

TSV05

TSV05 CTRL LT2
CVTSBA0

AH-T096A-MC
FICHE 1 OF 2

SEP 1982
COPYRIGHT © 1982
MADE IN USA



TSV05

TSV05 CTRL LT2
CVTSBA0

AH-T096A-MC
FICHE 2 OF 2

SEP 1982
COPYRIGHT © 1982
MADE IN USA



.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 RELIABILITY 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 SUCESSFULLY.

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 'DDDDD'.

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

EXAMPLE OF SWITCH USAGE:

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

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

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

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

2.3 FLAGS

FLAGS ARE USED TO SET UP CERTAIN OPERATIONAL PARAMETERS SUCH AS LOOPING ON ERROR. ALL FLAGS ARE CLEARED AT STARTUP AND REMAIN CLEARED UNTIL EXPLICITLY SET USING THE FLAGS SWITCH. FLAGS ARE ALSO CLEARED AFTER A START COMMAND UNLESS SET USING THE FLAG SWITCH. THE ZFLAGS COMMAND MAY ALSO BE USED TO CLEAR ALL FLAGS. WITH THE EXCEPTION OF THE START AND ZFLAGS COMMANDS, NO COMMANDS AFFECT THE STATE OF THE FLAGS; THEY REMAIN SET OR CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

FLAG	EFFECT
----	-----
HOE	HALT ON ERROR - CONTROL IS RETURNED TO RUNTIME SERVICES COMMAND MODE
LOE	LOOP ON ERROR
IER*	INHIBIT ALL ERROR REPORTS
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 SOHZ (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 TIMEOUT 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 'W 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/ONLINE 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      .ASCII /A/
002011      L$DEPO::                     ;0
002011      .ASCII /O/
002012      L$UNIT::                     ;NUMBER OF UNITS
002012      .WORD  0
002014      L$TIML::                     ;LONGEST TEST TIME
002014      .WORD  655.
002016      L$HPCP::                     ;POINTER TO H.W. QUES.
002016      .WORD  L$HARD
002020      L$SPCP::                     ;POINTER TO S.W. QUES.
002020      .WORD  L$SOFT
002022      L$HPTP::                     ;PTR. TO DEF. H.W. PTABLE
002022      .WORD  L$HW
002024      L$SPTP::                     ;PTR. TO S.W. PTABLE
002024      .WORD  L$SW
002026      L$LADP::                     ;DIAG. END ADDRESS
002026      .WORD  L$LAST
002030      L$STA::                      ;RESERVED FOR APT STATS
002030      .WORD  0
002032      L$CO::                      ;DIAGNOSTIC TYPE
002032      .WORD  0
002034      L$DTYP::                     ;APT EXPANSION
002034      .WORD  0
002036      L$APT::                      ;PTR. TO DISPATCH TABLE
002036      .WORD  0
002040      L$DTP::                      ;PTR. TO DISPATCH TABLE
002040      .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	CSEDIT	
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 077662DISPATCH 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 :++
46 : THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
47 : THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
48 : IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
49 :--
    BGNHW DFPTBL ;DEFAULT HARD-P-TABLE
    .WORD L10000-L$HW/2
L$HW::
DFPTBL::
50
51 .WORD 172520 ; 1ST (OF 2) REGISTERS.
52 .WORD 224 ; INTERRUPT VECTOR
53 .WORD PRI04 ; INTERRUPT PRIORITY.
54 ENDPHW
    L10000:
```

```
56                                     .SBTTL  SOFTWARE P-TABLE
57
58
59      :++
60      : THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
61      : PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
62      :--
62      002164      BGNSW      SFPTBL
        002164      .WORD      L10001-L$SW/2
        002166
        002166      L$SW::
        002166      SFPTBL::
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      002172      000017      LERRMAX::      .WORD      15.      ; ... 0 = ITERATE.
68      002174      000310      GERRMAX::      .WORD      200.      ; ...NZ = INHIBIT ITERATE.
69      002176
70      002176      L10001:      ENDSW      ; LOCAL (PER TEST) ERROR LIMIT
71
72      002176      ENDMOD      ; GLOBAL (PER UNIT) ERROR LIMIT
```

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

KT11

.SBTTL 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
```

; DEFINE MEMORY MANAGEMENT REGISTERS

```
.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      XS0ILC= BIT9       ;ILLEGAL COMMAND
83         000400      XS0ILA= BIT8       ;ILLEGAL ADDRESS
84         000200      XSOMOT= BIT7       ;TAPE IN MOTION
85         000100      XS0ONL= BIT6       ;TRANSPORT ON LINE
86         000040      XS0IE=  BIT5       ;INTERRUPT ENABLE
87         000020      XS0VCK= BIT4       ;VOLUME CHECK BIT
88         000010      XS0PED= BIT3       ;PHASE ENCODED DRIVE
89         000004      XS0WLK= BIT2       ;WRITE LOCKED
90         000002      XS0BOT= BIT1       ;BEGINNING OF TAPE
91         000001      XS0EOT= BIT0       ;END OF TAPE
92
93                                     :+
94                                     :BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 1
95                                     : (XST1)
96

```

```

97
98      100000      ;X1.DLT = BIT15 ;DATA LATE
99      040000      ;X1.SPARE= BIT14 ;NOT USED
100     020000      ;X1.COR = BIT13 ;CORRECTABLE DATA ERROR
101     017375      ;X1.MBZ = BIT12+BIT11+BIT10+BIT9+BIT7+BIT6+BIT5+BIT4+BIT3+BIT2+BIT0 ;ALWAYS 0
102     000400      ;X1.RBP = BIT8 ;READ BUS PARITY ERROR
103     000002      ;X1.UNC = BIT1 ;UNCORRECTABLE DATA OR HARD ERROR
104
105      ;+
106      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 2
107      ;(XST2)
108      ;-
109     100000      ;X2.OPM = BIT15 ;OPERATION IN PROGRESS (TAPE MOVING)
110     040000      ;X2.RCE = BIT14 ;RAM CHECKSUM ERROR
111     035400      ;X2.SPARE= BIT13+BIT12+BIT11+BIT9+BIT8 ;NOT USED BY TSV05 (ALWAYS=0)
112     002000      ;X2.WCF = BIT10 ;WRITE CLOCK FAILURE (FIFO NOT EMPTIED BY TRANSPORT)
113     000200      ;X2.EXTF = BIT7 ;IF WRITE CHAR CMD THEN = EXTENDED FEATURES ENABLED
114     000100      ;X2.BUFE = BIT6 ;IF WRITE CHAR CMD THEN = BUFFERING ENABLED
115     000077      ;X2.REV = 000077 ;IF WRITE CHAR CMD THEN = MICROCODE REVISION LEVEL
116     000007      ;X2.UNIT = BIT2+BIT1+BIT0 ;IF GET STATUS THEN = CURRENTLY SELECTED UNIT NO.
117
118      ;+
119      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 3
120      ;(XST3)
121      ;-
122     177400      ;X3.MDE = 177400 ;MICRO-DIAGNOSTIC ERROR CODE
123     000200      ;X3.SPARE= BIT7 ;NOT USED BY TSV05
124     000100      ;X3.OPI = BIT6 ;OPERATION INCOMPLETE
125     000040      ;X3.REV = BIT5 ;REVERSE
126     000020      ;X3.TRF = BIT4 ;TRANSPORT RESPONSE FAILURE
127     000010      ;X3.DCK = BIT3 ;DENSITY CHECK
128     000006      ;X3.MBZ =BIT2+BIT1 ;NOT USED ALWAYS 0
129     000001      ;X3.RIB = BIT0 ;REVERSE INTO BOT
130
131      ;+
132      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 4
133      ;(XST4)
134      ;-
135     100000      ;X4.HSP = BIT15 ;HIGH SPEED
136     040000      ;X4.RCE = BIT14 ;RETRY COUNT EXCEEDED
137     020000      ;X4.TSM = BIT13 ;TRANSPORT SPECIAL MODE
138     017400      ;X4.MBZ = BIT12+BIT11+BIT10+BIT9+BIT8 ;NOT USED ALWAYS 0
139     000377      ;X4.WRC = 000377 ;WRITE RETRY COUNT FIELD
140
141      ;+
142      ;TSSR TERMINATION CODES (BIT 0-2)
143      ;-
144
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      RMSGGBEG= 215      ;MESSAGE BUFFER BEGIN RAM ADDRESS
210      000234      RMSGGEN= 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      000000      BSELO  = 0      ;BYTE 0
240      000001      BSEL1  = 1      ;BYTE 1
241      000002      SEL2   = 2      ;WORD 2
242      000004      SELDATA = 4      ;WORD 3
243
244      ;+
245      ;BSELO SELECT CODES FOR WRITE SUBSYSTEM COMMAND
246      ;-
247      000000      PW.NOP      = 0      ;NO-OP
248      000001      PW.RDRAM    = 1      ;READ RAM
249      000002      PW.WTRAM    = 2      ;WRITE RAM
250      000003      PW.RFIFO    = 3      ;READ FIFO
251      000004      PW.WFIFO    = 4      ;WRITE FIFO
252      000005      PW.RDSTAT   = 5      ;READ STATUS
253      000006      PW.WCTL     = 6      ;WRITE TAPE CONTROL
254      000007      PW.WFMT     = 7      ;WRITE TAPE FORMAT
255      000010      PW.WMISC    = 10     ;WRITE MISCELLANEOUS
256      000011      PW.WNPR     = 11     ;WRITE NPR CONTROL
257      000020      PW.D22      = 20     ;DO MICROTEST 22
258      000021      PW.D11      = 21     ;DO MICROTEST 11
259      000022      PW.D13      = 22     ;DO MICROTEST 13
260      000023      PW.NO1311   = 23     ;DISABLE MICROTEST 11 AND 13
261      000024      PW.RDEXT    = 24     ;READ EXT. TAPE STATUS (NOT SUPPORTED BY ALL TRANSPORTS)
262
263      ;+
264      ;BSEL1 CODES FOR WRITE TAPE CONTROL
265      ;-
266      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	S0.ISPEED	= BIT7	:WORD #8 BYTE 0	ISPEED H
330	000100	S0.IRDY	= BIT6	:	IRDY L
331	000040	S0.IONL	= BIT5	:	IONL L
332	000020	S0.ILDPL	= BIT4	:	ILDPL L
333	000010	S0.IDBY	= BIT3	:	IDBY L
334	000004	S0.IRWD	= BIT2	:	IRWD L
335	000002	S0.IFBY	= BIT1	:	IFBY L
336	000001	S0.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

000000

002176 000000

;MACRO TO PERFORM XOR
;-

```

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

```

```

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.
;

```

```

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.

```

```
422                                     .SBTTL  GLOBAL DATA SECTION
423
424                                     :++
425                                     :THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
426                                     :IN MORE THAN ONE TEST.
427                                     :--
428
429                                     :
430                                     :THE FOLLOWING DATA ARE SET FOR EACH UNIT AT INIT TIME.
431                                     :SINGLE UNIT DEFAULTS (LISTED) ARE IN THE DEFAULT P-TABLE.
432
433 002200 000000 EPRTSW::          .WORD 0          ;PRINT SWITCH
434 002202 000000 UNITN::          .WORD 0          ;UNIT # UNDER TEST.
435 002204 000000 QVP::           .WORD 0          ;QUICK VERIFY FLAG.
436 002206 000000 CSRADDR::       .WORD 0          ;ADDRESS OF CSR FOR CURRENT DEVICE
437 002210 000224 IVEC::          .WORD 224        ;INTERRUPT VECTOR
438 002212 000200 IPRI::          .WORD PRI04      ;INTERRUPT PRIORITY.
439 002214 000000 TSTCNT::        .WORD 0          ;NUMBER OF TESTS RUN IN THIS PASS
440 002216 000000 LOOPCNT::       .WORD 0          ;REMAINING ITERATION COUNT FOR TEST
441 002220 000000 DEVCNT::        .WORD 0          ;NUMBER OF DEVICE UNDER TEST
442 002222 000000 FATFLG::        .WORD 0          ;SET IF FATAL ERROR IS DETECTED IN TEST
443 002224 000000 INTRECV::       .WORD 0          ;SET IF TAPE INTERRUPT WAS RECEIVED
444 002226 000000 EXTFEA::        .WORD 0          ;EXTENDED FEATURES SOFTWARE SW 0=OFF;1=ON
445 002230 000000 BENBSW::        .WORD 0          ;BUFFER ENABLE SWITCH SW 0=OFF;1=ON
446 002232 000000 EXPD::          .WORD 0          ;EXPECTED RAM DATA FOR PRAMPKT ROUTINE
447 002234 000000 RECV::          .WORD 0          ;RECEIVED RAM DATA FOR PRAMPKT ROUTINE
448 002236 000000 ERRHI::         .WORD 0          ;HIGH ADDRESS MEMORY ERROR
449 002240 000000 ERRLO::         .WORD 0          ;LOW ADDRESS MEMORY ERROR
450 002242 000000 RAMDATA::       .BLKW 16.        ;DATA READ FROM RAM PACKET OR MESSAGE BUF AREA
451 002302 000000 RAMSIZ::        .WORD 0          ;RAM DATA SIZE FOR PRAMPKT ROUTINE
452 002304 000000 RCVHIADD::      .WORD 0          ;RECEIVED BUFFER HIGH ADDRESS
453 002306 000000 RCVLOADD::      .WORD 0          ;RECEIVED BUFFER LOW ADDRESS
454 002310 000000 COUNT::        .WORD 0          ;TEST COUNT PATTERN
455 002312 000000 DATA::         .WORD 0          ;TEST DATA
456 002314 000000 TSTFLAG::      .WORD 0          ;TEST FLAG WORD
457 002316 000000 TSTPTR::       .WORD 0          ;TSTBLK POINTER
458 002320 000000 PRMNO::         .WORD 0          ;PRINT ROUTINE TEMP
459 002322 000000 EXPMSG::        .BLKB 100.        ;EXPECTED MESSAGE BUFFER DATA
460 002466 000000 RECMSG::        .BLKB 100.        ;RECEIVED MESSAGE BUFFER DATA
461 002632 000000 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
481 002754
482 002756
483 002760
484 002762
485 002764
486 002766
487 002770
488 002772
489 002774
490 002776
491 003000
492 003002
493 003004
494 003006
495 003010
496 003012
497 003014
498 003016
499 003020
500 003022
501 003024
502 003026
503 003030
504 003032
505 003034
506 003036
507 003040
508 003042
509 003044
510 003046
511 003050
512 003052
513 003054
514 003056
515 003060
516

000000
177777
000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000
177776
177775
177773
177767
177757
177737
177677
177577
177377
176777
175777
173777
167777
157777
137777
077777
125252
052525
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
560 CTABE::
561 ; ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:
562
563 0 = UNIT NOT TESTED
564 100000 = UNIT ONLINE, NO ERRORS
565 10XXXX = UNIT ONLINE, ENCOUNTERED XXXX ERRORS
566 160000 = UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
567 160001 = UNIT DROPPED, NOT IDLE AT START
568 14XXXX = UNIT DROPPED, ENCOUNTERED XXXX ERRORS
569
570 003176 000000 ERTABL: .BLKW 64.
571 003376 000000 ERTABE: .WORD 0
572
573 003400 000000 SKIPT: .WORD 0 ; 1=SKIP SUBTEST 0=NO SKIP OF SUBTEST

```

```
573 .SBTTL GLOBAL TEXT MESSAGES
574
575 :++
576 : THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
577 : MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
578 : MORE THAN ONE TEST.
579 :--
580
581
582 :+
583 : NAMES OF DEVICES SUPPORTED
584 :--
585
586 003402          DEVTYP <TSV05>
003402          L$DVTYP::
003402          124      123      126          .ASCIZ /TSV05/
          .EVEN
587
588
589 :+
590 : TEST DESCRIPTION
591 :--
592
593 003410          DESCRIPT <**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****>
003410          L$DESC::
003410          052      052      052          .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%A,%06%N/
```

```
644 004060 045 101 040 .ASCIZ /%A TSBA,TSSR REC'D: %06XA,%06/
645 004117 045 116 045 FUSI: .ASCII /%N%
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%
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: %06XA 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: %06XA, REC'D: %06/
659 004567 045 116 045 EXPGT2: .ASCIZ /%N% EXP'D: %06XA, %06%N% REC'D: %0XA, %06/
660 004643 045 101 040 DUAD12: .ASCIZ /%A REG(W) WRITTEN TO: %06XA REG(R) READ; EXP'D: %06XA, 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% ***** NO NXM ADDRESS--CANNOT TEST NXM TIMEOUT. *****%N'
669 005554 045 116 045 M8186: .ASCIZ '%N% ***** 11/23A SYSTEM *****%N'
670 005645 045 116 045 M8189: .ASCIZ '%N% ***** 11/23B SYSTEM *****%N'
671 .EVEN
672
673
674
```

```

676                                     .SBTTL  GLOBAL ERROR REPORT SECTION
677
678                                     :++
679                                     : THE GLOBAL ERROR REPORT SECTION CONTAINS THE PRINTB AND PRINTX
680                                     : CALLS THAT ARE USED IN MORE THAN ONE TEST.
681                                     : ASCII TEXT STRINGS ARE FOUND IN THE GLOBAL TEXT SECTION.
682                                     :--
683
684 005736      BGNMSG  NXRRERR      ;NON-EXISTANT DEVICE REGISTER.
685 005736      NXRRERR:
686 005736      PRINTX  #NXRX,NODEV  ;NODEV = NEXM ADDRESS.
687 005736      MOV     NODEV,-(SP)
688 005742      MOV     #NXRX,-(SP)
689 005746      MOV     #2,-(SP)
690 005752      MOV     SP,R0
691 005754      TRAP    C$PNTX
692 005756      ADD     #6,SP
693 005762      JSR     PC,EXTEND    ; PRINT EXTENSION IF REQUIRED.
694 005766      ENDMSG
695 005766      L10002:
696 005766      TRAP    C$MSG
697
698                                     :
699                                     : THIS ROUTINE APPENDS A UNIQUE EXTENSION (IF REQUIRED)
700                                     : TO ANY OF THE ABOVE ERROR SIGNATURES.
701
702 EXTEND: TST     (PC)+
703 EXTA:  0        ; 0 = NO EXTENSION.
704       BEQ      1$
705       JSR      PC,@EXTA
706 1$:  PRINTX    #NULCR
707       MOV      #NULCR,-(SP)
708       MOV      #1,-(SP)
709       MOV      SP,R0
710       TRAP    C$PNTX
711       ADD     #4,SP
712       RTS     PC

```

```

702                                     .SBTTL PRITSSR - PRINT TSSR CONTENTS
703
704                                     :+
705                                     :ROUTINE TO DISPLAY THE CONTENTS, AND BIT DEFINITIONS, OF
706                                     :THE TSSR REGISTER. THIS ROUTINE IS NORMALLY CALLED ONLY
707                                     :BY A MESSAGE PRINTING ROUTINE
708
709                                     :INPUTS:
710                                     :
711                                     :      R1      CONTENTS OF TSSR
712
713                                     :SUBORDINATE ROUTINES:
714                                     :
715                                     :      CHKAMB  CHECK FOR AMBIGUOUS CONTENTS
716
717                                     :-
718
719
720 PRITSSR:
721     SAVREG                          :SAVE GENERAL REGISTERS
722     MOV      R1,R4                  :SAVE THE TSSR CONTENTS
723     PRINTB   #TSSRFOR,R4           :PRINT THE CONTENTS OF TSSR
724     MOV      R4,-(SP)
725     MOV      #TSSRFOR,-(SP)
726     MOV      #2,-(SP)
727     MOV      SP,R0
728     TRAP     C$PNTB
729     ADD      #6,SP
730     MOV      R4,R0
731     JSR      PC,CHKAMB              :GET TSSR BACK FOR CHKAMB
732     BCS      5$                    :ARE CONTENTS AMBIGUOUS ?
733     PRINTX   #AMBTSSR              :BRANCH IF NOT
734     MOV      #AMBTSSR,-(SP)        :SHOW CONTENTS ARE AMBIGUOUS
735     MOV      #1,-(SP)
736     MOV      SP,R0
737     TRAP     C$PNTX
738     ADD      #4,SP
739     5$:     MOV      R4,R3          :CONTENTS OF TSSR
740     BIC      #HIADDR!FATERR!TERCLS,R3 :CLEAR ALL MULTIPLE BIT FIELDS
741     BEQ      20$                    :NO BITS ARE SET
742     MOV      #TMPBFR,P2            :TEMPORARY ASCII BUFFER
743     MOV      #TSSRBIT,R1          :ASCII EQUIVALENT OF BITS
744     10$:    TST      R3             :REMAINING BITS TO CONVERT
745     BEQ      15$                    :BRANCH WHEN ALL ARE DONE
746     CLC
747     ROL      R3                     :CLEAR CARRY FOR SHIFT
748     BCC      13$                    :SHIFT NEXT BIT TO CARRY
749     MOV      (R1),R0                :BRANCH IF BIT NOT SET
750     11$:    MOV      (R0)+,(R2)+    :POINTER TO BIT DEFINITION
751     BNE      11$                    :MOVE ASCII TO BUFFER
752     MOV      #'-1(R2)              :MOVE ALL BITS
753     13$:    TST      (R1)+          :INSERT A COMMA TO TERMINATE
754     BR       10$                   :POINT TO NEXT DESCRIPTION
755     15$:    CLRB      -(R2)         :GET THE REMAINING BITS
756     PRINTX   #TSSDEF,#TMPBFR      :TERMINATE THE LINE
757     MOV      #TMPBFR,-(SP)        :PRINT THE BIT DEFINITIONS
758     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:  .ASCIIZ  '%N% *****REPLACE M7196*****'
773
783 006415      045      116      045  TSSRFOR:  .ASCIIZ  '%N% TSSR = %06'
784 006435      045      116      045  TEXASC:   .ASCIIZ  '%N% Extended Address Bits = %06'
785 006476      045      116      045  TCOASC:   .ASCIIZ  '%N% Termination Class Code = %T'
786 006537      045      116      045  TFCASC:   .ASCIIZ  '%N% Fatal Termination Class Code = %T'
787 006606      045      116      045  TSSDEF:   .ASCIIZ  '%N% TSSR Bits Set: %T'
788 006635      045      116      045  AMBTSSR:  .ASCIIZ  '%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\$:	.ASCIIZ	'Normal Termination'
792	006741	124	145	162	2\$:	.ASCIIZ	'Termination Condition'
793	006767	124	141	160	3\$:	.ASCIIZ	'Tape Status Alert'
794	007011	106	165	156	4\$:	.ASCIIZ	'Function Reject'
795	007031	122	145	143	5\$:	.ASCIIZ	'Recoverable Error - Tape Position One Record Down'
796	007113	122	145	143	6\$:	.ASCIIZ	'Recoverable Error - Tape Was Not Moved'
797	007162	125	156	162	7\$:	.ASCIIZ	'Unrecoverable Error'
798	007206	106	141	164	8\$:	.ASCIIZ	'Fatal Controller Error'
799						.EVEN	
800							
801	007236	007246	007302	007313	TSFCOD:	.WORD	1\$,2\$,3\$,4\$
802	007246	111	156	164	1\$:	.ASCIIZ	'Internal Diagnostic Failure'
803	007302	122	145	163	2\$:	.ASCIIZ	'Reserved'
804	007313	102	165	163	3\$:	.ASCIIZ	'Bus Interface or Sanity Check Error'
805	007357	122	145	163	4\$:	.ASCIIZ	'Reserved'
806						.EVEN	

```

808 .SBTTL PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET
809
810
811 :+ THIS ROUTINE PRINTS THE ADDRESS AND CONTENTS OF A COMMAND PACKET.
812 : THIS ROUTINE IS NORMALLY ONLY CALLED FROM A PRINT ROUTINE.
813
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
      ADD #10,SP
834 15$: MOV R3,R0 ;GET HIGH ORDER ADDRESS
835 BEQ 20$ ;BR IF NOT ABOVE 28K.
836 MOV R4,R1 ;GET LOW ORDER ADDRESS
837 JSR PC,SETMAP ;SETUP PAR6 MAPPING FOR 18 BIT ADDRESS
838 MOV R0,R4 ;GET RETURNED PAR6 ADDRESS BIAS
839 20$: CLR R1 ;SAVE WORD NUMBER
840 25$: MOV (R4)+,R2 ;GET PACKET CONTENTS
841 PRINTB #PKTFRM,R1,R2 ;PRINT THE DATA
      MOV R2,-(SP)
      MOV R1,-(SP)
      MOV #PKTFRM,-(SP)
      MOV #3,-(SP)
      MOV SP,R0
      TRAP C$PNTB
      ADD #10,SP
842 INC R1 ;NEXT WORD NUMBER
843 CMP R1,R5 ;DONE ALL PACKET WORDS?
844 BLT 25$ ;LOOP TILL ALL DONE
845 RTS PC ;RETURN
846
847 007516 045 116 045 PKTFRM: .ASCII 'XN% Packet Word #XD1% = %06'
848 007554 045 116 045 PKTADD: .ASCII 'XN% Packet Address = %01%05'
849 .EVEN
850

```

```

852                                     .SBTTL PRIBXOR - PRINT EXPD, RECV AND XOR BYTE
853
854                                     :+
855                                     :PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE DATA BYTE
856                                     :THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
857
858                                     :INPUTS:
859                                     :
860                                     :      R1      RECEIVED DATA
861                                     :      R2      EXPECTED DATA
862
863                                     :OUTPUT:
864                                     :
865                                     :      R0      XOR OF EXPECTED/RECEIVED DATA
866                                     :
867                                     :-
868
869
870 PRIBXOR::
871     SAVREG                                ;SAVE THE REGISTERS
872     MOV      R2,R3                        ;EXPECTED DATA
873     XOR      R1,R3                        ;FORM THE EXCLUSIVE OR
874     MOV      #^C<377>,R0                 ;BYTE MASK
875     BIC      R0,R1                        ;SAVE LOW BYTE RECV
876     BIC      R0,R2                        ;SAVE LOW BYTE EXPD
877     BIC      R0,R3                        ;SAVE LOW BYTE XOR
878     PRINTB  #XORBFOR,R2,R1,R3 ;PRINT THE MESSAGE
879     MOV      R3,-(SP)
880     MOV      R1,-(SP)
881     MOV      R2,-(SP)
882     MOV      #XORBFOR,-(SP)
883     MOV      #4,-(SP)
884     MOV      SP,R0
885     TRAP     C$PNTB
886     ADD      #12,SP
887     MOV      R3,R0                        ;R0 HAS XOR ON RETURN
888     RTS      PC                          ;RETURN TO CALLER
889
890     .ASCIZ  '%X%A EXPD: %03%A RECV: %03%A XOR: %03%'
891     .EVEN
  
```

```

886                                     .SBTTL PRIOR - PRINT EXPD, RECV AND XOR
887
888                                     :+
889                                     :PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE TWO
890                                     :THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
891                                     :
892                                     :INPUTS:
893                                     :
894                                     :      R1      RECEIVED DATA
895                                     :      R2      EXPECTED DATA
896                                     :
897                                     :OUTPUT:
898                                     :
899                                     :      R0      XOR OF EXPECTED/RECEIVED DATA
900                                     :
901                                     :-
902
903
904 007742                               PRIOR::
905 007742                               SAVREG                               :SAVE THE REGISTERS
906 007746 010203                       MOV      R2,R3                       :EXPECTED DATA
907 007750                               XOR      R1,R3                       :FORM THE EXCLUSIVE OR
908 007760                               PRINTB   #XORFOR,R2,R1,R3 :PRINT THE MESSAGE
909                               007760 010346   MOV      R3,-(SP)
910                               007762 010146   MOV      R1,-(SP)
911                               007764 010246   MOV      R2,-(SP)
912                               007766 012746 010012  MOV      #XORFOR,-(SP)
913                               007772 012746 000004  MOV      #4,-(SP)
914                               007776 010600   MOV      SP,R0
915                               010000 104414   TRAP     C$PNTB
916                               010002 062706 000012  ADD      #12,SP
917                               010006 010300   MOV      R3,R0                :R0 HAS XOR ON RETURN
918                               010010 000207   RTS      PC                  :RETURN TO CALLER
919
920 010012 045 116 045 XORFOR: .ASCIZ '%N% EXPD: %06% RECV: %06% XOR: %06%'
921                               .EVEN
922
923

```

```

915 .SBTTL PRIEQU - PRINT BIT NUMBERS AS ASCII EQUIVALENT
916
917
918
919
920
921
922
923
924
925
926
927
928
929 010060
930 010060
931 010064 000207
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947 010066
948 010066
949 010072 010446 010116
    010074 012746 000002
    010100 012746 000002
    010104 010600
    010106 104414
    010110 062706 000006
950 010114 000207
951
952 010116 045 116 045 RAMFOR: .ASCII 'XN%A CONTROLLER RAM ADDRESS = %06'
953 .EVEN
954
955
956
957
958
959
960
961
962
963
964
965
  
```

```

      ROUTINE TO CONVERT BIT VALUES TO ASCII AND PRINT THE STRING
      THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
      INPUTS:
      R0      OCTAL VALUE TO CONVERT
      R1      TABLE OF POINTERS TO ASCII EQUIVALENT
      PRIEQU:
      SAVREG      ;SAVE THE REGISTERS
      RTS PC      ;RETURN TO CALLER

      .SBTTL PRIRAM - PRINT RAM ADDRESS
      PRINT CONTROLLER RAM ADDRESS.
      THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
      INPUTS:
      R4      RAM ADDRESS
      PRIRAM:
      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
      PRINTB #RAMFOR,R4 ;PRINT RAM ADDRESS IN ERROR
      MOV R4,-(SP)
      MOV #RAMFOR,-(SP)
      MOV #2,-(SP)
      MOV SP,R0
      TRAP C$PNTB
      ADD #6,SP
      RTS PC ;RETURN

      .SBTTL PRIADD - PRINT MEMORY ERROR ADDRESS
      PRINT MEMORY ADDRESS
      THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
      IMPLICIT INPUTS
      ERRHI - HIGH ORDER ADDRESS
      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 'XNZA MEMORY ERROR ADDRESS = %01X05'
979 .EVEN
980
981
982 .SBTTL PRITADD - PRINT MEMORY TEST ADDRESS
983
984
985
986
987
988
989
990
991
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
    MOV ERRHI,R0 ;SAVE R1-R5 UNTIL NEXT RETURN
    MOV ERRLO,R1 ;GET HIGH ADDRESS
    MOV R1,R2 ;GET LOW ADDRESS
    ROL R1 ;COPY LOW ADDRESS
    ROL R0 ;SHIFT BIT 15 TO C BIT
    PRINTB #PRIA0,R0,R2 ;SHIFT INTO HIGH ORDER
    MOV R2,-(SP) ;PRINT MEMORY ADDRESS IN ERROR
    MOV R0,-(SP)
    MOV #PRIA0,-(SP)
    MOV #3,-(SP)
    MOV SP,R0
    TRAP C$PNTB
    ADD #10,SP
    RTS PC ;RETURN

.PRIA0: .ASCIZ 'XNZA 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
    MOV ERRHI,R2 ;SAVE R1-R5 UNTIL NEXT RETURN
    MOV ERRLO,R1 ;GET HIGH ADDRESS
    MOV R1,R2 ;GET LOW ADDRESS
    ROL R1 ;COPY LOW ADDRESS
    ROL R0 ;SHIFT BIT 15 TO C BIT
    PRINTB #PRIT0,R1 ;SHIFT INTO HIGH ORDER
    MOV R1,-(SP) ;PRINT MEMORY ADDRESS LOW IN ERROR
    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				SPACE::	SAVREG			
1048	010466					MOV	#500,,SDELAY		:SAVE THE GENERAL REGISTERS
1049	010472	012737	000764	010660		MOV	#140010,80\$		:SET UP DELAY
1050	010500	012737	140010	010650		TST	R3		:SET UP COMMAND, SPACE FORWARD
1051	010506	005703				BMI	5\$		:CHECK FOR DIRECTION
1052	010510	100403				MOV	R3,90\$		:BR, IF REVERSE INDICATED
1053	010512	010337	010652			BR	10\$		:LOAD UP NUMBER OF RECORDS TO SPACE
1054	010516	000407				BIC	#BIT15,R3		:GO DO COMMAND
1055	010520	042703	100000		5\$:	MOV	R3,90\$		:CLEAR DIRECTION BIT
1056	010524	010337	010652			BIS	#BIT8,80\$		:LOAD UP NUMBER OF RECORDS TO SPACE
1057	010530	052737	000400	010650		MOV	#80\$,R4		:SET REVERSE BIT IN COMMAND PACKET
1058	010536	012704	010650		10\$:	MOV	R4,TSDB(R5)		:SET UP R4 WITH PACKET ADDRESS
1059	010542	010465	000000			JSR	PC,WAITF		:SEND OUT COMMAND
1060	010546	004737	016250		15\$:	BCS	20\$		:WAIT FOR SSR
1061	010552	103420				DELAY	250		:BR, IF SSR IS SET AND OK
1062	010554					MOV	#250,(PC)+		:DELAY ABOUT .25 SECONDS
	010554	012727	000250			.WORD	0		
	010560	000000				MOV	LSPLY,(PC)+		
	010562	013727	002116			.WORD	0		
	010566	000000				DEC	-6(PC)		
	010570	005367	177772			BNE	.-4		
	010574	001375							

	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 :
1100 :ROUTINE TO ISSUE A WRITE CHARACTERISTICS
1101 :COMMAND SO THAT OTHER COMMANDS WILL BE ACCEPTED
1102 :
1103 :INPUT:
1104 :
1105 :       R4      ADDRESS OF PACKET FROM TEST
1106 :       R5      FIRST DEVICE UNIBUS ADDRESS
1107 :       REQUIRES A CALL TO SOFINIT BE DONE PREVIOUSLY
1108 :
1109 :OUTPUT:
1110 :
1111 :       R0      TSSR CONTENTS
1112 :       CARRY   SET - WRITE CHARACTERISTICS COMMAND OK
1113 :              CLR - WRITE CHARACTERISTICS FAILED
1114 :
1115 :IMPLICIT OUTPUT:
1116 :
1117 :       MESSAGE BUFFER AND OTHER BUFFERS ALL SET UP
1118 :       SOFTWARE SWITCHES SET AS FOLLOWS:
1119 :       EXTFEA = EXTENDED FEATURES PRESENT
1120 :       BENBSW = BUFFER ENABLE SWITCH ON OR OFF
1121 :
1122 :
1123 :SIDE EFFECTS:
1124 :
1125 :
1126 :-
1127 :

```

```

1128 010662 WRTCHR::
1129 010662 SAVREG
1130 010666 005037 002230 CLR BENBSW ;SAVE THE GENERAL REGISTERS
1131 010672 005037 002226 CLR EXTFEA ;CLEAR BUFFER ENABLE SWITCH
1132 010676 010465 000000 10$: MOV R4,TSDB(R5) ;CLEAR EXTENDED FEATURES SW SWITCH
1133 010702 004737 016336 JSR PC,CHKTSSR ;SEND OUT COMMAND
1134 010706 103401 BCS 20$ ;WAIT FOR SSR
1135 010710 000435 BR 60$ ;BR, IF SSR IS SET AND OK
1136 010712 016501 000002 20$: MOV TSSR(R5),R1 ;BR IF TROUBLE CARRY = CLEAR
1137 010716 012702 000200 MOV #SSR,R2 ;READ TSSR
1138 010722 032701 000100 BIT #OFL,R1 ;SET UP EXPECTED
1139 010726 001402 BEQ 25$ ;WAS OFF LINE SET IN TSSR
1140 010730 052702 000100 BIS #OFL,R2 ;BR, IF NO OFL SET
1141 010734 020201 25$: CMP R2,R1 ;MAKE THEM LOOK ALIKE
1142 010736 001401 BEQ 40$ ;ARE THEY OK
1143 010740 000421 BR 60$ ;BR, IF EQUAL = OK
1144 010742 062704 000010 40$: ADD #8,R4 ;TROUBLE EXIT
1145 010746 011403 MOV (R4),R3 ;POINT TO WRT CHARA DATA PACKET
1146 010750 032763 000200 000012 BIT #X2.EXTF,XST2(R3) ;GET ADDRESS OF MESSAGE BUFFER
1147 010756 001402 BEQ 45$ ;EXTENDED FEATURES BIT SET?
1148 010760 005237 002226 INC EXTFEA ;BR IF NO
1149 010764 45$: ;SET EXTENDED FEATURES SW SWITCH
1150 010764 032763 000100 000012 BIT #X2.BUFE,XST2(R3) ;BUFFER ENABLE SWITCH SET
1151 010772 001402 BEQ 50$ ;BR, IF SWITCH NOT SET
1152 010774 005237 002230 INC BENBSW ;SET SOFTWARE SWITCH FOR ENABLED

```

SEQ 0054

1153	011000		50\$:			
1154	011000	000261		SEC		;SET CARRY NO TROUBLE
1155	011002	000401		BR	70\$	;EXIT
1156	011004	000241		CLC		;CARRY CLEAR = ERROR
1157	011006	016500	000002	60\$:		;RETURN TSSR CONTENTS
1158	011012	000207		70\$:	TSSR(R5),R0	;RETURN
1159				MOV		
1160				RTS	PC	

```

1162 .SBTTL REWIND - POSITION TAPE (REWIND) COMMAND
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 011014
1191 011014
1192 011020 012704 011110
1193 011024 010465 000000
1194 011030 012703 000550
1195 011034 004737 016250
1196 011040 103417
1197 011042
    011042 012727 000372
    011046 000000
    011050 013727 002116
    011054 000000
    011056 005367 177772
    011062 001375
    011064 005367 177756
    011070 001367
1198 011072 005303
1199 011074 001357
1200 011076 000241
1201 011100 010400
1202 011102 000207
1203
1204
1206 011110
1208 011110
1209 011110 102010
1210 011112 000000
1211
1212

```

```

      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
      MOV #RWPACK,R4
      MOV R4,TSDB(R5)
      MOV #360.,R3
      JSR PC,WAITF
      BCS 20$
      DELAY 250.
      MOV #250.,(PC)+
      .WORD 0
      MOV LSDLY,(PC)+
      .WORD 0
      DEC -6(PC)
      BNE -.4
      DEC -22(PC)
      BNE -.20
      DEC R3
      BNE 10$
      CLC
      20$: MOV R4,R0
           RTS PC
      RWPACK:
      .=<.+10>&177770
      .WORD 102010
      .WORD 0

```

```

      :SAVE R1-R5 UNTIL NEXT RETURN
      :GET PACKET ADDRESS
      :SEND PACKET ADDRESS TO EXECUTE
      :ENOUGH TIME FOR 2400 REEL TO REWIND
      :WAIT FOR SSR TO SET
      :LEAVE WHEN SSR IS SET
      :WAIT FOR .25 SECONDS
      :BUMP COUNTER DOWN
      :KEEP GOING
      :CLEAR CARRY TO SET ERROR
      :PASS THE PACKET ADDRESS
      :RETURN
      :POSTION COMMAND (REWIND)
      :NOT USED

```

TSV3 - GLOBAL AREAS MACRO M1113 25-MAY-82 09:19 PAGE 30-1 E 5
REWIND - POSITION TAPE (REWIND) COMMAND

SEQ 0056

1213
1214
1215

1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246 011114
1247 011114
1248 011120 012701 002242
1249 011124 012702 000201
1250 011130 005003
1251 011132 004737 016336
1252 011136 112765 000000 000000
1253 011144 004737 016336
1254 011150 010265 000000
1255 011154 004737 016336
1256 011160 116511 000000
1257 011164 122124
1258 011166 001401
1259 011170 005203
1260 011172 005202
1261 011174 020227 000210
1262 011200 003761
1263 011202 005703
1264 011204 001402
1265 011206 000241
1266 011210 000401
1267 011212 000261
1268 011214 012737 000010 002302
1269 011222 000207
1270

