PRI07, (R1)		:USE PRIORITY SEVEN	
6139	070066			SETPRI	#PRI00		:LOWER INTERRUPT BR LEVEL	
	070066	012700	000000					MOV #PRI00, R0
	070072	104441						TRAP C\$SPRI
6140	070074	000240		360\$:	NOP		:ALLOW CONTROL C	
6141	070076	005737	002224		TST INTRECV		:DID AN INTERRUPT OCCUR ?	
6142	070102	001001			BNE 370\$		:BRANCH IF YES	
6143	070104	000773			BR 360\$		:WAIT SOME MORE FOR INTERRUPT	
6144	070106			370\$:	PRINTF #T38INT		: "INTERRUPT RECEIVED"	
	070106	012746	072736					MOV #T38INT, -(SP)
	070112	012746	000001					MOV #1, -(SP)
	070116	010600						MOV SP, R0
	070120	104417						TRAP C\$PNTF
	070122	062706	000004					ADD #4, SP
6145	070126	016501	000002	MOV	TSSR(R5), R1		:READ TSSR STATUS	
6146	070132	032701	000100	BIT	#OFL, R1		:CHECK THE OFF-LINE BIT	
6147	070136	001011		BNE	380\$		:BR, IF DRIVE IS OFF-LINE	
6148	070140			PRINTF	#T38ONL		: "DRIVE IS NOW ON-LINE"	
	070140	012746	072766					MOV #T38ONL, -(SP)
	070144	012746	000001					MOV #1, -(SP)
	070150	010600						MOV SP, R0
	070152	104417						TRAP C\$PNTF
	070154	062706	000004					ADD #4, SP
6149	070160	000410		BR	390\$		:ALMOST DONE	
6150	070162			380\$:	PRINTF #T38OFL		: "DRIVE IS NOW OFF-LINE"	
	070162	012746	073022					MOV #T38OFL, -(SP)
	070166	012746	000001					MOV #1, -(SP)
	070172	010600						MOV SP, R0
	070174	104417						TRAP C\$PNTF
	070176	062706	000004					ADD #4, SP
6151	070202	005037	002224	390\$:	CLR INTRECV		:CLEAR INTERRUPT FLAG	
6152	070206	000137	067630		JMP 300\$		:TRY AGAIN	
6153	070212			25\$:	GMANIL T38MSG, T38DAT, -1, NO		:WAIT FOR OPERATOR TO MOUNT TAPE	
	070212	104443						TRAP C\$GMAN
	070214	000404						BR 10000\$
	070216	073774						.WORD T38DAT
	070220	000120						.WORD T\$CODE
	070222	073305						.WORD T38MSG
	070224	177777						.WORD -1
	070226							
6154	070226				BNCOMPLETE 25\$		:RETRY IF ERROR	10000\$:
	070226	103371						BCC 25\$
6155	070230	005737	073774	TST	T38DAT		:DID OPERATOR SAY 'YES' ?	
6156	070234	001002		BNE	27\$		:BRANCH IF YES	
6157	070236	000137	066702	JMP	2\$		:RETURN TO MAIN MENU	
6158	070242			27\$:				
6159	070242	004737	015774	JSR	PC, SFINIT		:DO SOFT INIT OF CONTROLLER	
6160	070246	103405		BCS	400\$		:BR IF SOFT INIT = OK	
6164	070250	010001		MOV	R0, R1		:SAVE CONTENTS OF TSSR	
6165	070252			ERRDF	ERRNO, SFIERR, SFIMSG		:DEVICE FATAL ERROR DURING INIT	
	070252	104455						TRAP C\$ERDF
	070254	001763						.WORD 1011
	070256	003652						.WORD SFIERR
	070260	012034						.WORD SFIMSG
6166	070262			400\$:	CKLOOP		:LOOP IF SELECTED	

6167	070262	104406	002202	072170	MOV	UNITN,T38DSW	:SET UNIT NUMBER	TRAP	C\$CLP1
6168	070272	012704	072150		MOV	#T38PK2,R4	:SUBROUTINE NEEDS PACKET ADDRESS		
6169	070276	004737	010662		JSR	PC,WRTCHR	:ISSUE WRITE CHARACTERISTICS		
6170	070302	103405			BCS	410\$	:BR, IF COMMAND ISSUED OK		
6174	070304	010001			MOV	R0,R1	:SAVE CONTENTS OF TSSR		
6175	070306				ERRHRD	ERRNO,WRTMSG,SFIMSG	:WRITE CHARACTERISTIC FAILED		
	070306	104456						TRAP	C\$ERHRD
	070310	001764						.WORD	1012
	070312	005056						.WORD	WRTMSG
	070314	012034						.WORD	SFIMSG
6176	070316				410\$:	CKLOOP	:LOOP IF SELECTED		
	070316	104406						TRAP	C\$CLP1
6177	070320	013701	071474		MOV	T38BFR+6,R1	:PICK UP XSTO CONTENTS		
6178	070324	010102			MOV	R1,R2	:SET UP EXPECTED		
6179	070326	052702	000004		BIS	#BIT2,R2	:SET UP THE WRITE LOCKED BIT		
6180	070332	020102			CMP	R1,R2	:ARE THEY CORRECT		
6181	070334	001406			BEQ	430\$	:BR, IF ALL IS WELL (OK)		
6185	070336				ERRHRD	ERRNO,T38WRL,EXPREC	:WRITE LOCKED BIT IS NOT SET ETC."		
	070336	104456						TRAP	C\$ERHRD
	070340	001765						.WORD	1013
	070342	072464						.WORD	T38WRL
	070344	015474						.WORD	EXPREC
6186	070346	005237	002222		INC	FATFLG	:SET FATAL FLAG		
6187	070352				430\$:	CKLOOP	:LOOP IF SELECTED		
	070352	104406						TRAP	C\$CLP1
6188	070354	005737	002222		TST	FATFLG	:WAS THE DRIVE NOT WRITE LOCKED		
6189	070360	001402			BEQ	435\$	:BR, IF FLAG NOT SET		
6190	070362	000137	066702		JMP	2\$	:RE-WRITE MENU		
6191	070366	017737	112532	072222	435\$:	MOV	@FREE,T38WR	:SET UP WRITE BUFFER ADDRESS	
6192	070374	012704	072220		MOV	#T38PK4,R4	:GET PACKET ADDRESS		
6193	070400	010465	000000		MOV	R4,TSDB(R5)	:SET THE PACKET ADDRESS		
6194	070404	004737	016250		JSR	PC,WAITF	:WAIT FOR SSR TO SET		
6195	070410	016501	000002		MOV	TSSR(R5),R1	:GET TSSR		
6196	070414	012702	100206		MOV	#SC!SSR!BIT1!BIT2,R2	:SET UP EXPECTED		
6197	070420	020102			CMP	R1,R2	:ARE THEY EQUAL (CORRECT)		
6198	070422	001404			BEQ	440\$	:BR, IF CORRECT STATUS		
6202	070424				ERRHRD	ERRNO,T38WRT,PKTSSR	:TSSR INCORRECT AFTER WRITE COMMAND		
	070424	104456						TRAP	C\$ERHRD
	070426	001766						.WORD	1014
	070430	072400						.WORD	T38WRT
	070432	012046						.WORD	PKTSSR
6203	070434				440\$:	CKLOOP	:LOOP ON ERROR, IF FLAG SET		
	070434	104406						TRAP	C\$CLP1
6204	070436	013701	071474		MOV	T38BFR+6,R1	:READ XSTO CONTENTS		
6205	070442	010102			MOV	R1,R2	:SET UPR EXPECTED		
6206	070444	052702	004000		BIS	#BIT11,R2	:SET THE WRITE LOCK ERROR BIT (XSTO)		
6207	070450	020102			CMP	R1,R2	:WAS THE BIT SET		
6208	070452	001404			BEQ	450\$	:BR, IF IT WAS (GOOD)		
6212	070454				ERRHRD	ERRNO,T38WLE,EXPREC	:WRITE LOCK ERROR BIT NOT SET"		
	070454	104456						TRAP	C\$ERHRD
	070456	001767						.WORD	1015
	070460	072525						.WORD	T38WLE
	070462	015474						.WORD	EXPREC
6213	070464				450\$:	CKLOOP	:LOOP IF SELECTED		
	070464	104406						TRAP	C\$CLP1
6214	070466	000137	066702		JMP	2\$	:GO BACK TO MENU		



```

6215
6216
6217
6218
6219 070472
6220 070472
    070472 012746 072305
    070476 012746 000001
    070502 010600
    070504 104414
    070506 062706 000004
6221 070512 004737 073776
6222 070516 004737 015774
6223 070522 103405
6227 070524 010001
6228 070526
    070526 104455
    070530 001770
    070532 003652
    070534 012034
6229 070536 013737 002202 072170 500$:
6230 070544 012704 072150
6231 070550 004737 010662
6232 070554 103405
6236 070556 010001
6237 070560
    070560 104456
    070562 001771
    070564 005056
    070566 012034
6238 070570
6239 070570 112737 000000 071461 510$:
6240 070576 112737 000020 071460
6241 070604 012704 071450
6242 070610 010465 000000
6243 070614 004737 016336
6244 070620 103405
6245 070622 010001
6249 070624
    070624 104455
    070626 001772
    070630 072646
    070632 012046
6250 070634
    070634 104406
6251 070636
    070636 012700 000340
    070642 104441
6252 070644 005037 071434
6253 070650 032737 000100 177560
6254 070656 001005
6255 070660 005237 071434
6256 070664 052737 000100 177560
6257 070672 012701 000060
6258 070676 011137 071436
6259 070702 012721 070730
6260 070706 011137 071440

```

```

:*****
:SERVO EXERCISER NO LONGER USED
:*****

```

```

30$: PRINTB #T38MS3 ;'EXE ANY OTHER MENU SELECTION TO STOP
    MOV #T38MS3,-(SP)
    MOV #1,-(SP)
    MOV SP,R0
    TRAP C$PNTB
    ADD #4,SP
    JSR PC,T38REST ;SET PACKET TO INITIAL VALUES
    JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
    BCS 500$ ;BR IF SOFT INIT = OK
    MOV R0,R1 ;SAVE CONTENTS OF TSSR
    ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
    TRAP C$ERDF
    .WORD 1016
    .WORD SFIERR
    .WORD SFIMSG
    MOV UNITN,T38DSW ;SET UNIT NUMBER
    MOV #T38PK2,R4 ;SUBROUTINE NEEDS PACKET ADDRESS
    JSR PC,WRTCHR ;ISSUE WRITE CHARACTERISTICS
    BCS 510$ ;BR, IF COMMAND ISSUED OK
    MOV R0,R1 ;SAVE CONTENTS OF TSSR
    ERRHRD ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTIC FAILED
    TRAP C$ERHRD
    .WORD 1017
    .WORD WRTMSG
    .WORD SFIMSG
    MOVB #0,T38BS1 ;CLEAR BIT #4
    MOVB #20,T38BS0 ;EXECUTE DRIVE TEST 22
    MOV #T38PACKET,R4 ;SET I/P NEW WRT. SUBSYS MEM. COMMAND
    MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS
    JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
    BCS 550$ ;BR IF CARRY SET (GOOD RETURN)
    MOV R0,R1 ;SAVE CONTENTS OF TSSR
    ERRDF ERRNO,T38SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
    TRAP C$ERDF
    .WORD 1018
    .WORD T38SSR
    .WORD PKTSSR
    550$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
    SETPRI #PRI07 ;RAISE THE PRIORITY
    MOV #PRI07,R0
    TRAP C$SPRI
    CLR TTION2 ;ASSUME INTERRUPTS ARE ENABLED
    BIT #100,#TTICSR ;ARE TTI INTERRUPTS ON ?
    BNE 555$ ;BRANCH IF YES
    INC TTION2 ;FLAG SET IF INTERRUPTS OFF
    BIS #100,#TTICSR ;ENABLE INTERRUPTS
    555$: MOV #TTIVEC,R1 ;START OF TTI VECTORS
    MOV (R1),TVSAV2 ;SAVE THE CURRENT TTI VECTOR
    MOV #590$,(R1)+ ;SET NEW INTERRUPT ROUTINE
    MOV (R1),TPSAV2 ;SAVE THE VECTOR PRIORITY