.SBTTL CKRAM - COMPARE RAM TO I/O PACKET

```

:ROUTINE TO READ THE FIRST 8 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 IS SET TO 8. FOR PRAMPKT ROUTINE

:SIDE EFFECTS:
      THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE

```

CKRAM::

```

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,#RMPKTEND     :ADDRESS OF NEXT RAM LOCATION
BLE 10$              :REACHED END YET ?
TST R3               :BRANCH TILL ALL READ
BEQ 30$              :WAS AN ERROR FOUND ?
CLC                  :BRANCH IF NOT
BR 50$               :CLEAR CARRY TO SHOW ERROR
SEC                  :AND EXIT
MOV #8.,RAMSIZ       :SHOW GOOD COMPARE
RTS PC               :SETUP RAMSIZ FOR PRAMPKT ROUTINE

```

```

1272 .SBTTL CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA
1273
1274
1275 :ROUTINE TO READ THE FIRST 8 OR 10 BYTES FROM RAM
1276 :MEMORY AND COMPARE THIS DATA TO A CHARACTERISTICS DATA BLOCK.
1277
1278 :INPUT:
1279
1280 R4 ADDRESS OF THE CHARACTERISTICS DATA
1281 R5 FIRST DEVICE UNIBUS ADDRESS
1282
1283 :OUTPUT:
1284
1285 CARRY SET - RAM MATCHES PACKET
1286 CLR - RAM DOES NOT MATCH PACKET
1287
1288 :IMPLICIT OUTPUT:
1289
1290 THE TABLE RAMDATA IS FILLED WITH THE
1291 DATA HELD IN RAM.
1292 RAMSIZ IS SET TO 8. OR 10. FOR PRAMPKT ROUTINE
1293
1294 :SIDE EFFECTS:
1295
1296 THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
1297
1298 :-
1299
1300 CKRAM2::
1301 SAVREG
1302 :SAVE THE GENERAL REGISTERS
1303 :ADDRESS TO SAVE THE RAM DATA
1304 :BYTE ADDRESS OF FIRST RAM DATA
1305 :CLEAR THE ERROR FLAG
1306 :WAIT FOR SSR
1307 :SET MAINTENANCE MODE
1308 :WAIT FOR SSR TO SET
1309 :SELECT NEXT RAM ADDRESS
1310 :WAIT FOR SSR TO SET
1311 :READ THE RAM DATA
1312 :COMPARE TO EXPECTED
1313 :BRANCH IF OK
1314 :SET ERROR FLAG
1315 :ADDRESS OF NEXT RAM LOCATION
1316 :ASSUME EXTFEA NOT SET
1317 :IS THE SOFTWARE EXTENDED FEATURES SET
1318 :BR, IF NOT SET
1319 :SET RAMSIZ FOR EXTEND FEATURES
1320 :AT END OF EXTENDED BUFFER
1321 :BR, IF NOT AT END YET
1322 :AT END BRANCH
1323 :REACHED END YET ?
1324 :BRANCH TILL ALL READ
1325 :WAS AN ERROR FOUND ?
1326 :BRANCH IF NOT
1327 :CLEAR CARRY TO SHOW ERROR
1328 :AND EXIT
1329 :SHOW GOOD COMPARE

```

TSV3 - GLOBAL AREAS MACRO M1113 25-MAY-82 09:19 PAGE 32-1 H 5
CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA

SEQ 0059

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 RECMSG 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 CKMSG::
1358 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1359 MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
1360 MOV R1,RCVLOADD ;SAVE RECV LOW ADDRESS
1361 TST KTENABLE ;TESTING ABOVE 28K?
1362 BEQ 10$ ;BR IF NO
1363 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
1364 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
1365 10$: CLR R4 ;WORD IN BUFFER
1366 CLR R3 ;CLEAR ERROR SEEN FLAG
1367 MOV R2,R5 ;GET EXPD BUFFER ADDRESS
1368 15$: MOV (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
1369 MOV (R1),RECMSG(R4) ;SAVE RECV FOR ERROR REPORT
1370 CMP (R2)+,(R1)+ ;EXPD EQUAL RECV?
1371 BEQ 25$ ;BR IF YES
1372 INC R3 ;SET ERROR SEEN FLAG
1373 25$: ADD #2,R4 ;POINT TO NEXT WORD ADDRESS
1374 CMP R4,#14 ;DONE FIRST 7 WORDS?
1375 BLE 15$ ;BR IF NO
1376 BIT #X2.EXTF,XST2(R5) ;IS EXTENDED FEATURES SET IN EXPD?
1377 BEQ 50$ ;BR IF NO
1378 CMP R4,#16 ;DONE EXTENDED FEATURES WORD?
1379 BLE 15$ ;BR IF NO
1380 50$: TST R3 ;ANY ERRORS SEEN?
1381 BEQ 55$ ;BR IF NO
1382 CLC ;SET FAILURE
1383 BR 60$
1384 55$: SEC ;SET SUCCESS
1385 60$: RTS PC ;RETURN
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 :      RECMSG   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 CKMSG2::
1416 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1417 CMP R3,#RECMSG-EXPMSG;@@D IS COUNT ABOVE MAX ALLOWED?
1418 BLE 5$ ;@@D BR IF NO
1419 MOV #REC SG-EXPMSG,R3;@@D
1420 PRINTF #DEBUGMSG ;@@D
1421 MOV #DEBUGMSG,-(SP)
1422 MOV #1,-(SP)
1423 MOV SP,R0
1424 TRAP C$PNTF
1425 ADD #4,SP
1426 5$: MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
1427 MOV R1,RCVLOAD ;SAVE RECV LOW ADDRESS
1428 TST KTENABLE ;TESTING ABOVE 28K?
1429 BEQ 10$ ;BR IF NO
1430 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
1431 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
1432 10$: CLR R4 ;WORD IN BUFFER
1433 CLR R5 ;CLEAR ERROR SEEN FLAG
1434 15$: MOVB (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
1435 MOVB (R1),RECMSG(R4) ;SAVE RECV FOR ERROR REPORT
1436 CMPB (R2)+,(R1)+ ;EXPD EQUAL RECV?
1437 BEQ 25$ ;BR IF YES
1438 INC R5 ;SET ERROR SEEN FLAG
1439 25$: ADD #1,R4 ;POINT TO NEXT BYTE
1440 CMP R4,R3 ;DONE ALL BYTES?
1441 BGE 50$ ;BR IF YES
1442 BR 15$ ;DO NEXT BYTE
1443 50$: TST R5 ;ANY ERRORS SEEN?
1444 BEQ 55$ ;BR IF NO

```

1440	011622	000241			CLC		;SET FAILURE
1441	011624	000401			BR	60\$	;
1442	011626	000261			SEC		;SET SUCCESS
1443	011630	000207			RTS	PC	;RETURN
1444							
1445	011632	120	122	117	DEBUGMSG:	.ASCIZ	'PROGRAM INTERNAL ERROR -CKMSG2 MESSAGE BUFFER EXCEEDED-';@ad
1446	011722	045	116	045	FERCM:	.ASCII	/ %NZA *** /
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 012034
      012034
1469 012034 004737 006024
1470 012040 004737 017202
1471 012044
      012044
      012044 104423
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484 012046
      012046
1485 012046 004737 006024
1486 012052 012700 000004
1487 012056 004737 007370
1488 012062
      012062
      012062 104423
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501 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      L10005:      ENDMMSG
      012100      TRAP     C$MSG
      012100 104423

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

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

1541

```

```

1543                                     .SBTTL  ADDSSR - PRINT TEST ADDRESS AND TSSR
1544                                     :+
1545                                     :PRINT ROUTINE TO PRINT THE CONTENTS OF
1546                                     :TSSR AND A MEMORY TEST ADDRESS
1547                                     :
1548                                     :INPUTS:
1549                                     :
1550                                     R5      FIRST DEVICE UNIBUS ADDRESS
1551                                     ERRHI   HIGH ORDER MEMORY TEST ADDRESS
1552                                     ERRLO   LOW ORDER MEMORY TEST ADDRESS
1553                                     :-
1554
1555 012126      BGNMSG  ADDSSR
1556 012126      ADDSSR::
1557 012126 004737 010274      JSR      PC,PRITADD      ;PRINT MEMORY TEST ADDRESS
1558 012132 016501 000002      MOV      TSSR(R5),R1    ;GET CURRENT TSSR
1559 012136 004737 006024      JSR      PC,PRITSSR     ;PRINT THE CONTENTS OF TSSR REGISTER
1560 012142      ENDMSG
1561 012142      L10010:
1562 012142 104423      TRAP      C$MSG
1563
1564                                     .SBTTL  MSGEXP - PRINT WRITE CHAR. EXPD-RECV MESSAGE BUFFERS
1565                                     :+
1566                                     :PRINT ROUTINE TO PRINT WRITE CHARACTERISTIC MESSAGE BUFFER
1567                                     :
1568                                     :IMPLICIT INPUTS:
1569                                     :
1570                                     EXPMSG  - EXPECTED MESSAGE BUFFER
1571                                     RECMSG  - RECEIVED MESSAGE BUFFER
1572                                     RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1573                                     RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1574                                     :-
1575 012144      BGNMSG  MSGEXP
1576 012144      MSGEXP::
1577 012144 012700 000007      MOV      #7,R0      ;ASSUME NO EXT FEATURES
1578 012150 005737 002226      TST      EXTFEA     ;EXT FEATURES SET?
1579 012154 001402      BEQ      $$              ;BR IF NO
1580 012156 012700 000010      MOV      #8.,R0     ;EXT FEATURE BUFFER IS 8 WORDS
1581 012162 004737 014552      JSR      PC,PRMSGEXP   ;PRINT EXPD/RECV MESSAGE BUFFERS
1582 012166      ENDMSG
1583 012166      L10011:
1584 012166      TRAP      C$MSG

```

1584			
1585			
1586			
1587			
1588			
1589			
1590			
1591			
1592			
1593			
1594			
1595			
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
1603	012311	045	116
1604			
1605			

```

.SBTTL FIFEXP - PRINT FIFO EXP/RCV DATA
;+
;PRINT ROUTINE TO PRINT FIFO EXP/RCV DATA
;
; R1 - BYTE COUNT
;
;IMPLICIT INPUTS:
;
; EXPMSG - EXPECTED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
; RECMSG - RECEIVED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
;-
;BGNMSG FIFEXP
FIFEXP:
PRINTX #FIF1MSG,R1 ;PRINT BYTES TRANSFERRED
MOV R1,-(SP)
MOV #FIF1MSG,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #6,SP
PRINTX #FIF2MSG ;PRINT HEADER MSG
MOV #FIF2MSG,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #4,SP
MOV R1,R0 ;GET BYTE COUNT
JSR PC,PRBYTEXP ;PRINT FIFO BYTES IN ERROR
ENDMSG
L10012:
TRAP C$MSG
FIF1MSG: .ASCII 'X% NUMBER OF BYTES TRANSFERRED = %d'
FIF2MSG: .ASCII 'X% FIFO DATA BYTES IN ERROR:'
;EVEN

```

```

.SBTTL MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS
;+
;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
;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 MSGSTAT
MSGSTAT::
MOV #STATCOD,R1 ;ASCII ADDRESS TABLE
10$: MOV (R1)+,R0 ;DONE ALL MSG LINES?
BEQ 20$ ;BR IF YES
PRINTX R0 ;PRINT STATUS BIT NAMES
MOV R0,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #4,SP
BR 10$ ;DO ANOTHER MSG LINE
20$: MOV #10.,R0 ;NUMBER OF WORDS IN A READ STATUS BUFFER
JSR PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
ENDMSG
L10013: TRAP C$MSG
STATCOD: .WORD 1$,2$,3$,4$,5$,6$,0
1$: .ASCIIZ '%X%A Tape Bus Signals in Word #8:'
2$: .ASCIIZ '%X%A PARERR<15> IEOT <12> IFMK <9> IRDY<6> IRWD<2>'
3$: .ASCIIZ '%X%A IRESV2<14> IIDENT<11> IHER <8> IONL<5> IFBY<1>'
4$: .ASCIIZ '%X%A IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>'
5$: .ASCIIZ '%X%A Tape Bus Signals in Word #9:'
6$: .ASCIIZ '%X%A DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>'
.EVEN
.SBTTL MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS
;+
;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
;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 MSGLOOP
MSGLOOP::
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$:  MOV      #10,R0      ;NUMBER OF WORDS IN A READ STATUS BUFFER
1660 013120 004737 014552  JSR      PC,PRMSGEXP    ;PRINT EXPD/RECV MESSAGE BUFFERS
1661 013124      ENDMSG
      013124      L10014: TRAP     C$MSG
1662 013124 104423
1663 013126 013146 013221 013320 LOOPCOD: .WORD 1$,2$,3$,4$,5$,6$,7$,0
1664 013146      045      116      045 1$: .ASCIIZ '%NZA Tape Bus Loopback Signals in Word #8:'
1665 013221      045      116      045 2$: .ASCIIZ '%NZA PARERR<15> IRESV2<14> IRESV1<13>'
1666 013320      045      116      045 3$: .ASCIIZ '%NZA IHISP=>IEOT<12> IWRT=>IIDENT<11> IREV =>ICER <10>'
1667 013417      045      116      045 4$: .ASCIIZ '%NZA IWFM =>IFMK<09> IEDIT=>IHER <08> IFAD =>ISPEED<07>'
1668 013516      045      116      045 5$: .ASCIIZ '%NZA ITADO=>IRDY<06> ITAD1=>IONL <05> IERASE=>ILDPA <04>'
1669 013615      045      116      045 6$: .ASCIIZ '%NZA IREW =>IDBY<03> IRWU =>IRWD <02> IFEN =>IFBY <01>'
1670 013714      045      116      045 7$: .ASCIIZ '%NZA IGO =>IFPT<00>'
1671      .EVEN
1672

```

```

1674 .SBTTL MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER
1675 :+
1676 :PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
1677 :
1678 :IMPLICIT INPUTS:
1679 :
1680 :EXPMSG - EXPECTED MESSAGE BUFFER
1681 :RCMSG - RECEIVED MESSAGE BUFFER
1682 :RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1683 :RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1684 :
1685 :
1686 :
1687 013742 BGNMSG MSGSUB
      013742 MSGSUB::
1688 013742 012700 000012 MOV #10.,R0 ;SIZE OF WRITE SUBSYSTEM BUFFER
1689 013746 004737 014552 JSR PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
1690 013752 ENDMSG
      013752
      013752 L10015:
      013752 104423 TRAP C$MSG

1691
1692
1693
1694
1695
1696 .SBTTL MEMADD - PRINT MEMORY ADDRESS DATA ERROR
1697 :+
1698 :PRINT ROUTINE TO PRINT MEMORY ADDRESS DATA COMPARE ERROR
1699 :
1700 :IMPLICIT INPUTS:
1701 :
1702 :ERRHI - MEMORY ERROR HIGH ORDER ADDRESS
1703 :ERRLO - MEMORY ERROR LOW ORDER ADDRESS
1704 :EXP - EXPECTED DATA
1705 :RCV - RECEIVED DATA
1706 :
1707 :
1708 013754 BGNMSG MEMADD
      013754 MEMADD::
1709 013754 004737 010160 JSR PC,PR1ADD ;PRINT MEMORY ADDRESS IN ERROR
1710 013760 013701 002232 MOV EXPD,R1 ;GET EXPD DATA
1711 013764 013702 002234 MOV RECV,R2 ;GET RECEIVED DATA
1712 013770 004737 007742 JSR PC,PR1XOR ;PRINT EXPD/RCV
1713 013774 ENDMSG
      013774
      013774 L10016:
      013774 104423 TRAP C$MSG

1714

```

```
1716 .SBTTL PRAMPKT - PRINT RAM AND PACKET DATA
1717
1718
1719 :PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
1720 :WHEN THE RAM DATA DOES NOT MATCH.
1721
1722 :INPUTS:
1723
1724 :R4 POINTER TO COMMAND PACKET
1725
1726 :IMPLICIT INPUTS:
1727
1728 :RAMDATA DATA AS READ FROM THE RAM
1729 :RAMSIZ NUMBER OF BYTES IN PACKET
1730 :IF RAMSIZ=0 THEN DEFAULT TO 8.
1731
1732 :IMPLICIT OUTPUTS:
1733
1734 :RAMSIZ SET TO 0
1735
1736
1737 PRAMPKT:
1738 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1739 MOV #RAMDATA,R1 ;DATA FROM THE RAM
1740 CLR R2 ;INIT BYTE NUMBER
1741 5$: CMPB (R1)+,(R4)+ ;COMPARE EXPECTED, RECEIVED
1742 BNE 7$ ;BR IF NO MATCH
1743 FORCERROR 7$,NOTSSR
1744 BR 10$
1745 7$: MOVB -1(R1),R5 ;GET RCV RAM DATA
1746 MOVB -1(R4),R3 ;GET EXPD PACKET DATA
1747 XOR R5,R3 ;XOR EXPD/RCV
1748 BIC #177400,R3 ;LOW BYTE ONLY
1749 MOVB -1(R1),RCV ;GET RECEIVED RAM DATA
1750 MOVB -1(R4),EXPD ;GET EXPECTED RAM DATA
1751 PRINTB #RAMASC,R2,RCV,EXPD,R3
1752 MOV R3,-(SP)
1753 MOV EXPD,-(SP)
1754 MOV RCV,-(SP)
1755 MOV R2,-(SP)
1756 MOV #RAMASC,-(SP)
1757 MOV #5,-(SP)
1758 MOV SP,R0
1759 TRAP C$PNTB
1760 ADD #14,SP
1761 10$: INC R2 ;UPDATE BYTE COUNT
1762 TST RAMSIZ ;DEFAULT TO 8.?
1763 BEQ 15$ ;BR IF YES
1764 CMP R2,RAMSIZ ;DONE ALL BYTES?
1765 BLE 5$ ;BR IF NO
1766 BR 25$
1767 15$: CMP R2,#8. ;DONE DEFAULT NUMBER OF BYTES?
1768 BLT 5$ ;BR IF NO
1769 20$: BLT 5$
1770 25$: CLR RAMSIZ ;SET DEFAULT RAMSIZ
1771 RTS PC ;RETURN
1772
1773 013776
1774 013776
1775 014002 012701 002242
1776 014006 005002
1777 014010 122124
1778 014012 001005
1779 014014
1780 014024 000436
1781 014026 116105 177777
1782 014032 116403 177777
1783 014036
1784 014046 042703 177400
1785 014052 116137 177777 002234
1786 014060 116437 177777 002232
1787 014066
1788 014066 010346
1789 014070 013746 002232
1790 014074 013746 002234
1791 014100 010246
1792 014102 012746 014156
1793 014106 012746 000005
1794 014112 010600
1795 014114 104414
1796 014116 062706 000014
1797 014122 005202
1798 014124 005737 002302
1799 014130 001404
1800 014132 020237 002302
1801 014136 003724
1802 014140 000403
1803 014142 020227 000010
1804 014146 002720
1805 014150 005037 002302
1806 014154 000207
1807 014156 045 116 045 RAMASC: .ASCIZ '%N%A BYTE: %D2%A RAM: %O3%A Packet: %O3%A XOR:%O3'
```

TSV3 - GLOBAL AREAS MACRO M1113 25-MAY-82 09:19 PAGE 40-1
PRAMPKT - PRINT RAM AND PACKET DATA

G 6

SEQ 0071

1764
1765
1766

.EVEN

```

1768                                     .SBTTL PRMESS - PRINT CONTENTS OF MESSAGE BUFFER
1769
1770
1771                                     :+
1772                                     :THIS ROUTINE PRINTS THE CONTENTS OF
1773                                     :THE 7 OR 8 WORD MESSAGE BUFFER RETURNED BY THE
1774                                     :TSV-05.
1775
1776                                     :INPUT:
1777                                     R0      LOW ORDER ADDRESS OF MESSAGE BUFFER
1778                                     R1      HIGH ORDER ADDRESS OF MESSAGE BUFFER
1779                                     NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
1780
1781                                     :THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
1782                                     :-
1783
1784
1785 PRMESS:
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805

```

014242	010005	
014250	005737	003134
014254	001001	
014256	005001	
014260	010103	
014262	006100	
014264	006101	
014266	010546	
014270	010146	
014272	012746	014420
014276	012746	000003
014302	010600	
014304	104415	
014306	062706	000010
014312	012746	014465
014316	012746	000001
014322	010600	
014324	104415	
014326	062706	000004
014332	005004	
014334	010501	
014336	010300	
014340	001403	
014342	004737	017316
014346	010005	
014350	012546	
014352	010446	
014354	012746	014523
014360	012746	000003
014364	010600	
014366	104415	
014370	062706	000010
014374	005204	
014376	020427	000007
014402	003005	

```

                                     SAVREG
                                     MOV R0,R5
                                     TST KTENABLE
                                     BNE 10$
                                     CLR R1
                                     MOV R1,R3
                                     ROL R0
                                     ROL R1
                                     PRINTX #PROASC,R1,R5
                                     MOV R5,-(SP)
                                     MOV R1,-(SP)
                                     MOV #PROASC,-(SP)
                                     MOV #3,-(SP)
                                     MOV SP,R0
                                     TRAP C$PNTX
                                     ADD #10,SP
                                     PRINTX #PRIASC
                                     MOV #PRIASC,-(SP)
                                     MOV #1,-(SP)
                                     MOV SP,R0
                                     TRAP C$PNTX
                                     ADD #4,SP
                                     CLR R4
                                     MOV R5,R1
                                     MOV R3,R0
                                     BEQ 20$
                                     JSR PC,SETMAP
                                     MOV R0,R5
                                     PRINTX #PRASC,R4,(R5)+
                                     MOV (R5)+,-(SP)
                                     MOV R4,-(SP)
                                     MOV #PRASC,-(SP)
                                     MOV #3,-(SP)
                                     MOV SP,R0
                                     TRAP C$PNTX
                                     ADD #10,SP
                                     INC R4
                                     CMP R4,#7
                                     BGT 50$

```

```

:SAVE THE REGISTERS
:SAVE LOW ORDER ADDRESS
:ADDRESS ABOVE 28K?
:BR IF YES
:SET HIGH ORDER ADDRESS TO 0
:SAVE HIGH ORDER ADDRESS
:SHIFT BIT15 TO C BIT
:SHIFT TO HIGH ORDER FOR PRINTOUT
:PRINT MESSAGE BUFFER ADDRESS

:PRINT HEADER FOR CONTENTS

:NUMBER OF THE NEXT WORD
:COPY LOW ORDER ADDRESS
:COPY HIGH ORDER ADDRESS
:BR IF NOT ABOVE 28K
:SETUP PAR ADDRESS IN R0
:GET PAR FORMAT ADDRESS ABOVE 28K
:PRINT THE CONTENTS OF MEMORY BUFFER

:NUMBER OF THE NEXT
:DONE ALL YET ?
: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	'%N%A	Message Buffer Address = %01%05'
1812	014465	045	116	045	PR1ASC:	.ASCIZ	'%N%A	Message Buffer Contents:'
1813	014523	045	116	045	PRASC:	.ASCIZ	'%N%A	Word%D1%A: %0'
1814						.EVEN		

```

1816 .SBTTL PRMSGEXP - PRINT EXPD/RECV MESSAGE BUFFERS
1817
1818
1819 :+
1820 :ROUTINE TO PRINT EXPECTED AND RECEIVED MESSAGE BUFFERS
1821
1822 :RO - NUMBER OF WORDS IN BUFFER
1823
1824 :IMPLICIT INPUTS:
1825
1826 :EXPMSG - EXPECTED MESSAGE BUFFER
1827 :RECMSG - RECEIVED MESSAGE BUFFER
1828 :RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1829 :RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1830
1831 PRMSGEXP::
1832 SAVREG
1833 MOV R0,R5 ;SAVE R1-R5 UNTIL NEXT RETURN
1834 MOV RCVLOADD,R0 ;SAVE NUMBER OF WORDS
1835 MOV R0,R4 ;GET RECV LOW ADDRESS
1836 MOV RCVHIADD,R1 ;COPY LOW ADDRESS
1837 ROL R0 ;GET RECV HIGH ADDRESS
1838 ROL R1 ;SHIFT BIT15 TO C BIT
1839 PRINTX #PRMSGC,R1,R4 ;SHIFT TO HIGH ORDER FOR PRINTOUT
1840 MOV R4,-(SP) ;PRINT MESSAGE BUFFER ADDRESS
1841 MOV R1,-(SP)
1842 MOV #PRMSGC,-(SP)
1843 MOV #3,-(SP)
1844 MOV SP,R0
1845 TRAP C$PNTX
1846 ADD #10,SP
1847 PRINTX #PRMSG1 ;PRINT HEADER FOR CONTENTS
1848 MOV #PRMSG1,-(SP)
1849 MOV #1,-(SP)
1850 MOV SP,R0
1851 TRAP C$PNTX
1852 ADD #4,SP
1853 CLR R4 ;NUMBER OF THE CURRENT WORD
1854 MOV #EXPMSG,R1 ;GET EXPD BUFFER ADDRESS
1855 MOV #RECMSG,R2 ;GET RECV BUFFER ADDRESS
1856 MOV (R1),R0 ;GET EXPD
1857 MOV (R2),R3 ;GET RECV
1858 XOR R0,R3 ;XOR EXPD/RECV
1859 PRINTX #PRMSG2,R4,(R1)+,(R2)+,R3
1860 MOV R3,-(SP)
1861 MOV (R2)+,-(SP)
1862 MOV (R1)+,-(SP)
1863 MOV R4,-(SP)
1864 MOV #PRMSG2,-(SP)
1865 MOV #5,-(SP)
1866 MOV SP,R0
1867 TRAP C$PNTX
1868 ADD #14,SP
1869 INC R4 ;NUMBER OF THE NEXT
1870 CMP R4,R5 ;DONE ALL YET?
1871 BGE 50$ ;BR IF YES
1872 BR 20$ ;DO ANOTHER
1873
1874 50$: RTS PC ;RETURN

```

```
1852
1853 014732      045      116      045 PRMSG0: .ASCIZ 'XNZA Message Buffer Address = %01X05'
1854 014777      045      116      045 PRMSG1: .ASCIZ 'XNZA Message Buffer Contents:'
1855 015035      045      116      045 PRMSG2: .ASCIZ 'XNZA    WORD #XD2XA EXPD: %06XA RECV: %06XA XOR: %06'
1856                                  .EVEN
1857
```

```

1859 .SBTTL PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER
1860
1861
1862 :+
1863 :ROUTINE TO PRINT ERROR BYTES IN MESSAGE BUFFERS
1864 :ONLY THE FIRST 8 ERRORS ENCOUNTERED ARE PRINTED DUE TO SCREEN SPACE
1865
1866 :R0 - NUMBER OF BYTES IN BUFFER
1867
1868 :IMPLICIT INPUTS:
1869
1870 :EXPMSG - EXPECTED MESSAGE BUFFER
1871 :RECMSG - RECEIVED MESSAGE BUFFER
1872
1872 015122 PRBYTEXP::
1873 015122 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1874 015126 010005 MOV R0,R5 ;SAVE NUMBER OF BYTES
1875 015130 005037 002320 CLR PRMNO ;INIT ERROR COUNT
1876 015134 005004 CLR R4 ;NUMBER OF THE CURRENT BYTE
1877 015136 012701 002322 MOV #EXPMSG,R1 ;GET EXPD BUFFER ADDRESS
1878 015142 012702 002466 MOV #RECMSG,R2 ;GET RECV BUFFER ADDRESS
1879 015146 111100 20$: MOV (R1),R0 ;GET EXPD BYTE
1880 015150 042700 177400 BIC #^C<377>,R0 ;CLEAR UPPER BYTE
1881 015154 110037 015470 MOV R0,PRBEXP ;SAVE FOR ERROR REPORT
1882 015160 111203 MOV (R2),R3 ;GET RECV BYTE
1883 015162 042703 177400 BIC #^C<377>,R3 ;CLEAR UPPER BYTE
1884 015166 110337 015472 MOV R3,PRBREC ;FOR ERROR REPORT
1885 015172 XOR R0,R3 ;XOR EXPD/RECV
1886 015202 122122 CMPB (R1)+,(R2)+ ;EXPD = RECV?
1887 015204 001431 BEQ 30$ ;BR IF YES
1888 015206 005237 002320 INC PRMNO ;UPDATE ERROR COUNT
1889 015212 023727 002320 000010 CMP PRMNO,#8. ;PRINTED 8?
1890 015220 101023 30$: BHI 30$ ;BR IF YES
1891 015222 27$: PRINTX #PRBMSG,R4,PRBEXP,PRBREC,R3
1892 015222 010346 MOV R3,-(SP)
1893 015224 013746 015472 MOV PRBREC,-(SP)
1894 015230 013746 015470 MOV PRBEXP,-(SP)
1895 015234 010446 MOV R4,-(SP)
1896 015236 012746 015336 MOV #PRBMSG,-(SP)
1897 015242 012746 000005 MOV #5,-(SP)
1898 015246 010600 MOV SP,R0
1899 015250 104415 TRAP C$PNTX
1900 015252 062706 000014 ADD #14,SP
1901 015256 1892 015256 FORCEEXIT 50$ ;aad
1902 015266 1893 015266 BR 35$ ;aad
1903 015270 1894 015270 30$: FORCERROR 27$,NOTSSR ;aad
1904 015270 1895 015270 35$: ;aad
1905 015300 1896 015300 INC R4 ;NUMBER OF THE NEXT
1906 015300 1897 015300 005204 CMP R4,R5 ;DONE ALL YET?
1907 015302 1898 015302 020405 BGE 50$ ;BR IF YES
1908 015304 1899 015304 002001 BR 20$ ;DO ANOTHER
1909 015306 1900 015306 000717 50$: PRINTX #PRBTOT,PRMNO ;PRINT TOTAL ERROR COUNT
1910 015310 1901 015310 MOV PRMNO,-(SP)
1911 015310 013746 002320 MOV #PRBTOT,-(SP)
1912 015314 012746 015423 MOV #2,-(SP)
1913 015320 012746 000002 MOV SP,R0
1914 015324 010600 TRAP C$PNTX
1915 015326 104415

```

1902	015330	062706	000006		ADD	#6,SP	
1903	015334	000207			RTS	PC	;RETURN
1904	015336	045	116	045	PRBMSG: .ASCIZ	'%N%A	BYTE #D2%A EXPD: %03%A RECV: %03%A XOR: %03'
1905	015423	045	116	045	PRBTOT: .ASCIZ	'%N%A	NUMBER OF BYTES IN ERROR = %D2'
1906					.EVEN		
1907	015470	000000			PRBEXP: .WORD	0	;EXPD
1908	015472	000000			PRBREC: .WORD	0	;RECV
1909							

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,PRIOR          ;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
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
:
:-
:
:      BGNMSG  EXPBREC
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
:
:-
:
:      BGNMSG  RAMERR
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

```

SEQ 0081

```

2031
2032      :-
2033      015552      BGNMSG  TIMEXP
2034      015552      TIMEXP::
2035      015552      PRINTX  #TIMSG0      :PRINT HEADER
2036      015552      MOV      #TIMSG0,-(SP)
2037      015552      MOV      #1,-(SP)
2038      015552      MOV      SP,R0
2039      015552      TRAP     C$PNTX
2040      015552      ADD      #4,SP
2041      015552      JSR      PC,PRIXOR      :PRINT THE DATA
2042      015552      ENDMSG
2043      015552      L10024:
2044      015552      TRAP     C$MSG
2045      015552
2046      015552      012746  015600
2047      015552      012746  000001
2048      015552      010600
2049      015552      104415
2050      015552      062706  000004
2051      015552      004737  007742
2052      015552
2053      015552      104423
2054      015552
2055      015600      045      116      045  TIMSG0: .ASCIZ  '%N% TIMER A STATUS IS IN BIT 3%N% TIMER B STATUS IS IN BIT 2'
2056      015600      .EVEN
    
```

```

2042                                     .SBTTL  BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS
2043
2044                                     :+
2045                                     :PRINT ROUTINE FOR TSSR ERRORS ON DATA TRANSFERS
2046                                     :
2047                                     :INPUTS:
2048                                     :
2049                                     :
2050                                     :      R1      CONTENTS OF TSSR
2051                                     :      R2      DATA WRITTEN (8 BITS)
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$PNTB
2065 015730 012602                        ADD     #6,SP
2066 015732 004737 006024                  MOV     (SP)+,R2                ;RESTORE R2
2067 015736                                JSR     PC,PRITSSR            ;DECODE TSSR CONTENTS
2068 015736                                ENDMSG
2069 015736 104423                        L10025:
2070 015740 045 116 045 XFERASC:          TRAP    C$MSG
2071                                     .ASCIIZ  '%N%A Data Transferred = %03'
  
```

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                                     :INPUTS:
2082
2083                                     :      R5          ADDRESS OF FIRST REGISTER
2084
2085                                     :OUTPUTS:
2086
2087                                     :      R0          CONTENTS OF TSSR, IF ERROR
2088                                     :      CARRY       SET IF INIT WAS OKAY
2089                                     :                  CLEAR IF FATAL ERROR
2090
2091                                     :CALLING SEQUENCE:
2092
2093                                     :      MOV        #ADDRESS,R5
2094                                     :      JSR        PC,SOFINIT
2095                                     :      BCS       CONTINUE
2096                                     :      ERDF
2097                                     :                      ;REPORT FATAL ERROR
2098
2099                                     :-
2100
2101 015774    SOFINIT::
2102 015774            SAVREG                                ; SAVE THE REGISTERS
2103 016000    MOV     #0,TSSR(R5)                          ; DO THE INIT.
2104 016006    JSR     PC,WAITF                             ; WAIT FOR SSR
2105 016012    MOV     TSSR(R5),R0                         ; GET THE TSSR REGISTER
2106 016016    MOV     R0,R4                               ; TSSR CONTENTS
2107 016020    BIC     #^C<HIADDR!OFL>,R4
2108 016024    BIS     #SSR!NBA,R4                        ; R4 HAS EXPECTED CONTENTS
2109 016030    CMP     R4,R0                              ; ONLY EXPECTED BITS SET ?
2110 016032    BEQ     $$                                  ; BRANCH IF OKAY
2111 016034    CLC                                         ; CLEAR THE CARRY FOR ERROR
2112 016036    BR      10$                                ; GO TO EXIT
2113 016040    SEC                                         ; SET THE CARRY BIT
2114 016042    RTS     PC                                 ; RETURN TO CALLER

```

```
2116 .SBTTL CHKAMB - CHECK TSSR FOR AMBIGUITY
2117
2118
2119
2120 : THIS ROUTINE TESTS THE CONTENTS OF THE TSSR REGISTER
2121 : FOR AMBIGUITY
2122
2123 : INPUT:
2124
2125 : R0 CONTENTS OF TSSR
2126
2127 : OUTPUT:
2128
2129 : R0 CONTENTS OF TSSR
2130
2131 : CARRY SET - NO AMBIGUITY
2132 : CLR - AMBIGUOUS CONTENTS
2133
2134 : -
2135
2136 CHKAMB: SAVREG ;SAVE THE GENERAL REGISTERS
2137 MOV R0,R4 ;CONTENTS OF TSSR
2138 BIT #SC,R0 ;IS BIT 15 SET ?
2139 BNE 5$ ;BRANCH IF YES
2140 BIT #^C<NBA!OFL!SSR!HIADDR>,R0 ;ANY OTHER BITS SET ?
2141 BNE 40$ ;MUST BE AN ERROR
2142 BR 45$ ;RETURN WITH SUCCESS
2143 5$: BIT #SSR,R0 ;IS READY BIT SET ?
2144 BNE 10$ ;BRANCH IF READY BIT IS SET.
2145 BIT #BIT5,R0 ;IS FATAL ERROR BIT SET ?
2146 BEQ 40$ ;ERROR IF NOT
2147 BIC #^CTERCLS,R4 ;CLEAR ALL BUT TERMINATION CODE
2148 CMP R4,#16 ;ALL THREE BITS MUST BE SET
2149 BNE 40$ ;ERROR IF NOT SET
2150 BR 45$ ;OK IF ALL ARE SET
2151 10$: BIT #BIT5,R0 ;IS FATAL ERROR BIT SET ?
2152 BEQ 45$ ;ERROR IF BIT IS SET WITH SSR
2153 BIT #BIT2!BIT1,R0 ;IS THIS A FUNCTION REJECT
2154 BNE 45$ ;BR, IF TSSR IS OK
2155 40$: CLC ;AMBIGUOUS CONTENTS
2156 BR 50$
2157 45$: SEC ;SHOW SUCCESS - NO AMBIGUITY
2158 50$: RTS PC ;RETURN TO CALLER
2159
2160
```

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	

```

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 104422 BREAK ; DO A SUPVSR BREAK FIRST.
2231 016254 012746 003000 TRAP CSBRK
2232 016260 016500 000002 1$: MOV #3000,-(SP) ;300 MSEC TIMER
2233 016264 105700 2$: 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 012727 000001 DELAY 1 ; WAIT 100 USEC
2237 016274 000000 MOV #1,(PC)+
2238 016276 013727 002116 .WORD 0
2239 016302 000000 MOV LSDLY,(PC)+
2240 016304 005367 177772 .WORD 0
2241 016310 001375 DEC -6(PC)
2242 016312 005367 177756 BNE -4
2243 016316 001367 DEC -22(PC)
2237 016320 005316 BNE -20
2238 016322 001356 DEC (SP)
2239 016324 000241 BNE 2$ ;REDUCE DELAY COUNT
2240 016326 000401 CLC ;RETRY UNTIL TIMER EXPIRES
2241 016330 000261 BR 4$ ; C = 0, CONTROLLER STILL RUNNING...
2242 016332 005326 3$: SEC ;...OR HUNG-UP AFTER 300 MSEC.
2243 016334 000207 4$: DEC (SP)+ ; C = 1, CONTROLLER IS STOPPED.
RTS PC ;RESTORE STACK WITHOUT CHANGING CARRY BIT
  
```

```

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
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

      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

      -

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
      CLC                ;SET FAILURE
      BR 20$
      SEC                ;SET SUCCESS
      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 TSTLOOP::
2320 016456 005737 002170 TST NOITS ; ITERATIONS INHIBITED?
2321 016462 001006 BNE 1$ ; YES.
2322 016464 005737 002204 TST QVP ; NO.
2323 016470 100403 BMI 1$ ; LOOPS DISALLOWED IN QUICK PASS.
2324 016472 005337 002216 DEC LOOPCNT ; BUMP LOOP COUNTER.
2325 016476 001002 BNE 2$
2326 016500 000241 1$: CLC ; LOOP DISALLOWED, OR DONE.
2327 016502 000401 BR 3$
2328 016504 000261 2$: SEC ; LOOP ENABLED.
2329 016506 000207 3$: RTS PC

```

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 016510
2360 016510
2361 016512
2362 016516
2363 016522
2364 016526
2365 016532
2366 016536
2367 016540
2368 016544
2369 016546
2370 016550
2371 016556
016556
016560
016562
016564
2372 016566
2373 016570
2374 016576
016576
016600
016602
016604
2375 016606
2376 016614
016614
016620
2377 016622

010046
005037 003154
005037 016756
005037 005772
105037 016144
013700 002202
006300
005737 003114
001430
100010
052760 160000 003176
104455
000001
003740
005736
000407
052760 160001 003176 3\$:
104455
000002
004335
000000
012737 177777 003112 2\$:
013700 002202
104451

```
.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
:
:
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     #160000,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
ERRDF   1,NXR,NXRERR  ; NO DEVICE HERE -- PRINT IT
TRAP    C$ERRDF
        .WORD      1
        .WORD      NXR
        .WORD      NXRERR
BR       2$
BIS     #160001,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
ERRDF   2,NOINIT      ; DEVICE NOT IDLE
TRAP    C$ERRDF
        .WORD      2
        .WORD      NOINIT
        .WORD      0
MOV     #-1,DUFLG     ; DROP THE UNIT
DODU    UNITN
MOV     UNITN,R0
TRAP    C$DODU
DOCLN                     ; ABORT THE PASS
```

2378	016622	104444			TRAP	C\$DCLN	
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	'%SXTXA Test'
2391						.EVEN	

```

2393                                     .SBTTL TSTEND - PRINT ERRORS RECEIVED
2394                                     :
2395                                     : AT END OF EACH TEST, PRINT THE NUMBER OF ERRORS RECEIVED
2396                                     : IF NORMAL ERROR REPORTING IS DISABLED (FLA:IER).
2397                                     :
2398 016720 TSTEND: RFLAGS RO
                016720 104421 TRAP C$RFLA
2399 016722 030027 020000 BIT RO,#IER
2400 016726 001412 BEQ 1$ ; BR IF 'IER' NOT SET.
2401 016730 PRINTF #ESUM,ERRK ; PRINT ERROR COUNT.
                016730 013746 016756 MOV ERRK,-(SP)
                016734 012746 016760 MOV #ESUM,-(SP)
                016740 012746 000002 MOV #2,-(SP)
                016744 010600 MOV SP,RO
                016746 104417 TRAP C$PNTF
                016750 062706 000006 ADD #6,SP
2402 016754 000207 1$: RTS PC
2403
2404 016756 000000 ERRK: 0 ; LOCAL ERROR COUNT.
2405 016760 045 101 040 ESUM: .ASCIZ /%A %DXA ERRORS/
2406 016777 105 122 122 EMAXDU: .ASCIZ /ERROR LIMIT REACHED -- DROPPING UNIT/
2407 .EVEN
2408
2409                                     .SBTTL INCERK - INCREMENT LOCAL ERROR COUNT
2410                                     :
2411                                     :+
2412                                     : ROUTINES TO INCREMENT LOCAL ERROR COUNT AND CHECK FOR LIMIT:
2413                                     :
2413 017044 005237 016756 INCERK: INC ERRK ; INCREMENT LOCAL ERROR COUNT
2414 017050 010046 MOV RO,-(SP) ; SAVE RO
2415 017052 013700 002202 MOV UNITN,RO ; GET UNIT NUMBER
2416 017056 006300 ASL RO ; ... AND MAKE IT A WORD OFFSET.
2417 017060 062700 003176 ADD #ERTABL,RO ; RO GETS ADDRESS OF ERROR TABLE ENTRY.
2418 017064 005210 INC (RO) ; INCREMENT THE DEVICE ERROR COUNT
2419 017066 032710 007777 BIT #7777,(RO) ; DID WE OVERFLOW THE FIELD?
2420 017072 001001 BNE 1$ ; BR IF NO.
2421 017074 005310 DEC (RO) ; YES -- BACK IT UP TO 7777.
2422 017076 012600 1$: MOV (SP)+,RO ; RESTORE RO
2423 017100 000207 RTS PC ; RETURN TO CALLER.
2424
2425 017102 010046 CKEMAX: MOV RO,-(SP) ; SAVE RO
2426 017104 013700 002202 MOV UNITN,RO ; GET UNIT NUMBER
2427 017110 006300 ASL RO ; ... AND MAKE IT A WORD OFFSET
2428 017112 016000 003176 MOV ERTABL(RO),RO ; GET ERROR TABLE ENTRY
2429 017116 042700 170000 BIC #170000,RO ; EXTRACT ERROR COUNT FIELD
2430 017122 020037 002174 CMP RO,GERRMAX ; IS GLOBAL LIMIT EXCEEDED FOR THIS UNIT?
2431 017126 103004 BHIS 1$ ; BR IF YES
2432 017130 023737 016756 002172 CMP ERRK,LERRMAX ; IS LOCAL LIMIT EXCEEDED FOR THIS TEST?
2433 017136 103417 BLO 2$ ; BR IF NO
2434 017140 1$: RFLAGS RO ; GET OPERATOR FLAGS
                017140 104421 TRAP C$RFLA
2435 017142 032700 000040 BIT #IDU,RO ; IS DROPPING INHIBITED?
2436 017146 001013 BNE 2$ ; BR IF YES.
2437 017150 012737 177777 003112 MOV #-1,DUFLG ; NO -- DROP THE UNIT
2438 017156 ERRDF 4,EMAXDU
                017156 104455 TRAP C$ERDF
                017160 000004 .WORD 4
                017162 016777 .WORD EMAXDU

```

2439	017164	000000		.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 017202 010046 CKDROP: MOV R0,-(SP)
2450 017204 FORCERR 1$,NOTSSR
2451 017214 RFLAGS R0
2452 017216 104421 TRAP C$RFLA
2453 017222 032700 000040 BIT #IDU,R0
2454 017224 001010 BNE 1$
2455 017226 011600 MOV (SP),R0
2456 017234 012737 177777 003112 MOV #-1,DUFLG
2457 017234 013700 002202 DODU UNITN
2458 017242 104451 MOV UNITN,R0
2459 017244 012600 TRAP C$DODU
2460 017246 000207 DOCLN ;ABORT THE PASS
2461 TRAP C$DCLN
2462 1$: MOV (SP)+,R0
2463 RTS PC
2464
2465                                     .SBTTL CONFIG - DETERMINE CONFIGURATION OF SYSTEM
2466                                     ;
2467                                     ; SUBROUTINE - DETERMINE CONFIGURATION OF TSV05 SYSTEM.
2468 017250 CONFIG:
2469 017250 004737 015774 JSR PC,SOFINIT
2470 017254 000207 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
: TST      KTF LG
: BEQ      10$
: MOV      R1,R2
: .REPT    6
: ASR      R0
: ROR      R1
: .ENDR
: BIC      #177,R1
: CMP      R1,KTF LG
: BHIS     10$
: MOV      R1,#KIPAR6
: BIC      #160000,R2
: ADD      #140000,R2
: MOV      R2,R0
: SEC
: BR       15$
10$: CLC
15$: RTS   PC
: SAVE R1-R4 UNTIL NEXT RETURN
: SYSTEM HAVE ABOVE 28K?
: BR IF NO
: SAVE LOW ORDER BITS
: CONVERT WORD ADDRESS TO 32W BLOCKS
: MAKE IT DOUBLE PRECISION
: ALINE FOR LOWER 4K BOUNDARY
: HIGHER THAN EXISTING MEMORY?
: BR IF YES
: SETUP MAPPING REGISTER PAR6
: SETUP DISPLACEMENT IN PAGE
: ADD IN PAR6 BIAS
: RETURN IN R0
: SET SUCCESS
:
: SET FAILURE
: 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 10$: MOV R3,(R1)+ ;STORE A BACKGROUND WORD
2561 DEC R2 ;DONE ALL MEMORY IN FREE SPACE?
2562 BGT 10$ ;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 30$: JSR PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
2574 MOV R3,(R0)+ ;STORE TEST PATTERN IN >28K ADDRESS
2575 CMP R0,#160000 ;END OF PAR6 MAPPING AREA?
2576 BLO 30$ ;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 35$: TST T23B ;11/23B?
2588 BEQ 45$ ;NO KEEP GOING
2589 CMP @#KIPAR6,#7600 ;REACHED 18 BITS?
2590 BHIS 40$ ;YES
2591 BR 45$ ;NO KEEP GOING
2592 40$: MOV #20,SR3 ;SET 22 BIT RELOCATION
2593 45$: JMP 30$ ;KEEP GOING ON ETC.
2594 50$: JSR PC,KTOFF ;DISABLE KT.
2595 55$: RTS PC
  
```

TSV3 - GLOBAL AREAS MACRO M1113 25-MAY-82 09:19 PAGE 60-1 I 8
FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN

SEQ 0099

2596
2597

```

2599 .SBTTL CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN
2600
2601 :+
2602 : COMPARE MEMORY WITH A BACKGROUND PATTERN
2603 :
2604 : INPUTS:
2605 :
2606 :     RO = BACKGROUND PATTERN
2607 :     FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
2608 :     KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
2609 :
2610 : OUTPUTS:
2611 :
2612 :     CARRY - SET IF NO ERROR
2613 :     CARRY - CLR IF ERROR
2614 :
2615 : IMPLICIT OUTPUTS:
2616 :
2617 :     ERRHI - ERROR HIGH ADDRESS
2618 :     ERRLO - ERROR LOW ADDRESS
2619 :     EXPD  - EXPECTED DATA
2620 :     RECV  - RECEIVED DATA
2621 :
2622 : CMPMEM:
2623 :     SAVREG
2624 :     MOV RO,R3 ;SAVE R1-R5 UNTIL NEXT RETURN
2625 :     JSR PC,KTOFF ;COPY TEST PATTERN
2626 :     MOV FREE,R1 ;DISABLE KT.
2627 :     MOV FRESIZ,R2 ;GET FIRST FREE LOCATION
2628 :     CMP R3,(R1) ;SIZE OF FREE SPACE BELOW 28K.
2629 :     BEQ 15$ ;FREE SPACE LOCATION EQUAL TO EXPD?
2630 :     MOV R1,ERRLO ;BR IF YES
2631 :     CLR ERRHI ;SAVE ADDRESS IN ERROR
2632 :     MOV R3,EXPD ;NO HIGH ADDRESS
2633 :     MOV (R1),RECV ;SAVE EXPD FOR ERROR REPORT
2634 :     BR 50$ ;SAVE RECV FOR ERROR REPORT
2635 :
2636 : 10$: TST (R1)+ ;POINT TO NEXT ADDRESS
2637 :     DEC R2 ;DONE ALL MEMORY IN FREE SPACE?
2638 :     BGT 10$ ;BR IF NO
2639 :     TST KTFLG ; GOT KT?
2640 :     BEQ 55$ ; NO. GET OUT.
2641 :     JSR PC,KTON ; YES. ENABLE KT.
2642 :     CLR R0 ;HIGH ORDER ADDRESS START
2643 :     MOV PST32W,R1 ;GET >28K START ADDRESS (IN 32W BLOCKS)
2644 :     .REPT 6
2645 :     ROL R1 ;CONVERT BLOCKS TO WORDS
2646 :     ROL R0 ;MAKE IT DOUBLE PRECISION
2647 :     .ENDR
2648 :     BIC #177,R1 ;ALINE 4K BOUNDARY
2649 :     MOV R0,-(SP) ;SAVE HIGH ORDER
2650 :     MOV R1,-(SP) ;SAVE LOW ORDER
2651 :     JSR PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
2652 :     MOV R0,R4 ;COPY ADDRESS BIASED TO PAR6
2653 :     MOV (SP)+,R1 ;RESTORE LOW ORDER IN NON PAR6 FORMAT
2654 :     MOV (SP)+,R0 ;RESTORE HIGH ORDER IN NON PAR6 FORMAT
2655 :     CMP R3,(R4) ;ABOVE 28K LOCATION EQUAL EXPD?
2656 :     BEQ 32$ ;BR IF YES
2657 :     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                                     .SBTTL  REGSAV - SAVE R1-R5 ON STACK
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
2704 020156 004736
2705 020160 012601
2706 020162 012602
2707 020164 012603
2708 020166 012604
2709 020170 012605
2710 020172 000207
2711
  
```

000012

```

: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 ;ROUTINE TO REQUEST AN 8 BIT DATA PATTERN FROM THE OPERATOR
2716 ;INPUTS:
2717 ;
2718 ;NONE.
2719 ;
2720 ;OUTPUTS:
2721 ;
2722 ;R0 OCTAL NUMBER FROM THE OPERATOR
2723 ;
2724 ;CALLING SEQUENCE:
2725 ;
2726 ;JSR PC,GETPAT
2727 ;-
2728
2729 GETPAT::
2730 SAVREG ;SAVE THE GENERAL REGISTERS
2731 1$: GMANID DATASC,PATDAT,0,377,0,377,NO
2732 TRAP CS$GMAN
2733 BR 10000$
2734 .WORD PATDAT
2735 .WORD T$CODE
2736 .WORD DATASC
2737 .WORD 377
2738 .WORD T$LOLIM
2739 .WORD T$HILIM
2740 10000$:
2741 BNCOMPLETE 1$ ;RETRY IF ERROR
2742 BCC 1$
2743 MOV PATDAT,R0 ;DATA PATTERN FROM OPERATOR
2744 RTS PC ;RETURN TO CALLER
2745
2746 ;+
2747 ;LOCAL DATA AREA
2748 ;-
2749
2750 PATDAT: .WORD 0 ;TEMPORARY STORAGE FOR DATA
2751 DATASC: .ASCIZ 'ENTER DATA PATTERN'
2752 .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 ASCII 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     TST      (R3)         ;END OF ASCII ?
2769     BEQ      3$           ;BRANCH IF ALL LINES DISPLAYED
2770     PRINTF   #SELASC,(R3)+ ;DISPLAY THE MENU
2771     MOV      (R3)+,-(SP)
2772     MOV      #SELASC,-(SP)
2773     MOV      #2,-(SP)
2774     MOV      SP,R0
2775     TRAP     C$PNTF
2776     ADD      #6,SP
2777     BR       2$
2778     3$:      GMANID      MENASC,MENRES,D,-1,0,-1,NO
2779     TRAP     C$GMAN
2780     BR       10001$
2781     .WORD    MENRES
2782     .WORD    T$CODE
2783     .WORD    MENASC
2784     .WORD    -1
2785     .WORD    T$LOLIM
2786     .WORD    T$HILIM
2787     10001$:  BNCOMPLETE  1$      ;RETRY IF ERROR
2788     BCC      1$
2789     MOV      MENRES,R0      ;GET THE OPERATOR'S REPLY
2790     CMP      R0,R1         ;COMPARE TO MAXIMUM ALLOWED
2791     BLOS     5$            ;BRANCH IF OK
2792     PRINTF   #MENERR       ;DISPLAY ERROR MESSAGE
2793     MOV      #MENERR,-(SP)
2794     MOV      #1,-(SP)
2795     MOV      SP,R0
2796     TRAP     C$PNTF
2797     ADD      #4,SP
2798     BR       1$           ;RETRY
2799     5$:      RTS          PC    ;RETURN TO CALLER
2800     MENERR:  .ASCII      '%N% *** Menu Selection Too Large ***'
2801     SELASC:  .ASCII      '%N%T'
2802     MENASC:  .ASCII      'Enter Menu Selection: '

```

TSV3 - GLOBAL AREAS MACRO M1113 25-MAY-82 09:19 PAGE 64-1
GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE

B 9

SEQ 0105

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 CHKMAN::
2809     SAVREG                ;SAVE THE REGISTERS
2810     MANUAL                ;SEE IF MANUAL INTERVENTION OK
2811     TRAP C$MANI
2812     BCOMPLETE 1$          ;BRANCH IF ALLOWED
2813     BCS 1$
2814     PRINTF #NOMAN         ;PRINT THE WARNING MESSAGE
2815     MOV #NOMAN,-(SP)
2816     MOV #1,-(SP)
2817     MOV SP,R0
2818     TRAP C$PNTF
2819     ADD #4,SP
2820     CLC                  ;CLEAR CARRY FOR ERROR
2821     RTS PC               ;RETURN
2822
2823 1$:
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
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
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
  
```

```

2819                                     .SBTTL  ENVIRN  - SETUP FREE DIAGNOSTIC SPACE
2820                                     :
2821                                     : SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
2822                                     :
2823  ENVIRN: MEMORY  R0
2824  020630 104431 TRAP  C$MEM
2825  020632 010037 003124 MOV  R0,FREE ; GET 1ST FREE ADDRESS...
2826  020636 062737 000002 003124 ADD  #2,FREE
2827  020644 011037 003126 MOV  (R0),FRESIZ ;...AND WORD COUNT.
2828  020650 162737 000004 003126 SUB  #4,FRESIZ
2829  020656 013702 002012 MOV  L$UNIT,R2 ; GET NUMBER OF UNITS
2830  020662 162737 000007 003126 10$: SUB  #7,FRESIZ ; TAKE AWAY 7 WORDS PER UNIT
2831  020670 005302 DEC  R2
2832  020672 001373 BNE  10$
2833  020674 013700 003124 MOV  FREE,R0 ;GET FIRST FREE ADDRESS
2834  020700 063700 003126 ADD  FRESIZ,R0 ;POINT TO LAST FREE ADDRESS
2835  020704 162700 000002 SUB  #2,R0 ;BACKUP 1 WORD
2836  020710 010037 003130 MOV  R0,FREEH1 ;STORE LAST FREE ADDRESS
2837  020714 000240 NOP
2838  020716 012701 177520 MOV  #BDVPCR,R1 ;*****
2839  020722 010102 MOV  R1,R2 ;GET BDV11 PCR ADDRESS
2840  020724 062702 000002 ADD  #2,R2 ;COPY TO R2
2841  020730 004737 016376 JSR  PC,XNXM ;SET THE RANGE
2842  020734 103001 BCC  15$ ;SEE IF WE HAVE ONE
2843  020736 000445 BR  40$ ;OK TO SET FLAGS
2844  020740 013701 177520 15$: MOV  BDVPCR,R1 ;RETURN WITH FLAGS CLEAR
2845  020744 062701 000001 ADD  #1,R1 ;SAVE PCR CONTENTS
2846  020750 012702 177520 MOV  #BDVPCR,R2 ;ADD ONE TO IT
2847  020754 005212 INC  (R2) ;GET BDV11 PCR ADDRESS
2848  020756 013703 177520 MOV  BDVPCR,R3 ;TRY TO WRITE TO IT
2849  020762 020103 CMP  R1,R3 ;GET RESULTS
2850  020764 001017 BNE  20$ ;DID IT CHANGE?
2851  020766 005237 003144 INC  T23A ;NO, MUST BE 11/23B
2852  020772 042737 170000 002120 BIC  #170000,L$HIME ;SET THE FLAG
2853  021000 000240 NOP ;SUPERVISOR COULD BE WRONG
2854  021002 PRINTF #M8186 ;BR 40$ FOR RELEASE
2855  021002 012746 005554 MOV  #M8186,-(SP) ;TELL THE SYSTEM TYPE
2856  021006 012746 000001 MOV  #1,-(SP)
2857  021012 010600 MOV  SP,R0
2858  021014 104417 TRAP  C$PNTF
2859  021016 062706 000004 ADD  #4,SP
2860  021022 000413 BR  40$ ;RETURN
2861  021024 005237 003146 20$: INC  T23B ;SET THE FLAG
2862  021030 000240 NOP ;BR 40$ FOR RELEASE
2863  021032 PRINTF #M8189 ;TELL THE SYSTEM TYPE
2864  021032 012746 005645 MOV  #M8189,-(SP)
2865  021036 012746 000001 MOV  #1,-(SP)
2866  021042 010600 MOV  SP,R0
2867  021044 104417 TRAP  C$PNTF
2868  021046 062706 000004 ADD  #4,SP
2869  021052 000207 40$: RTS  PC ;RETURN

```

```

2861                                     .SBTTL KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS
2862                                     ;+
2863                                     ;ROUTINE TO INIT KT-11
2864                                     ;-
2865
2866
2867
2868 KTINIT:
2869 021054 005037 003132 CLR KTFLG ; INIT >28K MEMORY FLAG
2870 021054 005037 003134 CLR KTENABLE ; INIT TEST >28K FLAG
2871 021060 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$,@#ERRVEC ; SET ERR VEC PTR.
2875 021106 005737 177572 TST @#SR0 ; GOT KT11?
2876 021112 000240 NOP ; (TRAP IF NO).
2877 021114 013737 002120 003132 MOV LSHIME,KTFLG ; YES. SET KT FLAG.
2878 021122 042737 000177 003132 BIC #177,KTFLG
2879 021130 010037 000004 MOV R0,@#ERRVEC ; RESTORE OLD ERR VEC PTR.
2880 021134 005000 CLR R0 ; R0 = AR DATA.
2881 021136 012701 172340 MOV #KIPAR0,R1 ; R1 = KI REGS PTR.
2882 021142 012761 077406 177740 1$: MOV #77406,-40(R1) ; SET DESCRIPTOR REG.
2883 021150 010021 MOV R0,(R1)+ ; SET KIPAR REG.
2884 021152 062700 000200 ADD #200,R0 ; BUMP AR DATA BY '4k'.
2885 021156 020027 002000 CMP R0,#2000 ; AT 'I/O'?
2886 021162 001367 BNE 1$ ; NO.
2887 021164 012741 177600 MOV #177600,-(R1) ; YES. SET KIPAR7 FOR I/O.
2888 021170 000405 BR 9$
2889
2890 021172 012716 021200 2$: MOV #6$, (SP) ; SET UP RETURN
2891 021176 000002 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 BOUNDARY.
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

```

```

2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952 021276
2953
2954 021276
2955 021302 005037 003136
2956 021306 005037 003140
2957 021312 005037 003142
2958 021316 005737 003146
2959 021322 001407
2960 021324 023727 002120 007777
2961 021332 103406
2962 021334 004737 021452
2963 021340 000427
2964 021342 005737 003144 1$:
2965 021346 001413
2966 021350 023727 002120 005777 2$:
2967 021356 101023
2968 021360 023727 002120 003777
2969 021366 103403
2970 021370 004737 021452
2971 021374 000411
2972 021376 023727 002120 001577 4$:
2973 021404 103410
2974 021406 004737 021452
2975 021412 062737 000077 003142
2976 021420 005237 003136 13$:
2977 021424 000411
2978 021426 000410 14$:
2979 021430
    021430 012746 005460
    021434 012746 000001
    021440 010600
    021442 104417
    021444 062706 000004
2980 021450 000207 15$:
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990 021452 013701 002120
2991 021456 062701 000200
2992 021462 042701 000177
2993 021466 010102

;+
; SUBROUTINE TO CHECK WETHER OR NOT WE'LL TEST NXM
;
; INPUTS:
; OUTPUTS:
; The NXMFLG is set if we can test.
; The NXMLO and NXMHI addresses are setup.
;-

MEMCK::

SAVREG
CLR NXMFLG
CLR NXMLO
CLR NXMHI
TST T23B
BEQ 1$
CMP L$HIME,#7777
BLO 2$
JSR PC,NXMTST
BR 13$
TST T23A
BEQ 4$
CMP L$HIME,#5777
BHI 14$
CMP L$HIME,#3777
BLO 4$
JSR PC,NXMTST
BR 13$
CMP L$HIME,#1577
BLO 14$
JSR PC,NXMTST
ADD #77,NXMHI
INC NXMFLG
BR 15$
BR 15$
PRINTF #NOMEM
MOV #NOMEM,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #4,SP
RTS PC

;SAVE THE REGISTERS
;CLEAR THE FLAG
;CLEAR THE TEST ADDRESS LO
;CLEAR THE TEST ADDRESS HI
;IS IT A 11/23B?
;NO
; GREATER THAN 128K
; NO
;SETUP THE ADDRESS
;SET THE FLAG AND EXIT
;IS IT A 11/23A?
;NO
;GREATER THAN 96K
;YES,23A/23B WITH 128K MEMORY
;GREATER THAN 64K BUT LESS THAN 92K?
;NO, CHECK 24K
;SETUP THE ADDRESS
;SET THE FLAG AND EXIT
;GREATER THAN 24K BUT LESS THAN 64K?
;NO, TELL THEM AND EXIT WITH FLAG CLEAR
;SETUP THE ADDRESS
;FOOL THE 11/02 & 11/03
;SET THE FLAG
;EXIT
;NOP FOR PRINTOUT
;TELL THEM & EXIT ***NO PRINT*****

;RETURN

;+
; SUBROUTINE TO SETUP THE NXM ADDRESS FOR TESTING
;
; OUTPUTS: NXMLO, NXMHI
; SETUP WITH NXM ADDRESS
;-

NXMTST: MOV L$HIME,R1
ADD #200,R1
BIC #177,R1
MOV R1,R2

;GET TOP OF MEMORY
;MAKE IT I/O BLOCK OR OTHER NXM
;RESAVE RESULTS

```

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                                     .SBTTL  PROTECTION TABLE
19 021546                             BGNPROT
    021546
20 021546 177777 177777 177777  L$PROT:: .WORD  -1, -1, -1, -1
21 021556                             ENDPROT
22

```

;NO DEVICE PROTECTION REQUIRED.

```
24                                     .SBTTL  INITIALIZE SECTION
25
26
27      ;++
28      ;THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
29      ;AT THE BEGINNING OF EACH PASS.
30
31      ;IF "START" OR "RESTART", SET QUICK-PASS FLAG AND BUS-INIT.
32      ;IF "CONTINUE", NOTHING IS REQUIRED.
33
34      ;--
35      ;+
36      ;INSERT TEMPORARY JUMP TO ODT
37      ;-
38      021556      BGNINIT
39      021556      L$INIT::
40      021556      005037 002226      40$: CLR      EXTFEA
41      021562      005037 003136      CLR      NXMFLG
42      021566      012737 006360      MOV      #EPRT1,EPRTSW      ;SET UP PRIMARY MESSAGE FOR REPLACEMENT
43      021574      005037 003154      CLR      SIFLAG      ;CLEAR "SOFT INIT" FLAG
44      021600      005037 003134      CLR      KTENABLE      ;CLEAR TEST ABOVE 28K FLAG
45      021604      005037 002302      CLR      RAMSIZ      ;CLEAR RAM SIZE FOR RAMERR ROUTINE
46      021610      012700 000036      READEF #EF.CONTINUE
47      021614      104447      MOV      #EF.CONTINUE,R0
48      021616      103023      TRAP     C$REFG
49      021620      023737 002202 002012      BNCOMPLETE 1$
50      021626      103070      BCC      1$
51      021630      005737 003112      CMP      UNITN,L$UNIT      ;UNIT IN RANGE?
52      021634      100472      BHIS     4$      ;BR IF NO.
53      021636      013701 002202      TST      DUFLG      ;DROPPED UNIT?
54      021642      006301      BMI      NXTU      ;BR IF YES
55      021644      005761 003176      MOV      UNITN,R1
56      021650      001516      ASL      R1
57      021652      032761 040000 003176      TST      ERTABL(R1)
58      021660      001060      BEQ      SETU      ;DROPPED?
59      021662      104432      BIT      #BIT14,ERTABL(R1)
60      021664      000416      BNE      NXTU      ;DO NOTHING IF "CONTINUE".
61      021666      012700 000035      EXIT     INIT
62      021672      104447      TRAP     C$EXIT
63      021674      103052      .WORD    L10030-.
64      021676      012700 000040      READEF #EF.NEW
65      021702      104447      MOV      #EF.NEW,R0
66      021704      103404      TRAP     C$REFG
67      021706      012700 000037      BNCOMPLETE NXTU      ;TAKE NEXT UNIT IF NOT NEW PASS.
68      021712      103031      BCC      NXTU
69      021714      012700 000040      READEF #EF.START
70      021716      104433      MOV      #EF.START,R0
71      021718      012700 000037      TRAP     C$REFG
72      021720      103031      BNCOMPLETE 2$
73      021722      012700 000037      BCS      2$
74      021724      012700 000037      READEF #EF.RESTART
75      021726      012700 000037      MOV      #EF.RESTART,R0
76      021728      012700 000037      TRAP     C$REFG
77      021730      012700 000037      BNCOMPLETE 31$
78      021732      012700 000037      BCC      31$
79      021734      012700 000037      BRESET
80      021736      012700 000037      TRAP     C$RESET      ;1ST PASS, BUS-INIT...
81      021738      012700 000037      ;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	O.OBT	::@ENTER THE DEBUGGER
72	021740	005037	003400		CLR	SKIPT	:CLEAR THE SUBTEST "SKIPPER"
73	021744			20\$:			
74	021744	012737	177777	002204	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,R0	
78	021766	005020		30\$:	CLR	(R0)+	:CLEAR THE ERROR TABLE
79	021770	020027	003376		CMP	R0,#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				4\$:			
85	022010						
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			NXTU:	BREAK		
	022022	104422			TRAP	C\$BRK	
89	022024	005237	002202		INC	UNITN	:...AND SET NEXT UNIT NUMBER.
90	022030	023737	002202	002012	CMP	UNITN,LSUNIT	
91	022036	103423			BLO	SETU	
92	022040	012737	177777	003112	MOV	#-1,DUFLG	
93	022046	000401			BR	11\$	
94	022050				DOCLN		:ABORT, NO MORE UNITS.
	022050	104444			TRAP	C\$DCLN	
95	022052	000240		11\$:	NOP		
96	022054			PASRPT:			
97	022054	023727	002012	000001	CMP	LSUNIT,#1	:HOW MANY UNITS SELECTED?
98	022062	101752			BLOS	NEWPAS	:BR IF ONLY 1
99	022064	005737	002220		TST	DEVCNT	:ARE ANY STILL RUNNING?
100	022070	001747			BEQ	NEWPAS	:BR IF NO
101	022072				RFLAGS	R0	
	022072	104421			TRAP	C\$RFLA	
102	022074	032700	000100		BIT	#ISR,R0	:SHOULD WE PRINT STATISTICS
103	022100	001343			BNE	NEWPAS	:BR IF NO
104							
105	022102				DORPT		
	022102	104424			TRAP	C\$DRPT	
106	022104	000741			BR	NEWPAS	
107	022106			10\$:			
108							
109	022106			SETU:	GPHARD	UNITN,R0	:GET UNIT N P-TABLE POINTER.
	022106	013700	002202		MOV	UNITN,R0	
	022112	104442			TRAP	C\$GPHRD	
110	022114				BNCOMPLETE	NXTU	:BR IF UNIT NOT AVAILABLE.
	022114	103342			BCC	NXTU	
111	022116	005037	003112		CLR	DUFLG	:CLEAR "DROPPED" FLAG.
112	022122	005237	002220		INC	DEVCNT	
113	022126	012001			MOV	(R0)+,R1	:GET 1ST REGISTER ADDRESS.
114	022130	010137	002206		MOV	R1,C\$RADDR	: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$PNTF
147 022236      ADD      #6,SP
148 022236 005037 003114      10$:      CLR      NODEV
149 022242 013701 002206      MOV      CSRADDR,R1      ;ADDRESS OF FIRST REGISTER
150 022246 010102      MOV      R1,R2      ;START OF REGISTERS
151 022250 062702 000002      ADD      #TSSR,R2      ;ADDRESS OF TSSR REGISTER
152 022254 004737 016376      JSR      PC,XNXM      ;TEST BOTH CONTROLLER REGISTERS...
153 022260 103005      BCC      2$      ;...AND BR IF ALL OK.
154 022262 010137 003114      MOV      R1,NODEV      ;FLAG DEVICE AS NON-EXISTENT
155 022266 012737 177777 003112      MOV      #-1,DUFLG      ;DROP THIS UNIT.
156 022274
157
158                   ;FINALLY, SET CPU PRIORITY AND WE'RE DONE.
159                   ;
160 022274 012700 000000      5$:      SETPRI      #PRI00      ;ENABLE INTERRUPTS.
161 022274 104444      MOV      #PRI00,R0
162 022300 104444      TRAP      C$SPRI
163 022302      ENDINIT
164 022302      L10030:
165 022302 104411      TRAP      C$INIT
166
167 022304 045 116 045 PUNIT: .ASCIZ      /%N%N%A***** TESTING UNIT %D2%A *****/
168 .EVEN
```

```
160 .SBTTL ADD AND DROP UNITS SECTIONS
161
162
163 :++
164 : THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
165 : TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
166 : OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
167 :--
168 022352 BGNAU
169 022352 L$AU::
170 022352 010001 MOV R0,R1 ; GET UNIT TO BE ADDED (R0)
171 022354 006301 ASL R1 ; MAKE IT A WORD INDEX
172 022356 052761 100000 003176 BIS #100000,ERTABL(R1) ; SET THE 'ACTIVE' BIT
173 022364 042761 040000 003176 BIC #40000,ERTABL(R1) ; CLEAR THE 'DROPPED' BIT
174 022372 PRINTF #1$,R0
175 022372 010046 MOV R0,-(SP)
176 022374 012746 022420 MOV #1$,-(SP)
177 022400 012746 000002 MOV #2,-(SP)
178 022404 010600 MOV SP,R0
179 022406 104417 TRAP C$PNTF
180 022410 062706 000006 ADD #6,SP
181 022414 EXIT AU
182 022414 000167 .WORD JSJMP
183 022416 000026 .WORD L10031-2-
184 022420 045 116 045 1$: .ASCIZ /%N% UNIT %D% ADDED/
185 .EVEN
186
187 ENDAU ; UNUSED.
188 L10031:
189 022446 TRAP C$AU
190 022446 104452
191
192 :++
193 : THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
194 : TO BE REMOVED FROM THE TEST LIST.
195 :
196 : SUPVSR DOES THE 'DROPPING'. THIS IS JUST TO TELL THE MAN.
197 : 'DROPPED' UNITS ARE RE-SELECTED ON OPERATOR 'STA' OR 'ADD'.
198 : COMMAND, OTHERWISE REMAIN INACTIVE. THE 'DISPLAY' COMMAND
199 : WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
200 : WHICH ARE STILL ACTIVE.
201 : UPON ENTRY, R0 CONTAINS THE UNIT TO BE DROPPED.
202
203 BGNDU
204 L$DU::
205 022450 MOV #-1,DUFLG
206 022450 MOV R0,R1
207 022456 010001 ASL R1
208 022460 006301 BIS #140000,ERTABL(R1) ; SAY DROPPED
209 022462 052761 140000 003176 240,240,240 ; ?????????
210 022470 000240 000240 000240
211 022476 PRINTF #1$,R0
212 022476 010046 MOV R0,-(SP)
213 022500 012746 022524 MOV #1$,-(SP)
214 022504 012746 000002 MOV #2,-(SP)
215 022510 010600 MOV SP,R0
216 022512 104417 TRAP C$PNTF
217 022514 062706 000006 ADD #6,SP
218 022520 EXIT DU
219 022520 000167 .WORD JSJMP
220 022522 000030 .WORD L10032-2-
```

197	022524	045	116	045	1\$:	.ASCIZ /%N% UNIT %D% 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	

				.SBTTL	CLEAN-UP AND REPORT CODING SECTIONS	
215				:++		
216				:	THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS	
217				:	EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).	
218				:	USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.	
219				--		
220						
221				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	MOV	#0,TSSR(R5)	;DO SOFT INIT
229	022656	004737	016250	JSR	PC,WAITF	
230	022662			1\$:		
231	022662			2\$:	ENDCLN	
232	022662			L10034:		
233	022662	104412		TRAP	C\$CLEAN	
234				:++		
235				:	THE REPORT CODING SECTION CONTAINS THE	
236				:	'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.	
237				--		
238	022664			BGNRPT		
239	022664			L\$RPT::		
240	022664	012746	023126	PRINTS	#DEVSUM	
241	022670	012746	000001	MOV	#DEVSUM,-(SP)	
242	022674	010600		MOV	#1,-(SP)	
243	022676	104416		MOV	SP,R0	
244	022700	062706	000004	TRAP	C\$PNTS	
245	022704	010246		ADD	#4,SP	
246	022706	010346		MOV	R2,-(SP)	
247	022710	010446		MOV	R3,-(SP)	
248	022712	012704	003176	MOV	R4,-(SP)	
249	022716	005003		MOV	#ERTABL,R4	; GET START OF ERROR TABLE.
250	022720	011402		CLR	R3	; CLEAR UNIT NUMBER
251	022722	001467		1\$: MOV	(R4),R2	; GET ERROR TABLE ENTRY & TEST IT.
252	022724	100066		BEQ	4\$	; ZERO IF UNIT NOT RUN
253	022726	032702	040000	BPL	4\$	
254	022732	001015		BIT	#BIT14,R2	; WAS UNIT DROPPED?
255	022734	042702	170000	BNE	2\$	; BR IF YES
256	022740			BIC	#^C7777,R2	; GET ERROR COUNT FIELD
257	022740	010246		PRINTS	#DEVONL,R3,R2	; PRINT
258	022742	010346		MOV	R2,-(SP)	
259	022744	012746	023163	MOV	R3,-(SP)	
260	022750	012746	000003	MOV	#DEVONL,-(SP)	
261	022754	010600		MOV	#3,-(SP)	
262	022756	104416		MOV	SP,R0	
263	022760	062706	000010	TRAP	C\$PNTS	
264	022764	000446		ADD	#10,SP	
265	022766	020227	160000	BR	4\$	
266	022772	001012		2\$: CMP	R2,#160000	; WAS UNIT NON-EXISTENT?
267	022774			BNE	3\$	; BR IF NO
268	022774	010346		PRINTS	#DEVNXR,R3	
269	022774	012746	023233	MOV	R3,-(SP)	
270				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$:   BIC      #^C7777,R2
260 023056      PRINTS    #DEVDRD,R3,R2
023056 010246      MOV      R2,-(SP)
023060 010346      MOV      R3,-(SP)
023062 012746 023376      MOV      #DEVDRD,-(SP)
023066 012746 000003      MOV      #3,-(SP)
023072 010600      MOV      SP,R0
023074 104416      TRAP     C$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 DEVDRD: .ASCIZ  /%A UNIT %D3%A DROPPED, ERRORS = %D%N/
276      .EVEN
277
278 023446      ENDMOD
279
280

```

TSV5 - HARDWARE TESTS MACRO M1113
CLEAN-UP AND REPORT CODING SECTIONS

25-MAY-82 09:19 PAGE 77

E 10

SEQ 0121

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      ;ASCII MESSAGE TO IDENTIFY TEST
51      ;DO INITIAL TEST SETUP
52      ;PERFORM 10 ITERATIONS
53      023446 012700 024102      MOV      #TST13ID,R0
54      023452 004737 016510      JSR      PC,TSTSETUP
55      023456 012737 000012 002216  MOV      #10.,LOOPCNT
56      023464      T13LOOP:      JSR      PC,T13REST
57      023464 004737 024356      ;SET PACKET TO START-UP VALUES
58
59      023470 012703 002764      MOV      #TSTBLK+10.,R3
60      023474 012704 024040      ;START OF TEST DATA
61      023500 012764 000010 000006  MOV      #T13PACKET,R4
62      023506      5$:          MOV      #8.,PKBCNT(R4)
63      023506 004737 015774      ;GET THE ADDRESS OF COMMAND PACKET
64      023512 103405      ;START WITH MINIMUM ALLOWABLE VALUE
65      023514 010001      JSR      PC,SOFINIT
66      023516      BCS      10$
67      023516 104455      ;WRITE TO TSSR TO SOFT INITIALIZE
68      023520 000144      ;BR IF SOFT INIT OKAY
69      023522 003652      ;SAVE CONTENTS OF TSSR
70      023524 012034      ;DEVICE FATAL DURING INIT
71
72      TRAP      C$ERDF
73      .WORD     100
74      .WORD     SFIERR
75      .WORD     SFIMSG
76
77      ;Do WRITE CHARACTERISTICS command.
78      10$:      CLR      FATFLG
79      023526 005037 002222      ;CLEAR FATAL ERROR FLAG
80      023532 010465 000000      MOV      R4,TSD(R5)
81      023536 004737 016336      ;SET THE PACKET ADDRESS TO EXECUTE
82      023542      JSR      PC,CHKTSSR
83      023542 103407      ;WAIT FOR SSR TO SET
84      023556 010001      ;ADDFORCE ERROR IF FORCER=1
85      023562      BCS      15$
86      023562 104455      ;BR IF CARRY SET (GOOD RETURN)
87      023564 000145      ;SAVE CONTENTS OF TSSR
88      023566 024267      ;DEVICE FATAL SSR FAILED TO SET
89      023570 012046      TRAP      C$ERDF
90      023572 005237 002222      .WORD     101
91      023576      12$:      ERRDF   ERRNO,T13SSR,PKTSSR
92
93      .WORD     T13SSR
94      .WORD     PKTSSR
95
96      15$:      INC      FATFLG
97      ;SET FATAL ERROR FLAG
98      ;LOOP ON ERROR, IF FLAG SET
  