```

```

6261 070712 012711 000340          MOV    #PRI07,(R1)          ;USE PRIORITY SEVEN
6262 070716          SETPRI #PRI00             ;LOWER INTERRUPT BR LEVEL
        070716 012700 000000          MOV    #PRI00,R0
        070722 104441          TRAP  C$SPRI
6263 070724 000240          560$: NOP                ;LOOP AWHILE
6264 070726 000776          BR     560$              ;STAY IN 'TIGHT' LOOP
6265          ;+
6266          ;PROCESS CONSOLE INTERRUPTS
6267          ;-
6268
6269 070730 010046          590$: MOV    R0,-(SP)      ;SAVE WORK REGISTER
6270 070732 113700 177562          MOVB  @#TIBFR,R0        ;GET THE OPERATOR INPUT
6271 070736 042700 000200          BIC   #200,R0          ;STRIP OFF PARITY BIT
6272 070742 122700 000015          CMPB  #15,R0           ;IS IT A CARRIAGE RETURN ?
6273 070746 001075          BNE   591$             ;JUST EXIT IF NOT
6274 070750 012766 066702 000002          MOV   #2$,2(SP)        ;RETURN TO MASTER MENU
6275 070756 005066 000004          CLR  4(SP)             ;FORCE PRIORITY 0
6276 070762 013737 071436 000060          MOV   TVSAV2,@#TTIVEC ;RESTORE VECTOR
6277 070770 013737 071440 000062          MOV   TPSAV2,@#TTIVEC+2 ;RESTORE SUPER PRIORITY
6278 070776 112737 000025 071460          MOVB  #25,T38BS0       ;STOP DRIVE TEST 22
6279 071004 112737 000000 071461          MOVB  #0,T38BS1        ;CLEAR BS1
6280 071012 012704 071450          MOV   #T38PACKET,R4    ;SET UP NEW WRT. SUBSYS MEM. COMMAND
6281 071016 010465 000000          MOV   R4,T$DB(R5)      ;SET THE PACKET ADDRESS
6282 071022 012737 176750 071442          MOV   #65000,T38DLY    ;SET UP DELAY COUNTER
6283 071030 004737 016250          592$: JSR   PC,WAITF     ;DO A WAIT FOR SSR
6284 071034 016501 000002          MOV   TSSR(R5),R1      ;CONTENTS OF TSSR REGISTER
6285 071040 032701 000200          BIT   #SSR,R1          ;CHECK FOR TSSR SET
6286 071044 001017          BNE  595$              ;KEEP GOING IF NOT SET
6287 071046          DELAY 250             ;CALL DELAY ROUTINE
        071046 012727 000250          MOV   #250,(PC)+
        071052 000000          .WORD 0
        071054 013727 002116          MOV   L$DLY,(PC)+
        071060 000000          .WORD 0
        071062 005367 177772          DEC  -6(PC)
        071066 001375          BNE  -4
        071070 005367 177756          DEC  -22(PC)
        071074 001367          BNE  -20
6288 071076 005337 071442          DEC  T38DLY            ;BUMP COUNTER DOWN
6289 071102 001352          BNE  592$              ;BR, IF MORE TIME LEFT
6290 071104 004737 016336          595$: JSR   PC,CHKTSSR    ;WAIT FOR SSR TO SET
6291 071110 103405          BCS  580$              ;BR IF CARRY SET (GOOD RETURN)
6292 071112 010001          MOV   R0,R1            ;SAVE CONTENTS OF TSSR
6296 071114          ERRDF ERRNO,T38SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
        071114 104455          TRAP C$ERDF
        071116 001773          .WORD 1019
        071120 072646          .WORD T38SSR
        071122 012046          .WORD PKTSSR
6297 071124          580$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
        071124 104406          TRAP C$CLP1
6298 071126 005737 071434          TST  TTION2            ;ARE SUPER INTERRUPTS ENABLED
6299 071132 001403          BEQ  591$              ;BR, IF YES
6300 071134 042737 000100 177560          BIC  #100,@#TTICSR     ;RESTORE REGISTER
6301 071142 012600          591$: MOV   (SP)+,R0      ;RESTORE REGISTER
6302 071144 000002          RTI                    ;RETURN
6303 071146          35$:
6304 071146 004737 073776          JSR   PC,T38REST       ;SET PACKET TO INITIAL VALUES
6305 071152 004737 015774          JSR   PC,SOFINIT       ;DO SOFT INIT OF CONTROLLER
    
```

```

6306 071156 103405      BCS      600$      ;BR IF SOFT INIT = OK
6310 071160 010001      MOV      RO,R1      ;SAVE CONTENTS OF TSSR
6311 071162      ERRDF     ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
        071162 104455      TRAP     C$ERDF
        071164 001774      .WORD   1020
        071166 003652      .WORD   SFIERR
        071170 012034      .WORD   SFIMSG
6312 071172      600$:    CKLOOP      ;LOOP IF SELECTED
        071172 104406      TRAP     C$CLP1
6313 071174 012701 071466      MOV      #T38BFR,R1 ;ADDRESS OF MESSAGE BUFFER
6314 071200 012702 125252      MOV      #125252,R2 ;ALTERNATING 1'S AND 0'S
6315
6316 071204 010221      601$:    MOV      R2,(R1)+ ;CLEAR OUT THE MESSAGE BUFFER
6317 071206 022701 072142      CMP      #T38EB,R1 ;END OF BUFFER YET
6318 071212 001401      BEQ     605$      ;BR, IF AT END OF BUFFER
6319 071214 000773      BR      601$      ;NOT AT END KEEP GOING
6320 071216 013737 002202 072170 605$:    MOV      UNITN,T38DSW ;SET UNIT NUMBER
6321 071224 012704 072150      MOV      #T38PK2,R4 ;SUBROUTINE NEEDS PACKET ADDRESS
6322 071230 004737 010662      JSR     PC,WRTCHR ;ISSUE WRITE CHARACTERISTICS
6323 071234 103405      BCS     610$      ;BR, IF COMMAND ISSUED OK
6327 071236 010001      MOV      RO,R1      ;SAVE CONTENTS OF TSSR
6328 071240      ERRHRD   ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTICSC FAILED
        071240 104456      TRAP     C$ERHRD
        071242 001775      .WORD   1021
        071244 005056      .WORD   WRTMSG
        071246 012034      .WORD   SFIMSG
6329 071250      610$:    CKLOOP      ;LOOP IF SELECTED
        071250 104406      TRAP     C$CLP1
6330 071252 112737 000000 071461      MOVB    #0,T38BS1 ;CLEAR BIT #4
6331 071260 112737 000024 071460      MOVB    #24,T38BS0 ;READ EXTENDED DRIVE STATUS
6332 071266 012704 071450      MOV     #T38PACKET,R4 ;SET UP NEW WRT. SUBSYS MEM. COMMAND
6333 071272 010465 000000      MOV     R4,TSDB(R5) ;SET THE PACKET ADDRESS
6334 071276 012737 000144 071442      MOV     #100.,T38DLY ;SET UP DELAY ROUTINE
6335 071304 004737 016250      JSR     PC,WAITF ;WAIT AWHILE FOR SSR TO SET
6336 071310 016501 000002      MOV     TSSR(R5),R1 ;SEE IF IT REALLY DID
6337 071314 032701 000200      BIT     #SSR,R1 ;JUST CHECK THAT BIT
6338 071320 001017      BNE     630$      ;BR, IF SSR IS SET
6339 071322      DELAY    250 ;DELAY ABOUT .25 SEC
        071322 012727 000250      MOV     #250,(PC)+
        071326 000000      .WORD   0
        071330 013727 002116      MOV     LSDLY,(PC)+
        071334 000000      .WORD   0
        071336 005367 177772      DEC     -6(PC)
        071342 001375      BNE     -4
        071344 005367 177756      DEC     -22(PC)
        071350 001367      BNE     -20
6340 071352 005337 071442      DEC     T38DLY ;START DELAY COUNT DOWN
6341 071356 001352      BNE     620$      ;BR, IF COUNTER IS NOT AT DONE
6342 071360 004737 016336      JSR     PC,CHKTSSR ;WAIT FOR SSR TO SET
6343 071364 103405      BCS     650$      ;BR IF CARRY SET (GOOD RETURN)
6344 071366 010001      MOV     RO,R1      ;SAVE CONTENTS OF TSSR
6348 071370      ERRDF     ERRNO,T38SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
        071370 104455      TRAP     C$ERDF
        071372 001776      .WORD   1022
        071374 072646      .WORD   T38SSR
        071376 012046      .WORD   PKTSSR
6349 071400      650$:    CKLOOP      ;LOOP ON ERROR, IF FLAG SET
    
```

6350	071400	104406							
6351	071402	012700	071506	MOV	#T38BFR+20,R0			TRAP	C\$CLP1
6352	071406	005001		CLR	R1				
6353	071410	005037	003134	CLR	KTENABLE				
6354	071414	004737	074034	JSR	PC,T38MBP				
6355	071420	000137	066702	JMP	2\$				
6356									
6357	071424	000137	000200	63\$: JMP	200				
6358	071430			64\$: EXIT	TST				
	071430	104432							
	071432	003054						TRAP	C\$EXIT
6359								.WORD	L10075-
6360									
6361									
6362									
6363									
6364									
6365									
6366									
6367									
6368									
6369									
6370	071434	000000		TTION2:	.WORD	0			
6371	071436	000000		TVSAV2:	.WORD	0			
6372	071440	000000		TPSAV2:	.WORD	0			
6373									
6374	071442	000000		T38DLY:	.WORD	0			
6376		071450							
6378	071450			T38PACKET:					
6379	071450	140006			.WORD	140006			
6380	071452	071460			.WORD	T38TAD			
6381	071454	000000			.WORD	0			
6382	071456	000012			.WORD	10.			
6383	071460			T38TAD:					
6384	071460	000		T38BS0:	.BYTE	0			
6385	071461	000		T38BS1:	.BYTE	0			
6386	071462	000000		T38BS2:	.WORD	0			
6387	071464	000000			.WORD	0			
6388	071466			T38BFR:	.BLKW	150.			
6389	072142	000000		T38EB:	.WORD				
6390									
6391									
6393		072150							
6395	072150			T38PK2:					
6396	072150	140004			.WORD	140004			
6397	072152	072160			.WORD	T38DTA			
6398	072154	000000			.WORD	0			
6399	072156	000012			.WORD	10.			
6400									
6401									
6402	072160			T38DTA:					
6403	072160	071466			.WORD	T38BFR			
6404	072162	000000			.WORD	0			
6405	072164	000400			.WORD	256.			
6406	072166	000000		T38EAI:	.WORD	0			
6407	072170	000000		T38DSW:	.WORD	0			

```

:MESSAGE BUFFER ADDRESS
:NO HIGH ORDER ADDRESS BITS
:NO KT11 STUFF EITHER
:GO PRINT MESSAGE BUFFER CONTENTS
:GO BACK TO MENU

:REALLY RETURN TO THE SUPERVISOR
:LEAVE TEST

:LOCAL TEXT MESSAGES FOR TEST
:-
:LOCAL STORAGE FOR THIS TEST
:-
:LOCAL STORAGE FOR THIS TEST
:-

:WORD SET IF SUPERVISOR TTI INTER OFF
:SAVE TTI VECTOR
:SAVE TTI PRIORITY

:DELAY COUNTER FOR TEST

:COMMAND PACKET FOR TEST
:WRITE SUBSYSTEM MEM. CMD, ACK,CVC=1
:ADDRESS OF CHARACTERISTICS BLOCK

:STARTING VALUE OF BLOCK SIZE
:CHARACTERISTICS DATA BLOCK
:BSEL0 BYTE
:BSEL1 BYTE
:BSEL1 WORD
:DATA
:MESSAGE BUFFER
:END OF BUFFER ADDRESS

:COMMAND PACKET FOR TEST
:WRITE CHARA. MEM. CMND., ACK,CVC=1
:ADDRESS OF SELECT DATA BLOCK

:STARTING VALUE OF BLOCK SIZE

:SELECT DATA BLOCK
:ADDRESS OF MESSAGE BUFFER

:LENGTH OF MESSAGE BUFFER
:EAI BIT WORD
:DRIVE SELECT WORD ETC
    
```

```

6409          072200
6411 072200  140212      T38PK3:  .=<. +10>8177770
6412 072202  000000      .WORD   140212      :MESSAGE BUFFER RELEASE COMMAND
6413 072204  000000      .WORD   0           :NOT USED
6414 072206  000000      .WORD   0           :NOT USED
6415 072210  000000      .WORD   0           :NOT USED
6416
6417          :WRITE TAPE PACKET
6418          :
6420          072220
6422 072220  140005      T38PK4:  .=<. +10>8177770
6423 072222  000000      T38WR:   .WORD   140005      :WRITE, ACK, CVC=1 COMMAND
6424 072224  000000      .WORD   0           :ADDRESS OF WRITE BUFFER
6425 072226  000400      T38SIZ:  .WORD   0           :MORE ADDRESS OF WRITE BUFFER
6426          .WORD   256.      :SIZE OF RECORD
6427
6428
6429
6430          :+
6431          :LOCAL TEXT MESSAGES FOR TEST
6432          :-
6433
6434
6435
6436
6437
6438 072230    123    164    141    T38NE:  .ASCIZ  'Stand-alone Manual Intervention Not Executed'
6439 072305    045    116    045    T38MS3: .ASCIZ  '%N% Type <RETURN> To Stop Servo Exerciser, Return To Menu'
6440 072400    124    123    123    T38WRT: .ASCIZ  'TSSR Not Correct After WRITE, With WRITE PROTECT On'
6441 072464    127    122    111    T38WRL: .ASCIZ  'WRITE LOCKED Bit Not Set In XST0'
6442 072525    127    122    111    T38WLE: .ASCIZ  'WRITE LOCK ERROR Bit Not Set In XST0'
6443 072572    127    122    111    T38NBA: .ASCIZ  'WRITE SUBSYSTEM MEMORY Command Not Accepted'
6444 072646    103    157    156    T38SSR: .ASCIZ  'Contents of TSSR Incorrect After WRITE SUBSYSTEM MEMORY'
6445 072736    045    116    045    T38INT: .ASCIZ  '%N% Interrupt Received'
6446 072766    045    116    045    T38ONL: .ASCIZ  '%N% Drive Is Now ON-LINE'
6447 073022    045    116    045    T38OFL: .ASCIZ  '%N% Drive Is Now OFF-LINE'
6448 073056    103    157    156    T38SST: .ASCIZ  'Contents Of TSSR Incorrect After MESSAGE BUFFER RELEASE'
6449 073146    045    116    045    T38MS1: .ASCIZ  '%N%Toggle ON-LINE Switch to Generate ATTENTION Interrupts'
6450 073241    045    116    045    T38MS2: .ASCIZ  '%N%Type RETURN To Return To Menu%N'
6451 073305    111    163    040    T38MSG: .ASCIZ  'Is Write-Protected Tape Mounted'
6452 073345    115    141    156    T38ID:  .ASCIZ  'Manual Intervention'
6453          .EVEN
6454 073372    073416  073470  073516  MIMENU: .WORD   1$,2$,3$,4$,5$,6$
6455 073406    073665  073730  073773  .WORD   8$,9$,10$,0
6456
6457 073416    012    123    105    1$:   .ASCIZ  '<12>'SELECT OPERATION FROM FOLLOWING OPTIONS:'
6458 073470    012    011    060    2$:   .ASCIZ  '<12>' 0 Display This Menu'
6459 073516    011    061    011    3$:   .ASCIZ  ' 1 Turn On All M7196 LED's'
6460 073550    011    062    011    4$:   .ASCIZ  ' 2 Turn Off All M7196 LED's'
6461 073603    011    063    011    5$:   .ASCIZ  ' 3 Offline/Online Attention'
6462 073637    011    064    011    6$:   .ASCIZ  ' 4 Write Protect Test'
6463 073665    011    065    011    8$:   .ASCIZ  ' 5 Print Extended Transport Status'
6464 073730    011    066    011    9$:   .ASCIZ  ' 6 Return to Diagnostic Supervisor'
6465 073773    000
6466          .EVEN
6467
6468          :+
    
```

```

6469          ;LOCAL STORAGE FOR THIS TEST
6470          ;--
6471
6472 073774   000000   T38DAT: .WORD   0           ;LOGICAL RESPONSE TO QUESTION
6473 073776
6474 073776
6475 074002   012701   071450   SAVREG           ;SAVE THE REGISTERS
6476 074006   012721   140206   MOV    #T38PACKET,R1 ;START OF THE PACKET
6477 074012   012721   071460   MOV    #140206,(R1)+ ;WRITE SUBSYSTEM MEM. WITH ACK,CVC=1
6478 074016   005021                MOV    #T38TAD,(R1)+ ;ADDRESS OF DATA BLOCK
6479 074020   012721   000006   CLR    (R1)+       ;EXTENDED ADDRESS
6480 074024   005021                MOV    #6,(R1)+     ;SIZE OF DATA BLOCK IN BYTES
6481 074026   005021                CLR    (R1)+       ;CLEAR BSELO AND BSEL1
6482 074030   005011                CLR    (R1)+       ;CLEAR SEL2
6483 074032   000207                CLR    (R1)        ;CLEAR DATA AREA
6484
6485
6486          ;+
6487          ;THIS ROUTINE PRINTS THE CONTENTS OF
6488          ;THE 256 BYTE MESSAGE BUFFER RETURNED BY THE
6489          ;TSV-05.
6490
6491          ;INPUT:
6492
6493          ;
6494          ;      RC      LOW ORDER ADDRESS OF MESSAGE BUFFER
6495          ;      R1      HIGH ORDER ADDRESS OF MESSAGE BUFFER
6496          ;NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
6497
6498          ;-
6499
6500
6501 074034
6502 074034
6503 074040   010005   003134   T38MBP: SAVREG           ;SAVE THE REGISTERS
6504 074042   005737   910$      MOV    R0,R5        ;SAVE LOW ORDER ADDRESS
6505 074046   001001                TST    KTENABLE     ;ADDRESS ABOVE 28K?
6506 074050   005001                BNE    910$         ;BR IF YES
6507 074052   010103                CLR    R1          ;SET HIGH ORDER ADDRESS TO 0
6508 074054   006100                MOV    R1,R3       ;SAVE HIGH ORDER ADDRESS
6509 074056   006101                ROL    R0          ;SHIFT BIT15 TO C BIT
6510 074060                ROL    R1          ;SHIFT TO HIGH ORDER FOR PRINTOUT
6511          PRINTX  #T38AS0,R1,R5 ;PRINT MESSAGE BUFFER ADDRESS
6512          074060   010546                MOV    R5,-(SP)
6513          074062   010146                MOV    R1,-(SP)
6514          074064   012746   074336                MOV    #T38AS0,-(SP)
6515          074070   012746   000003                MOV    #3,-(SP)
6516          074074   010600                MOV    SP,R0
6517          074076   104415                TRAP  C$PNTX
6518          074100   062706   000010                ADD    #10,SP
6519
6520          074104                PRINTX  #T38AS1           ;PRINT HEADER FOR CONTENTS
6521          074104   012746   074403                MOV    #T38AS1,-(SP)
6522          074110   012746   000001                MOV    #1,-(SP)
6523          074114   010600                MOV    SP,R0
6524          074116   104415                TRAP  C$PNTX
6525          074120   062706   000004                ADD    #4,SP
6526
6527          074124   010501                MOV    R5,R1       ;COPY LOW ORDER ADDRESS
6528          074126   010300                MOV    R3,R0       ;COPY HIGH ORDER ADDRESS
    
```

6514	074130	001403			BEQ	913\$		:BR IF NOT ABOVE 28K
6515	074132	004737	017316		JSR	PC,SETMAP		:SETUP PAR ADDRESS IN R0
6516	074136	010005			MOV	R0,R5		:GET PAR FORMAT ADDRESS ABOVE 28K
6517	074140	010537	074504	913\$:	MOV	R5,T38CNT		:HOLD ADDRESS
6518	074144	011504		911\$:	MOV	(R5),R4		:GET BUFFER ENTRY
6519	074146	022704	125252		CMP	#125252,R4		:CHECK FOR NO LOAD CONDITION
6520	074152	001417			BEQ	912\$		:BR, IF BUFFER WASN'T LOADED
6521	074154	010403			MOV	R4,R3		:MAKE COPY
6522	074156	042704	170377		BIC	#170377,R4		:ONLY BITS 11,10,9 AND 8 ARE SAVED
6523	074162	000241			CLC			:CLEAR CARRY
6524	074164	006004			ROR	R4		:11 TO 10 BIT POSITION
6525	074166	006004			ROR	R4		:10 TO 9 BIT POSITION
6526	074170	006004			ROR	R4		:9 TO 8 BIT POSITION
6527	074172	006004			ROR	R4		:8 TO 7 BIT POSITION
6528	074174	042703	177760		BIC	#177760,R3		:ONLY BITS 3,2,1 AND 0 ARE SAVED
6529	074200	060403			ADD	R4,R3		:OR'EM TOGETHER
6530	074202	010325			MOV	R3,(R5)+		:PUT BACK IN BUFFER
6531	074204	020527	072142		CMP	R5,#T38EB		:END OF BUFFER YET
6532	074210	001355			BNE	911\$		:BR, IF NOT AT END YET
6533	074212	013705	074504	912\$:	MOV	T38CNT,R5		:PUT ADDRESS BACK
6534	074216	012704	000001		MOV	#1,R4		:START BYTE NUMBER AT ONE
6535	074222			915\$:	PRINTX	#T38ASN,R4,(R5)+		:PRT MEM BUFFER W/NEWLINE
	074222	012546					MOV	(R5)+,-(SP)
	074224	010446					MOV	R4,-(SP)
	074226	012746	074460				MOV	#T38ASN,-(SP)
	074232	012746	000003				MOV	#3,-(SP)
	074236	010600					MOV	SP,R0
	074240	104415					TRAP	C\$PNTX
	074242	062706	000010				ADD	#10,SP
6536	074246	005037	074504		CLR	T38CNT		:CLEAR COUNTER
6537	074252	000412			BR	921\$		:SKIP OTHER PRINT
6538	074254			920\$:	PRINTX	#T38ASC,R4,(R5)+		:PRINT THE CONTENTS OF MEMOR: 3UFFER
	074254	012546					MOV	(R5)+,-(SP)
	074256	010446					MOV	R4,-(SP)
	074260	012746	074441				MOV	#T38ASC,-(SP)
	074264	012746	000003				MOV	#3,-(SP)
	074270	010600					MOV	SP,R0
	074272	104415					TRAP	C\$PNTX
	074274	062706	000010				ADD	#10,SP
6539	074300	005237	074504	921\$:	INC	T38CNT		:BUMP COUNTER
6540	074304	005204			INC	R4		:NUMBER OF THE NEXT
6541	074306	020427	000200		CMP	R4,#128.		:DONE ALL YET ?
6542	074312	003010			BGT	50\$		:BRANCH IF ALL DONE
6543	074314	023727	074504 000004		CMP	T38CNT,#4		:DONE FOUR YET
6544	074322	001401			BEQ	925\$		:BR, IF THREE DONE
6545	074324	000753			BR	920\$		:KEEP GOING
6546	074326	005037	074504	925\$:	CLR	T38CNT		:CLEAR COUNTER
6547	074332	000733			BR	915\$		:PRINT WITH NEW LINE
6548	074334	000207		50\$:	RTS	PC		:RETURN
6549								
6550	074336	045	116	045	T38AS0:	.ASCIZ	'%NZA Message Buffer Address = %01X05'	
6551	074403	045	116	045	T38AS1:	.ASCIZ	'%NZA Message Buffer Contents:'	
6552	074441	045	101	040	T38ASC:	.ASCIZ	'%A %D4%A: %03'	
6553	074460	045	116	045	T38ASN:	.ASCIZ	'%NZA Byte%D4%A: %03'	
6554						.EVEN		
6555	074504	000000			T38CNT:	.WORD		:COUNTER FOR PRINT
6556	074506				ENDTST			

074506  
074506 104401

SEQ 0279

L10075: TRAP C\$ETST



6558  
6559  
6560  
6561  
6562  
6563  
6564  
6565  
6566  
6567  
6568  
6569  
6570  
6571  
6572  
6573  
6574  
6575  
6576  
6577  
6578  
6579  
6580  
6581  
6582  
6583  
6584  
6585  
6586  
6587  
6588  
6589  
6590  
6591  
6592  
6593  
6594  
6595  
6596  
6601  
6602  
6603  
6604  
6605  
6606  
6607  
6608  
6609  
6610  
6611  
6612  
6616  
6617  
074510  
074510  
074510  
074510  
074512  
074514  
074520  
074522  
074526  
074532  
074536  
074540  
074544  
074544  
074550  
074552  
074554  
074554  
074556