```

```

79 023576 104406
80 023600 016501 000002
81 023604 012702 000200
82 023610 032701 000100
83 023614 001402
84 023616 052702 000100
85
86 023622
87 023622
88 023636 020201
89 023640 001404
90 023642
91 023642
92 023652
93 023652 104406
94
95 023654
96 023654 004737 015774
97 023660
98 023674 103405
99 023676 010001
100 023700
101 023700
102 023700 104455
103 023702 000147
104 023704 003652
105 023706 012034
106 023710 012704 000310
107 023714 005002
108 023716 105065 000000
109 023722 004737 016336
110 023726 010465 000000
111 023732 004737 016336
112 023736 116501 000000
113 023742
114 023752 120102
115 023754 001406
116 023756
117 023756 104455
118 023760 000150
119 023762 024155
120 023764 015530
121 023766 005237 002222
122 023772
123 023772 104406
124 023774
125 023774 104410
126 023776 000426

;If the NBA bit in the TSSR register is NOT=0 then Print Error.
25$:
    MOV TSSR(R5),R1
    MOV #SSR,R2
    BIT #OFL,R1
    BEQ 25$
    BIS #OFL,R2
    ;GET THE CONTENTS OF TSSR
    ;EXPECTED CONTENTS OF TSSR
    ;IS OFF-LINE BIT SET ?
    ;BRANCH IF NOT OFF-LINE
    ;SET OFF-LINE IN EXPECTED DATA
    TRAP C$CLP1

;Does expected match received ?
    ;OKAY IF MATCH
    ;@ad
    FORCERROR 27$
    CMP R2,R1
    BEQ 30$
    NEXT.ERRNO
    27$: ERRDF ERRNO,T13NBA,PKTSSR
    ;NBA NOT ZERO
    TRAP C$ERHRD
    .WORD 102
    .WORD T13NBA
    .WORD PKTSSR

30$: CKLOOP
    ;LOOP ON ERROR ?
    TRAP C$CLP1

;Write to TSSR register to soft initialize the controller
40$:
    JSR PC,SOFINIT
    FORCERROR 42$
    BCS 50$
    MOV R0,R1
    NEXT.ERRNO
    42$: ERRDF ERRNO,SFIERR,SFIMSG
    ;WRITE TO TSSR TO SOFT INITIALIZE
    ;@ad
    ;BR IF SOFT INIT OKAY
    ;SAVE CONTENTS OF TSSR
    ;DEVICE FATAL DURING INIT
    TRAP C$ERDF
    .WORD 103
    .WORD SFIERR
    .WORD SFIMSG

;If controller RAM 310-377 NOT=0 then Print Error
50$:
    MOV #310,R4
    CLR R2
    CLRB TSDB(R5)
    JSR PC,CHKTSSR
    60$: MOV R4,TSDB(R5)
    JSR PC,CHKTSSR
    MOVB TSBA(R5),R1
    FORCERROR 62$,NOTSSR
    CMPB R1,R2
    BEQ 70$
    NEXT.ERRNO
    62$: ERRDF ERRNO,T13MEM,RAMEXP
    ;START WITH LOC 310
    ;MEMORY EXPECTED SHOULD BE 000000
    ;SET MAINTENANCE MODE
    ;WAIT FOR SSR READY
    ;SELECT RAM ADDRESS
    ;WAIT FOR SSR READY
    ;READ LOC CONTENTS
    ;@ad
    ;CHECK MEMORY FOR 000000
    ;BRANCH IF DATA OKAY
    ;MEMORY NOT ZERO AFTER INIT.
    TRAP C$ERDF
    .WORD 104
    .WORD T13MEM
    .WORD RAMEXP

70$: INC FATFLG
    CKLOOP
    ;SET THE FATAL ERROR FLAG
    TRAP C$CLP1

    ESCAPE TST
    ;EXIT ON FATAL ERROR
    TRAP C$ESCAPE
    .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,TSTLOOP	;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\$:			
132	024034				EXIT	TST	
	024034	104432					TRAP C\$EXIT
	024036	000366					.WORD L10036-
133							
134							

```

136
137      ;+
138      ;LOCAL STORAGE FOR THIS TEST
139      ;-
143 024040      T13PACKET:
144 024040      100004      .WORD 100004      ;COMMAND PACKET FOR TEST
145 024042      024050      .WORD T13DATA      ;WRITE CHARACTERISTICS COMMAND, WITH ACK
146 024044      000000      .WORD 0      ;ADDRESS OF CHARACTERISTICS BLOCK
147 024046      000010      .WORD 8.      ;STARTING VALUE OF BLOCK SIZE
148
149 024050      T13DATA:
150 024050      024062      .WORD T13BFR      ;CHARACTERISTICS DATA BLOCK
151 024052      000000      .WORD 0      ;ADDRESS OF MESSAGE BUFFER
152 024054      000016      .WORD 14.      ;LENGTH OF MESSAGE BUFFER
153 024056      000000      000000      .WORD 0,0
154
155 024062      T13BFR: .BLKW 8.      ;MESSAGE BUFFER
156      ;LOCAL TEXT MESSAGES FOR TEST
157      ;-
158
159 024102      111      156      151      TST13ID:      .ASCIIZ 'Initialization After WRITE CHARACTERISTICS'
160 024155      111      156      143      T13MEM: .ASCIIZ 'Incorrect RAM Data After Init'
161      .EVEN
162 024214      127      122      111      T13NBA: .ASCIIZ 'WRITE CHARACTERISTICS Command Not Accepted'
163 024267      103      157      156      T13SSR: .ASCIIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
164

```

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

L10036: TRAP C\$ETST

190				.SBTTL TEST 2: BASIC WRITE SUBSYSTEM MEMORY COMMAND	
191				:*	
192				:	
193				:	
194				:THIS TEST VERIFIES THAT THE WRITE SUBSYSTEM MEMORY COMMAND WITH A	
195				:BSEL0 SELECT CODE OF 0 (NO-OP) EXECUTES CORRECTLY. IT ALSO	
196				:VERIFIES THAT A WRITE SUBSYSTEM MEMORY COMMAND WITH A NON-ZERO	
197				:MODE FIELD IS REJECTED. THE TEST FURTHER VERIFIES MICROPROGRAM	
198				:COMMAND DECODING AND HANDLING SEQUENCES.	
199				:	
200				:-	
201	024426			BGNTST	
	024426				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	MOV #20.,LOOPCNT	:PERFORM 20 ITERATIONS
209	024444			T14LOOP:	
210	024444			BGNSUB	:////////// BEGIN SUBTEST //////////
	024444				T2.1:
	024444	104402			TRAP C\$BSUB
211	024446	004737	026306	JSR PC,T14REST	:SET PACKET TO INITIAL VALUES
212	024452	004737	026344	JSR PC,T14RST	:SET PACKET TO INITIAL VALUES
213	024456			SETPRI #PRI00	:LOWER PRIORITY TO ALLOW INTERRUPTS
	024456	012700	000000		MOV #PRI00,R0
	024462	104441			TRAP C\$SPRI
214	024464			5\$:	
215	024464			BGNSEG	:>>>>>>>>> BEGIN SEGMENT >>>>>>>>>
	024464	104404			TRAP C\$BSEG
216	024466	004737	015774	JSR PC,SOFINIT	:DO SOFT INIT OF CONTROLLER
217	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
	024476	104455			TRAP C\$ERDF
	024500	000311			.WORD 201
	024502	003652			.WORD SFIERR
	024504	012034			.WORD SFIMSG
223	024506			10\$:	
224	024506	012704	025600	MOV #T14PK2,R4	:SUBROUTINE NEEDS PACKET ADDRESS
225	024512	004737	010662	JSR PC,WRTCHR	:ISSUE WRITE CHARACTERISTICS
226	024516	103405		BCS 11\$	:BR, IF COMMAND ISSUED OK
230	024520	010001		MOV R0,R1	:SAVE CONTENTS OF TSSR
231	024522			ERRHRD ERRNO,WRTMSG,SFIMSG	:WRITE CHARACTERISTICS FAILED
	024522	104456			TRAP C\$ERHRD
	024524	000312			.WORD 202
	024526	005056			.WORD WRTMSG
	024530	012034			.WORD SFIMSG
232	024532			11\$:	
233	024532	005037	002222	CLR FATFLG	:CLEAR FATAL ERROR FLAG
234	024536	005037	002224	CLR INTRECV	:CLEAR INTERRUPT RECEIVED FLAG
235	024542	012704	025160	MOV #T14PACKET,R4	:SET UP NEW WRT. SUBSYS MEM. COMMAND
236	024546	010465	000000	MOV R4,TSDB(R5)	:SET THE PACKET ADDRESS
237	024552	004737	016336	JSR PC,CHKTSSR	:WAIT FOR SSR TO SET
238	024556	103407		BCS 15\$	:BR IF CARRY SET (GOOD RETURN)
239	024560	010001		MOV R0,R1	:SAVE CONTENTS OF TSSR
243	024562			ERRDF ERRNO,T14SSR,PKTSSR	:DEVICE FATAL SSR FAILED TO SET
	024562	104455			TRAP C\$ERDF
	024564	000313			.WORD 203

Address	Hex	Dec	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op41
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

[illegible]

PC	Op	OpC	OpD	OpE	OpF	OpG	OpH	OpI	OpJ	OpK	OpL	OpM	OpN	OpO	OpP	OpQ	OpR	OpS	OpT	OpU	OpV	OpW	OpX	OpY	OpZ	OpAA	OpAB	OpAC	OpAD	OpAE	OpAF	OpAG	OpAH	OpAI	OpAJ	OpAK	OpAL	OpAM	OpAN	OpAO	OpAP	OpAQ	OpAR	OpAS	OpAT	OpAU	OpAV	OpAW	OpAX	OpAY	OpAZ	OpBA	OpBB	OpBC	OpBD	OpBE	OpBF	OpBG	OpBH	OpBI	OpBJ	OpBK	OpBL	OpBM	OpBN	OpBO	OpBP	OpBQ	OpBR	OpBS	OpBT	OpBU	OpBV	OpBW	OpBX	OpBY	OpBZ	OpCA	OpCB	OpCC	OpCD	OpCE	OpCF	OpCG	OpCH	OpCI	OpCJ	OpCK	OpCL	OpCM	OpCN	OpCO	OpCP	OpCQ	OpCR	OpCS	OpCT	OpCU	OpCV	OpCW	OpCX	OpCY	OpCZ	OpDA	OpDB	OpDC	OpDD	OpDE	OpDF	OpDG	OpDH	OpDI	OpDJ	OpDK	OpDL	OpDM	OpDN	OpDO	OpDP	OpDQ	OpDR	OpDS	OpDT	OpDU	OpDV	OpDW	OpDX	OpDY	OpDZ	OpEA	OpEB	OpEC	OpED	OpEE	OpEF	OpEG	OpEH	OpEI	OpEJ	OpEK	OpEL	OpEM	OpEN	OpEO	OpEP	OpEQ	OpER	OpES	OpET	OpEU	OpEV	OpEW	OpEX	OpEY	OpEZ	OpFA	OpFB	OpFC	OpFD	OpFE	OpFF	OpFG	OpFH	OpFI	OpFJ	OpFK	OpFL	OpFM	OpFN	OpFO	OpFP	OpFQ	OpFR	OpFS	OpFT	OpFU	OpFV	OpFW	OpFX	OpFY	OpFZ	OpGA	OpGB	OpGC	OpGD	OpGE	OpGF	OpGG	OpGH	OpGI	OpGJ	OpGK	OpGL	OpGM	OpGN	OpGO	OpGP	OpGQ	OpGR	OpGS	OpGT	OpGU	OpGV	OpGW	OpGX	OpGY	OpGZ	OpHA	OpHB	OpHC	OpHD	OpHE	OpHF	OpHG	OpHH	OpHI	OpHJ	OpHK	OpHL	OpHM	OpHN	OpHO	OpHP	OpHQ	OpHR	OpHS	OpHT	OpHU	OpHV	OpHW	OpHX	OpHY	OpHZ	OpIA	OpIB	OpIC	OpID	OpIE	OpIF	OpIG	OpIH	OpII	OpIJ	OpIK	OpIL	OpIM	OpIN	OpIO	OpIP	OpIQ	OpIR	OpIS	OpIT	OpIU	OpIV	OpIW	OpIX	OpIY	OpIZ	OpJA	OpJB	OpJC	OpJD	OpJE	OpJF	OpJG	OpJH	OpJI	OpJJ	OpJK	OpJL	OpJM	OpJN	OpJO	OpJP	OpJQ	OpJR	OpJS	OpJT	OpJU	OpJV	OpJW	OpJX	OpJY	OpJZ	OpKA	OpKB	OpKC	OpKD	OpKE	OpKF	OpKG	OpKH	OpKI	OpKJ	OpKK	OpKL	OpKM	OpKN	OpKO	OpKP	OpKQ	OpKR	OpKS	OpKT	OpKU	OpKV	OpKW	OpKX	OpKY	OpKZ	OpLA	OpLB	OpLC	OpLD	OpLE	OpLF	OpLG	OpLH	OpLI	OpLJ	OpLK	OpLL	OpLM	OpLN	OpLO	OpLP	OpLQ	OpLR	OpLS	OpLT	OpLU	OpLV	OpLW	OpLX	OpLY	OpLZ	OpMA	OpMB	OpMC	OpMD	OpME	OpMF	OpMG	OpMH	OpMI	OpMJ	OpMK	OpML	OpMM	OpMN	OpMO	OpMP	OpMQ	OpMR	OpMS	OpMT	OpMU	OpMV	OpMW	OpMX	OpMY	OpMZ	OpNA	OpNB	OpNC	OpND	OpNE	OpNF	OpNG	OpNH	OpNI	OpNJ	OpNK	OpNL	OpNM	OpNN	OpNO	OpNP	OpNQ	OpNR	OpNS	OpNT	OpNU	OpNV	OpNW	OpNX	OpNY	OpNZ	OpOA	OpOB	OpOC	OpOD	OpOE	OpOF	OpOG	OpOH	OpOI	OpOJ	OpOK	OpOL	OpOM	OpON	OpOO	OpOP	OpOQ	OpOR	OpOS	OpOT	OpOU	OpOV	OpOW	OpOX	OpOY	OpOZ	OpPA	OpPB	OpPC	OpPD	OpPE	OpPF	OpPG	OpPH	OpPI	OpPJ	OpPK	OpPL	OpPM	OpPN	OpPO	OpPP	OpPQ	OpPR	OpPS	OpPT	OpPU	OpPV	OpPW	OpPX	OpPY	OpPZ	OpQA	OpQB	OpQC	OpQD	OpQE	OpQF	OpQG	OpQH	OpQI	OpQJ	OpQK	OpQL	OpQM	OpQN	OpQO	OpQP	OpQQ	OpQR	OpQS	OpQT	OpQU	OpQV	OpQW	OpQX	OpQY	OpQZ	OpRA	OpRB	OpRC	OpRD	OpRE	OpRF	OpRG	OpRH	OpRI	OpRJ	OpRK	OpRL	OpRM	OpRN	OpRO	OpRP	OpRQ	OpRR	OpRS	OpRT	OpRU	OpRV	OpRW	OpRX	OpRY	OpRZ	OpSA	OpSB	OpSC	OpSD	OpSE	OpSF	OpSG	OpSH	OpSI	OpSJ
----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------

```

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          .EVEN
405

```

```

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
  
```

TSV5 - HARDWARE TESTS MACRO M1113 25-MAY-82 09:19 PAGE 85
TEST 2: BASIC WRITE SUBSYSTEM MEMORY COMMAND

D 11

SEQ 0133

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

```

012700 030450
004737 016510
012737 000012
005237 003150
004737 021276

```

002216

```

486 .SBTTL TEST 3: SUBTEST 1: GET STATUS SELECTED LOCATIONS
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514 026446          BGNSUB                      ;////////// BEGIN SUBTEST //////////
      026446          T3.1:                      TRAP      CSBSUB
      026446 104402
515
516
517
518 026450 004737 015774 ;Write to TSSR to soft initialize
519 026454 103405        JSR      PC,SOFINIT      ;DO SOFT INIT OF CONTROLLER
520 026456        BCS      15$                    ;BR IF SOFT INIT = OK
521 026456 010001        NEXT.ERRNO
522 026460        MOV      R0,R1                    ;SAVE CONTENTS OF TSSR
      026460 104455        ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      026462 000455        TRAP      CSERDF
      026464 003652        .WORD    301
      026466 012034        .WORD    SFIERR
                        .WORD    SFIMSG
523
524
525 026470        ;Do a WRITE CHARACTERISTICS to setup a message buffer
526 026470 012704 030240 15$:
527 026474 004737 031620      MOV      #T12PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
528 026500 005037 003134      JSR      PC,T12SWRT         ;RESTORE PACKET TO STARTING VALUES
529 026504 010465 000000      CLR      KTENABLE          ;TURN OFF KT-11
530 026510 004737 016336      MOV      R4,TSDB(R5)         ;SET THE PACKET ADDRESS
531 026514        JSR      PC,CHKTSSR                    ;WAIT FOR SSR TO SET
532 026530 103405        FORCERROR 17$
533 026532 010001        BCS      20$                    ;BR IF SSR SET IN CHKTSSR
534 026534        MOV      R0,R1                    ;SAVE CONTENTS OF TSSR
535 026534 104455        NEXT.ERRNO
      026534        ERRDF   ERRNO,T12WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                        TRAP      CSERDF

```

PC	PC+4	PC+8	PC+12	PC+16	PC+20	PC+24	PC+28	PC+32	PC+36	PC+40	PC+44	PC+48	PC+52	PC+56	PC+60	PC+64	PC+68	PC+72	PC+76	PC+80	PC+84	PC+88	PC+92	PC+96	PC+100	PC+104	PC+108	PC+112	PC+116	PC+120	PC+124	PC+128	PC+132	PC+136	PC+140	PC+144	PC+148	PC+152	PC+156	PC+160	PC+164	PC+168	PC+172	PC+176	PC+180	PC+184	PC+188	PC+192	PC+196	PC+200	PC+204	PC+208	PC+212	PC+216	PC+220	PC+224	PC+228	PC+232	PC+236	PC+240	PC+244	PC+248	PC+252	PC+256	PC+260	PC+264	PC+268	PC+272	PC+276	PC+280	PC+284	PC+288	PC+292	PC+296	PC+300	PC+304	PC+308	PC+312	PC+316	PC+320	PC+324	PC+328	PC+332	PC+336	PC+340	PC+344	PC+348	PC+352	PC+356	PC+360	PC+364	PC+368	PC+372	PC+376	PC+380	PC+384	PC+388	PC+392	PC+396	PC+400	PC+404	PC+408	PC+412	PC+416	PC+420	PC+424	PC+428	PC+432	PC+436	PC+440	PC+444	PC+448	PC+452	PC+456	PC+460	PC+464	PC+468	PC+472	PC+476	PC+480	PC+484	PC+488	PC+492	PC+496	PC+500	PC+504	PC+508	PC+512	PC+516	PC+520	PC+524	PC+528	PC+532	PC+536	PC+540	PC+544	PC+548	PC+552	PC+556	PC+560	PC+564	PC+568	PC+572	PC+576	PC+580	PC+584	PC+588	PC+592	PC+596	PC+600	PC+604	PC+608	PC+612	PC+616	PC+620	PC+624	PC+628	PC+632	PC+636	PC+640	PC+644	PC+648	PC+652	PC+656	PC+660	PC+664	PC+668	PC+672	PC+676	PC+680	PC+684	PC+688	PC+692	PC+696	PC+700	PC+704	PC+708	PC+712	PC+716	PC+720	PC+724	PC+728	PC+732	PC+736	PC+740	PC+744	PC+748	PC+752	PC+756	PC+760	PC+764	PC+768	PC+772	PC+776	PC+780	PC+784	PC+788	PC+792	PC+796	PC+800	PC+804	PC+808	PC+812	PC+816	PC+820	PC+824	PC+828	PC+832	PC+836	PC+840	PC+844	PC+848	PC+852	PC+856	PC+860	PC+864	PC+868	PC+872	PC+876	PC+880	PC+884	PC+888	PC+892	PC+896	PC+900	PC+904	PC+908	PC+912	PC+916	PC+920	PC+924	PC+928	PC+932	PC+936	PC+940	PC+944	PC+948	PC+952	PC+956	PC+960	PC+964	PC+968	PC+972	PC+976	PC+980	PC+984	PC+988	PC+992	PC+996	PC+1000	PC+1004	PC+1008	PC+1012	PC+1016	PC+1020	PC+1024	PC+1028	PC+1032	PC+1036	PC+1040	PC+1044	PC+1048	PC+1052	PC+1056	PC+1060	PC+1064	PC+1068	PC+1072	PC+1076	PC+1080	PC+1084	PC+1088	PC+1092	PC+1096	PC+1100	PC+1104	PC+1108	PC+1112	PC+1116	PC+1120	PC+1124	PC+1128	PC+1132	PC+1136	PC+1140	PC+1144	PC+1148	PC+1152	PC+1156	PC+1160	PC+1164	PC+1168	PC+1172	PC+1176	PC+1180	PC+1184	PC+1188	PC+1192	PC+1196	PC+1200	PC+1204	PC+1208	PC+1212	PC+1216	PC+1220	PC+1224	PC+1228	PC+1232	PC+1236	PC+1240	PC+1244	PC+1248	PC+1252	PC+1256	PC+1260	PC+1264	PC+1268	PC+1272	PC+1276	PC+1280	PC+1284	PC+1288	PC+1292	PC+1296	PC+1300	PC+1304	PC+1308	PC+1312	PC+1316	PC+1320	PC+1324	PC+1328	PC+1332	PC+1336	PC+1340	PC+1344	PC+1348	PC+1352	PC+1356	PC+1360	PC+1364	PC+1368	PC+1372	PC+1376	PC+1380	PC+1384	PC+1388	PC+1392	PC+1396	PC+1400	PC+1404	PC+1408	PC+1412	PC+1416	PC+1420	PC+1424	PC+1428	PC+1432	PC+1436	PC+1440	PC+1444	PC+1448	PC+1452
----	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------

TSV5 - HARDWARE TESTS MACRO M1113 25-MAY-82 09:19 PAGE 87-2 H 11
TEST 3: SUBTEST 1: GET STATUS SELECTED LOCATIONS

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 .SBTTL TEST 3: SUBTEST 2: MESSAGE PACKETS TO SELECTED LOCATIONS
589
590 :++
591 :TEST 3: SUBTEST 2:
592 :SUBTEST DESCRIPTION:
593 :
594 :   This subtest verifies the controller can deposit message packets
595 :   to all available memory locations.
596 :   Write Characteristics commands are executed with message
597 :   buffer addresses set to various addresses in each available
598 :   memory location.
599 :   The various addresses are determined by floating a 1 then a 0
600 :   through the address bits.
601 :
602 :TEST STEPS:
603 :
604 :   BEGIN
605 :   Write to TSSR to soft initialize
606 :   Do a WRITE CHARACTERISTICS to setup a message buffer to compare
607 :
608 :   REPEAT FOR SELECTED ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
609 :   BEGIN
610 :   Get a valid modulo-4 test address
611 :   Set the packet message buffer to the TEST ADDRESS
612 :   Do a WRITE CHARACTERISTICS
613 :   Restore the test message buffer to background pattern
614 :   END
615 :   END
616 :--
617
618 027006 BGNSUB ;////////// BEGIN SUBTEST ///////////
619 027006 104402 T3.2: TRAP C$BSUB
620
621 :Write to TSSR to soft initialize
622 027010 004737 015774 JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
623 027014 103405 BCS 15$ ;BR IF SOFT INIT = OK
624 027016 NEXT.ERRNO
625 027016 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
626 027020 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
627 027020 104455 TRAP C$ERDF
628 027022 000460 .WORD 304
629 027024 003652 .WORD SFIERR
630 027026 012034 .WORD SFIMSG
631
632 :Do a WRITE CHARACTERISTICS to setup a message buffer to compare
633 15$:
634 027030 012704 030240 MOV #T12PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
635 027034 004737 031620 JSR PC,T12SWRT ;SET PACKET TO WRITE CHARACTERISTICS
636 027040 004737 017274 JSR PC,KTOFF ;TURN OFF KT-11
637 027044 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS
638 027050 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
639 027054 FORCERROR 17$
640 027070 103405 BCS 20$ ;BR IF SSR SET IN CHKTSSR
641 027072 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
642 027074 NEXT.ERRNO

```

Line	Address	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464	Op46
------	---------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

```

682 .SBTTL TEST 3: SUBTEST 3: CHARACTERISTIC DATA SELECTED LOCATIONS
683
684 ++
685 TEST 3: SUBTEST 3:
686
687 SUBTEST DESCRIPTION:
688
689 This subtest verifies the controller can fetch a
690 Write Characteristics data block from all available
691 memory locations.
692 Write Characteristics commands are executed with
693 characteristic data blocks at various memory addresses.
694 The various memory addresses are determined by floating
695 a 1 then a 0 through the address bits.
696
697 TEST STEPS:
698
699 BEGIN
700 Write to TSSR to soft initialize
701
702 REPEAT FOR SELECTED VALID ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
703 BEGIN
704 Get a valid test address
705 Set the test packet characteristics data pointer to the
706 test address.
707 Store expected characteristic data in test address block
708 Do a WRITE CHARACTERISTIC command
709 END
710 END
711 --
712 BGNSUB
713
714 ;//////////////// BEGIN SUBTEST //////////////////
715 T3.3:
716 TRAP CSBSUB
717
718 ;Write to TSSR to soft initialize
719 JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
720 BCS 20$ ;BR IF SOFT INIT = OK
721 NEXT.ERRNO
722 MOV R0,R1 ;SAVE CONTENTS OF TSSR
723 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
724
725 TRAP CSERDF
726 .WORD 307
727 .WORD SFIERR
728 .WORD SFIMSG
729
730 ;Get a valid test address
731 20$: CLR FATFLG ;CLEAR FATAL ERROR FLAG
732 CLR T12KT ;TEST ABOVE 28K SWITCH
733 MOV #T12BLK,R3 ;POINT TO TEST PATTERN TABLE
734
735 T123LOOP:
736 CLR KTENABLE ;TURN OFF ABOVE 28K TEST FLAG
737 MOV (R3)+,R1 ;GET TEST PATTERN ADDRESS
738 MOV R1,R0 ;GET ADDRESS ALL '18 BITS'
739 BIC #177774,R0 ;LEAVE ONLY A17 AND A16
740 BIC #3,R1 ;GET RID OF A17 AND A16
741 TST T12KT ;TEST ABOVE 28K THIS TIME?
  
```

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	MOV	#1,KTENABLE	:TURN ON ABOVE 28K TEST FLAG
737	027400	004737	031316	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	MOV	T12LOADD,PKLOW(R4)	:STORE CHAR. DATA PTR LOW ADDRESS
744	027430	013764	030302	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
	027476	104455				TRAP C\$ERDF
	027500	000464				.WORD 308
	027502	030552				.WORD T12WRTSSR
	027504	012046				.WORD PKTSSR
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		BHIS	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	KTF LG	:ANY MEMORY ABOVE 28K ON SYSTEM?
763	027534	001407		BEQ	70\$	:BR IF NO
764	027536	012737	000001	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 \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
	027560					L10045:
	027560	104403				TRAP C\$ESUB
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			75\$:		
773						

```

775 .SBTTL TEST 3: SUBTEST 4: NXM TO SELECTED INVALID ADDRESSES
776
777 **
778 TEST 3: SUBTEST 4:
779 SUBTEST DESCRIPTION:
780
781 This subtest verifies the NXM error bit in the TSSR
782 register is set when attempting to fetch data (a characteristic
783 data block) from selected nonexistent locations.
784 If NXM fails to set it is likely that an LSI-11 Bus driver is
785 failing to assert an address line.
786 Addresses tested include all combinations of high-order address
787 bits (i.e bits 16-21).
788 *****
789 CAUTION
790
791 The LSI BUS drivers for all available address lines(16-21)
792 are only checked when running on a 11/23B system with more than
793 128K words of memory!
794 *****
795
796 TEST STEPS:
797
798 BEGIN
799 Write to TSSR to soft initialize
800 Do a write characteristic command
801 Invert the extended features switch
802
803 REPEAT FOR SELECTED NON-EXISTENT MEMORY ADDRESSES
804 BEGIN
805 Get an invalid test address
806 Set the test packet characteristics data pointer to the
807 test address.
808 Do a WRITE CHARACTERISTIC command
809 If TSSR register NXM bit not set then print error message
810 END
811 END
812
813
814 BGNSUB
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

Line	Address	Operation	Comments	Trap	Word
828	027624	003652			SFIERR
829	027626	012034			SFIMSG
830			;Do a WRITE CHARACTERISTIC command so to invert switch		
831	027630		11\$: CKLOOP	;LOOP IF SELECTED	
832	027630	104406		TRAP	C\$CLP1
833	027632	012704	030240		
834	027636	004737	031620		
835	027642	005037	003134		
836	027646	010465	000000		
837	027652	004737	016336		
838	027656				
839	027672	103405			
840	027674	010001			
841	027676				
842	027676	104455			
843	027700	000466			
844	027702	030552			
845	027704	012046			
846	027706				
847	027710	104406	021206		
848	027714	005037	002222		
849	027720	013737	003142	030302	
850	027726	013737	003140	030304	
851	027734				
852					
853					
854					
855					
856	027734	012704	030240		
857	027740	004737	031620		
858	027744	013764	030304	000002	
859	027752	013764	030302	000004	
860					
861					
862	027760	004737	017274		
863	027764	010465	000000		
864	027770	004737	016250		
865	027774				
866	030010	103407			
867	030012	010001			
868	030014				
869	030014				
870	030024	005237	002222		
871	030030				
	030030	104406			

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		000000			
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	

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	T12TBE: .WORD	177777

;+
 ;LOCAL TEXT MESSAGES FOR TEST
 ;-

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:
R0 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 KFLG ;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 000241      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
;      MOV      #RMPKTBEG,R2
;      CLR      R3
;      JSR      PC,CHKTSSR
;      MOVB     #0,TSDB(R5)
;      JSR      PC,CHKTSSR
;      MOV      R2,TSDB(R5)
;      JSR      PC,CHKTSSR
;      MOVB     TSBA(R5),(R1)
;      CPB      (R1)+,(R4)+
;      BEQ      20$
;      INC      R3
;      INC      R2
;      CMP      R2,#RMPKTBEG+2
;      BLT      10$
;      TST      R3
;      BEQ      30$
;      CLC
;      BR      50$
;      SEC
;      MOV      #2,RAMSIZ
;      RTS      PC
;
;      ;SAVE THE GENERAL REGISTERS
;      ;ADDRESS TO SAVE THE RAM DATA
;      ;BYTE ADDRESS OF FIRST RAM DATA
;      ;CLEAR THE ERROR FLAG
;      ;WAIT FOR SSR
;      ;SET MAINTENANCE MODE
;      ;WAIT FOR SSR TO SET
;      ;SELECT NEXT RAM ADDRESS
;      ;WAIT FOR SSR TO SET
;      ;READ THE RAM DATA
;      ;COMPARE TO EXPECTED
;      ;BRANCH IF OK
;      ;SET ERROR FLAG
;      ;ADDRESS OF NEXT RAM LOCATION
;      ;DONE 2 BYTES?
;      ;BR IF NO
;      ;WAS AN ERROR FOUND ?
;      ;BRANCH IF NOT
;      ;CLEAR CARRY TO SHOW ERROR
;      ;AND EXIT
;      ;SHOW GOOD COMPARE
;      ;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 T12LOAD,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 C\$ETST

TSV5 - HARDWARE TESTS MACRO M1113 25-MAY-82 09:19 PAGE 92
TEST 3: SUBTEST 4: NXM TO SELECTED INVALID ADDRESSES

I 12

SEQ 0151

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$: MOV #TST15ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
1202 032012 004737 016510 JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
1203 032016 012737 000005 002216 MOV #5,LOOPCNT ;PERFORM 5 ITERATIONS
1204 032024
1205
1206 :TEST 4, SUBTEST 1
1207
1208 :THIS SUBTEST WRITES THE ADDRESS (8 BITS) INTO THE
1209 :RAM MEMORY SINGLE WORD (8 BITS) MODE
1210
1211
1212
1213 032024 BGNSUB ;////////// BEGIN SUBTEST //////////
1214 032024 104402 T4.1:
1215 032026 012700 000000 SETPRI #PRI00 ;LOWER PRIORITY TO ALLOW INTERRUPTS
1216 032032 104441 TRAP C$BSUB
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 CHARACTERISTICS FAILED
1234 032114 000622 TRAP C$ERHRD
1235 032116 005056 .WORD 402
1236 032122 012703 000400 23$: MOV #256.,R3 ;STARTING ADDRESS FOR RAM WRITE

```

1237	032126	112737	000001	034061	MOV	#1,T15BS1	:SIZE OF TRANSFER	
1238	032134	112737	000002	034060	MOV	#2,T15BS0	:WRITE RAM 'COMMAND'	
1239	032142							
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		MOV	R3,T15S3	:DATA FOR WRITE (ADDRESS)	
1243	032156	010465	000000		MOV	R4,TSD8(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	
	032172	104456					TRAP	C\$ERHRD
	032174	000623					.WORD	403
	032176	034066					.WORD	T15SSR
	032200	012046					.WORD	PKTSSR
1251	032202				ESCAPE	SUB	:DON'T CONTINUE IF ERROR ON WRITE	
	032202	104410					TRAP	C\$ESCAPE
	032204	000122					.WORD	L10050-
1252	032206				CKLOOP		:SCOPE LOOP	
	032206	104406					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		MOV	R3,T15S3	:SET DATA PATTERN BACK IN SHAPE	
1261	032230	010337	034062		MOV	R3,T15S2	:ADDRESS FOR RAM READ	
1262	032234	112737	000001	034060	MOV	#1,T15BS0	:READ RAM COMMAND	
1263	032242	010465	000000		MOV	R4,TSD8(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	
	032256	104455					TRAP	C\$ERDF
	032260	000624					.WORD	404
	032262	034066					.WORD	T15SSR
	032264	012046					.WORD	PKTSSR
1271	032266				CKLOOP		:SCOPE LOOP	
	032266	104406					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	
	032302	104456					TRAP	C\$ERHRD
	032304	000625					.WORD	405
	032306	034345					.WORD	T15AM4
	032310	015502					.WORD	EXPBREC
1280	032312				CKLOOP		:SCOPE LOOP	
	032312	104406					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				CKLOOP		:SCOPE LOOP	
	032324	104406					TRAP	C\$CLP1

TSVS - HARDWARE TESTS MACRO M1113 25-MAY-82 09:19 PAGE 93-2 L 12
TEST 4: RAM EXERCISER TEST

1285 032326
032326
032326 104403
1286

ENDSUB

SEQ 0154

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

```

1288
1289 032330          BGNSUB          :////////// BEGIN SUBTEST //////////
      032330          T4.2:          TRAP      C$BSUB
      032330 104402
1290
1291          :+
1292          :
1293          :TEST 4, SUBTEST 2
1294          :
1295          :
1296          :
1297          :
1298          :
1299 032332 004737 034452      JSR      PC,T15REST      :RESTORE PACKET FOR WRITE CHARA
1300 032336 004737 034524      JSR      PC,T15RT2       :RESTORE PACKET FOR WRT SUB SYS MEM
1301 032342 004737 015774      JSR      PC,SOFINIT      :DO INITIALIZE ON CONTROLLER
1302 032346 103405      BCS      20$      :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      C$ERDF
      032354 000626          .WORD      406
      032356 003652          .WORD      SFIERR
      032360 012034          .WORD      SFIMSG
1308 032362
1309 032362 012704 033350      20$:      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      C$ERHRD
      032400 000627          .WORD      407
      032402 005056          .WORD      WRTMSG
      032404 012034          .WORD      SFIMSG
1317 032406
1318 032406 112737 000001 034061      25$:      MOVB      #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      MOVB      #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      C$ERHRD
      032460 000630          .WORD      408
      032462 034066          .WORD      T15SSR
      032464 012046          .WORD      PKTSSR
1332 032466          33$:      CKLOOP          :SCOPE LOOP
      032466 104406          TRAP      C$CLP1
1333
1334
1335 032470 005203          INC      R3      :NEXT ADDRESS
1336 032472 020327 010000      CMP      R3,#10000      :END OF RAM MEMORY CHECK
1337 032476 001357          BNE      30$      :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,TSD8(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,TSD8(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,TSD8(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					.WORD EXPBREC	
1387	032706			45\$:	CKLOOP	:SCOPE LOOP		

SEQ 0157

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

```

TEST 4: RAM EXERCISER TEST

1395 032722          BGNSUB          ;////////// BEGIN SUBTEST ///////////
      032722          T4.3:          TRAP      C$BSUB
      032722 104402

1396          :+
1397          :
1398          :TEST 4, SUBTEST 3
1399          :
1400          :
1401          :
1402          :
1403          :
1404 032724 005737 003400          THIS SUBTEST WRITES RAM WITH ALL ONES
1405 032730 001402          THEN WALKS AN ALL ZEROS WORD DOWN THROUGH MEMORY
1406 032732 000137 033326          TST      SKIP
1407 032736 004737 034452          BEQ      10$
1408 032742 004737 034524          JMP      50$
1409 032746 004737 015774          JSR      PC,T15REST
1410 032752 103405          JSR      PC,T15RT2
1414 032754 010001          JSR      PC,SOFINIT
1415 032756          BCS      20$
      032756 104455          MOV      R0,R1
      032760 000636          ERRDF    ERRNO,SFIERR,SFIMSG
      032762 003652          ;CHECK RUN MODE
      032764 012034          ;BR, IF NO SKIP
1416 032766          ;SKIP SUBTEST
1417 032766 012704 033350          ;RESTORE PACKET FOR WRITE CHARA
1418 032772 004737 010662          ;RESTORE PACKET FOR WRT SUB SYS MEM
1419 032776 103405          ;DO INITIALIZE ON CONTROLLER
1423 033000 010001          ;BR IF INIT WAS OK
1424 033002          ;CONTENTS OF TSSR REGISTER
      033002 104456          ;FATAL ERROR TSSR WAS NOT OK
      033004 000637          TRAP      C$ERDF
      033006 005056          .WORD    414
      033010 012034          .WORD    SFIERR
1425 033012          .WORD    SFIMSG
1426 033012 112737 000001 034061 20$: MOV      #T15PACKET,R4
1427 033020 012704 034050          JSR      PC,WRTCHR
1428 033024 012703 000400          BCS      25$
1429 033030 112737 000002 034060          MOV      R0,R1
1430 033036 112737 000377 034064          ERRHRD  ERRNO,WRTMSG,SFIMSG
1431 033044 010337 034062          ;SUBROUTINE NEEDS PACKET ADDRESS
1432 033050 010465 000000          ;ISSUE WRITE CHARACTERISTICS
1433 033054 004737 016336          ;BR, IF COMMAND ISSUED OK
1434 033060 103405          ;SAVE CONTENTS OF TSSR
1435 033062 010001          ;WRITE CHARACTERISTICS FAILED
1439 033064          TRAP      C$ERHRD
      033064 104456          .WORD    415
      033066 000640          .WORD    WRTMSG
      033070 034066          .WORD    SFIMSG
      033072 012046          25$: MOV      #1,T15BS1
1440 033074 104406          30$: MOV      #T15PK2,R4
1441          MOV      #256,R3
1442          MOV      #2,T15BS0
1443 033076 005203          MOV      #377,T15S3
1444 033100 020327 010000          30$: MOV      R3,T15S2
1445 033104 001357          MOV      R4,TSD8(R5)
          MOV      JSR      PC,CHKTSSR
          BCS      33$
          MOV      R0,R1
          ERRHRD  ERRNO,T15SSR,PKTSSR
          ;TSSR NOT CORRECT
          TRAP      C$ERHRD
          .WORD    416
          .WORD    T15SSR
          .WORD    PKTSSR
          33$: CKLOOP          ;SCOPE LOOP
          TRAP      C$CLP1
          INC      R3
          CMP      R3,#10000
          BNE      30$
          ;NEXT ADDRESS
          ;END OF RAM MEMORY CHECK
          ;BR, MORE RAM TO GO

```

```
TEST 4: RAM EXERCISER TEST

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	005303		DEC	R3
1496	033320	020327	000377	CMP	R3, #255.
1497	033324	001271		BNE	40\$
1498					
1499	033326		50\$:		
1500	033326			ENDSUB	
	033326				
	033326	104403			

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

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

```
TEST 4: RAM EXERCISER TEST

1502
1503 033330 004737 016456          JSR    PC,TSTLOOP
1504 033334 103002                BCC    63$
1505 033336 000137 032024          JMP    T15LOOP
1506 033342                63$:   EXIT    TST
      033342 104432
      033344 001216
1507
1508
1509
1510
1512                033350
1514 033350                T15PACKET:  .=<.+10>8177770
1515 033350 100204                .WORD  100204
1516 033352 033360                .WORD  T15DATA
1517 033354 000000                .WORD  0
1518 033356 000010                .WORD  8.
1519 033360                T15DATA:
1520 033360 033372                .WORD  T15BFR
1521 033362 000000                .WORD  0
1522 033364 000400                .WORD  256.
1523 033366 000000 000000        .WORD  0,0
1524 033372                T15BFR: .BLKW  150.
1525
1526                ;WRITE SUBSYSTEM MEMORY COMMAND PACKET
1527
1529                034050
1531 034050                T15PK2:   .=<.+10>8177770
1532 034050 100206                .WORD  100206
1533 034052 034060                .WORD  T15BF2
1534 034054 000000                .WORD  0
1535 034056 000006                .WORD  6.
1536
1537                .EVEN
1538 034060                T15BF2:
1539 034060                T15BS0: .BYTE  0
1540 034061                T15BS1: .BYTE  0
1541 034062 000000                T15S2: .WORD  0
1542 034064 000000                T15S3: .WORD  0
1543
1544
1545
```

```
;DO WE NEED TO ITERATE TEST ?
;BRANCH IF NOT
;EXECUTE AGAIN
;ALL DONE THIS TEST

TRAP    C$EXIT
.WORD    L10047-.

;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
```

```

1547
1548
1549      ;+
1550      ;LOCAL TEXT MESSAGES FOR TEST
1551      ;--
1552 034066      127      122      111  T15SSR: .ASCIIZ 'WRITE SUBSYSTEM MEMORY Command Not Accepted'
1553 034142      127      122      111  T15AM2: .ASCIIZ 'WRITE SUBSYSTEM MEMORY COMMAND Failed On All Ones Word Read Back'
1554 034243      127      122      111  T15AM3: .ASCIIZ 'WRITE SUBSYSTEM MEMORY COMMAND Failed On All Zeros Word Read Back'
1555 034345      127      122      111  T15AM4: .ASCIIZ 'WRITE SUBSYSTEM MEMORY COMMAND Failed On Address Test'
1556 034433      122      101      115  TST15ID: .ASCIIZ '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      T15REST:
1566      SAVREG
1567      MOV      #T15PACKET,R1
1568      MOV      #100204,(R1)+
1569      MOV      #T15DATA,(R1)+
1570      CLR      (R1)+
1571      MOV      #8,(R1)+
1572      MOV      #T15BFR,(R1)+
1573      CLR      (R1)+
1574      MOV      #256,(R1)+
1575      CLR      (R1)+
1576      CLR      (R1)
1577      CLR      T15BFR
1578      RTS      PC
1579
1580
1581      T15RT2:
1582      SAVREG
1583      MOV      #T15PK2,R1
1584      MOV      #100206,(R1)+
1585      MOV      #T15BF2,(R1)+
1586      CLR      (R1)+
1587      MOV      #6,(R1)+
1588      CLR      (R1)+
1589      CLR      (R1)+
1590      CLR      (R1)
1591      RTS      PC
1592      ENDTST
1593
1594      L10047:
1595      TRAP      CSETST
  
```

```

1595 .SBTTL TEST 5: EXTENDED FEATURES SWITCH AND TIMERS A,B
1596
1597 :++
1598 : TEST DESCRIPTION:
1599 :
1600 : This test verifies the Invert Extended Features function
1601 : can logically invert the Extended features switch and
1602 : that the internal timers A and B operate correctly.
1603 :
1604 : TEST STEPS:
1605 :
1606 : REPEAT FOR LOOPCNT
1607 : BEGIN
1608 : Do Subtest 1 - Verify Extended Features Switch
1609 : Do Subtest 2 - Verify Timers A,B
1610 : END
1611 :--
1612
1613 034564 BGNTST
1614 034564
1618 034564 012700 036642 MOV #TST16ID,RO ;ASCII MESSAGE TO IDENTIFY TEST
1619 034570 004737 016510 JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
1620 034574 012737 000012 002216 MOV #10.,LOOPCNT ;PERFORM 10 ITERATIONS
1621 034602
1622 T16LOOP:
  
```

```

1624 .SBTTL TEST 5: SUBTEST 1: VERIFY EXTENDED FEATURES TEST
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672 034602
      034602
      034602 104402
1673
1674
1675 034604
1676
1677 034604 004737 015774
1678 034610 103405

      **
      TEST 5: SUBTEST 1:
      SUBTEST DESCRIPTION:
          This subtest verifies that the Invert Sense of Extended features
          Switch function (Write Subsystem Memory,Write Misc command)
          operates properly.
          First the state of the Extended Features switch is read in the
          message packet supplied by the write characteristics command.
          Then, the sense of the switch is logically inverted.
          A Write characteristics command is executed and it is verified
          that the Extended status register (XST4) is returned when
          in Extended mode, and not returned if not in extended mode.
          The subtest also verifies that specifying a Message Buffer
          address with any of bits 21-19 ,set will cause the command to
          be rejected.

      TEST STEPS:

      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
              (* Verify Extended Features switch can be Inverted to SET *)
              Do Write Subsystem Write Miscellaneous to SET Extended Features.
              DO a WRITE CHARACTERISTICS with an extended characteristic word
              Compare the controller ram to the extended characteristic word
              If Data word in controller ram NOT= to word sent Then Print Error
              If Message Buffer Data Length NOT= 12. Then Print Error
          ELSE
              (* Verify Extended Features switch can be Inverted to CLEAR *)
              Do Write Subsystem Write Miscellaneous to CLEAR Extended Features.
              Do a WRITE CHARACTERISTICS without an extended characteristic word
              If Message Buffer Data Length NOT= 10. Then Print Error
          END-IF
          (* Verify Function Reject when Message Buffer 21-19 are non-zero *)
          Write to TSSR register to soft initialize the controller
          REPEAT FOR MESSAGE BUFFER ADDRESS bits <21:19> FROM 0 TO 7
          DO a WRITE CHARACTERISTICS with a message address bit<21:19> non-zero
          If TSSR termination code NOT= Function Reject Then Print Error
          END-REPEAT
      END
      --
      BGNSUB
          :////////// BEGIN SUBTEST //////////
          T5.1:
          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
  
```

1679	034612	010001			MOV R0,R1	;SAVE CONTENTS OF TSSR	
1680	034614				ERRDF ERRNO,SFIERR,SFIMSG	;DEVICE FATAL DURING INIT	
	034614	104455				TRAP	C\$ERDF
	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	C\$ERDF
	034666	000765				.WORD	501
	034670	036712				.WORD	T16SSR
	034672	012046				.WORD	PKTSSR
1691	034674	005237	002222		INC FATFLG	;SET FATAL ERROR FLAG	
1692	034700			15\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET	
	034700	104406				TRAP	C\$CLP1
1693					If Extended Features Hardware Switch Clear then:		
1694				:	(* Verify Extended Features switch can be Inverted to SET *)		
1695				:	REPEAT FOR TEST PATTERNS IN TSTBLK TABLE		
1696				:			
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,TSD8(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	C\$ERDF
	034770	000766				.WORD	502
	034772	036747				.WORD	T162SSR
	034774	012046				.WORD	PKTSSR
1713	034776	005237	002222		INC FATFLG	;SET FATAL ERROR FLAG	
1714	035002			40\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET	
	035002	104406				TRAP	C\$CLP1
1715					DO a WRITE CHARACTERISTICS with an extended characteristic word		
1716				:			
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	62\$:	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			72\$:	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			80\$:	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			92\$:	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			100\$:	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			112\$:	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			120\$:	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      240$: INC      FATFLG      ;SET FATAL ERROR FLAG
1779 035342      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      035342 104406      TRAP      C$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      250$: INC      FATFLG      ;SET FATAL ERROR FLAG
1791 035416      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      035416 104406      TRAP      C$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
  
```

Address	Offset	Value	Label	Operation	Comment	Word	Length
1799	035440	037162				.WORD	T16LEN
	035442	015474				.WORD	EXPREC
	035444		270\$:	CKLOOP	;LOOP ON ERROR, IF FLAG	SET	
	035444	104406				TRAP	C\$CLP1
1800							
1801							
1802							
1803							
1804	035446						
1805							
1806	035446	012737	000001	002312			
1807							
1808	035454						
1809	035454	012704	040170				
1810	035460	012764	000016	000006			
1811	035466	013700	002312				
1812		000003					
1813							
1814							
1815	035500	010037	040202				
1816	035504	010465	000000				
1817	035510	004737	016250				
1818	035514						
1819	035530	103407					
1820	035532	010001					
1821	035534						
1822	035534						
	035534	104455					
	035536	000776					
	035540	036712					
	035542	012046					
1823	035544	005237	002222				
1824	035550						
	035550	104406					
1825							
1826							
1827	035552	016501	000002				
1828	035556	010102					
1829	035560	042702	000016				
1830	035564	052702	000006				
1831	035570						
1832	035600	020102					
1833	035602	001404					
1834	035604						
1835	035604						
	035604	104456					
	035606	000777					
	035610	037274					
	035612	012046					
1836	035614						
	035614	104406					
1837	035616						
1838	035626	005237	002312				
1839	035632	023727	002312	000007			
1840	035640	101002					
1841	035642	000137	035454				
1842							

1843 035646
1844 035646
035646
035646

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

.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:

Write to TSSR register to soft initialize the controller TRAP CSBSUB

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\$;ADDFORCE ERROR IF FORCER=1
 BCS 15\$;BR IF CARRY SET (GOOD RETURN)
 MOV R0,R1 ;SAVE CONTENTS OF TSSR
 NEXT.ERRNO

035662
035662 104402
035662 004737 015774
035664 103405
035670 010001
035672 104455
035674 000777
035676 003652
035700 012034
035702
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$:  ERRDF  ERRNO,T16SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
      035752 104455                                TRAP  C$ERDF
      035754 001000                                .WORD 512
      035756 036712                                .WORD T16SSR
      035760 012046                                .WORD PKTSSR
1908 035762 005237 002222
1909 035766      15$:  INC    FATFLG                ;SET FATAL ERROR FLAG
      035766 104406      CKLOOP                ;LOOP ON ERROR, IF FLAG SET
                                          TRAP  C$CLP1

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$:  ERRDF  ERRNO,T162SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
      036040 104455                                TRAP  C$ERDF
      036042 001001                                .WORD 513
      036044 036747                                .WORD T162SSR
      036046 012046                                .WORD PKTSSR
1924 036050 005237 002222
1925 036054      40$:  INC    FATFLG                ;SET FATAL ERROR FLAG
      036054 104406      CKLOOP                ;LOOP ON ERROR, IF FLAG SET
                                          TRAP  C$CLP1

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
1938 036120      72$:  ERRHRD  ERRNO,T16T01,TIMEXP      ;REPORT ERROR
      036120 104456                                TRAP  C$ERHRD
      036122 001002                                .WORD 514
      036124 037411                                .WORD T16T01
      036126 015552                                .WORD TIMEXP
1939 036130      80$:  CKLOOP                ;LOOP ON ERROR, IF FLAG SET
      036130 104406      TRAP  C$CLP1

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 104406 120$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                        TRAP      C$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      FORCERROR 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$:      ERRHRD      ERRNO,T16T28,TIMEXP ;REPORT ERROR
                        TRAP      C$ERHRD
                        036262 104456      .WORD      516
                        036264 001004      .WORD      T16T28
                        036266 037510      .WORD      TIMEXP
1968 036272 104406 180$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                        TRAP      C$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      FORCERROR 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 104406 220$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                        TRAP      C$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$:              ERRHRD  ERRNO,T16T53,TIMEXP ;REPORT ERROR
                                TRAP      C$ERHRD
                                .WORD     518
                                .WORD     T16T53
                                .WORD     TIMEXP
                                TRAP      C$CLP1
                                036424 104456
                                036426 001006
                                036430 037610
                                036432 015552
1997 036434 280$:              CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
                                036434 104406
1998 ;                          Do a Write Control RESET TIMER with 78 microsecond delay
1999 036436 012700 000001      MOV      #MS.RSD,R0 ;RESET TIMER COMMAND
2000 036442 013701 036640      MOV      T16D78,R1 ;78 MICROSECOND DELAY
2001 036446 004737 040122      JSR      PC,T16WMISC ;SETUP T16PK2 COMMAND PACKET
2002 036452 012704 040240      MOV      #T16PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
2003 036456 010465 000000      MOV      R4,TSD8(R5) ;SET THE PACKET ADDRESS TO EXECUTE
2004 036462 004737 015336      JSR      PC,CHKTSSR ;WAIT FOR SSR TO SET
2005 036466                      FORCERROR 312$ ;@@DFORCE ERROR IF FORCER=1
2006 036502 103407              BCS      320$ ;BR IF CARRY SET (GOOD RETURN)
2007 036504 010001              MOV      R0,R1 ;SAVE CONTENTS OF TSSR
2008 036506                      NEXT.ERRNO
2009 036506 312$:              ERRDF  ERRNO,T162SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD     519
                                .WORD     T162SSR
                                .WORD     PKTSSR
                                036506 104455
                                036510 001007
                                036512 036747
                                036514 012046
2010 036516 005237 002222      INC      FATFLG ;SET FATAL ERROR FLAG
2011 036522 320$:              CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
                                036522 104406
2012 ;                          If Timer A NOT= 1 Then Print Error
2013 ;                          If Timer B NOT= 0 Then Print Error
2014 036524 005002              CLR      R2 ;INIT EXPD
2015 036526 052702 000010      BIS      #S2.ATIM,R2 ;TIMER A EXPD=1
2016 036532 042702 000004      BIC      #S2.BTIM,R2 ;TIMER B EXPD=0
2017 036536 012700 040232      MOV      #T16BFSTA,R0 ;GET RECV READ STATUS
2018 036542 016001 000002      MOV      2(R0),R1 ;GET RECV BYTE 2
2019 036546 042701 177763      BIC      #^C<S2.ATIM!S2.BTIM>,R1 ;SAVE TIMER A:B RECV ONLY
2020 036552                      FORCERROR 372$,NOTSSR ;@D
2021 036562 020201              CMP      R2,R1 ;EXPD EQUAL RECV?
2022 036564 001404              BEQ      380$ ;BR IF YES
2023 036566                      NEXT.ERRNO
2024 036566 372$:              ERRHRD  ERRNO,T16T78,TIMEXP ;REPORT ERROR
                                TRAP      C$ERHRD
                                .WORD     520
                                .WORD     T16T78
                                .WORD     TIMEXP
                                036566 104456
                                036570 001010
                                036572 037710
                                036574 015552
2025 036576 380$:              CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
                                036576 104406
2026 ;
2027 036600                      ENDSUB ;////////// END SUBTEST //////////
                                L10055:
                                TRAP      C$ESUB
                                036600 104403
2028 ;
2029 036602 005737 002222      TST      FATFLG ;ANY FATAL ERRORS ?
2030 036606 001402              BEQ      460$ ;BRANCH IF NOT

```

2031 036610 004737 017202
2032 036614 004737 016456 460\$: JSR PC,CKDROP
2033 036620 103002 JSR PC,TSTLOOP
2034 036622 000137 034602 BCC 465\$
2035 036626 465\$: JMP T16LOOP
2036
2037
2038 036626 EXIT TST
036626 104432
036630 001524
2039
2040

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

:/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:      .ASCIIZ  'Extended Features Switch and Timers A,B'
2054 036712      127      122      111  T16SSR: .ASCIIZ  'WRITE CHARACTERISTICS Failed'
2055 036747      127      122      111  T162SSR: .ASCIIZ  'WRITE SUBSYSTEM (Write Misc) Failed'
2056 037013      127      122      111  T163SSR: .ASCIIZ  'WRITE SUBSYSTEM (Read Status) Failed'
2057 037060      102      165      163  T16TSBA: .ASCIIZ  'Bus Address Register (TSBA) Incorrect after Write Characteristics'
2058 037162      104      141      164  T16LEN: .ASCIIZ  'Data Field Length in Message Buffer Incorrect after Write Characteristics'
2059 037274      124      123      123  T16REJ: .ASCIIZ  'TSSR Function Reject Not Returned When Non-Existent Buffer Address Specified'
2060 037411      124      151      155  T16T01: .ASCIIZ  'Timer A,B Incorrect after Reset Timer with 1 microsecond Delay'
2061 037510      124      151      155  T16T28: .ASCIIZ  'Timer A,B Incorrect after Reset Timer with 28 microsecond Delay'
2062 037610      124      151      155  T16T53: .ASCIIZ  'Timer A,B Incorrect after Reset Timer with 53 microsecond Delay'
2063 037710      124      151      155  T16T78: .ASCIIZ  '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,ENB,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$: CLR      (R1)+      ;CLEAR A BYTE
2091 040074 005302      DEC      R2      ;DONE?
2092 040076 003375      BGT      10$      ;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 BSEL0
2102	040116	105010		CLRB	(R0)	:CLEAR BSEL1
2103	040120	000207		RTS	PC	:RETURN
2104						
2105						
2106						
2107						
2108						
2109						
2110						
2111						
2112						
2113	040122					
2114	040122					
2115	040126	004737	040056	SAVREG		:SAVE R1-R5 UNTIL NEXT RETURN
2116	040132	012702	040250	JSR	PC,T16CLRBUF	:CLEAR MESSAGE BUFFER
2117	040136	112722	000010	MOV	#T16DT2,R2	:WRITE SUBSYSTEM DATA BUFFER
2118	040142	110022		MOVB	#PW.WMISC,(R2)+	:STORE WRITE MISCELLANEOUS IN BSEL0
2119	040144	110112		MOVB	R0,(R2)+	:STORE WRITE MISC CODE IN BSEL1
2120	040146	000207		MOVB	R1,(R2)	:STORE DELAY (RESET TIMER) IN BSEL2
2121				RTS	PC	:RETURN
2122						
2123						
2124	040150					
2125	040150	012700	040250			
2126	040154	112720	000010	MOV	#T16DT2,R0	:WRITE SUBSYSTEM DATA BUFFER
2127	040160	112710	000200	MOVB	#PW.WMISC,(R0)+	:STORE WRITE MISCELLANEOUS IN BSEL0
2128	040164	000207		MOVB	#MS.EXT,(R0)	:STORE INVERT EXTENDED FEATURES IN BSEL1
2129				RTS	PC	:RETURN
2130						
2131						
2132						
2133						
2134		040170				
2135						
2136						
2137						
2138						
2139	040170					
2140	040170	100004				
2141	040172	040200				
2142	040174	000000				
2143	040176	000012				
2144						
2145	040200					
2146	040200	040212				
2147	040202	000000				
2148	040204	000024				
2149	040206	000000				
2150	040210	000000				
2151						
2152						
2153						
2154						
2155	040212					
2156	040212	000000				
2157	040214	000000				

```

:
:  SETUP T16PK2 PACKET FOR WRITE MISC.
:
:  INPUT:
:  R0      CONTAINS WRITE MISC FUNCTION CODE (BSEL1)
:  R1      CONTAINS DELAY (TIMES 800 NS) FOR BSEL2
:
:  T16WMISC:
:  SAVREG
:  JSR      PC,T16CLRBUF      :CLEAR MESSAGE BUFFER
:  MOV      #T16DT2,R2      :WRITE SUBSYSTEM DATA BUFFER
:  MOVB     #PW.WMISC,(R2)+  :STORE WRITE MISCELLANEOUS IN BSEL0
:  MOVB     R0,(R2)+        :STORE WRITE MISC CODE IN BSEL1
:  MOVB     R1,(R2)         :STORE DELAY (RESET TIMER) IN BSEL2
:  RTS      PC              :RETURN
:
:  SETUP T16PK2 PACKET FOR WRITE MISC. INVERT EXTENDED FEATURES SWITCH
:
:  T16SEXT:
:  MOV      #T16DT2,R0      :WRITE SUBSYSTEM DATA BUFFER
:  MOVB     #PW.WMISC,(R0)+ :STORE WRITE MISCELLANEOUS IN BSEL0
:  MOVB     #MS.EXT,(R0)    :STORE INVERT EXTENDED FEATURES IN BSEL1
:  RTS      PC              :RETURN
:
:  .=<.+10>B177770
:
:  WRITE CHARACTERISTICS COMMAND PACKET
:
:  T16PACKET:
:  .WORD    100004          :COMMAND PACKET FOR TEST
:  .WORD    T16DATA        :WRITE CHARACTERISTICS COMMAND, WITH ACK
:  .WORD    0              :ADDRESS OF CHARACTERISTICS BLOCK
:  .WORD    10.            :MESSAGE PACKET SIZE
:
:  T16DATA:
:  .WORD    T16BFR         :CHARACTERISTICS DATA BLOCK
:  .WORD    0              :ADDRESS OF MESSAGE BUFFER
:  .WORD    20.            :LENGTH OF MESSAGE BUFFER
:  .WORD    0              :ESS,ENB,EAI,ERI
:  .WORD    0              :EXTENDED FEATURES WORD
:
:  MESSAGE BUFFER
:
:  T16BFR:
:  .WORD    0              :BEGIN MESSAGE BUFFER
:  .WORD    0              :MESSAGE TYPE
:  .WORD    0              :DATA FIELD LENGTH

```

2158	040216	000000	.WORD	0	:RBCR
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	000	.BYTE	0	:BSEL0
2180	040251	000	.BYTE	0	:BSEL1
2181	040252	000000	.WORD	0	:SEL2
2182	040254		.BLKB	64.	:WRITE FIFO DATA OUTPUT BUFFER
2183					
2184					
2185	040354		ENDTST		
	040354				L10053:
	040354	104401			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 END
2250
2250 040404 BGNSUB ://////////////// BEGIN SUBTEST //////////////////
2251 040404 104402 T6.1: TRAP C$BSUB
2252
2253 040406 004737 015774 5$: Write to TSSR register to soft initialize the controller
2254 040406 103405 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
2255 040412 010001 BCS 10$ ;BR IF SOFT INIT OKAY
2256 040414 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2257 040416 104455 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
2258 040416 001130 TRAP C$ERDF
2259 040420 003652 .WORD 600
2260 040422 012034 .WORD SFIERR
2261 040424 012034 .WORD SFIMSG
2262
2263 040426 005037 002222 10$: Do a WRITE CHARACTERISTICS to setup a message buffer
2264 040432 012704 050200 CLR FATFLG ;CLEAR FATAL ERROR FLAG
2265 040436 004737 010662 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
2266 040442 103407 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
2267 040456 010001 FORCERROR 42$ ;@@DFORCE ERROR IF FORCER=1
2268 040460 010001 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
2269 040462 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2270 040462 104455 NEXT.ERRNO
2271 040464 001131 ERRDF ERRNO,T17SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
2272 040466 046625 TRAP C$ERDF
2273 040470 012046 .WORD 601
2274 040472 005237 002222 50$: INC FATFLG ;SET FATAL ERROR FLAG
2275 040476 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
2276 040476 104406 TRAP C$CLP1
2277
2278 ; Do a Write Subsystem READ STATUS
2279
2280

```

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,TSD8(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			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			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.OTRDY,(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	:@@D	
2299	040644	103404		BCS	90\$	:BR IF YES	
2300	040646			NEXT.ERRNO			
2301	040646			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 .SBTTL TEST 6: SUBTEST 2: FIFO WRITE SINGLE BYTE TEST
2313
2314 :++
2315 : TEST 6: SUBTEST 2:
2316 :
2317 : SUBTEST DESCRIPTION:
2318 :
2319 : This subtest verifies the ability of the FIFO to correctly
2320 : pass a single data byte from input to output. For each
2321 : of 256 data values (0-377 octal) the following is done:
2322 : 1. Initial FIFO status is checked
2323 : 2. The Write FIFO function, specifying a count of
2324 : one byte to be written is executed.
2325 : 3. Read Status is executed and FIFO status is checked.
2326 : 4. Read FIFO is executed and the data and final status
2327 : is checked.
2328 :
2329 : TEST STEPS:
2330 :
2331 : BEGIN
2332 : Write to TSSR to soft initialize
2333 : Do a WRITE CHARACTERISTICS to setup a message buffer
2334 : Do a Write Subsystem READ STATUS
2335 : If Input Ready NOT=1 Then Print Error
2336 : If Output Ready NOT=0 Then Print Error
2337 : If Data In Miss NOT=0 Then Print Error
2338 :
2339 : REPEAT FOR DATA FROM 0 TO 377 OCTAL
2340 : BEGIN
2341 : Do a Write Subsystem WRITE NPR to set tape direction out
2342 : Do a Write Subsystem WRITE FIFO with byte count equal to 1
2343 : Do a Write Subsystem READ STATUS
2344 : If Input Ready NOT=1 Then Print Error
2345 : If Output Ready NOT=1 Then Print Error
2346 : If Data In Miss NOT=0 Then Print Error
2347 : Do Write Subsystem READ FIFO with byte count equal to 1
2348 : If Data read from FIFO NOT= to Data sent Then Print Error
2349 : Do a Write Subsystem READ STATUS
2350 : If Input Ready NOT=1 Then Print Error
2351 : If Output Ready NOT=0 Then Print Error
2352 : If Data In Miss NOT=0 Then Print Error
2353 : END
2354 : END
2355 :--
2356 040674 BGNSUB ;////////// BEGIN SUBTEST //////////
      040674 T6.2:
      040674 104402 TRAP C$BSUB

2357
2358 :
2359 040676 $s: Write to TSSR register to soft initialize the controller
2360 040676 004737 015774 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
2361 040702 103405 BCS 10$ ;BR IF SOFT INIT OKAY
2362 040704 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2363 040706 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
      040706 104455 TRAP C$ERDF
      040710 001133 .WORD 603
      040712 003652 .WORD SFIFRR
  
```

Address	Offset	Hex	Hex	Instruction	Comment	Trap	Word
2364	040714	012034			Do a WRITE CHARACTERISTICS to setup a message buffer		
2365	040716	005037	002222	10\$: CLR FATFLG	;CLEAR FATAL ERROR FLAG		
2366	040722	012704	050200	MOV #T17PACKET,R4	;GET THE ADDRESS OF COMMAND PACKET		
2367	040726	004737	010662	JSR PC,WRTCHR	;DO WRITE CHARACTERISTICS COMMAND		
2368	040732			FORCERROR 42\$	;@@DFORCE ERROR IF FORCER=1		
2369	040746	103407		BCS 50\$	;BR IF CARRY SET (GOOD RETURN)		
2370	040750	010001		MOV R0,R1	;SAVE CONTENTS OF TSSR		
2371	040752			NEXT.ERRNO			
2372	040752			42\$: 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	50\$: INC FATFLG	;SET FATAL ERROR FLAG		
2374	040766			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,TSD(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			NEXT.ERRNO			
2384	041030			62\$: 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	70\$: INC FATFLG	;SET FATAL ERROR FLAG		
2386	041044			CKLOOP	;LOOP UN 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	;@@D		
2404	041134	103404		BCS 90\$	;BR IF YES		
2405	041136			NEXT.ERRNO			
2406	041136			82\$: 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					;REPEAT LABEL
2412						
2413	041156	012700	000100		Do a Write Subsystem WRITE NPR to set tape direction out	
2414	041162	004737	050026		MOV #NP.OUT,R0	;SET TAPE DIRECTION OUT
2415	041166	012704	050350		JSR PC,T17SNPR	;SETUP T17PK2 FOR WRITE NPR
2416	041172	010465	000000		MOV #T17PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET
2417	041176	004737	016336		MOV R4,TSDB(R5)	;SET THE PACKET ADDRESS TO EXECUTE
2418	041202				JSR PC,CHKTSSR	;WAIT FOR SSR TO SET
2419	041216	103407			FORCERROR 102\$	;ADDFORCE ERROR IF FORCER=1
2420	041220	010001			BCS 105\$	;BR IF CARRY SET (GOOD RETURN)
2421	041222				MOV R0,R1	;SAVE CONTENTS OF TSSR
2422	041222				NEXT.ERRNO	
	041222	104455			102\$: ERRDF ERRNO,T174SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET
	041224	001137				TRAP C\$ERDF
	041226	046773				.WORD 607
	041230	012046				.WORD T174SSR
2423	041232	005237	002222			.WORD PKTSSR
2424	041236				105\$: INC FATFLG	;SET FATAL ERROR FLAG
	041236	104406			CKLOOP	;LOOP ON ERROR, IF FLAG SET
2425						TRAP C\$CLP1
2426	041240	012700	000001		Do a Write Subsystem WRITE FIFO with byte count equal to 1	
2427	041244	012701	002312		MOV #1,R0	;WRITE 1 BYTE
2428	041250	004737	050052		MOV #DATA,R1	;FIFO WRITE DATA ADDRESS
2429	041254	012704	050350		JSR PC,T17WFIF	;SETUP T17PK2 FOR WRITE FIFO
2430	041260	010465	000000		MOV #T17PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET
2431	041264	004737	016336		MOV R4,TSDB(R5)	;SET THE PACKET ADDRESS TO EXECUTE
2432	041270				JSR PC,CHKTSSR	;WAIT FOR SSR TO SET
2433	041304	103407			FORCERROR 107\$	;ADDFORCE ERROR IF FORCER=1
2434	041306	010001			BCS 110\$	;BR IF CARRY SET (GOOD RETURN)
2435	041310				MOV R0,R1	;SAVE CONTENTS OF TSSR
2436	041310				NEXT.ERRNO	
	041310	104455			107\$: ERRDF ERRNO,T175SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET
	041312	001140				TRAP C\$ERDF
	041314	047036				.WORD 608
	041316	012046				.WORD T175SSR
2437	041320	005237	002222			.WORD PKTSSR
2438	041324				110\$: INC FATFLG	;SET FATAL ERROR FLAG
	041324	104406			CKLOOP	;LOOP ON ERROR, IF FLAG SET
2439						TRAP C\$CLP1
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\$	;ADDFORCE 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

Line	Address	Offset	Label	Instruction	Comment	Trap	CSCLP1
2450	041372	046726					
	041374	012046					
2451	041376	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
	041402	104406		CKLOOP	;LOOP ON ERROR, IF FLAG SET		
2452							
2453	041404	004737	050146	Set WORDS 0-7 of expd message buffer = to rcv since not testing			
2454	041410	012701	046402	JSR PC,T17SETEXP	;SET WORDS 0-7 EXPD=RCV		
2455	041414	012702	050242	MOV #T17EXSTA,R1	;GET EXPECTED READ STATUS		
2456	041420	012221		MOV #T17BFSTA,R2	;GET RCV READ STATUS		
2457	041422	011211		MOV (R2)+,(R1)+	;SET EXPD WORD #8 = RCV TEMP		
2458	041424	052711	000020	MOV (R2),(R1)	;SET EXPD WORD #9 = RCV TEMP		
2459	041430	052711	000040	BIS #S2.INRDY,(R1)	;SET EXP INPUT READY= 1		
2460	041434	042711	000200	BIS #S2.OTRDY,(R1)	;SET EXP OUTPUT READY= 1		
2461				BIC #S2.DIM,(R1)	;SET EXP DATA IN MISS = 0		
2462				If Input Ready NOT=1 then Print Error			
2463	041440	005000		If Output Ready NOT=1 or Data in Miss NOT=0 Then Print Error			
2464	041442	012701	050222	CLR R0	;HIGH RCV ADDRESS FOR CKMSG2		
2465	041446	012702	046362	MOV #T17BFR,R1	;LOW RCV ADDRESS FOR CKMSG2		
2466	041452	012703	000024	MOV #T17EXP,R2	;EXPD ADDRESS		
2467	041456	004737	011500	MOV #20,R3	;NUMBER OF BYTES TO COMPARE		
2468	041462			JSR PC,CKMSG2	;EXPD EQUAL RCV?		
2469	041472	103404		FORCERROR 132\$,NOTSSR	;00D		
2470	041474			BCS 140\$	;BR IF YES		
2471	041474			NEXT.ERRNO			
	041474	104456		ERRHRD ERRNO,T173CMP,MSGSTAT	;REPORT ERROR		
	041476	001142					
	041500	047323					
	041502	012350					
2472	041504						
	041504	104406					
2473							
2474							
2475	041506	012700	000001	Do Write Subsystem READ FIFO with byte count equal to 1			
2476	041512	004737	050106	MOV #1,R0	;SET READ BYTE COUNT		
2477	041516	012704	050350	JSR PC,T17RFIF	;SETUP T17PK2 FOR READ FIFO		
2478	041522	010465	000000	MOV #T17PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET		
2479	041526	004737	016336	MOV R4,TSDB(R5)	;SET THE PACKET ADDRESS TO EXECUTE		
2480	041532			JSR PC,CHKTSSR	;WAIT FOR SSR TO SET		
2481	041546	103407		FORCERROR 142\$	;00DFORCE ERROR IF FORCER=1		
2482	041550	010001		BCS 150\$	;BR IF CARRY SET (GOOD RETURN)		
2483	041552			MOV R0,R1	;SAVE CONTENTS OF TSSR		
2484	041552			NEXT.ERRNO			
	041552	104455		ERRDF ERRNO,T176SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET		
	041554	001143					
	041556	047102					
	041560	012046					
2485	041562	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
2486	041566			CKLOOP	;LOOP ON ERROR, IF FLAG SET		
	041566	104406					
2487							
2488	041570	004737	050146	Set WORDS 0-7 of expd message buffer = to rcv since not testing			
2489	041574	012701	046402	JSR PC,T17SETEXP	;SET WORDS 0-7 EXPD=RCV		
2490	041600	012702	050242	MOV #T17EXSTA,R1	;GET EXPECTED READ STATUS		
2491	041604	013721	002312	MOV #T17BFSTA,R2	;GET RCV READ STATUS		
2492	041610	011211		MOV DATA,(R1)+	;SET EXPD WORD #8 = COUNT DATA		
2493				MOV (R2),(R1)	;SET EXPD WORD #9 = RCV (NOT TESTING)		
				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$:      ERRHRD  ERRNO,T172CMP,MSGSUB      ;REPORT ERROR
                TRAP      C$ERHRD
                .WORD     612
                .WORD     T172CMP
                .WORD     MSGSUB
                TRAP      C$CLP1
                .WORD     104456
                .WORD     001144
                .WORD     047227
                .WORD     013742
2504 041656 160$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                TRAP      C$CLP1
                .WORD     104406
2505
2506      :      Do a Write Subsystem READ STATUS
2507 041660 004737 047764  JSR      PC,T17SRD      ;SETUP PACKET FOR READ STATUS
2508 041664 012704 050350  MOV      #T17PK2,R4      ;GET WRITE SUBSYSTEM COMMAND PACKET
2509 041670 010465 000000  MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
2510 041674 004737 016336  JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2511 041700      FORCERROR 162$      ;@@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
2515 041720 162$:      ERRDF  ERRNO,T173SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
                TRAP      C$ERDF
                .WORD     613
                .WORD     T173SSR
                .WORD     PKTSSR
                TRAP      C$CLP1
                .WORD     002222
2516 041730 005237 002222  INC      FATFLG      ;SET FATAL ERROR FLAG
2517 041734 170$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                TRAP      C$CLP1
                .WORD     104406
2518      :      Set WORDS 0-7 of expd message buffer = to recv since not testing
2519 041736 004737 050146  JSR      PC,T17SETEXP      ;SET WORDS 0-7 EXPD=RECV
2520 041742 012701 046402  MOV      #T17EXSTA,R1      ;GET EXPECTED READ STATUS
2521 041746 012702 050242  MOV      #T17BFSTA,R2      ;GET RECV READ STATUS
2522 041752 012221      MOV      (R2)+,(R1)+      ;SET EXPD WORD #8 = RECV TEMP
2523 041754 011211      MOV      (R2),(R1)      ;SET EXPD WORD #9 = RECV TEMP
2524 041756 052711 000020  BIS      #S2.INRDY,(R1)      ;SET EXP INPUT READY= 1
2525 041762 042711 000040  BIC      #S2.OUTRDY,(R1)      ;SET EXP OUTPUT READY= 0
2526 041766 042711 000200  BIC      #S2.DIM,(R1)      ;SET EXP DATA IN MISS = 0
2527      :      If Input Ready NOT=1 then Print Error
2528      :      If Output Ready NOT=0 or Data in Miss NOT=0 Then Print Error
2529 041772 005000      CLR      R0      ;HIGH RECV ADDRESS FOR CKMSG2
2530 041774 012701 050222  MOV      #T17BFR,R1      ;LOW RECV ADDRESS FOR CKMSG2
2531 042000 012702 046362  MOV      #T17EXP,R2      ;EXPD ADDRESS
2532 042004 012703 000024  MOV      #20.,R3      ;NUMBER OF BYTES TO COMPARE
2533 042010 004737 011500  JSR      PC,CKMSG2      ;EXPD EQUAL RECV?
2534 042014      FORCERROR 172$,NOTSSR      ;@@D
2535 042024 103404      BCS      180$      ;BR IF YES
2536 042026      NEXT.ERRNO
2537 042026 172$:      ERRHRD  ERRNO,T174CMP,MSGSTAT      ;REPORT ERROR
                TRAP      C$ERHRD
                .WORD     614
                .WORD     T174CMP
                .WORD     104456
                .WORD     001146
                .WORD     047407

```

SEQ 0186

2538	042034	012350			
	042036		180\$:	CKLOOP	
	042036	104406			
2539	042040			FORCEXIT	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 MSGSTAT
;                                TRAP C$CLP1
;@ad
;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 .SBTTL TEST 6: SUBTEST 3: FIFO WRITE MULTIPLE BYTES TEST
2555
2556
2557 : **
2558 : TEST 6: SUBTEST 3:
2559 : SUBTEST DESCRIPTION:
2560
2561 : This subtest verifies the ability of the FIFO to correctly
2562 : pass a multiple data bytes from input to output.
2563 : The following sequence is done with various data patterns
2564 : and byte counts from 2 to 64.
2565 : 1. Initial FIFO status is checked
2566 : 2. The Write FIFO function.
2567 : 3. Read Status is executed and FIFO status is checked.
2568 : 4. Read FIFO is executed and the data and final status
2569 : is checked.
2570
2571 : TEST STEPS:
2572
2573 : BEGIN
2574 : Write to TSSR to soft initialize
2575 : Do a WRITE CHARACTERISTICS to setup a message buffer
2576 : Do a Write Subsystem READ STATUS
2577 : If Input Ready NOT=1 Then Print Error
2578 : If Output Ready NOT=0 Then Print Error
2579 : If Data In Miss NOT=0 Then Print Error
2580 : If Last Word NOT=0 Then Print Error
2581 : REPEAT FOR DATA 0 TO 377, 377 TO 0, FLOATING 1'S,0'S AND ALL 1'S/0'S
2582 : REPEAT FOR BYTE COUNT 2 TO 64 DECIMAL
2583 : BEGIN
2584 : Do a Write Subsystem WRITE NPR to set tape direction out
2585 : Do a Write Subsystem WRITE FIFO
2586 : Do a Write Subsystem READ STATUS
2587 : If Input Ready NOT=1 Then Print Error
2588 : If Output Ready NOT=1 Then Print Error
2589 : If Data In Miss NOT=0 Then Print Error
2590 : If Last Word NOT=0 Then Print Error
2591 : Do Write Subsystem READ FIFO
2592 : If Data read from FIFO NOT= to Data sent Then Print Error
2593 : Do a Write Subsystem READ STATUS
2594 : If Input Ready NOT=1 Then Print Error
2595 : If Output Ready NOT=0 Then Print Error
2596 : If Data In Miss NOT=0 Then Print Error
2597 : If Last Word NOT=0 Then Print Error
2598 : END
2599 : END
2600
2601 042104 BGNSUB ;////////// BEGIN SUBTEST //////////
2602 042104 T6.3: TRAP CSBSUB
2603 042104 104402
2604
2605 : Write to TSSR register to soft initialize the controller
2606 : JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
2607 : BCS 10$ ;BR IF SOFT INIT OKAY
2608 : MOV R0,R1 ;SAVE CONTENTS OF TSSR
2609 : ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
  