074510  
074510  
104421  
001403  
012700 076503  
000402  
012700 077632  
004737 016510  
004737 020500  
103402  
000137 075720  
074544  
004737 015774  
103405  
010001  
104455  
002115

```
.SBTTL TEST 11: CONFIGURATION TYPEOUT

:THIS IS A STANDALONE ROUTINE THAT PRINTS OUT ON THE CONSOLE TERMINAL
:THE CONFIGURATION OF THE M7196 MODULE AND TSV05 SUBSYSTEM. SPECIFICALLY,
:THE FOLLOWING INFORMATION IS PRESENTED:

:
:1.0 STATE OF THE EXTENDED FEATURES SWITCH ON THE M7196: ON (EXTENDED
:   FEATURES ENABLED) OR OFF (EXTENDED FEATURES DISABLED),
:
:2.0 STATE OF THE BUFFERING ENABLE SWITCH ON THE M7196: ON
:   (BUFFERING ENABLED) OR OFF (BUFFERING DISABLED),
:
:3.0 MICROCODE REVISION LEVEL OF THE M7196,
:
:4.0 NUMBER OF TAPE TRANSPORTS CONNECTED TO THE CONTROLLER,
:
:5.0 UNIT SELECT CODE AND STATE (ONLINE/OFFLINE, WRITE ENABLED/PROTECTED)
:   OF EACH CONNECTED TRANSPORT. IN ADDITION, THE PROGRAM WILL INDICATE,
:   FOR EACH ON-LINE TRANSPORT, WHETHER OR NOT IT IS EQUIPPED WITH THE
:   EXTENDED TAPE STATUS READOUT FEATURE.
:
:THE OPERATOR IS EXPECTED TO READ THE PRINTOUT AND VERIFY THAT IT MATCHES
:THE ACTUAL CONFIGURATION AT HAND. IF, FOR EXAMPLE, THE PROGRAM INDICATES
:THAT IT "SEES" TWO TRANSPORTS CONNECTED WHEN IN FACT ONLY ONE IS PRESENT,
:THE OPERATOR MUST INTERPRET THIS AS AN ERROR AND ATTEMPT TO FIND THE
:CAUSE (BAD CABLE, FAULTY UNIT-SELECT DECODING IN THE TRANSPORT, ETC.).
:[SINCE THE CONTROLLER CAN ONLY ACCESS UNIT 0 IF IT IS IN "STANDARD"
:MODE, THE PROGRAM WILL FORCE THE MODULE INTO EXTENDED MODE VIA THE
:WRITE SUBSYSTEM MEMORY COMMAND IN ORDER TO SCAN FOR CONNECTED TRANSPORTS.]
:
:THIS ROUTINE, WHEN ITS ACTIONS ARE COMPLETED, WILL EXIT BACK TO THE
:DIAGNOSTIC SUPERVISOR SO THAT IF ADDITIONAL UNITS (CONTROLLERS) ARE
:SELECTED (E.G., FROM THE INITIAL STARTUP DIALOG), THE ROUTINE WILL BE
:REENTERED SO THAT THEIR CONFIGURATIONS CAN BE PRINTED.
```

```

BGNTST
RFLAGS R0
:GET OPERATOR FLAGS T11::
TRAP CSRFLA
:BR, IF OK TO RUN
:"TEST NOT EXECUTED"
:JUMP OUT OF TEST IF NOT
:TEST ID MESSAGE
:DO THE COMMON SETUP
:IS MANUAL INTERVENTION ALLOWED?
:BR, IF MANUAL INTERVENTION ALLOWED
:JUMP TO OUT IF NOT

10$: MOV #T39NE,R0
11$: JSR PC,TSTSETUP
:JSR PC,CHKMAN
BCS 20$
JMP 64$

20$: JSR PC,SOFINIT
BCS 25$
MOV R0,R1
ERRDF ERRNO,SFIERR,SFIMSG
:DO SOFT INIT OF CONTROLLER
:BR IF SOFT INIT = OK
:SAVE CONTENTS OF TSSR
:DEVICE FATAL ERROR DURING INIT
TRAP CSERDF
.WORD 1101
```



```

075000 104415
075002 062706 000004
6642 075006 042701 177700
6643 075012 010137 077570
6644 075016
075016 013746 077570
075022 012746 077513
075026 012746 000002
075032 010600
075034 104415
075036 062706 000006
6645 075042 004737 015774
6646 075046 103405
6650 075050 010001
6651 075052
075052 104455
075054 002117
075056 003652
075060 012034
6652 075062
075062 104406
6653 075064 013737 002202 076450
6654 075072 012704 076430
6655 075076 004737 010662
6656 075102 103405
6660 075104 010001
6661 075106
075106 104456
075110 002120
075112 005056
075114 012034
6662 075116
075116 104406
6663 075120 005737 002226
6664 075124 001036
6665 075126 112737 000200 075741
6666 075134 112737 000010 075740
6667 075142 012704 075730
6668 075146 010465 000000
6669 075152 004737 016336
6670 075156 103405
6671 075160 010001
6675 075162
075162 104456
075164 002121
075166 077205
075170 012046
6676 075172
075172 104406
6677 075174 012704 076430
6678
6679
6680
6681
6682
6683
6684 075200 004737 010662

130$: BIC #177700,R1 ;ONLY LEAVE MICROCODE REV LEVEL
MOV R1,T39RL ;LOAD UP REV LEVEL
PRINTX #T39MCL,T39RL ;'MICROCODE REVISION LEVEL =000XXX'
TRAP C$PNTX
ADD #4,SP
MOV T39RL,-(SP)
MOV #T39MCL,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #6,SP
JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
BCS 140$ ;BR IF SOFT INIT = OK
MOV R0,R1 ;SAVE CONTENTS OF TSSR
ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
TRAP C$ERDF
.WORD 1103
.WORD SFIERR
.WORD SFIMSG
140$: CKLOOP ;LOOP IF SELECTED
TRAP C$CLP1
MOV UNITN,T39DSW ;SET UNIT NUMBER
MOV #T39PK2,R4 ;SUBROUTINE NEEDS PACKET ADDRESS
JSR PC,WRTCHR ;ISSUE WRITE CHARACTERISTICS
BCS 150$ ;BR, IF COMMAND ISSUED OK
MOV R0,R1 ;SAVE CONTENTS OF TSSR
ERRHRD ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTIC FAILED
TRAP C$ERHRD
.WORD 1104
.WORD WRTMSG
.WORD SFIMSG
150$: CKLOOP ;LOOP IF SELECTED
TRAP C$CLP1
TST EXTFEA ;CHECK FOR EXTENDED FEATURES SW SWITCH
BNE 174$ ;BR IF SWITCH IS ON
MOVB #200,T39BS1 ;WRITE MISCELLANEOUS CONT/READ STATUS
MOVB #10,T39BS0 ;FUNCTION SELECTION BIT (TURN ON EXTFEA HW SWITCH)
MOV #T39PACKET,R4 ;WRITE SUBSYS MEM PACKET
MOV R4,TSDB(R5) ;ISSUE COMMAND
JSR PC,CHKTSSR ;WAIT FOR SSR
BCS 160$ ;BR, IF NO ERROR
MOV R0,R1 ;ERROR, SAVE TSSR
ERRHRD ERRNO,T39NBA,PKTSSR ;TSSR NOT CORRECT AFTER WRT. MISCELLANEOUS
TRAP C$ERHRD
.WORD 1105
.WORD T39NBA
.WORD PKTSSR
160$: CKLOOP ;LOOP IF SELECTED
TRAP C$CLP1
MOV #T39PK2,R4 ;SUBROUTINE NEEDS PACKET ADDRESS
*****
;WRITE CHARACTERISTICS COMMAND (CALL TO WRTCHR)
*****
JSR PC,WRTCHR ;ISSUE WRITE CHARACTERISTICS
    
```

6685	075204	103405				BCS	170\$			:BR, IF COMMAND ISSUED OK		
6689	075206	010001				MOV	R0,R1			:SAVE CONTENTS OF TSSR		
6690	075210					ERRHRD	ERRNO,WRTMSG,SFIMSG			:WRITE CHARACTERISTIC FAILED		
	075210	104456								TRAP	C\$ERHRD	
	075212	002122								.WORD	1106	
	075214	005056								.WORD	WRTMSG	
	075216	012034								.WORD	SFIMSG	
6691	075220				170\$:	CKLOOP				:SCOPE LOOP		
	075220	104406								TRAP	C\$CLP1	
6692	075222	005037	002202		174\$:	CLR	UNITN			:SET TO DRIVE 0		
6693	075226	013737	002202	076450	175\$:	MOV	UNITN,T39DSW			:SET UNIT NUMBER		
6694	075234	012704	076430			MOV	#T39PK2,R4			:SUBROUTINE NEEDS PACKET ADDRESS		
6695	075240	004737	010662			JSR	PC,WRTCHR			:ISSUE WRITE CHARACTERISTICS		
6696	075244	103405				BCS	180\$			:BR, IF COMMAND ISSUED OK		
6700	075246	010001				MOV	R0,R1			:SAVE CONTENTS OF TSSR		
6701	075250					ERRHRD	ERRNO,WRTMSG,SFIMSG			:WRITE CHARACTERISTIC FAILED		
	075250	104456								TRAP	C\$ERHRD	
	075252	002123								.WORD	1107	
	075254	005056								.WORD	WRTMSG	
	075256	012034								.WORD	SFIMSG	
6702	075260				180\$:	CKLOOP				:LOOP IF SELECTED		
	075260	104406								TRAP	C\$CLP1	
6703												
6704	075262	016501	000002		190\$:	MOV	TSSR(R5),R1			:GET TSSR STATUS		
6705	075266	032701	000100			BIT	#OFL,R1			:CHECK FOR OFF-LINE		
6706	075272	001414				BEQ	200\$			:BR, IF DRIVE IS ON-LINE		
6707	075274					PRINTX	#T39OF2,UNITN			: 'DRIVE NUMBER XX IS OFF-LINE'		
	075274	013746	002202							MOV	UNITN,-(SP)	
	075300	012746	076744							MOV	#T39OF2,-(SP)	
	075304	012746	000002							MOV	#2,-(SP)	
	075310	010600								MOV	SP,R0	
	075312	104415								TRAP	C\$PNTX	
	075314	062706	000006							ADD	#6,SP	
6708	075320	000137	075654			JMP	250\$			:DO NOT TRY TO GET ANYMORE INFO.		
6709	075324				200\$:	PRINTX	#T39ON2,UNITN			: 'DRIVE NUMBER XX IS ON-LINE'		
	075324	013746	002202							MOV	UNITN,-(SP)	
	075330	012746	077010							MOV	#T39ON2,-(SP)	
	075334	012746	000002							MOV	#2,-(SP)	
	075340	010600								MOV	SP,R0	
	075342	104415								TRAP	C\$PNTX	
	075344	062706	000006							ADD	#6,SP	
6710	075350	013701	075754			MOV	T39BFR+6,R1			:READ EXTENDED STATUS (XST0)		
6711	075354	032701	000004			BIT	#BIT2,R1			:IS DRIVE WRITE PROTECTED		
6712	075360	001013				BNE	210\$			:BR, IF WRITE PROTECTED		
6713	075362					PRINTX	#T39WPN,UNITN			: 'DRIVE NUMBER IS NOT WRT PRO'		
	075362	013746	002202							MOV	UNITN,-(SP)	
	075366	012746	077126							MOV	#T39WPN,-(SP)	
	075372	012746	000002							MOV	#2,-(SP)	
	075376	010600								MOV	SP,R0	
	075400	104415								TRAP	C\$PNTX	
	075402	062706	000006							ADD	#6,SP	
6714	075406	000412				BR	220\$			:SKIP OVER		
6715	075410				210\$:	PRINTX	#T39WRT,UNITN			: 'DRIVE NUMBER XX IS WRT PRO'		
	075410	013746	002202							MOV	UNITN,-(SP)	
	075414	012746	077053							MOV	#T39WRT,-(SP)	
	075420	012746	000002							MOV	#2,-(SP)	
	075424	010600								MOV	SP,R0	



TEST 11: CONFIGURATION TYPEOUT									
	075704	010600							
	075706	104415							MOV
	075710	062706	000004						TRAP
6747	075714	000137	000200						ADD
6748	075720			64\$:	JMP	200			
	075720	104432			EXIT	TST			
	075722	001736							
6749									
6750									
6751									
6752									
6753									
6754									
6755									
6756	075724	000000		T39DLY:	.WORD	0			
6758		075730							
6760	075730								
6761	075730	140006		T39PACKET:					
6762	075732	075740			.WORD	140006			
6763	075734	000000			.WORD	T39TAD			
6764	075736	000012			.WORD	0			
6765	075740				.WORD	10.			
6766	075740	000		T39TAD:					
6767	075741	000		T39BS0:	.BYTE	0			
6768	075742	000000		T39BS1:	.BYTE	0			
6769	075744	000000		T39BS2:	.WORD	0			
6770	075746				.WORD	0			
6771				T39BFR:	.BLKW	150.			
6772									
6774		076430							
6776	076430			T39PK2:					
6777	076430	140004			.WORD	140004			
6778	076432	076440			.WORD	T39DTA			
6779	076434	000000			.WORD	0			
6780	076436	000012			.WORD	10.			
6781									
6782									
6783	076440			T39DTA:					
6784	076440	075746			.WORD	T39BFR			
6785	076442	000000			.WORD	0			
6786	076444	000400			.WORD	256.			
6787	076446	000000		T39EAI:	.WORD	0			
6788	076450	000000		T39DSW:	.WORD	0			
6790		076460							
6792	076460	140012		T39PK3:	.WORD	140012			
6793	076462	000000			.WORD	0			
6794									
6795									
6796									
6798		076470							
6800	076470	140005		T39PK4:	.WORD	140005			
6801	076472	000000		T39WR:	.WORD	0			
6802	076474	000000			.WORD	0			
6803	076476	000400		T39SIZ:	.WORD	256.			
6804									
6805									
6806									

```

;RETURN TO SUPERVISOR
;EXIT THIS SECTION

TRAP      CSEXIT
.WORD     L10076-.

;DELAY COUNTER FOR TEST

;COMMAND PACKET FOR TEST
;WRITE SUBSYSTEM MEM. CMD, ACK,CVC=1
;ADDRESS OF CHARACTERISTICS BLOCK

;STARTING VALUE OF BLOCK SIZE
;CHARACTERISTICS DATA BLOCK
;BSEL0 BYTE
;BSEL1 BYTE
;BSEL1 WORD
;DATA
;MESSAGE BUFFER

;COMMAND PACKET FOR TEST
;WRITE CHARA. MEM. CMND., ACK,CVC=1
;ADDRESS OF SELECT DATA BLOCK

;STARTING VALUE OF BLOCK SIZE

;SELECT DATA BLOCK
;ADDRESS OF MESSAGE BUFFER

;LENGTH OF MESSAGE BUFFER
;EAI BIT WORD
;DRIVE SELECT WORD ETC

;MESSAGE BUFFER RELEASE COMMAND
;NOT USED

;WRITE, ACK, CVC=1 COMMAND
;ADDRESS OF WRITE BUFFER
;MORE ADDRESS OF WRITE BUFFER
;SIZE OF RECORD
    
```

```

6807
6808
6809          ;+
6810          ;LOCAL TEXT MESSAGES FOR TEST
6811          ;-
6812
6813
6814
6815 076500      045      116      000  T39NFL: .ASCIZ  '%N'
6816 076503      123      164      141  T39NE:  .ASCIZ  'Stand-alone Configuration Typeout Not Executed'
6817 076562      045      116      045  T39ETS: .ASCIZ  '%N% Extended Tape Status Available, Drive Number %D2'
6818 076651      045      116      045  T39ETN: .ASCIZ  '%N% Extended Tape Status NOT Available, Drive Number %D2'
6819 076744      045      116      045  T39OF2: .ASCIZ  '%N% Drive Number %D2% Is Off-Line'
6820 077010      045      116      045  T39ON2: .ASCIZ  '%N% Drive Number %D2% Is On-Line'
6821 077053      045      116      045  T39WRT: .ASCIZ  '%N% Drive Number %D2% Is Write Protected'
6822 077126      045      116      045  T39WPN: .ASCIZ  '%N% Drive Number %D2% Is NOT Write Protected'
6823 077205      127      122      111  T39NBA: .ASCIZ  'WRITE SUBSYSTEM MEMORY Command Not Accepted'
6824 077261      103      157      156  T39SSR: .ASCIZ  'Contents of TSSR Incorrect After WRITE SUBSYSTEM MEMORY'
6825
6826 077351      045      116      045  T39SFS: .ASCIZ  '%N% State Of Extended Features Switch  ='
6827 077423      045      116      045  T39SBS: .ASCIZ  '%N% State Of Buffering Switch          ='
6828 077475      045      101      040  T39OFF: .ASCIZ  '%A OFF'
6829 077504      045      101      040  T39ON:  .ASCIZ  '%A ON '
6830 077513      045      116      045  T39MCL: .ASCIZ  '%N% M7196 Microcode Revision Level    =%02'
6831
6832 077570      000000          T39RL: .WORD    0
6833
6834
6835
6836
6837          ;+
6838          ;LOCAL STORAGE FOR THIS TEST
6839          ;-
6840 077572      000000          T39DAT: .WORD    0          ;LOGICAL RESPONSE TO QUESTION
6841 077574
6842 077574          SAVREG
6843 077600      012701      075730      MOV      #T39PACKET,R1          ;SAVE THE REGISTERS
6844 077604      012721      140006      MOV      #140006,(R1)+          ;START OF THE PACKET
6845 077610      012721      075740      MOV      #T39TAD,(R1)+          ;WRITE SUBSYSTEM MEM. WITH ACK,CVC=1
6846 077614      005021          CLR      (R1)+                  ;ADDRESS OF DATA BLOCK
6847 077616      012721      000006      MOV      #6,(R1)+              ;EXTENDED ADDRESS
6848 077622      005021          CLR      (R1)+                  ;SIZE OF DATA BLOCK IN BYTES
6849 077624      005021          CLR      (R1)+                  ;CLEAR BSEL0 AND BSEL1
6850 077626      005011          CLR      (R1)+                  ;CLEAR SEL2
6851 077630      000207          CLR      (R1)                   ;CLEAR DATA AREA
6852
6853
6854          ;+
6855          ;LOCAL TEXT MESSAGES FOR TEST
6856          ;-
6857 077632      103      157      156  TST39ID: .ASCIZ  'Configuration Typeout'
6858
6859 077660          .EVEN
6860          077660          ENDTST
6860          077660      104401          L10076: TRAP    C$ETST

```





```

077732 003652
077734 012034
6922 077736 012700 100414 5$: MOV #SCMENU,R0 ;MENU OF SCOPE LOOP SELECTIONS
6923 077742 012701 000010 MOV #8.,R1 ;MAXIMUM ALLOWED SELECTION
6924 077746 004737 020256 JSR PC,GETSEL ;GO GET THE OPERATORS SELECTION
6925 077752 005700 TST R0 ;WAS ZERO SPECIFIED?
6926 077754 001760 BEQ 2$ ;REPEAT MENU IF YES.
6927 077756 020027 000007 CMP RO,#7 ;EXTENDED TSSR?
6928 077762 001015 BNE 3$ ;BRANCH IF NOT
6929 077764 005737 002226 TST EXTFEA ;CHECK FOR EXTENDED FEATURES SET
6930 077770 001012 BNE 3$ ;BR, IF IT IS ON
6931 077772 PRINTF #EXFMSG ;WARN OPERATOR EXTENDED FEATURES CLEAR
077772 012746 101177 MOV #EXFMSG,-(SP)
077776 012746 000001 MOV #1,-(SP)
100002 010600 MOV SP,R0
100004 104417 TRAP C$PNTF
100006 062706 000004 ADD #4,SP
6932 100012 000137 077716 JMP 2$ ;GO BACK TO BASIC MENU
6933 100016 010004 3$: MOV RO,R4 ;SAVE THE MENU SELECTION
6934 100020 SETPRI #PRI07 ;RAISE THE PRIORITY
100020 012700 000340 MOV #PRI07,R0
100024 104441 TRAP C$SPRI
6935 100026 005037 100406 CLR TTION ;ASSUME INTERRUPTS ARE ENABLED
6936 100032 032737 000100 177560 BIT #100,@TTICSR ;ARE TTI INTERRUPTS ON?
6937 100040 001005 BNE 4$ ;BRANCH IF YES
6938 100042 005237 100406 INC TTION ;FLAG SET IF INTERRUPTS OFF
6939 100046 052737 000100 177560 BIS #100,@TTICSR ;ENABLE INTERRUPTS
6940 100054 012701 000060 4$: MOV #TTIVEC,R1 ;START OF TTI VECTORS
6941 100060 011137 100410 MOV (R1),TVECSAV ;SAVE THE CURRENT TTI VECTOR
6942 100064 012721 100310 MOV #60$,(R1)+ ;SET NEW INTERRUPT ROUTINE
6943 100070 011137 100412 MOV (R1),TPRISAV ;SAVE THE VECTOR PRIORITY
6944 100074 012711 000340 MOV #PRI07,(R1) ;USE PRIORITY SEVEN
6945 100100 SETPRI #PRI00 ;LOWER INTERRUPT BR LEVEL
100100 012700 000000 MOV #PRI00,R0
100104 104441 TRAP C$SPRI
6946 100106 006304 ASL R4 ;CONVERT TO WORD OFFSET
6947 100110 000174 100114 JMP @6$(R4) ;JUMP TO PROPER LOOP
6948 100114 077716 6$: .WORD 2$ ;RETYPE THE MENU
6949 100116 100136 .WORD 10$ ;TSBA READ ACCESS
6950 100120 100146 .WORD 15$ ;TSSR READ ACCESS
6951 100122 100160 .WORD 20$ ;TSSR WRITE ACCESS
6952 100124 100200 .WORD 25$ ;TSDB HIGH BYTE WRITE ACCESS
6953 100126 100224 .WORD 30$ ;TSDB LOW BYTE WRITE ACCESS
6954 100130 100250 .WORD 35$ ;TSDB MAINTENANCE MODE
6955 100132 100270 .WORD 40$ ;TSDBX WRITE ACCESS
6956 100134 100402 .WORD 65$ ;LEAVE THE TEST
6957
6958
6959 100136 105065 000000 10$: CLRB TSDB(R5) ;ENTER MAINTENANCE MODE
6960 100142 011500 12$: MOV (R5),R0 ;READ TSBA REGISTER
6961 100144 000776 BR 12$ ;LOOP UNTIL HALTED
6962
6963
6964 100146 012703 000002 15$: MOV #TSSR,R3 ;ADDRESS OF TSSR REGISTER
6965 100152 060503 ADD R5,R3 ;POINT TO TSV05'S REGISTERS
6966 100154 011300 18$: MOV (R3),R0 ;READ TSSR REGISTER
6967 100156 000776 BR 18$ ;LOOP UNTIL STOPPED

```

```

6968
6969 100160 004737 020174      20$: JSR    PC,GETPAT      ;READ THE DATA PATTERN
6970 100164 010001              MOV    R0,R1           ;DATA PATTERN FOR LOOP
6971 100166 012703 000002      MOV    #TSSR,R3        ;ADDRESS OF TSSR
6972 100172 060503              ADD    R5,R3           ;POINT TO TSV05'S REGISTERS
6973 100174 010113              22$: MOV    R1,(R3)      ;WRITE DATA TO TSSR
6974 100176 000776              BR     22$            ;LOOP
6975
6976
6977 100200 105065 000000      25$: CLRB   TSDB(R5)     ;ENTER MAINTENANCE MODE
6978 100204 004737 020174      JSR    PC,GETPAT      ;READ THE DATA PATTERN
6979 100210 010001              MOV    R0,R1           ;DATA PATTERN FOR LOOP
6980 100212 012703 000001      MOV    #TSDBH,R3      ;ADDRESS OF HIGH BYTE OF TSDB
6981 100216 060503              ADD    R5,R3           ;POINT TO TSV05'S REGISTERS
6982 100220 110113              27$: MOV    R1,(R3)      ;WRITE THE DATA TO TSDB, HIGH BYTE
6983 100222 000776              BR     27$            ;LOOP UNTIL STOPPED
6984
6985
6986 100224 105065 000000      30$: CLRB   TSDB(R5)     ;ENTER MAINTENANCE MODE
6987 100230 004737 020174      JSR    PC,GETPAT      ;READ THE DATA PATTERN
6988 100234 010001              MOV    R0,R1           ;DATA PATTERN FOR LOOP
6989 100236 012703 000000      MOV    #TSDB,R3       ;ADDRESS OF TSSR
6990 100242 060503              ADD    R5,R3           ;POINT TO TSV05'S REGISTERS
6991 100244 110113              32$: MOV    R1,(R3)      ;WRITE DATA TO TSSR, LOW BYTE
6992 100246 000776              BR     32$            ;LOOP UNTIL HALTED BY OPERATOR
6993
6994 100250 004737 020174      35$: JSR    PC,GETPAT      ;READ THE DATA PATTERN
6995 100254 010001              MOV    R0,R1           ;DATA PATTERN FOR LOOP
6996 100256 012703 000000      MOV    #TSDB,R3       ;SELECT TSDB
6997 100262 060503              ADD    R5,R3           ;POINT TO TSV05'S REGISTERS
6998 100264 010113              37$: MOV    R1,(R3)      ;WRITE THE DATA PATTERN
6999
7000 100266 000776              BR     37$            ;LOOP UNTIL HALTED
7001
7002 100270 004737 020174      40$: JSR    PC,GETPAT      ;READ THE DATA PATTERN
7003 100274 010001              MOV    R0,R1           ;SAVE THE DATA PATTERN
7004 100276 012703 000003      MOV    #TSSRH,R3      ;BYTE ADDRESS OF TSSR, HIGH BYTE
7005 100302 060503              ADD    R5,R3           ;POINT TO TSV05'S REGISTERS
7006 100304 110113              42$: MOV    R1,(R3)      ;WRITE THE DATA TO REGISTER
7007 100306 000776              BR     42$            ;LOOP UNTIL HALTED
7008
7009
7010
7011      ;+
7012      ;PROCESS CONSOLE INTERRUPTS
7013      ;-
7014 100310 010046              60$: MOV    R0,-(SP)      ;SAVE WORK REGISTER
7015 100312 113700 177562      MOV    @#TTIBFR,R0    ;GET THE OPERATOR INPUT
7016 100316 042700 000200      BIC    #200,R0        ;STRIP OFF PARITY BIT
7017 100322 122700 000015      CMPB   #15,R0         ;IS IT A CARRIAGE RETURN ?
7018 100326 001021              BNE    61$            ;JUST EXIT IF NOT
7019 100330 012766 077716 000002  MOV    #2$,2(SP)      ;RETURN TO MASTER MENU
7020 100336 005066 000004      CLR    4(SP)          ;FORCE PRIORITY ZERO
7021 100342 013737 100410 000060  MOV    TVECSAV,@#TTIVEC ;RESTORE SUPERVISOR VECTOR
7022 100350 013737 100412 000062  MOV    TPRISAV,@#TTIVEC+2 ;RESTORE SUPERVISOR PRIORITY
7023 100356 005737 100406      TST    TTION          ;ARE SUPERVISOR INTERRUPTS ENABLED ?
7024 100362 001403              BEQ    61$            ;BRANCH IF YES