```

	042116	104455							TRAP	C\$ERDF
	042120	001146							.WORD	614
	042122	003652							.WORD	SFIERR
	042124	012034							.WORD	SFIMSG
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		INC FATFLG				:SET FATAL ERROR FLAG	
2619	042176			50\$:	CKLOOP				:LOOP ON ERROR, IF FLAG SET	
	042176	104406							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,TSD(BR5)				: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		INC FATFLG				:SET FATAL ERROR FLAG	
2631	042254			70\$:	CKLOOP				:LOOP ON ERROR, IF FLAG SET	
	042254	104406							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.OUTRDY,(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.IIW,(R1)				:SET EXP LAST WORD (IIW)=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							

2652	042352				NEXT.ERRNO				
2653	042352				82\$:	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	TRAP C\$CLP1
	042362	104406							
2655									
2656									
2657									
2658									
2659									
2660									
2661									
2662	042364	012737	000001	002314		MOV	#1,TSTFLAG	;TEST PATTERN FLAG	
2663	042372				95\$:				
2664	042372	012737	000002	002310		MOV	#2,COUNT	;GET FIRST BYTE COUNT	
2665	042400				100\$:				
2666									
2667	042400	012700	000100						
2668	042404	004737	050026						
2669	042410	012704	050350						
2670	042414	010465	000000						
2671	042420	004737	016336						
2672	042424								
2673	042440	103407							
2674	042442	010001							
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				CKLOOP		;LOOP ON ERROR, IF FLAG SET	TRAP C\$CLP1
2679									
2680	042462	004737	050126						
2681	042466	012701	046504						
2682	042472	013702	002310						
2683	042476	022737	000001	002314					
2684	042504	001005							
2685	042506	005000							
2686	042510	110021			110\$:				
2687	042512	005200							
2688	042514	005302							
2689	042516	003374							
2690	042520	022737	000002	002314	115\$:				
2691	042526	001006							
2692	042530	012700	000377						
2693	042534	110021			120\$:				
2694	042536	005300							
2695	042540	005302							
2696	042542	003374							
2697	042544	022737	000003	002314	125\$:				
2698	042552	001005							

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,T17WFI		: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
	042636	104455					TRAP	C\$ERDF
	042640	001153					.WORD	619
	042642	047036					.WORD	T175SSR
	042644	012046					.WORD	PKTSSR
2715	042646	005237	002222		INC	FATFLG		:SET FATAL ERROR FLAG
2716	042652			150\$:	CKLOOP			:LOOP ON ERROR, IF FLAG SET
	042652	104406					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
	042714	104455					TRAP	C\$ERDF
	042716	001154					.WORD	620
	042720	046726					.WORD	T173SSR
	042722	012046					.WORD	PKTSSR
2728	042724	005237	002222		INC	FATFLG		:SET FATAL ERROR FLAG
2729	042730			160\$:	CKLOOP			:LOOP ON ERROR, IF FLAG SET
	042730	104406					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				:		If Input Ready NOT=1 then Print Error		
2742				:		If Output Ready NOT=1 or Data in Miss NOT=0 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	:@@D	
2749	043024	103404		BCS	170\$	:BR IF YES	
2750	043026			NEXT.ERRNO			
2751	043026			162\$:	ERRHRD ERRNO,T173CMP,MSGSTAT	:REPORT ERROR	
	043026	104456				TRAP	C\$ERHRD
	043030	001155				.WORD	621
	043032	047323				.WORD	T173CMP
	043034	012350				.WORD	MSGSTAT
2752	043036			170\$:	CKLOOP	:LOOP ON ERROR, IF FLAG SET	
	043036	104406				TRAP	C\$CLP1
2753				:	Do Write Subsystem READ FIFO		
2754							
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	
	043104	104455				TRAP	C\$ERDF
	043106	001156				.WORD	622
	043110	047102				.WORD	T176SSR
	043112	012046				.WORD	PKTSSR
2765	043114	005237	002222	INC	FATFLG	:SET FATAL ERROR FLAG	
2766	043120			180\$:	CKLOOP	:LOOP ON ERROR, IF FLAG SET	
	043120	104406				TRAP	C\$CLP1
2767				:	If Data read from FIFO NOT= to Data sent Then Print Error		
2768							
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	:@@D	
2775	043154	103406		BCS	200\$	:BR IF YES	
2776	043156			NEXT.ERRNO			
2777	043156	013701	002310	192\$:	MOV COUNT,R1	:GET BYTE COUNT	
2778	043162			ERRHRD	ERRNO,T175CMP,FIFEXP	:REPORT ERROR	
	043162	104456				TRAP	C\$ERHRD
	043164	001157				.WORD	623
	043166	047472				.WORD	T175CMP
	043170	012170				.WORD	FIFEXP
2779	043172			200\$:	CKLOOP	:LOOP ON ERROR, IF FLAG SET	
	043172	104406				TRAP	C\$CLP1
2780				:	Do a Write Subsystem READ STATUS		
2781							
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      R0,R1      ;SAVE CONTENTS OF TSSR
2789 043234      NEXT.ERRNO
2790 043234      212$:  ERRDF  ERRNO,T173SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                        043234 104455      TRAP      C$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      220$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                        043250 104406      TRAP      C$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      R0 ;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      232$:  ERRHRD  ERRNO,T174CMP,MSGSTAT ;REPORT ERROR
                        043346 104456      TRAP      C$ERHRD
                        043350 001161      .WORD      625
                        043352 047407      .WORD      T174CMP
                        043354 012350      .WORD      MSGSTAT
2814 043356      240$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                        043356 104406      TRAP      C$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$:  ENDSUB
2825 043430      ;////////// END SUBTEST //////////
                        043430 104403      L10061:
                        043430      TRAP      C$ESUB
2826 043432 005737 002222      TST      FATFLG ;ANY FATAL ERRORS ?
2827 043436 001402      BEQ      260$ ;BRANCH IF NOT
2828 043440 004737 017202      JSR      PC,CKDROP ;TRY TO DROP THE UNIT
2829 043444      260$:
2830
2831
2832

```

TSV5 - HARDWARE TESTS MACRO M1113 25-MAY-82 09:19 PAGE 107-6 L 15
TEST 6: SUBTEST 3: FIFO WRITE MULTIPLE BYTES TEST

SEQ 0193

2833

```

2835 .SBTTL TEST 6: SUBTEST 4: FIFO Verify ILW Status
2836
2837
2838 :++
2839 : TEST 6: SUBTEST 4:
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:
      043444 104402 TRAP CSBSUB

2858
2859 : Write to TSSR register to soft initialize the controller
2860 043446 5$:
2861 043446 004737 015774 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
2862 043452 103405 BCS 10$ ;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
      043456 104455 TRAP CSERDF
      043460 001161 .WORD 625
      043462 003652 .WORD SFIERR
      043464 012034 .WORD SFIMSG

2865 : Do a WRITE CHARACTERISTICS to setup a message buffer
2866 043466 005037 002222 10$: CLR FATFLG ;CLEAR FATAL ERROR FLAG
2867 043472 012704 050200 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
2868 043476 004737 010662 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
2869 043502 103407 FORCERROR 42$ ;@@DFORCE ERROR IF FORCER=1
2870 043516 010001 BCS 50$ ;BR IF CARRY SET (GOOD RETURN)
2871 043520 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2872 043522 42$: NEXT.ERRNO
2873 043522 104455 ERRDF ERRNO,T17SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      043522 104455 TRAP CSERDF
      043524 001162 .WORD 626
      043526 046625 .WORD T17SSR
      043530 012046 .WORD PKTSSR

2874 043532 005237 002222 50$: INC FATFLG ;SET FATAL ERROR FLAG
2875 043536 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      043536 104406 TRAP CSCLP1

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\$	;@ADFORCE 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			ERRDF	ERRNO,T176SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	043604	104455					TRAP C\$ERDF
	043606	001163					.WORD 627
	043610	047102					.WORD T176SSR
	043612	012046					.WORD PKTSSR
2888	043614	005237	002222	INC	FATFLG	;SET FATAL ERROR FLAG	
2889	043620			CKLOOP		;LOOP ON ERROR, IF FLAG SET	
	043620	104406					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\$	;@ADFORCE 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			NEXT.ERRNO			
2900	043662			ERRDF	ERRNO,T173SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	043662	104455					TRAP C\$ERDF
	043664	001164					.WORD 628
	043666	046726					.WORD T173SSR
	043670	012046					.WORD PKTSSR
2901	043672	005237	002222	INC	FATFLG	;SET FATAL ERROR FLAG	
2902	043676			CKLOOP		;LOOP ON ERROR, IF FLAG SET	
	043676	104406					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.OUTRDY,(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	;@AD	
2922	043772	103404		BCS	180\$	;BR IF YES	
2923	043774			NEXT.ERRNO			
2924	043774			ERRHRD	ERRNO,T176CMP,MSGSTAT	;REPORT ERROR	
	043774	104456					TRAP C\$ERHRD
	043776	001165					.WORD 629
	044000	047546					.WORD T176CMP

SEQ 0196

2925 044002 012350
 044004 104406 180\$: CKLOOP
 2926 044006
 2927 044006 ENDSUB
 044006 104403
 2928 044010 C05737 002222 TST FATFLG
 2929 044014 001402 BEQ 260\$
 2930 044016 004737 017202 JSR PC,CKDROP
 2932 044022 260\$:
 2933
 2934

.WORD MSGSTAT
 ;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 .SBTTL TEST 6: SUBTEST 5: FIFO Verify Input Ready
2937
2938
2939 :++
2940 : TEST 6: SUBTEST 5:
2941 : SUBTEST DESCRIPTION:
2942 :
2943 : This subtest verifies that writing 64. bytes into the FIFO
2944 : without reading any out causes the Input Ready status to
2945 : negate. The Subtest then verifies that writing a 65th byte
2946 : into the FIFO causes the Data In Miss status to assert.
2947 : Next it is verified that the original 64 bytes can be read
2948 : out correctly and that the data has not been corrupted.
2949 :
2950 : TEST STEPS:
2951 : BEGIN
2952 : Write to TSSR to soft initialize
2953 : Do a WRITE CHARACTERISTICS to setup a message buffer
2954 : Do a Write Subsystem WRITE NPR to set tape direction out
2955 : Do a Write Subsystem WRITE FIFO 64. bytes incrementing pattern
2956 : Do a Write Subsystem READ STATUS
2957 : If Input Ready NOT=0 Then Print Error
2958 : If Output Ready NOT=1 Then Print Error
2959 : If Data In Miss NOT=0 Then Print Error
2960 : Do a Write Subsystem WRITE FIFO 1 byte for a total of 65. written
2961 : Do a Write Subsystem READ STATUS
2962 : If Input Ready NOT=0 Then Print Error
2963 : If Output Ready NOT=1 Then Print Error
2964 : If Data In Miss NOT=1 Then Print Error
2965 : Do Write Subsystem READ FIFO
2966 : If Data read from FIFO NOT= to Data sent Then Print Error
2967 : Do a Write Subsystem READ STATUS
2968 : If Input Ready NOT=1 Then Print Error
2969 : If Output Ready NOT=0 Then Print Error
2970 : If Data In Miss NOT=1 Then Print Error
2971 : END
2972 :--
2973
2974 044022 BGNSUB ;////////// BEGIN SUBTEST //////////
2975 044022 104402 T6.5: TRAP C$BSUB
2976
2977 : Write to TSSR register to soft initialize the controller
2978 044024 004737 015774 5$: JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
2979 044030 103405 BCS 10$ ;BR IF SOFT INIT OKAY
2980 044032 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
2981 044034 104455 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
2982 044034 001165 TRAP C$ERDF
2983 044036 003652 .WORD 629
2984 044040 012034 .WORD SFIERR
2985 044042 005037 002222 10$: Do a WRITE CHARACTERISTICS to setup a message buffer
2986 044044 012704 050200 CLR FATFLG ;CLEAR FATAL ERROR FLAG
2987 044050 004737 010662 MOV #T17PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
2988 044054 004737 010662 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
2989 044060 004737 010662 FORCERROR 42$ ;@ADFORCE ERROR IF FORCER=1

```

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
	044100	104455					TRAP	C\$ERDF
	044102	001166					.WORD	630
	044104	046625					.WORD	T17SSR
	044106	012046					.WORD	PKTSSR
2991	044110	005237	002222		INC	FATFLG		:SET FATAL ERROR FLAG
2992	044114			50\$:	CKLOOP			:LOOP ON ERROR, IF FLAG SET
	044114	104406					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\$:	NEXT.ERRNO			
	044162	104455			ERRDF	ERRNO,T174SSR,PKTSSR		:DEVICE FATAL SSR FAILED TO SET
	044164	001167					TRAP	C\$ERDF
	044166	046773					.WORD	631
	044170	012046					.WORD	T174SSR
3005	044172	005237	002222		INC	FATFLG		:SET FATAL ERROR FLAG
3006	044176			105\$:	CKLOOP			:LOOP ON ERROR, IF FLAG SET
	044176	104406					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			MOVB	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\$:	NEXT.ERRNO			
	044300	104455			ERRDF	ERRNO,T175SSR,PKTSSR		:DEVICE FATAL SSR FAILED TO SET
	044302	001170					TRAP	C\$ERDF
	044304	047036					.WORD	632
	044306	012046					.WORD	T175SSR
3028	044310	005237	002222		INC	FATFLG		:SET FATAL ERROR FLAG
3029	044314			150\$:	CKLOOP			:LOOP ON ERROR, IF FLAG SET

```

044314 104406                                TRAP    C$CLP1
3030
3031      :      Do a Write Subsystem READ STATUS
3032      :      If Input Ready NOT=0      Then Print Error
3033      :      If Output Ready NOT=1     Then Print Error
3034      :      If Data In Miss NOT=0     Then Print Error
3035 044316 004737 047764      JSR      PC,T17SRD      ;SETUP PACKET FOR READ STATUS
3036 044322 012704 050350      MOV      #T17PK2,R4      ;GET WRITE SUBSYSTEM COMMAND PACKET
3037 044326 010465 000000      MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
3038 044332 004737 016336      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
3039 044336      FORCERROR      157$      ;ADD FORCE 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
                                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
3046      :      Set WORDS 0-7 of expd message buffer = to recv 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.OTRDY,(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$,NOTSSR      ;ADD
3061 044462 103404      BCS      170$      ;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
                                044464 104456
                                044466 001172
                                044470 047323
                                044472 012350
3064 044474 170$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
3065      :
3066      :      Do a Write Subsystem WRITE FIFO 1 byte for a total of 65. written
3067      :      MOV      #1,R0      ;FIFO BYTE COUNT
3068 044476 012700 000001      MOV      #T17WFDATA,R1      ;FIFO WRITE DATA ADDRESS
3069 044502 012701 046504      JSR      PC,T17WFIF      ;SETUP T17PK2 FOR WRITE FIFO
3070 044506 004737 050052      MOV      #T17PK2,R4      ;GET WRITE SUBSYSTEM COMMAND PACKET
3071 044512 012704 050350      MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS TO EXECUTE
3072 044516 010465 000000      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
3073 044522 004737 016336      FORCERROR      172$      ;ADD FORCE ERROR IF FORCER=1
3074 044526      BCS      180$      ;BR IF CARRY SET (GOOD RETURN)
3075 044542 103407

```

```

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 180$: INC     FATFLG      ;SET FATAL ERROR FLAG
3080 044562             CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$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,TSD8(R5)     ;SET THE PACKET ADDRESS TO EXECUTE
3089 044600 004737 016336 JSR     PC,CHKTSSR      ;WAIT FOR SSR TO SET
3090 044604             FORCERROR 187$      ;ADD FORCE 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 190$: INC     FATFLG      ;SET FATAL ERROR FLAG
3096 044640             CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C$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=RCV
3099 044646 012701 046402 MOV     #T17EXSTA,R1    ;GET EXPECTED READ STATUS
3100 044652 012702 050242 MOV     #T17BFSTA,R2    ;GET RCV READ STATUS
3101 044656 012221 MOV     (R2)+,(R1)+ ;SET EXPD WORD #8 = RCV TEMP
3102 044660 011211 MOV     (R2),(R1) ;SET EXPD WORD #9 = RCV 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 RCV ADDRESS FOR CKMSG2
3107 044700 012701 050222 MOV     #T17BFR,R1      ;LOW RCV 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 RCV?
3111 044720             FORCERROR 192$,NOTSSR ;ADD
3112 044730 103404 BCS     200$      ;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			ERRDF	ERRNO,T176SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET
	045010	104455				TRAP C\$ERDF
	045012	001176				.WORD 638
	045014	047102				.WORD T176SSR
	045016	012046				.WORD PKTSSR
3127	045020	005237	002222	INC	FATFLG	;SET FATAL ERROR FLAG
3128	045024			CKLOOP		;LOOP ON ERROR, IF FLAG SET
	045024	104406				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	MOV	COUNT,R1	;GET BYTE COUNT
3140	045066			ERRHRD	ERRNO,T175CMP,FIFEXP	;REPORT ERROR
	045066	104456				TRAP C\$ERHRD
	045070	001177				.WORD 639
	045072	047472				.WORD T175CMP
	045074	012170				.WORD FIFEXP
3141	045076			240\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET
	045076	104406				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			ERRDF	ERRNO,T173SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET
	045140	104455				TRAP C\$ERDF
	045142	001200				.WORD 640
	045144	046726				.WORD T173SSR
	045146	012046				.WORD PKTSSR
3156	045150	005237	002222	INC	FATFLG	;SET FATAL ERROR FLAG
3157	045154			CKLOOP		;LOOP ON ERROR, IF FLAG SET
	045154	104406				TRAP C\$CLP1
3158				:	Set WORDS 0-7 of expd message buffer = to recv since not testing	
3159	045156	004737	050146	JSR	PC,T17SETEXP	;SET WORDS 0-7 EXPD=RECV

3160	045162	012701	046402	MOV	#T17EXSTA,R1	;GET EXPECTED READ STATUS	
3161	045166	012702	050242	MOV	#T17BFSTA,R2	;GET RECV READ STATUS	
3162	045172	012221		MOV	(R2)+,(R1)+	;SET EXPD WORD #8 = RECV TEMP	
3163	045174	011211		MOV	(R2),(R1)	;SET EXPD WORD #9 = RECV TEMP	
3164	045176	052711	000020	BIS	#S2.INRDY,(R1)	;SET EXP INPUT READY= 1	
3165	045202	042711	000040	BIC	#S2.OTRDY,(R1)	;SET EXP OUTPUT READY= 0	
3166	045206	052711	000200	BIS	#S2.DIM,(R1)	;SET EXP DATA IN MISS = 1	
3167	045212	005000		CLR	R0	;HIGH RECV ADDRESS FOR CKMSG2	
3168	045214	012701	050222	MOV	#T17BFR,R1	;LOW RECV ADDRESS FOR CKMSG2	
3169	045220	012702	046362	MOV	#T17EXP,R2	;EXPD ADDRESS	
3170	045224	012703	000024	MOV	#20.,R3	;NUMBER OF BYTES TO COMPARE	
3171	045230	004737	011500	JSR	PC,CKMSG2	;EXPD EQUAL RECV?	
3172	045234			FORCERROR	272\$,NOTSSR	;ADD	
3173	045244	103404		BCS	280\$	;BR IF YES	
3174	045246			NEXT.ERRNO			
3175	045246			ERRHRD	ERRNO,T174CMP,MSGSTAT	;REPORT ERROR	
	045246	104456					TRAP C\$ERHRD
	045250	001201					.WORD 641
	045252	047407					.WORD T174CMP
	045254	012350					.WORD MSGSTAT
3176	045256			280\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET	
	045256	104406					TRAP C\$CLP1
3177							
3178	045260			ENDSUB		////////// END SUBTEST //////////	
	045260					L10063:	
	045260	104403					TRAP C\$ESUB
3179							
3180	045262	005737	002222	TST	FATFLG	;ANY FATAL ERRORS ?	
3181	045266	001402		BEQ	300\$	;BRANCH IF NOT	
3182	045270	004737	017202	JSR	PC,CKDROP	;TRY TO DROP THE UNIT	
3183	045274			300\$:			
3184							
3185							
3186							

```
.SBITL 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 FIFO 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 //////////
      16.6:

```

TRAP CSBSUB

```

55:      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 SERDF
.WORD     641
.WORD     SFIERR
.WORD     SFIMSG

```

```

: Do a WRITE CHARACTERISTICS to setup a message buffer
IOS: CLR FATELG : CLEAR FATAL ERROR FLAG

```

```

DO WRITE CHARACTERISTICS TO SETOP & 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$             :00DFORCE ERROR IF FORCER=1

```

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,T17SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET
	045352	104455				TRAP	C\$ERDF
	045354	001202				.WORD	642
	045356	046625				.WORD	T17SSR
	045360	012046				.WORD	PKTSSR
3243	045362	005237	002222	INC	FATFLG		;SET FATAL ERROR FLAG
3244	045366		50\$:	CKLOOP			;LOOP ON ERROR, IF FLAG SET
	045366	104406				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
	045430	104455				TRAP	C\$ERDF
	045432	001203				.WORD	643
	045434	046662				.WORD	T172SSR
	045436	012046				.WORD	PKTSSR
3255	045440	005237	002222	INC	FATFLG		;SET FATAL ERROR FLAG
3256	045444		70\$:	CKLOOP			;LOOP ON ERROR, IF FLAG SET
	045444	104406				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
	045506	104455				TRAP	C\$ERDF
	045510	001204				.WORD	644
	045512	046726				.WORD	T173SSR
	045514	012046				.WORD	PKTSSR
3270	045516	005237	002222	INC	FATFLG		;SET FATAL ERROR FLAG
3271	045522		80\$:	CKLOOP			;LOOP ON ERROR, IF FLAG SET
	045522	104406				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	MOV	2(R2),2(R1)		;SET EXPD WORD #9 = RECV (NOT TESTING)
3280	045564	005000	000002	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
	045620	104456				TRAP C\$ERHRD
	045622	001205				.WORD 645
	045624	047654				.WORD T177CMP
	045626	012350				.WORD MSGSTAT
3289	045630	100\$:		CKLOOP		;LOOP ON ERROR, IF FLAG SET
	045630	104406				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,R4	;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\$	;@D FORCE 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
	045676	104455				TRAP C\$ERDF
	045700	001206				.WORD 646
	045702	046773				.WORD T174SSR
	045704	012046				.WORD PKTSSR
3302	045706	005237	002222	INC	FATFLG	;SET FATAL ERROR FLAG
3303	045712	120\$:		CKLOOP		;LOOP ON ERROR, IF FLAG SET
	045712	104406				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		MOV	B,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,COUNT	;GET FIRST BYTE COUNT
3316				:	Do a Write Subsystem WRITE FIFO with the current byte count	
3317	045744			150\$:		;REPEAT LOOP LABEL
3318	045744	013700	002310	MOV	COUNT,R0	;FIFO BYTE COUNT
3319	045750	012701	046402	MOV	#T17EXSTA,R1	;FIFO WRITE DATA ADDRESS
3320	045754	004737	050052	JSR	PC,T17WFIF	;SETUP T17PK2 FOR WRITE FIFO
3321	045760	012704	050350	MOV	#T17PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET
3322	045764	010465	000000	MOV	R4,TSDB(R5)	;SET THE PACKET ADDRESS TO EXECUTE
3323	045770	004737	016336	JSR	PC,CHKTSSR	;WAIT FOR SSR TO SET
3324	045774			FORCERROR	152\$	;@D FORCE 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  '175SSR
      046022      012046                                .WORD  PKTSSR
3329 046024      005237  002222
3330 046030      160$:  INC      FATFLG                ;SET FATAL ERROR FLAG
      046030      104406      CKLOOP                ;LOOP ON ERROR, IF FLAG SET
                                          TRAP  C$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
3343 046106      170$:  INC      FATFLG                ;SET FATAL ERROR FLAG
      046106      104406      CKLOOP                ;LOOP ON ERROR, IF FLAG SET
                                          TRAP  C$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
3358 046164      180$:  INC      FATFLG                ;SET FATAL ERROR FLAG
      046164      104406      CKLOOP                ;LOOP ON ERROR, IF FLAG SET
                                          TRAP  C$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
  
```



```

3401
3402      ;+
3403      ;LOCAL STORAGE FOR THIS TEST
3404      ;+
3405
3406      T17MSK:
3407      ;MASK OF UNTESTED BITS IN READ STATUS BYTES
3408      ;UNTESTED BITS ARE SET TO 1
3409      ;BYTE 0 MASK
3410      ;BYTE 1 MASK (PAKERR,IRESV2,IRESV1)
3411      ;BYTE 2 (TIMER A,TIMER B,UNDEFINED<1:0>)
3412      ;MAKE IT EVEN
3413
3414      T17EXP:
3415      ;BEGIN EXPECTED DATA BUFFER
3416      ;MESSAGE TYPE
3417      ;DATA FIELD LENGTH
3418      ;RBPGR
3419      ;XST0
3420      ;XST1
3421      ;XST2
3422      ;XST3
3423      ;XST4 (ALWAYS PRESENT FOR WRITE SUB.)
3424      ;EXPECTED READ STATUS AND WRITE FIFO DATA
3425      ;END EXPECTED DATA BUFFER
3426
3427      T17EXSTA: .BLKB 66.
3428      T17EXEND:
3429
3430      T17WFDATA: .BLKB 66.
3431      ;WRITE FIFO EXPECTED DATA BUFFER
3432
3433      ;+
3434      ;LOCAL TEXT MESSAGES FOR TEST
3435      ;+
3436
3437      TST17ID:
3438      ;ASCIIZ 'FIFO Exerciser'
3439      T17SSR: .ASCIIZ 'WRITE CHARACTERISTICS Failed'
3440      T172SSR: .ASCIIZ 'WRITE SUBSYSTEM (Write Misc) Failed'
3441      T173SSR: .ASCIIZ 'WRITE SUBSYSTEM (Read Status) Failed'
3442      T174SSR: .ASCIIZ 'WRITE SUBSYSTEM (Write Npr) Failed'
3443      T175SSR: .ASCIIZ 'WRITE SUBSYSTEM (Write FIFO) Failed'
3444      T176SSR: .ASCIIZ 'WRITE SUBSYSTEM (Read FIFO) Failed'
3445      T171CMP: .ASCIIZ 'FIFO Status in WORD #9 Incorrect after Initialize'
3446      T172CMP: .ASCIIZ 'Read FIFO Data not equal to Write FIFO , Data is in WORD #8'
3447      T173CMP: .ASCIIZ 'FIFO Status (In WORD #9) Incorrect after WRITE FIFO'
3448      T174CMP: .ASCIIZ 'FIFO Status (In WORD #9) Incorrect after READ FIFO'
3449      T175CMP: .ASCIIZ 'Read FIFO Data not equal to Write FIFO Data'
3450      T176CMP: .ASCIIZ 'FIFO Status (In WORD #9) Incorrect after READ FIFO from an Empty FIFO'
3451      T177CMP: .ASCIIZ 'FIFO Status (In WORD #9) Incorrect after RESET FIFO'
3452      .EVEN
3453
3454      ;+
3455      ; CLEAR MESSAGE BUFFER
3456      ;+
3457
3458      T17CLRBUF:
3459      SAVREG
3460      MOV #T17BFR,R1
3461      MOV #T17BEND-T17BFR,R2
3462      10$: CLR (R1)+
3463      DEC R2
3464      BGT 10$
3465      RTS PC
3466
3467      ;SAVE R1-R5 UNTIL NEXT RETURN
3468      ;GET MESSAGE BUFFER ADDRESS
3469      ;SIZE OF MESSAGE BUFFER IN BYTES
3470      ;CLEAR A BYTE
3471      ;DONE?
3472      ;BR IF NO
3473      ;RETURN
3474
3475      012701 050222
3476      012702 000120
3477      105021
3478      005302
3479      003375
3480      000207
  
```

```

3458
3459
3460      ;+
3461      ; SETUP T17PK2 PACKET FOR READ STATUS
3462      ;-
3462      T17SRD:
3463      JSR      PC,T17CLRBUF      ;CLEAR MESSAGE BUFFER
3464      MOV      #T17DT2,R0        ;WRITE SUBSYSTEM DATA BUFFER
3465      MOV      #PW.RDSTATUS,(R0)+ ;STORE READ STATUS COMMAND IN BSEL0
3466      CLRB      (R0)              ;CLEAR BSEL1
3467      RTS      PC                ;RETURN
3468
3469      ;+
3470      ; SETUP T17PK2 PACKET FOR WRITE MISC RESET FIFO
3471      ;-
3472      T17RSFIF:
3473      JSR      PC,T17CLRBUF      ;CLEAR MESSAGE BUFFER
3474      MOV      #T17DT2,R0        ;WRITE SUBSYSTEM DATA BUFFER
3475      MOV      #PW.WMISC,(R0)+   ;STORE WRITE MISCELLANEOUS IN BSEL0
3476      MOV      #MS.RSFIF!MS.RSTAP,(R0) ;STORE BSEL1 CLEAR FIFO CODES
3477      RTS      PC                ;RETURN
3478
3479      ;+
3480      ; SETUP T17PK2 PACKET FOR WRITE NPR
3481      ;-
3482      INPUT:
3483      R0 CONTAINS BSEL1 NPR DATA
3484
3485      SETS NP.WRP SINCE IF 0 IT WRITES WRONG PARITY.
3486      ;-
3487      T17SNPR:
3488      JSR      PC,T17CLRBUF      ;CLEAR MESSAGE BUFFER
3489      MOV      #T17DT2,R1        ;WRITE SUBSYSTEM DATA BUFFER
3490      MOV      #PW.WNPR,(R1)+    ;STORE WRITE NPR IN BSEL0
3491      BIS      #NP.WRP,R0        ;DON'T WRITE WRONG PARITY
3492      MOV      R0,(R1)           ;STORE NPR DATA IN BSEL1
3493      RTS      PC                ;RETURN
3494
3495      ;+
3496      ; SETUP T17PK2 PACKET FOR WRITE FIFO
3497      ;-
3498      INPUT:
3499      R0 CONTAINS BYTE COUNT
3500      R1 CONTAINS DATA PATTERN BLOCK ADDRESS
3501      ;-
3502      T17WFIF:
3503      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
3504      JSR      PC,T17CLRBUF      ;CLEAR MESSAGE BUFFER
3505      MOV      #T17DT2,R2        ;WRITE SUBSYSTEM DATA BUFFER
3506      MOV      #PW.WFIFO,(R2)+   ;STORE WRITE FIFO IN BSEL0
3507      MOV      R0,(R2)+          ;STORE BYTE COUNT IN BSEL1
3508      CLR      (R2)+             ;CLEAR SEL2 (UNUSED)
3509      MOV      (R1)+,(R2)+       ;STORE DATA PATTERN BYTE
3510      DEC      R0                ;DONE ALL BYTES?
3511      BGT      10$               ;BR IF NO
3512      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      JSR      PC,T17CLRBUF      ;CLEAR MESSAGE BUFFER
3522 050112      MOV      #T17DT2,R1      ;WRITE SUBSYSTEM DATA BUFFER
3523 050116      MOV      #PW,RFIFO,(R1)+ ;STORE READ FIFO IN BSEL0
3524 050122      MOV      RO,(R1)+        ;STORE BYTE COUNT IN BSEL1
3525 050124      RTS      PC              ;RETURN
3526
3527      ;+
3528      ; CLEAR EXPECTED DATA MESSAGE BUFFER
3529      ;
3530 050126 T17CLEXP:
3531 050132      MOV      #T17EXP,R1      ;GET EXPD ADDRESS
3532 050136      MOV      #T17EXEND-T17EXP,R0 ;GET EXPD SIZE
3533 050140      10$:      CLR      (R1)+    ;CLEAR A BYTE
3534 050142      DEC      RO              ;DONE?
3535 050144      BGT      10$             ;BR IF NO
3536      RTS      PC                    ;RETURN
3537
3538      ;+
3539      ;Set WORDS 0-7 of expd message buffer = to rcv since not testing
3540      ;
3541 050146 T17SETEXP:
3542 050152      MOV      #T17EXP,R2      ;GET EXPD
3543 050156      MOV      #T17BFR,R3      ;GET READ STATUS RECV BUFFER
3544 050162      MOV      #8,R0          ;SET WORDS 0-7 EXP=RCV
3545 050164      5$:      MOV      (R3)+,(R2)+ ;SET EXPD=RCV
3546 050166      DEC      RO              ;DONE WORDS 0-7 WORDS?
3547 050170      BGT      5$             ;BR IF NO
3548      RTS      PC                    ;RETURN
3549
3550      .=<.+10>&177770
3551
3552      ;
3553      ;WRITE CHARACTERISTICS COMMAND PACKET
3554      ;
3555 050200 T17PACKET:
3556 050200      .WORD      100004        ;COMMAND PACKET FOR TEST
3557 050202      .WORD      T17DATA      ;WRITE CHARACTERISTICS COMMAND, WITH ACK
3558 050204      .WORD      0           ;ADDRESS OF CHARACTERISTICS BLOCK
3559 050206      .WORD      10.         ;MINIMUM MESSAGE PACKET SIZE
3560
3561 050210 T17DATA:
3562 050210      .WORD      T17BFR      ;CHARACTERISTICS DATA BLOCK
3563 050212      .WORD      0           ;ADDRESS OF MESSAGE BUFFER
3564 050214      .WORD      20.         ;LENGTH OF MESSAGE BUFFER
3565 050216      .WORD      0           ;ESS,ENB,EAI,ERI
3566 050220      .WORD      0           ;EXTENDED FEATURES UNIT NO. E1..
3567
3568      ;
3569      ;MESSAGE BUFFER FOR ALL TEST 6 COMMANDS
3570      ;
3571 050222 T17BFR:
3572 050222      .WORD      0           ;BEGIN MESSAGE BUFFER
3573 050224      .WORD      0           ;MESSAGE TYPE
3574      .WORD      0           ;DATA FIELD LENGTH

```

3574	050226	000000	.WORD	0	;RBPCR
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	000	.BYTE	0	;BSEL0
3596	050361	000	.BYTE	0	;BSEL1
3597	050362	000000	.WORD	0	;SEL2
3598	050364		.BLKB	66.	;WRITE FIFO DATA OUTPUT BUFFER
3599					
3600	050466		ENDTST		
	050466				L10056:
	050466	104401			TRAP CSETST
3601					