```

```
7025 100364 042737 000100 177560      BIC      #100,@#TTICSR      ;TURN OFF TTI INTERRUPTS
7026 100372 012600      61$:    MOV      (SP)+,R0      ;RESTORE REGISTER
7027 100374 000002      RTI                               ;RETURN FROM INTERRUPT
7028
7029 100376      64$:
7030 100376      63$:    EXIT      TST      ;EXIT THE TEST
      100376 104432
      100400 000736
7031 100402 000137 000200      65$:    JMP      200      ;RETURN TO SUPERVISOR      TRAP      C$EXIT
      .WORD      L10077-.
7032
7033      ;+
7034      ;LOCAL STORAGE FOR THIS TEST
7035      ;-
7036
7037 100406 000000      TTION:      .WORD      0      ;WORD SET IF SUPERVISOR TTI INTER OFF
7038 100410 000000      TVECSAV:    .WORD      0      ;SAVE TTI VECTOR
7039 100412 000000      TPRISAV:    .WORD      0      ;SAVE TTI PRIORITY
7040
7041
7042      ;+
7043      ;MENU FOR OPERATOR INPUT FOR SCOPE LOOPS
7044      ;-
7045
7046
7047 100414 100446 100521 100547      SCMENU:    .EVEN
7048 100430 100720 100756 101024      .WORD      1$,2$,3$,4$,5$,6$
      .WORD      7$,8$,9$,10$,11$,12$,0
7049
7050
7051 100446      012      123      105      1$:    .ASCIZ    <12>'SELECT SCOPE LOOP FROM FOLLOWING OPTIONS:'
7052 100521      012      011      060      2$:    .ASCIZ    <12>'      0      Display This Menu'
7053 100547      011      061      011      3$:    .ASCIZ    '      1      TSBA Read Access'
7054 100573      011      062      011      4$:    .ASCIZ    '      2      TSSR Read Access'
7055 100617      011      063      011      5$:    .ASCIZ    '      3      Initialize (TSSR Write Access,'
7056 100661      011      064      011      6$:    .ASCIZ    '      4      TSDB High Byte Write Access'
7057 100720      011      065      011      7$:    .ASCIZ    '      5      TSDB Low Byte Write Access'
7058 100756      011      066      011      8$:    .ASCIZ    '      6      TSDB Maintenance Mode Write Access'
7059 101024      011      067      011      9$:    .ASCIZ    '      7      TSDBX (TSSR High Byte) Write Access'
7060 101073      011      070      011     10$:    .ASCIZ    '      8      Return to Diagnostic Supervisor'
7061 101136      000      11$:    .ASCIZ    ''
7062 101137      124      171      160     12$:    .ASCIZ    'Type RETURN To Stop Scope Loops'
7063 101177      045      116      045     EXFMSG: .ASCIZ    '%NZA *** Extended Features Switch Not On *** '
7064 101255      123      164      141     T4ONE:  .ASCIZ    'Stand-alone Scope Loops Not Executed'
7065 101322      123      143      157     TST40ID: .ASCIZ    'Scope Loops'
7066
7067 101336      .EVEN
      101336      .ENDTST
      101336      104401
7068 101340      ENDMOD
      .WORD      L10077:      TRAP      C$ETST
```

```

1          .TITLE  TSV6 - PARAMETER CODING
7
12
18
19 101340  BGNMOD  TSV6
    101340  TSV6::
20
21
22          .SBTTL  HARDWARE PARAMETER CODING SECTION
23
24          :++
25          : THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
26          : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
27          : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
28          : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
29          : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
30          : WITH THE OPERATOR.
31          :--
32 101340  BGNHRD
    101340  .WORD  L10100-L$HARD/2
    101342  L$HARD::
33
34 101342  GPRMA  HPM1,0,0,160010,177776,YES      ;GET TSBA/TSDB REGISTER ADDRESS.
    101342  .WORD  T$CODE
    101344  .WORD  HPM1
    101346  .WORD  T$LLOLIM
    101350  .WORD  T$HILIM
35 101352  GPRMA  HPM2,2,0,0,776,YES              ;GET VECTOR ADDRESS.
    101352  .WORD  T$CODE
    101354  .WORD  HPM2
    101356  .WORD  T$LLOLIM
    101360  .WORD  T$HILIM
36          ;GPRMD  HPM3,4,0,340,0,7,YES          ;GET INTERRUPT PRIORITY.
37 101362  ENDHRD
    .EVEN
    101362  L10100:
38 101362  104    105    126  HPM1:  .ASCIZ  'DEVICE ADDRESS (TSBA/TSDB) '
39 101416  111    116    124  HPM2:  .ASCIZ  'INTERRUPT VECTOR '
40 101442  111    116    124  HPM3:  .ASCIZ  'INTERRUPT PRIORITY '
41          .EVEN
42

```

```

44                                     .SBTTL SOFTWARE PARAMETER CODING SECTION
45
46                                     :++
47                                     : THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
48                                     : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
49                                     : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
50                                     : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
51                                     : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
52                                     : WITH THE OPERATOR.
53                                     :--
54 101472                                BGNSFT
55 101472 000003                          .WORD L10101-L$SOFT/2
56 101474                                L$SOFT::
57                                     : GPRML SPM1,0,-1,YES ; GET TRANSPORT TEST FLAG.
58 101474 001130                          : GPRML SPM4,2,-1,YES ; GET ITERATION CONTROL.
59 101476 101532                          .WORD T$CODE
60 101500 177777                          .WORD SPM4
61                                     : GPRMD SPM6,4,D,7777,0,7777,YES ; GET LOCAL ERROR LIMIT
62                                     : GPRMD SPM7,6,D,7777,0,7777,YES ; GET GLOBAL ERROR LIMIT
63                                     ENDSFT
64 101502                                L10101:
65                                     .EVEN
66
67 101502 105 116 101 SPM1: .ASCIZ 'ENABLE TRANSPORT TESTS '
68 101532 111 116 110 SPM4: .ASCIZ 'INHIBIT ITERATIONS '
69 101562 120 105 122 SPM6: .ASCIZ 'PER TEST ERROR LIMIT '
70 101612 120 105 122 SPM7: .ASCIZ 'PER UNIT ERROR LIMIT '
71                                     .SBTTL PATCH AREA
72
73                                     :
74                                     : FINALLY A GENEROUS PATCH AREA.
75                                     :
76                                     : AND AN ADJUSTMENT TO ACCOUNT FOR THE 'LASTAD BIT7' HACK
77                                     : DESCRIBED IN 'SUPPRG.MEM' (FOR REV C).
78                                     :
79
80 101642                                PATCH::
81                                     .B' KW 32.
82 102000 102000                          .= !377+1
83                                     LASTAD ;SET LAST USED ADDRESS.
84 102000 000000                          .EVEN
85 102002 000000                          .WORD 0
86 102004                                L$LAST::
87 102004                                .WORD 0
88                                     ENDMOD
89                                     .END
    
```

ADSSR 012126 G	CSAU = 000052	DEVDR0 023376	FREEHI 003130	INTPCPC 016150
ADR = 000020 G	CSAUTO= 000061	DEVNRD 023315	FRESIZ 003126 G	INTFLA 016145
AMBTSS 006635	CSBRK = 000022	DEVNXR 023233	FUSI 004117	INTMAS 016144
ASSEMB= 000010	CSBSEG= 000004	DEVONL 023163	FSAU = 000015	INTR 016216 G
A1716 = 000003	CSBSUB= 000002	DEVSUM 023126	FSAUTO= 000020	INTREC 002224 G
BADDAT 003156 G	CSCEFG= 000045	DFPTBL 002156 G	F\$BGN = 000040	INTVEC 016146
BADSSR 015700 G	CSCLCK= 000062	DIAGMC= 000000	F\$CLEA= 000007	INTX 004300
BDVPCR= 177520 G	CSCLEA= 000012	DICEB = 000001	F\$DU = 000016	INVERT 021206 G
BENBSW 002230 G	CS\$CLOS= 000035	DSBINT 016204	F\$END = 000041	IOKCKI= 000200
BIE = 040000	CSCLP1= 000006	DUAD12 004643	F\$HARD= 000004	IOKSTP= 000001
BIT0 = 000001 G	CS\$VEC= 000036	DUFLG 003112 G	F\$HW = 000013	IPRI 002212 G
BIT00 = 000001 G	CS\$DCLN= 000044	DUMMY 003062	F\$INIT= 000006	ISR = 000100 G
BIT01 = 000002 G	CS\$DODU= 000051	EF.CON= 000036 G	F\$JMP = 000050	IVEC 002210 G
BIT02 = 000004 G	CS\$DRPT= 000024	EF.NEW= 000035 G	F\$MOD = 000000	IXE = 004000 G
BIT03 = 000010 G	CS\$DU = 000053	EF.PWR= 000034 G	F\$MSG = 000011	ISAU = 000041
BIT04 = 000020 G	CS\$EDIT= 000003	EF.RES= 000037 G	F\$PROT= 000021	ISAUTO= 000041
BIT05 = 000040 G	CS\$ERDF= 000055	EF.STA= 000040 G	F\$PWR = 000017	ISCLN = 000041
BIT06 = 000100 G	CS\$ERHR= 000056	EMAXDU 016777	F\$RPT = 000012	ISDU = 000041
BIT07 = 000200 G	CS\$ERRO= 000060	EN = 000000	F\$SEG = 000003	ISHRD = 000041
BIT08 = 000400 G	CS\$ERSF= 000054	ENAINI 016152	F\$SOFT= 000005	ISINIT= 000041
BIT09 = 001000 G	CS\$ERSO= 000057	ENVIRN 020630	F\$SRV = 000010	ISMOD = 000041
BIT1 = 000002 G	CS\$ESCA= 000010	EPRTSW 002200 G	F\$SUB = 000002	ISMSG = 000041
BIT10 = 002000 G	CS\$ESEG= 000005	EPRT1 006360	F\$SW = 000014	ISPTAB= 000041
BIT11 = 004000 G	CS\$ESUB= 000003	EPRT2 006360	F\$TEST= 000001	ISPWR = 000041
BIT12 = 010000 G	CS\$ETST= 000001	ERRCM 011733	GDDAT 003160 G	ISRPT = 000041
BIT13 = 020000 G	CS\$EXIT= 000032	ERRHI 002236 G	GERRMA 002174 G	ISSEG = 000041
BIT14 = 040000 G	CS\$GETB= 000026	ERRK 016756	GETPAT 020174 G	ISSETU= 000041
BIT15 = 100000 G	CS\$GETW= 000027	ERRLO 002240 G	GETSEL 020256 G	ISSFT = 000041
BIT2 = 000004 G	CS\$GMAN= 000043	ERRNO = 002261	G\$CNT0= 000200	ISSRV = 000041
BIT3 = 000010 G	CS\$GPHR= 000042	ERRVEC= 000004 G	G\$DELM= 000372	ISSUB = 000041
BIT4 = 000020 G	CS\$GPLO= 000030	ERTABE 003376	G\$DISP= 000003	ISTST = 000041
BIT5 = 000040 G	CS\$GPRI= 000040	ERTABL 003176	G\$EXCP= 000400	JSJMP = 000167
BIT6 = 000100 G	CS\$INIT= 000011	ESUM 016760	G\$HILI= 000002	KIPAR0= 172340
BIT7 = 000200 G	CS\$INLP= 000020	EVL = 000004 G	G\$LOLI= 000001	KIPAR1= 172342
BIT8 = 000400 G	CS\$MANI= 000050	EXBCNT= 000010	G\$NO = 000000	KIPAR2= 172344
BIT9 = 001000 G	CS\$MEM = 000031	EXFMSG 101177	G\$OFFS= 000400	KIPAR3= 172346
BOE = 000400 G	CS\$MSG = 000023	EXPBRE 015502 G	G\$OFSI= 000376	KIPAR4= 172350
BRINIT 004457	CS\$OPEN= 000034	EXPD 002232 G	G\$PRMA= 000001	KIPAR5= 172352
BSELO = 000000	CS\$PNTB= 000014	EXPGOT 004533	G\$PRMD= 000002	KIPAR6= 172354
BSEL1 = 000001	CS\$PNTF= 000017	EXPGT2 004567	G\$PRML= 000000	KIPAR7= 172356
CHKAMB 016044	CS\$PNTS= 000016	EXPMSG 002322 G	G\$RADA= 000140	KIPDR0= 172300
CHKMAN 020500 G	CS\$PNTX= 000015	EXPREC 015474 G	G\$RADB= 000000	KIPDR1= 172302
CHKTSS 016336	CS\$QIO = 000377	EXTA 005772	G\$RADL= 000120	KIPDR2= 172304
CKDROP 017202	CS\$RDBU= 000007	EXTEND 005770	G\$RADO= 000020	KIPDR3= 172306
CKEMAX 017102	CS\$REFG= 000047	EXTFEA 002226 G	G\$XFER= 000004	KIPDR4= 172310
CKMSG 011360 G	CS\$RESE= 000033	E\$END = 002100	G\$YES = 000010	KIPDR5= 172312
CKMSG2 011500 G	CS\$REVI= 000003	E\$LOAD= 000035	HIADDR= 001400	KIPDR6= 172314
CKRAM 011114 G	CS\$RFLA= 000021	FATERR= 000060	HOE = 100000 G	KIPDR7= 172316
CKRAM2 011224 G	CS\$RPT = 000025	FATFLG 002222 G	HPM1 101362	KTENAB 003134 G
CMDPKT 021260 G	CS\$SEFG= 000046	FERCM 011722	HPM2 101416	KTFLG 003132 G
CMPMEM 017660	CS\$SPRI= 000041	FIFEXP 012170 G	HPM3 101442	KTINIT 021054
CONFIG 017250	CS\$SVEC= 000037	FIF1MS 012242	IBE = 010000 G	KTOFF 017274
COUNT 002310 G	CS\$TPRI= 000013	FIF2MS 012311	IDU = 000040 G	KTON 017256
CSRADD 002206 G	DATA 002312 G	FILLME 017422	IER = 020000 G	LERRMA 002172 G
CTAB 003164 G	DATASC 020232	FNOINT 004215	IFAUULT 004256	LERRNO= 000000
CTABE 003176 G	DEBUGM 011632	FORCER 002176 G	INCERK 017044	LISTAL= 000001
CTABM 003164 G	DEVCNT 002220 G	FREE 003124 G		

SYMBOL TABLE

LOE = 040000 G  
 LOOPCN 002216 G  
 LGOPCO 013126  
 LOOPFL 003162 G  
 LOT = 000010 G  
 LSACP 002110 G  
 LSAPT 002036 G  
 LSAU 022352 G  
 LSAUT 002070 G  
 LSAUTO 022556 G  
 LSCCP 002106 G  
 LSCLEA 022636 G  
 LSCO 002032 G  
 LSDEPO 002011 G  
 LSDESC 003410 G  
 LSDESP 002076 G  
 LSDEVP 002060 G  
 LSDISP 002124 G  
 LSDLY 002116 G  
 LSDTP 002040 G  
 LSDTYP 002034 G  
 LSDU 022450 G  
 LSDUT 002072 G  
 LSDVTY 003402 G  
 LSEF 002052 G  
 LSEVI 002044 G  
 LSETP 002102 G  
 LSEXP1 002046 G  
 LSEXP4 002064 G  
 LSEXP5 002066 G  
 LSHARD 101342 G  
 LSHIME 002120 G  
 LSHPCP 002016 G  
 LSHPTP 002022 G  
 LSHW 002156 G  
 LSICP 002104 G  
 LSINIT 021556 G  
 LSLADP 002026 G  
 LSLAST 102004 G  
 LSLOAD 002100 G  
 LSLUN 002074 G  
 LSMREV 002050 G  
 LSNAME 002000 G  
 LSPRIO 002042 G  
 LSPROT 021546 G  
 LSPRT 002112 G  
 LSREPP 002062 G  
 LSREV 002010 G  
 LSRPT 022664 G  
 LSSOFT 101474 G  
 LSSPC 002056 G  
 LSSPCP 002020 G  
 LSSPTP 002024 G  
 LSSTA 002030 G  
 LSSW 002166 G  
 LSTEST 002114 G  
 LSTIML 002014 G

LSUNIT 002012 G  
 L10000 002164  
 L10001 002176  
 L10002 005766  
 L10003 012044  
 L10004 012062  
 L10005 012100  
 L10006 012106  
 L10007 012124  
 L10010 012142  
 L10011 012166  
 L10012 012240  
 L10013 012410  
 L10014 013124  
 L10015 013752  
 L10016 013774  
 L10017 015500  
 L10020 015506  
 L10021 015514  
 L10022 015526  
 L10023 015550  
 L10024 015576  
 L10025 015736  
 L10026 016246  
 L10030 022302  
 L10031 022446  
 L10032 022554  
 L10033 022634  
 L10034 022662  
 L10035 023124  
 L10036 024424  
 L10037 026416  
 L10040 024700  
 L10041 025144  
 L10042 031772  
 L10043 026772  
 L10044 027262  
 L10045 027560  
 L10046 030170  
 L10047 034562  
 L10050 032326  
 L10051 032720  
 L10052 033326  
 L10053 040354  
 L10054 035646  
 L10055 036600  
 L10056 050466  
 L10057 040660  
 L10060 042070  
 L10061 043430  
 L10062 044006  
 L10063 045260  
 L10064 046324  
 L10065 051746  
 L10066 062574  
 L10067 053752  
 L10070 055240 G

L10071 056510  
 L10072 057754  
 L10073 066644  
 L10074 065004  
 L10075 074506  
 L10076 077660  
 L10077 101336  
 L10100 101362  
 L10101 101502  
 MEMADD 013754 G  
 MEMCK 021276 G  
 MENASC 020447  
 MENERR 020374  
 MENRES 020476  
 MIMENU 073372  
 MMVEC = 000250  
 MSA.FR= 000006  
 MSA.NO= 000000  
 MSA.NR= 000004  
 MSA.VO= 000002  
 MSGEXP 012144 G  
 MSGLOO 013064 G  
 MSGSTA 012350 G  
 MSGSUB 013742 G  
 MS.ATT= 000006  
 MS.EXT= 000200  
 MS.RSD= 000001  
 MS.RSF= 000020  
 MS.RST= 000010  
 M8186 005554  
 M8189 005645  
 NBA = 002000  
 NEWPAS 022010  
 NODEV 003114 G  
 NOEXTF 030164  
 NOINIT 004335  
 NOINTR 004221  
 NOITS 002170 G  
 NOMAN 020534  
 NOMEM 005460  
 NP.IR = 000200  
 NP.LOO= 000040  
 NP.OUT= 000100  
 NP.WRP= 000020  
 NSI 004152  
 NSINIT 004407  
 NUL 004527  
 NULCR 004530  
 NXM = 004000  
 NXMFLG 003136 G  
 NXMHI 003142 G  
 NXMLO 003140 G  
 NXMTST 021452  
 NXR 003740  
 NXRERR 005736 G  
 NXRX 003777  
 NXTU 022022

OFL = 000100  
 ONEFIL= 000000  
 OSAPTS= 000000  
 OSAU = 000001  
 OSBGNR= 000001  
 OSBGNS= 000001  
 OSDU = 000001  
 OSERRT= 000000  
 OSGNSW= 000001  
 OSPOIN= 000001  
 OSSETU= 000000  
 PASRPT 022054  
 PATCH 101642 G  
 PATDAT 020230  
 PC.ERA= 002400  
 PC.IER= 002000  
 PC.NOO= 001000  
 PC.REL= 000000  
 PC.REW= 000400  
 PKBCNT= 000006  
 PKHI = 000004  
 PKLOW = 000002  
 PKTADD 007554  
 PKTFRM 007516  
 PKTGET 012064 G  
 PKTMES 012110 G  
 PKTRAM 004745 G  
 PKTSSR 012046 G  
 PNT = 001000 G  
 PRAMPK 013776  
 PRASC 014523  
 PRBEXP 015470  
 PRBMSG 015336  
 PRBREC 015472  
 PRBTOT 015423  
 PRBYTE 015122 G  
 PRI = 002000 G  
 PRIADD 010160  
 PRIA0 010230  
 PRIBX0 007612 G  
 PRIEQU 010060  
 PRIPKT 007370 G  
 PRIRAM 010066  
 PRITAD 010274  
 PRITSS 006024  
 PRITO 010356  
 PRIT1 010421  
 PRIXOR 007742 G  
 PRI00 = 000000 G  
 PRI01 = 000040 G  
 PRI02 = 000100 G  
 PRI03 = 000140 G  
 PRI04 = 000200 G  
 PRI05 = 000240 G  
 PRI06 = 000300 G  
 PRI07 = 000340 G  
 PRMESS 014242

PRMNO 002320 G  
 PRMSGE 014552 G  
 PRMSG0 014732  
 PRMSG1 014777  
 PRMSG2 015035  
 PROASC 014420  
 PRIASC 014465  
 PST32W 003152 G  
 PUNIT 022304  
 PW.D11= 000021  
 PW.D13= 000022  
 PW.D22= 000020  
 PW.NOP= 000000  
 PW.NO1= 000023  
 PW.RDE= 000024  
 PW.RDR= 000001  
 PW.RDS= 000005  
 PW.RFI= 000003  
 PW.WCT= 000006  
 PW.WFI= 000004  
 PW.WFM= 000007  
 PW.WMI= 000010  
 PW.WNP= 000011  
 PW.WTR= 000002  
 P.ACK = 100000  
 P.CMD = 000037  
 P.CONT= 000012  
 P.CVC = 040000  
 P.FMT = 000140  
 P.FORM= 000011  
 P.GETS= 000017  
 P.IE = 000200  
 P.INIi= 000013  
 P.MODE= 007400  
 P.OPP = 020000  
 P.POSI= 000010  
 P.READ= 000001  
 P.SWB = 010000  
 P.WRIT= 000005  
 P.WRTC= 000004  
 P.WRTS= 000006  
 QVP 002204 G  
 RAMASC 014156  
 RAMDAT 002242 G  
 RAMERR 015510 G  
 RAMEXP 015530 G  
 RAMFOR 010116  
 RAMSIZ 002302 G  
 RAMTAD 015516 G  
 RCVHIA 002304 G  
 RCVLOA 002306 G  
 RDERR 005206  
 RECMSG 002466 G  
 RECV 002234 G  
 REGSAV 020140  
 RETERR 005372  
 REWIND 011014 G