```

3603 .SBTTL TEST 7: STATIC TRANSPORT BUS INTERFACE TEST
3604
3605 :+ TEST DESCRIPTION:
3606 :
3607 :
3608 : TEST STEPS:
3609 :
3610 : REPEAT FOR LOOPCNT
3611 : BEGIN
3612 : Write to TSSR register to soft initialize the controller
3613 : Do WRITE CHARACTERISTICS to check for Extended Features Switch
3614 : If Extended Features Hardware Switch Clear then:
3615 :   Do Write Subsystem Write Miscellaneous to Set Extended Features.
3616 : Do WRITE CHARACTERISTICS to select reserved unit 7
3617 : Do a Write Subsystem READ STATUS
3618 : If any transport interface signals are asserted then Print Error
3619 : END
3620 :--
3621
3622
3623 050470 BGNTST
3624 050470
3628 050470 012700 051176 T7::
3629 050474 004737 016510 MOV #TST18ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
3630 050500 012737 000012 002216 JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
3631 050506 MOV #10.,LOOPCNT ;PERFORM 10 ITERATIONS
3632 :
3633 : T18LOOP: Write to TSSR register to soft initialize the controller
3634 : 5$: JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
3635 : 103405 BCS 10$ ;BR IF SOFT INIT OKAY
3636 : 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
3637 : 050516 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
3638 : 050516 104455 TRAP C$ERDF
3639 : 050520 001274 .WORD 700
3640 : 050522 003652 .WORD SFIERR
3641 : 050524 012034 .WORD SFIMSG
3642 :
3643 : 10$: Do WRITE CHARACTERISTICS to check for Extended Features Switch
3644 : CLR FATFLG ;CLEAR FATAL ERROR FLAG
3645 : MOV #T18PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
3646 : JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
3647 : FORCERROR 12$ ;@@DFORCE ERROR IF FORCER=1
3648 : BCS 15$ ;BR IF CARRY SET (GOOD RETURN)
3649 : MOV R0,R1 ;SAVE CONTENTS OF TSSR
3650 : NEXT.ERRNO
3651 : 12$: ERRDF ERRNO,T18SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
3652 : 050562 104455 TRAP C$ERDF
3653 : 050564 001275 .WORD 701
3654 : 050566 051235 .WORD T18SSR
3655 : 050570 012046 .WORD PKTSSR
3656 : 3648 050572 005237 002222 15$: INC FATFLG ;SET FATAL ERROR FLAG
3657 : 050576 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
3658 : TRAP C$CLP1
3659 :
3660 : If Extended Features Hardware Switch Clear then:
3661 : Do Write Subsystem Write Miscellaneous to Set Extended Features.
3662 : MOV #T18BFR,R1 ;MESSAGE BUFFER ADDRESS
3663 : 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	RO,R1	:SAVE CONTENTS OF TSSR
3663	050654				NEXT.ERRNO		
3664	050654		22\$:		ERRDF	ERRNO,T182SSR,PKTSSR	:DEVICE FATAL SSR FAILED TO SET
	050654	104455					TRAP C\$ERDF
	050656	001276					.WORD 702
	050660	051272					.WORD T182SSR
	050662	012046					.WORD PKTSSR
3665	050664	005237	002222		INC	FATFLG	:SET FATAL ERROR FLAG
3666	050670		30\$:		CKLOOP		:LOOP ON ERROR, IF FLAG SET
	050670	104406					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	051600		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	RO,R1	:SAVE CONTENTS OF TSSR
3676	050726				NEXT.ERRNO		
3677	050726		42\$:		ERRDF	ERRNO,T18SSR,PKTSSR	:DEVICE FATAL SSR FAILED TO SET
	050726	104455					TRAP C\$ERDF
	050730	001277					.WORD 703
	050732	051235					.WORD T18SSR
	050734	012046					.WORD PKTSSR
3678	050736	005237	002222		INC	FATFLG	:SET FATAL ERROR FLAG
3679	050742		50\$:		CKLOOP		:LOOP ON ERROR, IF FLAG SET
	050742	104406					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,RO	:SIZE OF MESSAGE BUFFER IN BYTES
3684	050754	105021			CLRB	(R1)+	:CLEAR A BYTE
3685	050756	005300			DEC	RO	: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	RO,R1	:SAVE CONTENTS OF TSSR
3695	051022				NEXT.ERRNO		
3696	051022		62\$:		ERRDF	ERRNO,T183SSR,PKTSSR	:DEVICE FATAL SSR FAILED TO SET
	051022	104455					TRAP C\$ERDF
	051024	001300					.WORD 704
	051026	051336					.WORD T183SSR
	051030	012046					.WORD PKTSSR

```

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    70$
      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      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:      ;MASK OF UNUSED BITS IN READ STATUS BYTES
3732 051142      377      ;BYTE 0 MASK
3733 051143      037      ;BYTE 1
3734 051144      100      ;BYTE 2
3735 051145      000      ;MAKE IT EVEN
3736
3737 051146      T18EXP:      ;EXPECTED DATA BUFFER
3738 051146      000000      ;MESSAGE TYPE
3739 051150      000000      ;DATA FIELD LENGTH
3740 051152      000000      ;RBPGR
3741 051154      000000      ;XST0
3742 051156      000000      ;XST1
3743 051160      000000      ;XST2
3744 051162      000000      ;XST3
3745 051164      000000      ;XST4 (ALWAYS PRESENT FOR WRITE SUB)
3746 051166      000000      ;READ STATUS BYTE 1/0
3747 051170      000000      ;READ STATUS BYTE 2
3748
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: .ASCII 'Static Transport Bus Interface'
3757 051235      127      122      111 T18SSR: .ASCII 'WRITE CHARACTERISTICS Failed'
3758 051272      127      122      111 T182SSR: .ASCII 'WRITE SUBSYSTEM (Write Misc) Failed'
3759 051336      127      122      111 T183SSR: .ASCII 'WRITE SUBSYSTEM (Read Status) Failed'
3760 051403      124      162      141 T18CMP: .ASCII '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+2,2(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:
3816 051660 100004 .WORD 100004 ;COMMAND PACKET FOR TEST
3817 051662 051670 .WORD T18DATA ;WRITE CHARACTERISTICS COMMAND, WITH ACK
3818 051664 000000 .WORD 0 ;ADDRESS OF CHARACTERISTICS BLOCK
3819 051666 000012 .WORD 10. ;MESSAGE PACKET MINIMUM SIZE
3820
3821 051670 T18DATA:
3822 051670 051702 .WORD T18BFR ;CHARACTERISTICS DATA BLOCK
3823 051672 000000 .WORD 0 ;ADDRESS OF MESSAGE BUFFER
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:
3830 051702 000000 .WORD 0 ;MESSAGE BUFFER
3831 051704 000000 .WORD 0 ;MESSAGE TYPE
3832 051706 000000 .WORD 0 ;DATA FIELD LENGTH
3833 051710 000000 .WORD 0 ;RBPGR
3834 051712 000000 .WORD 0 ;XST0
3835 051714 000000 .WORD 0 ;XST1
3836 051716 000000 .WORD 0 ;XST2
3837 051720 000000 .WORD 0 ;XST3
3838 051722 000000 .WORD 0 ;XST4 (ALWAYS PRESENT FOR WRITE SUBSYSTEM
3839 051724 000000 .WORD 0 ;READ STATUS BYTE 1/0 RETURNED
3840 ;READ STATUS BYTE 2
3841
3842 ;WRITE SUBSYSTEM READ STATUS COMMAND PACKET

```

3844		051730			
3846	051730		T18PK2:	.=<.+10>&177770	
3847	051730	100006		.WORD P.WRTSUB!P.ACK	:WRITE SUBSYSTEM WITH ACK
3848	051732	051740		.WORD T18DT2	:LOW ADDRESS OF DATA BLOCK
3849	051734	000000		.WORD 0	:HIGH ADDRESS OF DATA BLOCK
3850	051736	000010		.WORD 8.	:BUFFER EXTENT
3851					
3852	051740		T18DT2:		:DATA BLOCK
3853	051740	000		.BYTE 0	:BSEL0
3854	051741	000		.BYTE 0	:BSEL1
3855	051742	000000		.WORD 0	:SEL2
3856	051744	000000		.WORD 0	:DATA
3857					
3858					
3859	051746		ENDTST		
	051746				
	051746	104401			

L10065: TRAP C\$ETST

```

3861 .SBTTL TEST 8: TRANSPORT BUS INTERFACE LOOPBACK TEST
3862
3863 :++
3864 : TEST DESCRIPTION:
3865 :
3866 : This test verifies the controller's Transport Bus
3867 : drivers, receivers, and signal loopback logic. Note
3868 : that the Static Transport Bus test must have run
3869 : correctly for this test to provide meaningful results.
3870 :
3871 : TEST STEPS:
3872 :
3873 : REPEAT FOR LOOPCNT
3874 : BEGIN
3875 : Do Subtest 1 - Loopback Control signals test
3876 : Do Subtest 2 - Loopback Read/Write signals test
3877 : Do Subtest 3 - Loopback Write Strobe test
3878 : Do Subtest 4 - Loopback Read Strobe test
3879 :
3880 : END
3881 :--
3882 051750 BGNTST
3883 051750
3887 051750 012700 060162 MOV #TST19ID,R0 T8::
3888 051754 004737 016510 JSR PC,TSTSETUP ;ASCII MESSAGE TO IDENTIFY TEST
3889 051760 012737 000012 002216 MOV #10.,LOOPCNT ;DO INITIAL TEST SETUP
3890 051766 T19LOOP: ;PERFORM 10 ITERATIONS
3891
  
```

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

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

.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

[illegible]

```

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 R0,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 104406 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.I1TAD,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 R0,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 104406 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 R0,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
052452 104455
052454 001446
052456 060612
052460 012046

```

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,TSD8(R5)		:SET THE PACKET ADDRESS TO EXECUTE
4035	052504	004737	016336		JSR	PC,CHKTSSR		:WAIT FOR SSR TO SET
4036	052510				FORCERROR	122\$		:@DFFORCE 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,TSD8(R5)		:SET THE PACKET ADDRESS TO EXECUTE
4049	052562	004737	016336		JSR	PC,CHKTSSR		:WAIT FOR SSR TO SET
4050	052566				FORCERROR	132\$		:@DFFORCE 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=RECV (NOT TESTING)
4058	052630	012701	060062		MOV	#T19EXSTA,R1		:GET EXPECTED READ STATUS
4059	052634	012702	062352		MOV	#T19BFSTA,R2		:GET RECV READ STATUS
4060	052640	011211			MOV	(R2),(R1)		:SET EXPD WORD #8 = RECV 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 = RECV (NOT TESTING)
4065	052664	005000			CLR	R0		:HIGH RECV ADDRESS FOR CKMSG2
4066	052666	012701	062332		MOV	#T19BFR,R1		:LOW RECV 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 RECV?
4070	052706				FORCERROR	152\$,NOTSSR		:@D
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				.WORD	809
	052724	061263				.WORD	T197CMP
	052726	013064				.WORD	MSGLOOP
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.ITAD,R0	; ITADO MUST ALWAYS =0	
4082	052760	042700	000040		BIC #WC.ITAD1,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				:@ad	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\$	; @adFORCE 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				:@ad	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\$	; @adFORCE 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$      ;ADD FORCE 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	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			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			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			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 ERROR 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		362\$:	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						
4234	053752			ENDSUB		;////////// END SUBTEST //////////
	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 TEST 8: SUBTEST 2:
4251
4252 SUBTEST DESCRIPTION:
4253
4254 This subtest verifies the Read/Write data loopback path.
4255 The Read/Write data signals are IR<7:0> and IW<7:0>
4256 respectively.
4257
4258 TEST STEPS:
4259
4260 REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
4261 BEGIN
4262 Write to TSSR register to soft initialize the controller
4263 Do WRITE CHARACTERISTICS to check for Extended Features Switch
4264 If Extended Features Hardware Switch Clear then:
4265 Do Write Subsystem Write Miscellaneous to Set Extended Features.
4266 Do WRITE CHARACTERISTICS to select reserved unit 7 and setup BUFFER
4267 Do a Write Subsystem WRITE NPR to set tape direction out and Loopback
4268 Do a WRITE NPR to set loopback and tape direction OUT
4269 Do a WRITE FIFO with byte count equal to 1 and Tape direction OUT
4270 Do a READ FIFO with tape direction OUT to load tape out write latch
4271 Do a WRITE NPR to set loopback and tape direction IN
4272 Do a WRITE FIFO with byte count equal to 1 and Tape direction IN
4273 to strobe loopback data into FIFO.
4274 Do a READ FIFO with tape direction IN to read data
4275 If Data read from FIFO NOT= to Data sent Then Print Error
4276 Do a Write Subsystem READ STATUS
4277 If Input Ready NOT=1 Then Print Error
4278 If Output Ready NOT=0 Then Print Error
4279 If Data In Miss NOT=0 Then Print Error
4280 END
4281
4282 --
4283 BGNSUB ;//////////////// BEGIN SUBTEST //////////////////
4284 053766 ;
4285 053766 T8.2:
4286 053766 104402 TRAP C$BSUB
4287
4288 5$: Write to TSSR register to soft initialize the controller
4289
4290 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
4291 BCS 10$ ;BR IF SOFT INIT OKAY
4292 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4293 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
4294
4295 104455 TRAP C$ERDF
4296 054000 .WORD 816
4297 054002 .WORD SFIERR
4298 054004 .WORD SFIMSG
4299 054006 012034
4300
4301 10$: Do WRITE CHARACTERISTICS to check for Extended Features Switch
4302 CLR FATFLG ;CLEAR FATAL ERROR FLAG
4303 MOV #T19PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
4304 JSR PC,WRTCHR ;DO WRITE CHARACTERISTICS COMMAND
4305 FORCERROR 12$ ;@ADFORCE ERROR IF FORCER=1
4306 BCS 15$ ;BR IF CARRY SET (GOOD RETURN)
4307 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4308 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      15$:  INC  FATFLG      ;SET FATAL ERROR FLAG
4299 054060      15$:  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      30$:  INC  FATFLG      ;SET FATAL ERROR FLAG
4315 054152      30$:  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      50$:  INC  FATFLG      ;SET FATAL ERROR FLAG
4325 054220      50$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      054220 104406                                     TRAP  C$CLP1
4326
4327
4328      ; REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE
4329 054222 012703 002752      100$:  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			ERRDF	ERRNO,T194SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET
	054314	104455				TRAP	C\$ERDF
	054316	001464				.WORD	820
	054320	060371				.WORD	T194SSR
	054322	012046				.WORD	PKTSSR
4345	054324	005237	002222	INC	FATFLG		;SET FATAL ERROR FLAG
4346	054330			CKLOOP			;LOOP ON ERROR, IF FLAG SET
	054330	104406				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			ERRDF	ERRNO,T195SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET
	054402	104455				TRAP	C\$ERDF
	054404	001465				.WORD	821
	054406	060434				.WORD	T195SSR
	054410	012046				.WORD	PKTSSR
4359	054412	005237	002222	INC	FATFLG		;SET FATAL ERROR FLAG
4360	054416			CKLOOP			;LOOP ON ERROR, IF FLAG SET
	054416	104406				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			ERRDF	ERRNO,T196SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET
	054464	104455				TRAP	C\$ERDF
	054466	001466				.WORD	822
	054470	060500				.WORD	T196SSR
	054472	012046				.WORD	PKTSSR
4372	054474	005237	002222	INC	FATFLG		;SET FATAL ERROR FLAG
4373	054500			CKLOOP			;LOOP ON ERROR, IF FLAG SET
	054500	104406				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\$				
4382	054544	103407		BCS	150\$				;@@DFORCE ERROR IF FORCER=1
4383	054546	010001		MOV	R0,R1				;BR IF CARRY SET (GOOD RETURN)
4384	054550			NEXT.ERRNO					;SAVE CONTENTS OF TSSR
4385	054550			ERRDF	ERRNO,T194SSR,PKTSSR	142\$:			;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			CKLOOP		150\$:			;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			ERRDF	ERRNO,T195SSR,PKTSSR	162\$:			;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			CKLOOP		170\$:			;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			ERRDF	ERRNO,T196SSR,PKTSSR	182\$:			;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			CKLOOP		190\$:			;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	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

4423	054772	012702	060042	MOV	#T19EXP,R2	;EXPD ADDRESS	
4424	054776	012703	000022	MOV	#18,R3	;NUMBER OF BYTES TO COMPARE	
4425	055002	004737	011500	JSR	PC,CKMSG2	;EXPD EQUAL RECV?	
4426	055006			FORCERROR	202\$,NOTSSR	;@@D	
4427	055016	103404		BCS	210\$	;BR IF YES	
4428	055020			NEXT.ERRNO			
4429	055020			202\$:	ERRHRD ERRNO,T199CMP,MSGSUB	;REPORT ERROR	
	055020	104456				TRAP	C\$ERHRD
	055022	001472				.WORD	826
	055024	061440				.WORD	T199CMP
	055026	013742				.WORD	MSGSUB
4430	055030			210\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET	
	055030	104406				TRAP	C\$CLP1
4431				:	Do a Write Subsystem READ STATUS		
4432				:	If Input Ready NOT=1 Then Print Error		
4433				:	If Output Ready NOT=0 Then Print Error		
4434				:	If Data In Miss NOT=0 Then Print Error		
4435	055032	004737	061760	JSR	PC,T19SRD	;SETUP PACKET FOR READ STATUS	
4436	055036	012704	062460	MOV	#T19PK2,R4	;SET WRITE SUBSYSTEM COMMAND PACKET	
4437	055042	010465	000000	MOV	R4,TSD8(R5)	;SET THE PACKET ADDRESS TO EXECUTE	
4438	055046	004737	016336	JSR	PC,CHKTSSR	;WAIT FOR SSR TO SET	
4439	055052			FORCERROR	212\$	;@@D FORCE ERROR IF FORCER=1	
4440	055066	103407		BCS	220\$	;BR IF CARRY SET (GOOD RETURN)	
4441	055070	010001		MOV	R0,R1	;SAVE CONTENTS OF TSSR	
4442	055072			NEXT.ERRNO			
4443	055072			212\$:	ERRDF ERRNO,T193SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	055072	104455				TRAP	C\$ERDF
	055074	001473				.WORD	827
	055076	060324				.WORD	T193SSR
	055100	012046				.WORD	PKTSSR
4444	055102	005237	002222	INC	FATFLG	;SET FATAL ERROR FLAG	
4445	055106			220\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET	
	055106	104406				TRAP	C\$CLP1
4446	055110	004737	062220	JSR	PC,T19SETEXP	;SET WORDS 0-7 EXPD=RECV (NOT TESTING)	
4447	055114	012701	060062	MOV	#T19EXSTA,R1	;GET EXPECTED READ STATUS	
4448	055120	012702	062352	MOV	#T19BFSTA,R2	;GET RECV READ STATUS	
4449	055124	012221		MOV	(R2)+,(R1)+	;SET EXPD WORD #8 = RECV TEMP	
4450	055126	011211		MOV	(R2),(R1)	;SET EXPD WORD #9 = RECV TEMP	
4451	055130	052711	000020	BIS	#S2.INRDY,(R1)	;SET EXP INPUT READY= 1	
4452	055134	042711	000040	BIC	#S2.OTRDY,(R1)	;SET EXP OUTPUT READY= 0	
4453	055140	042711	000200	BIC	#S2.DIM,(R1)	;SET EXP DATA IN MISS = 0	
4454	055144	005000		CLR	R0	;HIGH RECV ADDRESS FOR CKMSG2	
4455	055146	012701	062332	MOV	#T195fR,R1	;LOW RECV ADDRESS FOR CKMSG2	
4456	055152	012702	060042	MOV	#T19EXP,R2	;EXPD ADDRESS	
4457	055156	012703	000024	MOV	#20,R3	;NUMBER OF BYTES TO COMPARE	
4458	055162	004737	011500	JSR	PC,CKMSG2	;EXPD EQUAL RECV?	
4459	055166			FORCERROR	232\$,NOTSSR	;@@D	
4460	055176	103404		BCS	240\$	;BR IF YES	
4461	055200			NEXT.ERRNO			
4462	055200			232\$:	ERRHRD ERRNO,T196CMP,MSGSTAT	;REPORT ERROR	
	055200	104456				TRAP	C\$ERHRD
	055202	001474				.WORD	828
	055204	061200				.WORD	T196CMP
	055206	012350				.WORD	MSGSTAT
4463	055210			240\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET	
	055210	104406				TRAP	C\$CLP1
4464							

```
4465
4466
4467 055212          FORCEEXIT          255$
4468 055222 013703 002316 MOV TSTPTR,R3
4469 055226 020327 003062 CMP R3,#TBLEND
4470 055232 103002          BHIS 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:

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 828
 .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

12\$:

ERRDF ERRNO,T19SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET

104402

004737 015774

103405

010001

104455

001474

003652

012034

005037 002222

012704 062310

004737 010662

103407

010001

055332

055332

Line	Address	Offset	Label	Instruction	Comment	Trap	CS
4534	055332	104455				TRAP	C\$ERDF
4535	055334	001475				.WORD	829
	055336	060223				.WORD	T19SSR
	055340	012046				.WORD	PKTSSR
4536	055342	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
4537	055346	104406		CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
4538					If Extended Features Hardware Switch Clear then:		
4539					Do Write Subsystem Write Miscellaneous to Set Extended Features.		
4540	055350	012701	062332	MOV #T19BFR,R1	;MESSAGE BUFFER ADDRESS		
4541	055354	032761	000200	000012	BIT #X2.EXTF,XST2(R1)	;EXTENDED FEATURES SWITCH SET?	
4542	055362	001026		BNE 30\$	;BR IF YES		
4543	055364	004737	062162	JSR PC,T19SEXT	;SETUP PACKET FOR WRITE MISC INVERT		
4544	055370	012704	062460	MOV #T19PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET		
4545	055374	010465	000000	MOV R4,TSDB(R5)	;SET THE PACKET ADDRESS TO EXECUTE		
4546	055400	004737	016336	JSR PC,CHKTSSR	;WAIT FOR SSR TO SET		
4547	055404			FORCERROR 22\$	;@@DFORCE ERROR IF FORCER=1		
4548	055420	103407		BCS 30\$	;BR IF CARRY SET (GOOD RETURN)		
4549	055422	010001		MOV R0,R1	;SAVE CONTENTS OF TSSR		
4550	055424			NEXT.ERRNO			
4551	055424			ERRDF ERRNO,T192SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	TRAP	C\$ERDF
4552	055426	001476				.WORD	830
4553	055430	060260				.WORD	T192SSR
4554	055432	012046				.WORD	PKTSSR
4555	055434	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
4556	055440	104406		CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
4557					Do WRITE CHARACTERISTICS to select reserved unit 7		
4558	055442	012704	062310	MOV #T19PACKET,R4	;GET THE ADDRESS OF COMMAND PACKET		
4559	055446	004737	010662	JSR PC,WRTCHR	;DO WRITE CHARACTERISTICS COMMAND		
4560	055452			FORCERROR 42\$	;@@DFORCE ERROR IF FORCER=1		
4561	055466	103407		BCS 50\$	;BR IF CARRY SET (GOOD RETURN)		
4562	055470	010001		MOV R0,R1	;SAVE CONTENTS OF TSSR		
4563	055472			NEXT.ERRNO			
4564	055472			ERRDF ERRNO,T19SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	TRAP	C\$ERDF
4565	055474	001477				.WORD	831
4566	055476	060223				.WORD	T19SSR
4567	055500	012046				.WORD	PKTSSR
4568	055502	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
4569	055506	104406		CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
4570					REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE		
4571	055510	012703	002752	MOV #TSTBLK,R3	;GET FIRST PATTERN ADDRESS		
4572	055514	012337	002312	MOV (R3)+,DATA	;GET A TEST PATTERN		
4573	055520	042737	177400	BIC #^C<377>,DATA	;DATA IS BYTE		
4574	055526	010337	002316	MOV R3,TSTPTR	;SETUP CURRENT TSTBLK POINTER		
4575					Do a WRITE NPR to set loopback and tape direction OUT		
4576	055532	012700	000100	MOV #NP.OUT,R0	;SET TAPE DIRECTION OUT		
4577	055536	052700	000040	BIS #NP.LOOP,R0	;SET LOOPBACK		
4578	055542	004737	062022	JSR PC,T19SNPR	;SETUP T19PK2 FOR WRITE NPR		
4579	055546	012704	062460	MOV #T19PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET		
4580	055552	010465	000000	MOV R4,TSDB(R5)	;SET THE PACKET ADDRESS TO EXECUTE		
4581	055556	004737	016336	JSR PC,CHKTSSR	;WAIT FOR SSR TO SET		
4582	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			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\$		:ADD FORCE 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			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\$		:ADD FORCE 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			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\$		:ADD FORCE 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 160$: INC FATFLG ;SET FATAL ERROR FLAG
4621 056046 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056046 104406 TRAP C$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$: NEXT.ERRNO
      056114 104455 ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      056116 001504 TRAP C$ERDF
      056120 060612 .WORD 836
      056122 012046 .WORD T198SSR
      056124 005237 002222 .WORD PKTSSR
4633 056124 005237 002222 180$: INC FATFLG ;SET FATAL ERROR FLAG
4634 056130 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056130 104406 TRAP C$CLP1
4635 ;
4636 ; Do a WRITE NPR to set loopback and tape direction IN
4637 056132 005000 CLR R0 ;CLR NP.OUT TO SET TAPE DIRECTION IN
4638 056134 052700 000040 BIS #NP.LOOP,R0 ;SET LOOPBACK
4639 056140 004737 062022 JSR PC,T19SNPR ;SETUP T19PK2 FOR WRITE NPR
4640 056144 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4641 056150 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4642 056154 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4643 056160 FORCERROR 182$ ;@@DFORCE ERROR IF FORCER=1
4644 056174 103407 BCS 190$ ;BR IF CARRY SET (GOOD RETURN)
4645 056176 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
4646 056200
4647 056200 182$: NEXT.ERRNO
      056200 104455 ERRDF ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      056202 001505 TRAP C$ERDF
      056204 060371 .WORD 837
      056206 012046 .WORD T194SSR
      056210 005237 002222 .WORD PKTSSR
4648 056210 005237 002222 190$: INC FATFLG ;SET FATAL ERROR FLAG
4649 056214 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056214 104406 TRAP C$CLP1
4650 ;
4651 056216 012700 000001 Do a WRITE FIFO with byte count equal to 1 and Tape direction IN
4652 056222 012701 002312 MOV #1,R0 ;WRITE 1 BYTE
4653 056226 004737 062066 MOV #DATA,R1 ;FIFO WRITE DATA ADDRESS
4654 056232 012704 062460 JSR PC,T19WFIF ;SETUP T19PK2 FOR WRITE FIFO
4655 056236 010465 000000 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4656 056242 004737 016336 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4657 056246 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4658 056262 103407 FORCERROR 202$ ;@@DFORCE ERROR IF FORCER=1
4659 056264 010001 BCS 210$ ;BR IF CARRY SET (GOOD RETURN)
      MOV R0,R1 ;SAVE CONTENTS OF TSSR
  
```

```

4660 056266
4661 056266 202$: NEXT.ERRNO
      056266 104455 ERRDF ERRNO,T195SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      056270 001506 TRAP C$ERDF
      056272 060434 .WORD 838
      056274 012046 .WORD T195SSR
      056276 005237 002222 .WORD PKTSSR
4662 056276 005237 002222 210$: INC FATFLG ;SET FATAL ERROR FLAG
4663 056302 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056302 104406 TRAP C$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,TSD8(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4669 056324 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4670 056330 FORCERROR 222$ ;@@@FORCE 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$: NEXT.ERRNO
      056350 104455 ERRDF ERRNO,T196SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      056352 001507 TRAP C$ERDF
      056354 060500 .WORD 839
      056356 012046 .WORD T196SSR
      056356 012046 .WORD PKTSSR
4675 056360 005237 002222 230$: INC FATFLG ;SET FATAL ERROR FLAG
4676 056364 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      056364 104406 TRAP C$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$: NEXT.ERRNO
      056450 104456 ERRHRD ERRNO,T19WSTR,MSGSUB ;REPORT ERROR
      056452 001510 TRAP C$ERHRD
      056454 061523 .WORD 840
      056456 013742 .WORD T19WSTR
      056456 013742 .WORD MSGSUB
4692 056460 104406 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
4701
255$:

```

4702 056510
056510
056510

104403

ENDSUB

:/:/:/:/:/:/ END SUBTEST /:/:/:/:/
L10071:

TRAP C\$ESUB

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

TST FATFLG
BEQ 260\$
JSR PC,CKDROP

260\$:

;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
 4744
 4745
 4746
 4747
 4748
 4749
 4750
 4751
 4752
 4753
 4754
 4755
 4756
 4757
 4758
 4759

.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

BGNSUB

////////// BEGIN SUBTEST //////////
 T8.4:

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 840
 .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\$;@ADDFORCE 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

104402

015774

104455

001510

003652

012034

005037 002222

012704 062310

004737 010662

103407

010001

Line	Address	Offset	Label	Operation	Comment	Trap	CS
	056602	104455				TRAP	C\$ERDF
	056604	001511				.WORD	841
	056606	060223				.WORD	T19SSR
	056610	012046				.WORD	PKTSSR
4760	056612	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
4761	056616			15\$: CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
4762	056616	104406					
4763				; If Extended Features Hardware Switch Clear then:			
4764	056620	012701	062332		Do Write Subsystem Write Miscellaneous to Set Extended Features.		
4765	056624	032761	000200	000012	MOV #T19BFR,R1	;MESSAGE BUFFER ADDRESS	
4766	056632	001026			BIT #X2.EXTF,XST2(R1)	;EXTENDED FEATURES SWITCH SET?	
4767	056634	004737	062162		BNE 30\$	;BR IF YES	
4768	056640	012704	062460		JSR PC,T19SEXT	;SETUP PACKET FOR WRITE MISC INVERT	
4769	056644	010465	000000		MOV #T19PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET	
4770	056650	004737	016336		MOV R4,TSDB(R5)	;SET THE PACKET ADDRESS TO EXECUTE	
4771	056654				JSR PC,CHKTSSR	;WAIT FOR SSR TO SET	
4772	056670	103407			FORCERROR 22\$	;@@DFORCE ERROR IF FORCER=1	
4773	056672	010001			BCS 30\$	;BR IF CARRY SET (GOOD RETURN)	
4774	056674				MOV R0,R1	;SAVE CONTENTS OF TSSR	
4775	056674				NEXT.ERRNO		
	056674	104455			22\$: ERRDF ERRNO,T192SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	056676	001512				TRAP	C\$ERDF
	056700	060260				.WORD	842
	056702	012046				.WORD	T192SSR
4776	056704	005237	002222		INC FATFLG	;SET FATAL ERROR FLAG	
4777	056710				30\$: CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP C\$CLP1
4778	056710	104406					
4779	056712	012704	062310		; Do WRITE CHARACTERISTICS to select reserved unit 7		
4780	056716	004737	010662		MOV #T19PACKET,R4	;GET THE ADDRESS OF COMMAND PACKET	
4781	056722				JSR PC,WRTCHR	;DO WRITE CHARACTERISTICS COMMAND	
4782	056736	103407			FORCERROR 42\$	;@@DFORCE ERROR IF FORCER=1	
4783	056740	010001			BCS 50\$	;BR IF CARRY SET (GOOD RETURN)	
4784	056742				MOV R0,R1	;SAVE CONTENTS OF TSSR	
4785	056742				NEXT.ERRNO		
	056742	104455			42\$: ERRDF ERRNO,T19SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	056744	001513				TRAP	C\$ERDF
	056746	060223				.WORD	843
	056750	012046				.WORD	T19SSR
4786	056752	005237	002222		INC FATFLG	;SET FATAL ERROR FLAG	
4787	056756				50\$: CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP C\$CLP1
4788	056756	104406					
4789					; REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE		
4790	056760	012703	002752		MOV #TSTBLK,R3	;GET FIRST PATTERN ADDRESS	
4791	056764	012337	002312	002312	100\$: MOV (R3)+,DATA	;GET A TEST PATTERN	
4792	056770	042737	177400		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
	057052	104455				TRAP	C\$ERDF
	057054	001514				.WORD	844
	057056	060371				.WORD	T194SSR
	057060	012046				.WORD	PKTSSR
4806	057062	005237	002222	105\$: INC	FATFLG		;SET FATAL ERROR FLAG
4807	057066			CKLOOP			;LOOP ON ERROR, IF FLAG SET
	057066	104406				TRAP	C\$CLP1
4808				:	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
	057134	104455				TRAP	C\$ERDF
	057136	001515				.WORD	845
	057140	060612				.WORD	T198SSR
	057142	012046				.WORD	PKTSSR
4819	057144	005237	002222	120\$: INC	FATFLG		;SET FATAL ERROR FLAG
4820	057150			CKLOOP			;LOOP ON ERROR, IF FLAG SET
	057150	104406				TRAP	C\$CLP1
4821				:	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
	057222	104455				TRAP	C\$ERDF
	057224	001516				.WORD	846
	057226	060434				.WORD	T195SSR
	057230	012046				.WORD	PKTSSR
4833	057232	005237	002222	140\$: INC	FATFLG		;SET FATAL ERROR FLAG
4834	057236			CKLOOP			;LOOP ON ERROR, IF FLAG SET
	057236	104406				TRAP	C\$CLP1
4835				:	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 160$: INC FATFLG ;SET FATAL ERROR FLAG
4847 057320 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      057320 104406 TRAP C$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,TSD8(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4854 057344 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4855 057350 FORCERROR 182$ ;@0DFORCE 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 NEXT.ERRNO
4859 057370 182$: ERRDF ERRNO,T194SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      057370 104455 TRAP C$ERDF
      057372 001520 .WORD 848
      057374 060371 .WORD T194SSR
      057376 012046 .WORD PKTSSR
4860 057400 005237 002222 190$: INC FATFLG ;SET FATAL ERROR FLAG
4861 057404 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      057404 104406 TRAP C$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,T19WFM ;SETUP T9PK2 FOR WRITE FORMAT
4865 057414 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4866 057420 010465 000000 MOV R4,TSD8(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4867 057424 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4868 057430 FORCERROR 202$ ;@0DFORCE 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 NEXT.ERRNO
4872 057450 202$: ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      057450 104455 TRAP C$ERDF
      057452 001521 .WORD 849
      057454 060612 .WORD T198SSR
      057456 012046 .WORD PKTSSR
4873 057460 005237 002222 210$: INC FATFLG ;SET FATAL ERROR FLAG
4874 057464 CKLOOP ;LOOP ON ERROR, IF FLAG SET
      057464 104406 TRAP C$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,T19WFM ;SETUP T9PK2 FOR WRITE FORMAT
4878 057476 012704 062460 MOV #T19PK2,R4 ;GET WRITE SUBSYSTEM COMMAND PACKET
4879 057502 010465 000000 MOV R4,TSD8(R5) ;SET THE PACKET ADDRESS TO EXECUTE
4880 057506 004737 016336 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
4881 057512 FORCERROR 222$ ;@0DFORCE 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 NEXT.ERRNO
4885 057532 222$: ERRDF ERRNO,T198SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
  
```

Line	Address	Offset	Label	Instruction	Comment	Trap	Trap Word
	057532	104455				TRAP	C\$ERDF
	057534	001522				.WORD	850
	057536	060612				.WORD	T198SSR
	057540	012046				.WORD	PKTSSR
4886	057542	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
4887	057546			CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
4888	057546	104406					
4889	057550	012700	000001	;	Do a READ FIFO with tape direction IN to read data		
4890	057554	004737	062046	MOV #1,R0	;SET READ BYTE COUNT		
4891	057560	012704	062460	JSR PC,T19RFIF	;SETUP T19PK2 FOR READ FIFO		
4892	057564	010465	000000	MOV #T19PK2,R4	;GET WRITE SUBSYSTEM COMMAND PACKET		
4893	057570	004737	016336	MOV R4,TSD8(R5)	;SET THE PACKET ADDRESS TO EXECUTE		
4894	057574			JSR PC,CHKTSSR	;WAIT FOR SSR TO SET		
4895	057610	103407		FORCERROR 282\$	;@@@FORCE ERROR IF FORCER=1		
4896	057612	010001		BCS 290\$	;BR IF CARRY SET (GOOD RETURN)		
4897	057614			MOV R0,R1	;SAVE CONTENTS OF TSSR		
4898	057614			NEXT.ERRNO			
	057614	104455		ERRDF ERRNO,T196SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	TRAP	C\$ERDF
	057616	001523				.WORD	851
	057620	060500				.WORD	T196SSR
	057622	012046				.WORD	PKTSSR
4899	057624	005237	002222	INC FATFLG	;SET FATAL ERROR FLAG		
4900	057630			CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
4901	057630	104406					
4902	057632	004737	062220	;	If Data read from FIFO NOT= to Data sent Then Print Error		
4903	057636	012701	060062	JSR PC,T19SETEXP	;SET WORDS 0-7 EXPD=RECV (NOT TESTING)		
4904	057642	012702	062352	MOV #T19EXSTA,R1	;GET EXPECTED READ STATUS		
4905	057646	013711	002312	MOV #T19BFSTA,R2	;GET RECV READ STATUS		
4906	057652	016261	000002	MOV DATA,(R1)	;SET EXPD WORD #8 = DATA		
4907	057660	005000		MOV 2(R2),2(R1)	;SET EXPD WORD #9 = RECV (NOT TESTING)		
4908	057662	012701	062332	CLR R0	;HIGH RECV ADDRESS FOR CKMSG2		
4909	057666	012702	060042	MOV #T19BFR,R1	;LOW RECV ADDRESS FOR CKMSG2		
4910	057672	012703	000022	MOV #T19EXP,R2	;EXPD ADDRESS		
4911	057676	004737	011500	MOV #18.,R3	;NUMBER OF BYTES TO COMPARE		
4912	057702			JSR PC,CKMSG2	;EXPD EQUAL RECV?		
4913	057712	103404		FORCERROR 302\$,NOTSSR	;@@@		
4914	057714			BCS 310\$	;BR IF YES		
4915	057714			NEXT.ERRNO			
	057714	104456		ERRHRD ERRNO,T19RSTR,MSGSUB	;REPORT ERROR	TRAP	C\$ERHRD
	057716	001524				.WORD	852
	057720	061630				.WORD	T19RSTR
	057722	013742				.WORD	MSGSUB
4916	057724			310\$: CKLOOP	;LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
4917	057724	104406					
4918	057726			FORCEXIT 355\$	;@@@		
4919	057736	013703	002316	MOV TSTPTR,R3	;RESTORE CURRENT TSTBLK POINTER		
4920	057742	020327	003062	CMP R3,#TBLEND	;END OF TSTBLK?		
4921	057746	103002		BHIS 355\$	;BR IF YES		
4922	057750	000137	056764	JMP 100\$	;DO ANOTHER TSTBLK PATTERN		
4923	057754			355\$:			
4924	057754			ENDSUB	;;;;;;;;;;;;; END SUBTEST ;;;;;;;;;;;;;;		
4925	057754				L10072:		

SEQ 0244

4927 057754 104403
4928 057756 005737 002222
4929 057762 001402
4930 057764 004737 017202
4931 057770 360\$:
4932
4933 057770
057770 104432
057772 002602
4934
4935

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
4991 060223 127 122 111
4992 060260 127 122 111
4993 060324 127 122 111

;+
;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
;-

T19ECTL:
WC.IGO
WC.IFEN
WC.IRWU
WC.IREW
WF.IERASE*256.
WC.ITAD
WC.IOTAD
WC.IFAD
WF.IEDIT*256.
WF.IWFM*256.
WF.IREV*256.
WF.IWRT*256.
WF.IHISP*256.
.WORD 0
.WORD 0
.WORD 0

;WRITE CONTROL DRIVE SIGNALS
;IGO==>IFPT DATA<0>
;IFEN==>IFBY DATA<1>
;IRWU==>IRWD DATA<2>
;IREW==>IDBY DATA<3>
;IFAD==>ILDPA DATA<4>
;ITAD1==>ICNL DATA<5>
;ITAD0==>IRDY DATA<6>
;IERASE==>ISPEED DATA<7>
;IEDIT==>IHER DATA<8>
;IWFM==>IFMK DATA<9>
;IREV==>ICER DATA<10>
;IWRT==>IIDENT DATA<11>
;IHISP==>IEOT DATA<12>
;IRESV2 (UNUSED)DATA<13>
;IRESV1 (UNUSED)DATA<14>
;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

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

;+
;LOCAL TEXT MESSAGES FOR TEST
;-

TST19ID: .ASCII 'Transport Bus Interface Loopback'
T19SSR: .ASCII 'WRITE CHARACTERISTICS Failed'
T192SSR: .ASCII 'WRITE SUBSYSTEM (Write Misc) Failed'
T193SSR: .ASCII 'WRITE SUBSYSTEM (Read Status) Failed'

```

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'

.EVEN

5009						
5010						
5011						
5012						
5013						
5014						
5015	061734					
5016	061734					
5017	061740	012701	062332			
5018	061744	012702	000120			
5019	061750	105021				
5020	061752	005302				
5021	061754	003375				
5022	061756	000207				
5023						
5024						
5025						
5026						
5027	061760					
5028	061760	004737	061734			
5029	061764	012700	062470			
5030	061770	112720	000005			
5031	061774	105010				
5032	061776	000207				
5033						
5034						
5035						
5036						
5037	062000					
5038	062000	004737	061734			
5039	062004	012700	062470			
5040	062010	112720	000010			
5041	062014	112710	000030			
5042	062020	000207				
5043						
5044						
5045						
5046						
5047						
5048						
5049						
5050						

```

: +
: CLEAR MESSAGE BUFFER
: -
T19CLRBUF:
: SAVREG
: MOV #T19BFR,R1
: MOV #T19BEND-T19BFR,R2
10$: CLRB (R1)+
: DEC R2
: BGT 10$
: RTS PC
: SAVE R1-R5 UNTIL NEXT RETURN
: GET MESSAGE BUFFER ADDRESS
: SIZE OF MESSAGE BUFFER IN BYTES
: CLEAR A BYTE
: DONE?
: BR IF NO
: RETURN

: +
: SETUP T19PK2 PACKET FOR READ STATUS
: -
T19SRD:
: JSR PC,T19CLRBUF
: MOV #T19DT2,R0
: MOVB #PW.RDSTATUS,(R0)+
: CLRB (R0)
: RTS PC
: CLEAR MESSAGE BUFFER
: WRITE SUBSYSTEM DATA BUFFER
: STORE READ STATUS COMMAND IN BSEL0
: CLEAR BSEL1
: RETURN

: +
: SETUP T19PK2 PACKET FOR WRITE MISC Reset Tape Status F-FLOPS
: -
T19RSFIF:
: JSR PC,T19CLRBUF
: MOV #T19DT2,R0
: MOVB #PW.WMISC,(R0)+
: MOVB #MS.RSFIF:MS.RSTAP,(R0)
: RTS PC
: CLEAR MESSAGE BUFFER
: WRITE SUBSYSTEM DATA BUFFER
: STORE WRITE MISCELLANEOUS IN BSEL0
: STORE BSEL1 CLEAR FIFO CODES
: RETURN

: +
: SETUP T19PK2 PACKET FOR WRITE NPR
: INPUT:
: R0 CONTAINS BSEL1 NPR DATA
: SETS NP.WRP SINCE IF 0 IT WRITES WRONG PARITY.