RMCHBE=	000167	S1.IID=	004000	TST40I	101322	T10	066646	G	T15BFR	033372	
RMCHEN=	000200	S1.I1R=	020000	TSV2	002000	T11	074510	G	T15BF2	034060	
RMMSGB=	000215	S1.I2R=	040000	TSV3	002176	T12	077662	G	T15BS0	034060	
RMMSGG=	000234	S1.PAR=	100000	TSV4	021546	T12BFR	030262		T15BS1	034061	
RMPKTB=	000201	S2.ATI=	000010	TSV5	023446	T12BKG	030737		T15DAT	033360	
RMPKTE=	000210	S2.BTI=	000004	TSV6	101340	T12BLK	030314		T15LOO	032024	
RMR	= 010000	S2.DIM=	000200	TT1BFR=	177562	T12CHA	031730		T15PAC	033350	
RWPACK	011110	S2.ILW=	000100	TT1CSR=	177560	T12CKR	031510	G	T15PK2	034050	
SC	= 100000	S2.INR=	000020	TTION	100406	T12CON	031316		T15RES	034452	
SCE	= 020000	S2.OUT=	000040	TTION2	071434	T12DAT	030250		T15RT2	034524	
SCHERR	005300	S2.UND=	000003	TTIVEC=	000060	T12DPR	031116		T15SSR	034066	
SCME	005013	TBLEND=	003062	G	TVECSA	100410	T12GET	030476	T15S2	034062	
SCMENU	100414	TCOASC	006476		TVSAV2	071436	T12HIA	030302	T15S3	034064	
SDELAY	010660	TCOCOD	006676		TSARGC=	000001	T12KT	030310	T16BEN	040240	
SELASC	020442	TEMP1	003116	G	TSCODE=	001130	T12LOA	030304	T16BFR	040212	
SELDAT=	000004	TEMP2	003120	G	TSERRN=	002261	T12LOO	026446	T16BFS	040232	
SEL2	= 000002	TERCLS=	000016		TSEXCP=	000000	T12MSG	030641	T16CLR	040056	
SETMAP	017316	TESTNO=	000014		TSFLAG=	000040	T12NIN	031025	T16DAT	040200	
SETU	022106	TEXASC	006435		TSGMAN=	000000	T12NXM	031207	T16DT2	040250	
SFFMSG	012102	TFCASC	006537		TSHILI=	000776	T12PAC	030240	T16D01	036632	
SFHERR	003705	TIMEXP	015552	G	TSLAST=	000001	T12PAR	030306	T16D28	036634	
SFIERR	003652	TIMSGO	015600		TSLOLI=	000000	T12SET	031666	T16D53	036636	
SFIMSG	012034	TINERR	012021		TSLSYM=	010000	T12SWR	031620	T16D78	036640	
SFPTBL	002166	TMPBFR	002632	G	TSLTNO=	000014	T12TBE	030446	T16LEN	037162	
SIFLAG	003154	TNAM	016704		TSNEST=	177777	T12WRT	030552	T16LOO	034602	
SIMSG	011766	TPRISA	100412		TSNS0 =	000000	T121LO	026560	T16PAC	040170	
SKIPT	003400	TPSAV2	071440		TSNS1 =	000005	T122LO	027114	T16PK2	040240	
SOFINI	015774	TRANST	002166	G	TSNS2 =	000002	T123LO	027334	T16REJ	037274	
SPACE	010466	TSBA =	000000	G	TSNS3 =	000003	T124LO	027734	T16RES	040010	
SPM1	101502	TSBAH =	000001	G	TSPTNU=	000000	T124TS	030312	T16SEX	040150	
SPM4	101532	TSDB =	000000	G	TS\$AVL=	177777	T13BFR	024062	T16SRD	040102	
SPM6	101562	TSDBH =	000001	G	TS\$EGL=	177777	T13DAT	024050	T16SSR	036712	
SPM7	101612	TSFCOD	007236		TS\$EKO=	010000	T13LCO	023464	T16TSB	037060	
SRO	= 177572	TSREJ =	000006		TS\$SUBN=	000000	T13MEM	024155	T16T01	037411	
SR1	= 177574	TSSDEF	006606		TSTAGL=	177777	T13NBA	024214	T16T28	037510	
SR2	= 177576	TSSR =	000002	G	TSTAGN=	010102	T13PAC	024040	T16T53	037610	
SR3	= 172516	TSSRBI	003502	G	T\$TEMP=	000000	T13RES	024356	T16T78	037710	
SSR	= 000200	TSSRFO	006415		T\$TEST=	000014	T13SSR	024267	T16WMI	040122	
STATCO	012412	TSSRH =	000003	G	T\$TSTM=	177777	T14BFR	025176	T162SS	036747	
SVCGBL=	000000	TSSX	004020		T\$TSTS=	000001	T14BS0	025170	T163SS	037013	
SVCINS=	000000	TSTBLK	002752	G	T\$SAU =	010031	T14BS1	025171	T17BEN	050342	
SVC\$SUB=	000001	TSTCNT	002214	G	T\$SAUT=	010033	T14BS2	025172	T17BFR	050222	
SVCTAG=	000000	TSTEND	016720		T\$SCLE=	010034	T14DAT	025170	T17BFS	050242	
SVCTST=	000001	TSTFLA	002314	G	T\$SDU =	010032	T14DTA	025610	T17CLE	050126	
S\$LSYM=	010000	TSTLOO	016456	G	T\$SHAR=	010100	T14LOO	024444	T17CLR	047740	
SO.IDB=	000010	TSTPTR	002316	G	T\$SHW =	010000	T14NBA	025622	T17DAT	050210	
SO.IFB=	000002	TSTSET	016510	G	T\$SINI=	010030	T14NIN	026063	T17DT2	050360	
SO.IFP=	000001	TST12I	030450		T\$MSG=	010025	T14PAC	025160	T17EXE	046504	
SO.ILD=	000020	TST13I	024102		T\$PRO=	010027	T14PK2	025600	T17EXP	046362	
SO.ION=	000040	TST14I	026241		T\$SRPT=	010035	T14RES	026306	T17EXS	046402	
SO.IRD=	000100	TST15I	034433		T\$SSEG=	010000	T14RST	026344	T17LOO	040404	
SO.IRW=	000004	TST16I	036642		T\$SOF=	010101	T14SSR	025773	T17MSK	046356	
SO.ISP=	000200	TST17I	046606		T\$SSRV=	010026	T14TSB	026155	T17PAC	050200	
S1.ICE=	002000	TST18I	051176		T\$SSUB=	010074	T142RE	025676	T17PK2	050350	
S1.IEO=	010000	TST19I	060162		T\$SSW =	010001	T15AM2	034142	T17RFI	050106	
S1.IFM=	001000	TST20I	065162		T\$STES=	010077	T15AM3	034243	T17RSF	050004	
S1.IHE=	000400	TST39I	077632		T1	023446	G	T15AM4	034345	T17SET	050146



SYMBOL TABLE	
T17SNP	050026
T17SRD	047764
T17SSR	046625
T17WFD	046504
T17WFI	050052
T171CM	047145
T172CM	047227
T172SS	046662
T173CM	047323
T173SS	046726
T174CM	047407
T174SS	046773
T175CM	047472
T175SS	047036
T176CM	047546
T176SS	047102
T177CM	047654
T18BFR	051702
T18CMP	051403
T18DAT	051670
T18DT2	051740
T18EXP	051146
T18LOO	050506
T18MSK	051142
T18PAC	051660
T18PK2	051730
T18SET	051550
T18SMI	051526
T18SRD	051506
T18SSR	051235
T18XS	051172
T182SS	051272
T183SS	051336
T19BEN	062452
T19BFC	057776
T19BFR	062332
T19BFS	062352
T19CLE	062200
T19CLR	061734
T19CNV	062244
T19DAT	062320
T19DT2	062470
T19EXE	060162
T19EXP	060042
T19XS	060062
T19LOO	051766
T19MSK	060036
T19PAC	062310
T19PK2	062460
T19PRE	057774
T19RFI	062046
T19RSF	062000
T19RST	061630
T19SET	062220
T19SEX	062162
T19SNP	062022
T19SRD	061760
T19SSR	060223
T19WCT	062122
T19WFI	062066
T19WFM	062142
T19WST	061523
T191CM	060660
T192CM	060742
T192SS	060260
T193CM	061036
T193SS	060324
T194SS	060371
T195CM	061124
T195SS	060434
T196CM	061200
T196SS	060500
T197CM	061263
T197SS	060543
T198CM	061351
T198SS	060612
T199CM	061440
T2	024426 G
T2.1	024444
T2.2	024714
T20BEN	066522
T20BFR	066402
T20BFS	066422
T20CLE	066314
T20CLR	066102
T20CWP	066007
T20DAT	066370
T20DT2	066540
T20EXE	065162
T20EXP	065042
T20EXS	065062
T20LOO	062614
T20MSK	065036
T20PAC	066360
T20PK2	066530
T20RFI	066166
T20RSF	065706
T20SET	066334
T20SEX	066276
T20SRD	066126
T20SSR	065211
T20SWP	065577
T20WFI	066206
T20WFM	066242
T20WMI	066262
T20WNP	066146
T202SS	065246
T203SS	065312
T204SS	065357
T205SS	065422
T206SS	065466
T208SS	065531
T23A	003144 G
T23B	003146 G
T3	026420 G
T3BFLG	003150 G
T3.1	026446
T3.2	027006
T3.3	027276
T3.4	027574
T38ASC	074441
T38ASN	074460
T38ASO	074336
T38AS1	074403
T38BFR	071466
T38BSO	071460
T38BS1	071461
T38BS2	071462
T38CNT	074504
T38DAT	073774
T38DLY	071442
T38DSW	072170
T38DTA	072160
T38EAI	072166
T38EB	072142
T38ID	073345
T38INT	072736
T38MBP	074034
T38MSG	073305
T38MS1	073146
T38MS2	073241
T38MS3	072305
T38NBA	072572
T38NE	072230
T38OFL	073022
T38ONL	072766
T38PAC	071450
T38PK2	072150
T38PK3	072200
T38PK4	072220
T38RES	073776
T38SIZ	072226
T38SSR	072646
T38SST	073056
T38TAD	071460
T38WLE	072525
T38WR	072222
T38WRL	072464
T38WRT	072400
T39BFR	075746
T39BSO	075740
T39BS1	075741
T39BS2	075742
T39DAT	077572
T39DLY	075724
T39DSW	076450
T39DTA	076440
T39EAI	076446
T39ETN	076651
T39ETS	076562
T39MCI	077513
T39NBA	077205
T39NE	076503
T39NFL	076500
T39OFF	077475
T39OF2	076744
T39ON	077504
T39ON2	077010
T39PAC	075730
T39PK2	076430
T39PK3	076460
T39PK4	076470
T39RES	077574
T39RL	077570
T39SBS	077423
T39SFS	077351
T39SIZ	076476
T39SSR	077261
T39TAD	075740
T39WPN	077126
T39WR	076472
T39WRT	077053
T4	031774 G
T4.1	032024
T4.2	032330
T4.3	032722
T4ONE	101255
T5	034564 G
T5.1	034602
T5.2	035662
T6	040356 G
T6.1	040404
T6.2	040674
T6.3	042104
T6.4	043444
T6.5	044022
T6.6	045274
T7	050470 G
T8	051750 G
T8.1	051766
T8.2	053766
T8.3	055254
T8.4	056524
T9	062576 G
T9.1	062614
UAM	= 000200 G
UNITN	= 002202 G
UNREC	= 000006
USI	004123
WAITF	016250 G
WC.IFA	= 000200
WC.IFE	= 000002
WC.IGO	= 000001
WC.IRE	= 000010
WC.IRW	= 000004
WC.IOT	= 000100
WC.IIT	= 000040
WC.ISR	= 000020
WF.IED	= 000010
WF.IER	= 000004
WF.IHI	= 000200
WF.IRE	= 000040
WF.IWF	= 000020
WF.IWR	= 000100
WF.I3R	= 000002
WF.I4R	= 000001
WRTCHR	010662 G
WRTERR	005113
WRTMSG	005056
WSMBK	021270 G
XFERAS	015740
XNXM	016376
XORBFO	007674
XORFOR	010012
XST0	= 000006 G
XST1	= 000010 G
XST2	= 000012 G
XST3	= 000014 G
XST4	= 000016 G
XSOBOT	= 000002
XSOEOT	= 000001
XSOIE	= 000040
XSOILA	= 000400
XSOILC	= 001000
XSOLET	= 020000
XSOMOT	= 000200
XSONEF	= 002000
XSOONL	= 000100
XSOPEL	= 000010
XSORLL	= 010000
XSORLS	= 040000
XSOVMK	= 100000
XSOVCK	= 000020
XSOVLE	= 004000
XSOVLK	= 000004
XXCOMM	003122 G
X\$ALWA	= 000000
X\$FALS	= 000040
X\$OFFS	= 000400
X\$TRUE	= 000020
X1.COR	= 020000
X1.DLT	= 100000
X1.MBZ	= 017375
X1.RBP	= 000400
X1.SPA	= 040000
X1.UNC	= 000002
X2.BUF	= 000100
X2.EXT	= 000200
X2.OPM	= 100000
X2.RCE	= 040000
X2.REV	= 000077
X2.SPA	= 035400
X2.UNI	= 000007
X2.WCF	= 002000
X3.DCK	= 000010

X3.MBZ= 000006  
X3.MDE= 177400  
X3.OPI= 000100

X3.REV= 000040  
X3.RIB= 000001  
X3.SPA= 000200

X3.TRF= 000020  
X4.HSP= 100000

X4.MBZ= 017400  
X4.RCE= 040000

X4.TSM= 020000  
X4.WRC= 000377

. ABS. 102004 000  
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 31032 WORDS ( 122 PAGES)

DYNAMIC MEMORY: 20346 WORDS ( 78 PAGES)

ELAPSED TIME: 00:38:31

CVTSBA0, CVTSBA0=SVC/ML, TSV1B, TSV22B, TSV3B, TSV4, TSV55B, TSV6