```

```

5051
5052 062022      ;T19SNPR:
5053 062022      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
5054 062026      MOV      #T19DT2,R1        ;WRITE SUBSYSTEM DATA BUFFER
5055 062032      MOV      #PW.WNPR,(R1)+    ;STORE WRITE NPR IN BSEL0
5056 062036      BIS      #NP.WRP,R0        ;DON'T WRITE WRONG PARITY
5057 062042      MOV      R0,(R1)           ;STORE NPR DATA IN BSEL1
5058 062044      RTS      PC                ;RETURN
5059
5060      ;+
5061      ; SETUP T19PK2 PACKET FOR READ FIFO
5062      ;
5063      ; INPUT:
5064      ; R0 CONTAINS SEL2 BYTE COUNT
5065      ;-
5066 062046      ;T19RFIF.
5067 062046      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
5068 062052      MOV      #T19DT2,R1        ;WRITE SUBSYSTEM DATA BUFFER
5069 062056      MOV      #PW.RFIFO,(R1)+    ;STORE READ FIFO IN BSEL0
5070 062062      MOV      R0,(R1)+          ;STORE BYTE COUNT IN BSEL1
5071 062064      RTS      PC                ;RETURN
5072
5073      ;+
5074      ; SETUP T19PK2 PACKET FOR WRITE FIFO
5075      ;
5076      ; INPUT:
5077      ; R0 CONTAINS BYTE COUNT
5078      ; R1 CONTAINS DATA PATTERN BLOCK ADDRESS
5079      ;-
5080 062066      ;T19WFIF:
5081 062072      SAVREG      ;SAVE R1-R5 UNTIL NEXT RETURN
5082 062076      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
5083 062102      MOV      #T19DT2,R2        ;WRITE SUBSYSTEM DATA BUFFER
5084 062106      MOV      #PW.WFIFO,(R2)+    ;STORE WRITE FIFO IN BSEL0
5085 062110      MOV      R0,(R2)+          ;STORE BYTE COUNT IN BSEL1
5086 062112      CLR      (R2)+             ;CLEAR SEL2 (UNUSED)
5087 062114      MOV      (R1)+,(R2)+       ;STORE DATA PATTERN BYTE
5088 062116      DEC      R0                ;DONE ALL BYTES?
5089 062120      BGT      10$              ;BR IF NO
5090      RTS      PC                      ;RETURN
5091
5092      ;+
5093      ; SETUP T19PK2 FOR WRITE CONTROL
5094      ;
5095      ; INPUT:
5096      ; R0 CONTAINS DRIVING DATA PATTERN
5097      ;-
5098 062122      ;T19WCTL:
5099 062126      JSR      PC,T19CLRBUF      ;CLEAR MESSAGE BUFFER
5100 062132      MOV      #T19DT2,R'        ;WRITE SUBSYSTEM DATA BUFFER
5101 062136      MOV      #PW.WCTL,(R1)+    ;STORE WRITE CONTROL IN BSEL0
5102      MOV      R0,(R1)+          ;STORE DATA WORD IN BSEL1
5103      RTS      PC                      ;RETURN
5104
5105      ;+
5106      ; SETUP T19PK2 FOR WRITE FORMAT TRANSPORT REGISTER
5107      ;
5108      ; INPUT:
5109      ; R0 CONTAINS DRIVING DATA PATTERN
5110      ;-

```

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
      MOV      #PW.WFMT,(R1)+   ;STORE WRITE FORMAT IN BSEL0
      MOV      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
      MOV      #PW.WMISC,(R0)+  ;STORE WRITE MISCELLANEOUS IN BSEL0
      MOV      #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$:   CLR      (R1)+           ;CLEAR A BYTE
      DEC      R0               ;DONE?
      BGT      10$              ;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      ;RBPOR
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      ;
5212      ;WRITE SUBSYSTEM READ STATUS COMMAND PACKET
5213      ;
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

```

TSVS - HARDWARE TESTS MACRO M1113 25-MAY-82 09:19 PAGE 119-5 F 4
TEST 8: SUBTEST 4 LOOPBACK READ STROBE TEST

SEQ 0250

5226 062474

.BLKB 64.

;WRITE FIFO DATA OUTPUT BUFFER

5227

5228

5229 062574

ENDTST

L10066:

062574

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      MOV      #10.,LOOPCNT      ;PERFORM 10 ITERATIONS
5300 062614      T20LOOP:
5301
5302 062614      BGNSUB      ;////////// BEGIN SUBTEST //////////
5303 062614 104402      T9.1:      TRAP      C$BSUB
5304 062616      :      Write to TSSR register to soft initialize the controller
5305 062616 004737 015774      5$:      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      C$ERDF
5310 062630 001604      .WORD      900
5311 062632 003652      .WORD      SFIERR
5312 062634 012034      .WORD      SFIMSG
5313
5314 062636 005037 002222      :      Do WRITE CHARACTERISTICS to check for Extended Features Switch
5315 062642 012704 066360      10$:      CLR      FATFLG      ;CLEAR FATAL ERROR FLAG
5316 062646 004737 010662      MOV      #T20PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
5317 062652      JSR      PC,WRTCHR      ;DO WRITE CHARACTERISTICS COMMAND
5318 062652 103407      FORCERROR      12$      ;@@DFORCE ERROR IF FORCER=1
5319 062670 010001      BCS      15$      ;BR IF CARRY SET (GOOD RETURN)
5320 062672      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
5321 062672      NEXT.ERRNO
5322 062672 104455      12$:      ERRDF      ERRNO,T20SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
5323 062674 001605      TRAP      C$ERDF
5324 062676 065211      .WORD      901
5325 062700 012046      .WORD      T20SSR
5326 062702 005237 002222      .WORD      PKTSSR
5327 062706 104406      15$:      INC      FATFLG      ;SET FATAL ERROR FLAG
5328 062706      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
5329 062706      TRAP      C$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 104455      22$:      ERRDF      ERRNO,T202SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
5345 062766 001606      TRAP      C$ERDF
5346 062770 065246      .WORD      902
5347      .WORD      T202SSR

```

5334	062772	012046							
5335	062774	005237	002222						
	063000	104406		30\$:	INC	FATFLG		;SET FATAL ERROR FLAG	.WORD PKTSSR
					CKLOOP			;LOOP ON ERROR, IF FLAG SET	
								TRAP	C\$CLP1
5336				:	Do WRITE CHARACTERISTICS to select reserved unit 7				
5337	063002	012704	066360		MOV	#T20PACKET,R4		;GET THE ADDRESS OF COMMAND PACKET	
5338	063006	004737	010662		JSR	PC,WRTCHR		;DO WRITE CHARACTERISTICS COMMAND	
5339	063012				FORCERROR	42\$		;@@DFORCE ERROR IF FORCER=1	
5340	063026	103407			BCS	50\$		;BR IF CARRY SET (GOOD RETURN)	
5341	063030	010001			MOV	R0,R1		;SAVE CONTENTS OF TSSR	
5342	063032				NEXT.ERRNO				
5343	063032			42\$:	ERRDF	ERRNO,T20SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET	
	063032	104455						TRAP	C\$ERDF
	063034	001607						.WORD	903
	063036	065211						.WORD	T20SSR
	063040	012046						.WORD	PKTSSR
5344	063042	005237	002222		INC	FATFLG		;SET FATAL ERROR FLAG	
5345	063046			50\$:	CKLOOP			;LOOP ON ERROR, IF FLAG SET	
	063046	104406						TRAP	C\$CLP1
5346									
5347									
5348				:	REPEAT FOR ALL TEST PATTERNS IN TSTBLK TABLE				
5349	063050	012703	002752		MOV	#TSTBLK,R3		;GET FIRST PATTERN ADDRESS	
5350	063054	012337	002312	100\$:	MOV	(R3)+,DATA		;GET A TEST PATTERN	
5351	063060	042737	177400		BIC	#^C<377>,DATA		;DATA IS BYTE	
5352	063066	010337	002316		MOV	R3,TSTPTR		;SETUP CURRENT TSTBLK POINTER	
5353				:	Do a WRITE NPR to set loopback and tape direction OUT and				
5354				:	and SET Write Wrong Parity.				
5355	063072	012700	000100		MOV	#NP.OUT,R0		;SET TAPE DIRECTION OUT	
5356	063076	052700	000040		BIS	#NP.LOOP,R0		;SET LOOPBACK	
5357	063102	042700	000020		BIC	#NP.WRP,R0		;SET WRITE WRONG PARITY (INVERTED)	
5358	063106	004737	066146		JSR	PC,T20WNPR		;SETUP T20PK2 FOR WRITE NPR	
5359	063112	012704	066530		MOV	#T20PK2,R4		;GET WRITE SUBSYSTEM COMMAND PACKET	
5360	063116	010465	000000		MOV	R4,TSDB(R5)		;SET THE PACKET ADDRESS TO EXECUTE	
5361	063122	004737	016336		JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET	
5362	063126				FORCERROR	102\$		;@@DFORCE ERROR IF FORCER=1	
5363	063142	103407			BCS	105\$		;BR IF CARRY SET (GOOD RETURN)	
5364	063144	010001			MOV	R0,R1		;SAVE CONTENTS OF TSSR	
5365	063146				NEXT.ERRNO				
5366	063146			102\$:	ERRDF	ERRNO,T204SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET	
	063146	104455						TRAP	C\$ERDF
	063150	001610						.WORD	904
	063152	065357						.WORD	T204SSR
	063154	012046						.WORD	PKTSSR
5367	063156	005237	002222		INC	FATFLG		;SET FATAL ERROR FLAG	
5368	063162			105\$:	CKLOOP			;LOOP ON ERROR, IF FLAG SET	
	063162	104406						TRAP	C\$CLP1
5369				:	Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)				
5370	063164	012700	000001		MOV	#WF.I4RES,R0		;IRESV4==>IRSTR = 1	
5371	063170	004737	066242		JSR	PC,T20WFM		;SETUP T20PK2 FOR WRITE FORMAT	
5372	063174	012704	066530		MOV	#T20PK2,R4		;GET WRITE SUBSYSTEM COMMAND PACKET	
5373	063200	010465	000000		MOV	R4,TSDB(R5)		;SET THE PACKET ADDRESS TO EXECUTE	
5374	063204	004737	016336		JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET	
5375	063210				FORCERROR	112\$		;@@DFORCE ERROR IF FORCER=1	
5376	063224	103407			BCS	120\$		;BR IF CARRY SET (GOOD RETURN)	
5377	063226	010001			MOV	R0,R1		;SAVE CONTENTS OF TSSR	
5378	063230				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
5381 063244 120$: INC FATFLG ;SET FATAL ERROR FLAG
      063244 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                     TRAP C$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
5395 063332 160$: INC FATFLG ;SET FATAL ERROR FLAG
      063332 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                     TRAP C$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
5409 063414 180$: INC FATFLG ;SET FATAL ERROR FLAG
      063414 104406 CKLOOP ;LOOP ON ERROR, IF FLAG SET
                                     TRAP C$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      200$:  INC  FATFLG  ;SET FATAL ERROR FLAG
5423 063474      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063474 104406                                     TRAP  C$CLP1
5424      ; Do a WRITE FORMAT to set IRESV4==>IRSTR = 1 (sets read strobe high)
5425 063476 012700 000001      MOV  #WF.14RES,R0  ;IRESV4==>IRSTR = 1
5426 063502 004737 066242      JSR  PC,T20WFM1  ;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      220$:  INC  FATFLG  ;SET FATAL ERROR FLAG
5436 063556      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063556 104406                                     TRAP  C$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      240$:  INC  FATFLG  ;SET FATAL ERROR FLAG
5448 063634      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      063634 104406                                     TRAP  C$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
      063732 104406 260$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      063732 104406 TRAP C$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 T20PK2 FOR WRITE MISC
5469 063744 012704 066530 MOV #T20PK2,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$: ERRDF ERRNO,T202SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      064000 104455 TRAP C$ERDF
      064002 001620 .WORD 912
      064004 065246 .WORD T202SSR
      064006 012046 .WORD PKTSSR
5477 064010 005237 002222 INC FATFLG ;SET FATAL ERROR FLAG
5478 064014 104406 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,T20SETEXP ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
5482 064022 012701 065062 MOV #T20EXSTA,R1 ;GET EXPECTED READ STATUS
5483 064026 012702 066422 MOV #T20BFSTA,R2 ;GET RECV READ STATUS
5484 064032 011211 MOV (R2),(R1) ;SET EXPD WORD #8 = RECV TEMP
5485 064034 016261 000002 000002 MOV 2(R2),2(R1) ;SET EXPD WORD #9 = RECV (NOT TESTED)
5486 064042 042711 100000 BIC #S1.PARERR,(R1) ;SET EXP PAR ERR =0
5487 064046 005000 CLR R0 ;HIGH RECV ADDRESS FOR CKMSG2
5488 064050 012701 066402 MOV #T20BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
5489 064054 012702 065042 MOV #T20EXP,R2 ;EXPD ADDRESS
5490 064060 012703 000024 MOV #20.,R3 ;NUMBER OF BYTES TO COMPARE
5491 064064 004737 011500 JSR PC,CKMSG2 ;EXPD EQUAL RECV?
5492 064070 FORCERROR 302$,NOTSSR ;@@
5493 064100 103404 BCS 320$ ;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 104406 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 T20PK2 FOR WRITE FORMAT
5501 064124 012704 066530 MOV #T20PK2,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				:	Do a WRITE NPR to set loopback and tape direction OUT and		
5512				:	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      R0,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      R0          ;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      R0,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,R0      ;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      R0,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          ; Do a Write Subsystem READ STATUS
5582          ;
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)

```

```

TEST 9: READ/WRITE DATA PARITY TEST

5589 064640 010001      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
5590 064642      NEXT.ERRNO
5591 064642      452$:  ERRDF  ERRNO,T203SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD     920
                                .WORD     T203SSR
                                .WORD     PKTSSR
5592 064652 005237 002222      INC      FATFLG      ;SET FATAL ERROR FLAG
5593 064656      460$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
5594 064660      :      If Read Data parity error NOT=0 Then Print Error
5595 064660 004737 066334      JSR      PC,T20SETEXP ;SET WORDS 0-7 EXPD=RECV (NOT TESTING)
5596 064664 012701 065062      MOV      #T20EXSTA,R1 ;GET EXPECTED READ STATUS
5597 064670 012702 066422      MOV      #T20BFSTA,R2 ;GET RECV READ STATUS
5598 064674 011211      MOV      (R2),(R1) ;SET EXPD WORD #8 = RECV TEMP
5599 064676 016261 000002 000002      MOV      2(R2),2(R1) ;SET EXPD WORD #9 = RECV (NOT TESTED)
5600 064704 042711 100000      BIC      #S1.PARERR,(R1) ;SET EXP PAR ERR =0
5601 064710 005000      .LR      R0 ;HIGH RECV ADDRESS FOR CKMSG2
5602 064712 012701 066402      MOV      #T20BFR,R1 ;LOW RECV ADDRESS FOR CKMSG2
5603 064716 012702 065042      MOV      #T20EXP,R2 ;EXPD ADDRESS
5604 064722 012703 000024      MOV      #20,R3 ;NUMBER OF BYTES TO COMPARE
5605 064726 004737 011500      JSR      PC,CKMSG2 ;EXPD EQUAL RECV?
5606 064732      FORCERROR 472$,NOTSSR ;@@D
5607 064742 103404      BCS      480$ ;BR IF YES
5608 064744      NEXT.ERRNO
5609 064744      472$:  ERRHRD ERRNO,T20CWP,MSGSTAT ;REPORT ERROR
                                TRAP      C$ERHRD
                                .WORD     921
                                .WORD     T20CWP
                                .WORD     MSGSTAT
5610 064754      480$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
                                TRAP      C$CLP1
5611 064756
5612 064756      FORCEEXIT 555$ ;@@D
5613 064766 013703 002316      MOV      TSTPTR,R3 ;RESTORE CURRENT TSTBLK POINTER
5614 064772 020327 003062      CMP      R3,#TBLEND ;END OF TSTBLK?
5615 064776 103002      BHIS 555$ ;BR IF YES
5616 065000 000137 063054      JMP      100$ ;DO ANOTHER TSTBLK PATTERN
5617 065004      555$:
5618 065004      ENDSUB ;///////////////// END SUBTEST ///////////////////
5619 065004      L10074: TRAP      C$ESUB
5620 065004 104403
5621 065006 005737 002222      TST      FATFLG ;ANY FATAL ERRORS ?
5622 065012 001402      BEQ      560$ ;BRANCH IF NOT
5623 065014 004737 017202      JSR      PC,CKDROP ;TRY TO DROP THE UNIT
5624 065020      560$:
5625 065020 004737 016456      JSR      PC,TSTLOOP ;DO ITERATIONS?
5626 065024 103002      BCC      565$ ;BR IF NO
5627 065026 000137 050506      JMP      T18LOOP ;LOOP UNTIL ITERATIONS DONE
5628 065032      565$:
5629 065032      EXIT      TST ;///////////////// EXIT TEST ///////////////////
                                TRAP      C$EXIT
                                .WORD     L10073-
5630 065032 104432
5631 065034 001610

```

```

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: .ASCIIZ 'Read/Write Data Parity'
T20SSR: .ASCIIZ 'WRITE CHARACTERISTICS Failed'
T202SSR: .ASCIIZ 'WRITE SUBSYSTEM (Write Misc) Failed'
T203SSR: .ASCIIZ 'WRITE SUBSYSTEM (Read Status) Failed'
T204SSR: .ASCIIZ 'WRITE SUBSYSTEM (Write Npr) Failed'
T205SSR: .ASCIIZ 'WRITE SUBSYSTEM (Write FIFO) Failed'
T206SSR: .ASCIIZ 'WRITE SUBSYSTEM (Read FIFO) Failed'
T208SSR: .ASCIIZ 'WRITE SUBSYSTEM (Write Format) Failed'
T20SWP: .ASCIIZ 'Read Data Parity Error (PARERR) Failed to Set after Write Wrong Parity'
T20RSF: .ASCIIZ 'Read Data Parity Error (PARERR) Failed to Clear after RESET FIFO'
T20CWP: .ASCIIZ '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 BSEL0
5692	066142	105010		CLRB	(R0)	:CLEAR BSEL1
5693	066144	000207		RTS	PC	:RETURN
5694						
5695						
5696						
5697						
5698						
5699						
5700						
5701						
5702						
5703	066146					
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 BSEL0
5707	066162	110011		MOVB	R0,(R1)	:STORE NPR DATA IN BSEL1
5708	066164	000207		RTS	PC	:RETURN
5709						
5710						
5711						
5712						
5713						
5714						
5715						
5716	066166					
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 BSEL0
5720	066202	110021		MOVB	R0,(R1)+	:STORE BYTE COUNT IN BSEL1
5721	066204	000207		RTS	PC	:RETURN
5722						
5723						
5724						
5725						
5726						
5727						
5728						
5729	066206					
5730	066206					
5731	066212	004737	066102	SAVREG		:SAVE R1-R5 UNTIL NEXT RETURN
5732	066216	012702	066540	JSR	PC,T20CLRBUF	:CLEAR MESSAGE BUFFER
5733	066222	112722	000004	MOV	#T20DT2,R2	:WRITE SUBSYSTEM DATA BUFFER
5734	066226	110022		MOVB	#PW.WFIFO,(R2)+	:STORE WRITE FIFO IN BSEL0
5735	066230	005022		MOVB	R0,(R2)+	:STORE BYTE COUNT IN BSEL1
5736	066232	112122		CLR	(R2)+	:CLEAR SEL2 (UNUSED)
5737	066234	005300		10\$: MOVB	(R1)+,(R2)+	:STORE DATA PATTERN BYTE
5738	066236	003375		DEC	R0	:DONE ALL BYTES?
5739	066240	000207		BGT	10\$	:BR IF NO
5740				RTS	PC	:RETURN
5741						
5742						
5743						
5744						
5745						
5746						

5747	066242			T20WFM:	JSR	PC,T20CLRBUF	:CLEAR MESSAGE BUFFER
5748	066242	004737	066102		MOV	#T20DT2,R1	:WRITE SUBSYSTEM DATA BUFFER
5749	066246	012701	066540		MOVB	#PW.WFMT,(R1)+	:STORE WRITE FORMAT IN BSEL0
5750	066252	112721	000007		MOVB	RO,(R1)+	:STORE DATA WORD IN BSEL1
5751	066256	110021			RTS	PC	:RETURN
5752	066260	000207					
5753				:+			
5754				:-			
5755				:-			
5756				:-			
5757				:-			
5758	066262			T20WMISC:	MOV	#T20DT2,R1	:WRITE SUBSYSTEM DATA BUFFER
5759	066262	012701	066540		MOVB	#PW.WMISC,(R1)+	:STORE WRITE MISCELLANEOUS IN BSEL0
5760	066266	112721	000010		MOVB	RO,(R1)	:STORE INVERT EXTENDED FEATURES IN BSEL1
5761	066272	110011			RTS	PC	:RETURN
5762	066274	000207					
5763				:+			
5764				:-			
5765				:-			
5766	066276			T20SEXT:	MOV	#T20DT2,RO	:WRITE SUBSYSTEM DATA BUFFER
5767	066276	012700	066540		MOVB	#PW.WMISC,(RO)+	:STORE WRITE MISCELLANEOUS IN BSEL0
5768	066302	112720	000010		MOVB	#MS.EXT,(RO)	:STORE INVERT EXTENDED FEATURES IN BSEL1
5769	066306	112710	000200		RTS	PC	:RETURN
5770	066312	000207					
5771				:+			
5772				:-			
5773				:-			
5774	066314			T20CLEXP:	MOV	#T20EXP,R1	:GET EXPD ADDRESS
5775	066314	012701	065042		MOV	#T20EXEND-T20EXP,RO	:GET EXPD SIZE
5776	066320	012700	000120	10\$:	CLRB	(R1)+	:CLEAR A BYTE
5777	066324	105021			DEC	RO	:DONE?
5778	066326	005300			BGT	10\$	:BR IF NO
5779	066330	003375			RTS	PC	:RETURN
5780	066332	000207					
5781				:+			
5782				:-			
5783				:-			
5784				:-			
5785	066334			T20SETEXP:	MOV	#T20EXP,R2	:GET EXPD
5786	066334	012702	065042		MOV	#T20BFR,R3	:GET READ STATUS RECV BUFFER
5787	066340	012703	066402		MOV	#8,RO	:SET WORDS 0-7 EXP=RECV
5788	066344	012700	000010	5\$:	MOV	(R3)+,(R2)+	:SET EXPD=RECV
5789	066350	012322			DEC	RO	:DONE WORDS 0-7 WORDS?
5790	066352	005300			BGT	5\$	:BR IF NO
5791	066354	003375			RTS	PC	:RETURN
5792	066356	000207					
5793							
5794							
5795							
5799							
5800				:-			
5801				:-			
5802	066360			T20PACKET:			:COMMAND PACKET FOR TEST
5803	066360	100004			.WORD	100004	:WRITE CHARACTERISTICS COMMAND, WITH ACK
5804	066362	066370			.WORD	T20DATA	:ADDRESS OF CHARACTERISTICS BLOCK
5805	066364	000000			.WORD	0	
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
5832                .=<.+10>&177770
5833 066530 066530
5834
5835 066530          T20PK2:          ;WRITE SUBSYSTEM WITH ACK
5836 066530 100006   .WORD P.WRTSUB!P.ACK ;LOW ADDRESS OF DATA BLOCK
5837 066532 066540   .WORD T20DT2      ;HIGH ADDRESS OF DATA BLOCK
5838 066534 000000   .WORD 0          ;MINIMUM MESSAGE PACKET SIZE
5839 066536 000012   .WORD 10.
5840
5841 066540          T20DT2:          ;DATA BLOCK
5842 066540 000      .BYTE 0          ;BSEL0
5843 066541 000      .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
5870
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
5938 066646
5942 066646 RFLAGS R0 T10:: ;GET OPERATOR FLAGS
5943 066646 104421 TRAP CSRFLA
5944 066650 001403 BEQ 21$ ;BR, IF OK TO RUN
5945 066652 012700 072230 MOV #T38NE,R0 ;"TEST NOT EXECUTED"
5946 066660 000402 BR 3$ ;JUMP IF NOT FIRST TEST
5947 066660 012700 073345 21$: MOV #T38ID,R0 ;TEST ID MESSAGE
5948 066664 004737 016510 3$: 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$:
5956 066702 005037 002222 2$: CLR FATFLG ;CLEAR THE FATAL ERROR FLAG
5957 066706 012737 176750 071442 MOV #65000,,T38DLY ;SET UP DELAY COUNTER
5958 066714 004737 015774 5$: JSR PC,SOFINIT ;DO A SOFT INIT
5959 066720 103427 BCS 23$ ;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
5964 066732 012727 000250 MOV #250,(PC)+
5965 066736 000000 .WORD 0
5966 066740 013727 002116 MOV LSDLY,(PC)+
5967 066744 000000 .WORD 0
5968 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

23$: MOV #MIMENU,R0
      MOV #6,R1
      JSR PC,GETSEL
      MOV R0,R4
      ASL R4
      JMP @6$(R4)
6$: .WORD 2$
    .WORD 10$
    .WORD 15$
    .WORD 20$
    .WORD 25$
    .WORD 35$
    .WORD 63$
10$: PRINTF #T38MS2

BNE .-4
DEC .-22(PC)
BNE .-20

:TRAP C$ERDF
: .WORD 1001
: .WORD SFIERR
: .WORD SFIMSG

:MENU OF MANUAL INTERVENTIONS
:MAXIMUM ALLOWED SELECTION
:GO GET THE OPERATORS SELECTION
:GET NUMBER FROM ROUTINE
:CONVERT TO WORD OFFSET
:JUMP TO PROPER LOOP
:RETYPE THE MENU
: 1 TURN ON LED'S
: 2 TURN OFF LED'S
: 3 ONLINE ATTENTION
: 4 WRITE PROTECT
: 5 EXTENDED TRANSPORT STATUS
: 6 LEAVE THE TEST
:TELL OPERATOR TO CNTRL-C FOR EXIT
      MOV #T38MS2,-(SP)
      MOV #1,-(SP)
      MOV SP,R0
      TRAP C$PNTF
      ADD #4,SP

:SET PACKET TO INITIAL VALUES
:DO SOFT INIT OF CONTROLLER
:BR IF SOFT INIT = OK
:SAVE CONTENTS OF TSSR
:DEVICE FATAL ERROR DURING INIT
      TRAP C$ERDF
      .WORD 1002
      .WORD SFIERR
      .WORD SFIMSG

:SET UNIT NUMBER

:SUBROUTINE NEEDS PACKET ADDRESS
:ISSUE WRITE CHARACTERISTICS
:BR, IF COMMAND ISSUED OK
:SAVE CONTENTS OF TSSR
:WRITE CHARACTERISTICS FAILED
      TRAP C$ERHRD
      .WORD 1003
      .WORD WRTMSG
      .WORD SFIMSG

110$: MOV #0,T38BS1
      MOV #11,T38BS0
      MOV #T38PACKET,R4

:NOTE: THIS COMMAND TURNS ON THE PROCESSOR FAIL LED

      MOV R4,TSDB(R5)

:BUMP COUNTER DOWN
:BR, IF MORE TIME LEFT
:REPORT FATAL ERROR

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

Address	Offset	Hex	Dec	Label	Instruction	Comment	Register	Value
067410	001756							
067412	005056							
067414	012034							
6054								
6055								
6056								
6057								
6058								
6059	067416			210\$:				
6060	067416	112737	000000	071461	MOVB	#0,T38BS1		
6061	067424	112737	000025	071460	MOVB	#25,T38BS0		
6062	067432	012704	071450		MOV	#T38PACKET,R4		
6063	067436	010465	000000		MOV	R4,TSDB(R5)		
6064	067442	004737	016336		JSR	PC,CHKTSSR		
6065	067446	103405			BCS	250\$		
6066	067450	010001			MOV	R0,R1		
6070	067452				ERRDF	ERRNO,T38SSR,PKTSSR		
	067452	104455						
	067454	001757						
	067456	072646						
	067460	012046						
6071	067462			250\$:	CKLOOP			
6072	067462	104406						
	067464				SETPRI	#PRI07		
	067464	012700	000340					
	067470	104441						
6073	067472	005037	071434		CLR	TTION2		
6074	067476	032737	000100	177560	BIT	#100,@#TTICSR		
6075	067504	001005			BNE	710\$		
6076	067506	005237	071434		INC	TTION2		
6077	067512	052737	000100	177560	BIS	#100,@#TTICSR		
6078	067520	012701	000060		MOV	#TTIVEC,R1		
6079	067524	011137	071436		MOV	(R1),TVSAV2		
6080	067530	012721	070730		MOV	#590\$,(R1)+		
6081	067534	011137	071440		MOV	(R1),TPSAV2		
6082	067540	012711	000340		MOV	#PRI07,(R1)		
6083	067544				SETPRI	#PRI00		
	067544	012700	000000					
	067550	104441						
6084	067552	000240		260\$:	NOP			
6085	067554	000776			BR	260\$		
6086								
6087								
6088	067556			20\$:	PRINTF	#T38MS2		
	067556	012746	073241					
	067562	012746	000001					
	067566	010600						
	067570	104417						
	067572	062706	000004					
6089	067576				SETPRI	#PRI00		
	067576	012700	000000					
	067602	104441						
6090	067604	005037	002224		CLR	INTRECV		
6091	067610	004737	015774		JSR	PC,SOFINIT		

TESTS MACR
INTERVENTION

MACRO M1113

25-MAY-82 09:19 PAGE 122-5

L 5

Address	Offset	Hex	Label	Assembly	Comments	Trap	CS
067620	104455					TRAP	C\$ERDF
067622	001760					.WORD	1008
067624	003652					.WORD	SF IERR
067626	012034					.WORD	SF IMSG
6098	067630		300\$:				
6099	067630	013737	002202	072170	MOV UNITN,T38DSW		
6100	067636	012737	000040	072166	MOV #BITS,T38EAI		
6101	067644	012704	072150		MOV #T38PK2,R4		
6102	067650	004737	010662		JSR PC,WRTCHR		
6103	067654	103405			BCS 310\$		
6107	067656	010001			MOV R0,R1		
6108	067660				ERRHRD ERRNO,WRTMSG,SF IMSG		
	067660	104456					
	067662	001761				TRAP	C\$ERHRD
	067664	005056				.WORD	1009
	067666	012034				.WORD	WRTMSG
						.WORD	SF IMSG
6109	067670		310\$:				
6110	067670	012704	072200		MOV #T38PK3,R4		
6111	067674	010465	000000		MOV R4,TSDB(R5)		
6112	067700	004737	016250		JSR PC,WAITF		
6113	067704	005002			CLR R2		
6114	067706	016501	000002		MOV TSSR(R5),R1		
6115	067712	032701	000100		BIT #OFL,R1		
6116	067716	001402			BEQ 320\$		
6117	067720	052702	000100		BIS #OFL,R2		
6118	067724	052702	000200		BIS #SSR,R2		
6119	067730	020201			CMP R2,R1		
6120	067732	001404			BEQ 350\$		
6124	067734				ERRHRD ERRNO,T38SST,PKTSSR		
	067734	104456					
	067736	001762				TRAP	C\$ERHRD
	067740	073056				.WORD	1010
	067742	012046				.WORD	T38SST
						.WORD	PKTSSR
6125	067744		350\$:		CKLOOP		
6126	067744	104406			PRINTF #T38MS1		
	067746	012746	073146				
	067752	012746	000001				
	067756	010600					
	067760	104417					
	067762	062706	000004				
6127	067766				PRINTF #T38MS2		
	067766	012746	073241				
	067772	012746	000001				
	067776	010600					
	070000	104417					
	070002	062706	000004				
6128	070006				SETPRI #PRI07		
	070006	012700	000340				
	070012	104441					
6129	070014	005037	071434		CLR TTION2		
6130	070020	032737	000100	177560	BIT #100,@TTICSR		
6131	070026	001005			BNE 720\$		
6132	070030	005237	071434		INC TTION2		
6133	070034	052737	000100	177560	BIS #100,@TTICSR		
6134	070042						

TEST	10: MANUAL	INTERVENTION				
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	; "INTERUPT 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			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					10000\$:
6154	070226			BNCOMPLETE	25\$	;RETRY IF ERROR
	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					
6159	070242	004737	015774	27\$:	JSR PC, SOFINIT	;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

Address	Offset	Hex	Label	Instruction	Comment	Trap	Trap Value
6167	070262	104406				TRAP	C\$CLP1
6168	070264	013737	002202	MOV UNITN,T38DSW	:SET UNIT NUMBER		
6169	070272	012704	072170	MOV #T38PK2,R4	:SUBROUTINE NEEDS PACKET ADDRESS		
6170	070276	004737	010662	JSR PC,WRTCHR	:ISSUE WRITE CHARACTERISTICS		
6174	070302	103405		BCS 410\$	:BR, IF COMMAND ISSUED OK		
6175	070304	010001		MOV R0,R1	:SAVE CONTENTS OF TSSR		
6176	070306	104456		ERRHRD ERRNO,WRTMSG,SFIMSG	:WRITE CHARACTERISTICS FAILED	TRAP	C\$SERHRD
	070310	001764				.WORD	1012
	070312	005056				.WORD	WRTMSG
	070314	012034				.WORD	SFIMSG
6177	070316	104406		410\$: CKLOOP	:LOOP IF SELECTED		
6178	070320	013701	071474	MOV T38BFR+6,R1	:PICK UP XSTO CONTENTS	TRAP	C\$CLP1
6179	070324	010102		MOV R1,R2	:SET UP EXPECTED		
6180	070326	052702	000004	BIS #BIT2,R2	:SET UP THE WRITE LOCKED BIT		
6181	070332	020102		CMP R1,R2	:ARE THEY CORRECT		
6182	070334	001406		BEQ 430\$	:BR, IF ALL IS WELL (OK)		
6183	070336	104456		ERRHRD ERRNO,T38WRL,EXPREC	:WRITE LOCKED BIT IS NOT SET ETC."	TRAP	C\$SERHRD
	070340	001765				.WORD	1013
	070342	072464				.WORD	T38WRL
	070344	015474				.WORD	EXPREC
6186	070346	005237	002222	INC FATFLG	:SET FATAL FLAG		
6187	070352	104406		430\$: CKLOOP	:LOOP IF SELECTED		
6188	070354	005737	002222	TST FATFLG	:WAS THE DRIVE NOT WRITE LOCKED	TRAP	C\$CLP1
6189	070360	001402		BEQ 435\$	:BR, IF FLAG NOT SET		
6190	070362	000137	066702	JMP 2\$	:RE-WRITE MENU		
6191	070366	017737	112532	MOV @FREE,T38WR	:SET UP WRITE BUFFER ADDRESS		
6192	070374	012704	072220	MOV #T38PK4,R4	:GET PACKET ADDRESS		
6193	070400	010465	000000	MOV R4,TSD(BR5)	: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	104456		ERRHRD ERRNO,T38WRT,PKTSSR	:TSSR INCORRECT AFTER WRITE COMMAND	TRAP	C\$SERHRD
	070426	001766				.WORD	1014
	070430	072400				.WORD	T38WRT
	070432	012046				.WORD	PKTSSR
6203	070434	104406		440\$: CKLOOP	:LOOP ON ERROR, IF FLAG SET	TRAP	C\$CLP1
6204	070436	013701	071474	MOV T38BFR+6,R1	:READ XSTO CONTENTS		
6205	070442	010102		MOV R1,R2	:SET UP 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	104456		ERRHRD ERRNO,T38WLE,EXPREC	:WRITE LOCK ERROR BIT NOT SET"	TRAP	C\$SERHRD
	070456	001767				.WORD	1015
	070460	072525				.WORD	T38WLE
	070462	015474				.WORD	EXPREC
6213	070464	104406		450\$: CKLOOP	:LOOP IF SELECTED		
6214	070466	000137	066702	JMP 2\$	:GO BACK TO MENU	TRAP	C\$CLP1

```

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$: MOV UNITN,T38DSW
6230 070544 012704 072150 MOV #T38PK2,R4
6231 070550 004737 010662 JSR PC,WRTCHR
6232 070554 103405 BCS 510$
6236 070556 010001 MOV R0,R1
6237 070560
      070560 104456
      070562 001771
      070564 005056
      070566 012034
6238 070570
6239 070570 112737 000000 071461 510$: MOV #0,T38BS1
6240 070576 112737 000020 071460 MOV #20,T38BS0
6241 070604 012704 071450 MOV #T38PACKET,R4
6242 070610 010465 000000 MOV R4,TSDB(R5)
6243 070614 004737 016336 JSR PC,CHKTSSR
6244 070620 103405 BCS 550$
6245 070622 010001 MOV R0,R1
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 CHARACTERISTICS FAILED
TRAP C\$ERHRD
.WORD 1017
.WORD WRTMSG
.WORD SFIMSG

510\$: MOV #0,T38BS1 ;CLEAR BIT #4
MOV #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
TRAP C\$CLP1
MOV #PRI07,R0
TRAP C\$SPRI

555\$: 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
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								
6267								
6268								
6269	070730	010046		590\$:	MOV	R0,-(SP)		:SAVE WORK REGISTER
6270	070732	113700	177562		MOVB	@#TTIBFR,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,T38DLY		: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	R0,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	R0,R1		:SAVE CONTENTS OF TSSR	
6328	071240			ERRHRD	ERRNO,WRTMSG,SFIMSG		:WRITE CHARACTERISTICS 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	MOV	#0,T38BS1	:CLEAR BIT #4	
6331	071260	112737	000024	071460	MOV	#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,T38DB(R5)		:SET THE PACKET ADDRESS	
6334	071276	012737	000144	071442	MOV	#100,T38DLY	:SET UP DELAY ROUTINE	
6335	071304	004737	016250	620\$:	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	L\$DLY,(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	630\$:	JSR	PC,CHKTSSR	:WAIT FOR SSR TO SET	
6343	071364	103405		BCS	650\$		:BR IF CARRY SET (GOOD RETURN)	
6344	071366	010001		MOV	R0,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	

SEQ 0275

6350 071400 104406
 6351 071402 012700 071506
 6352 071406 005001
 6353 071410 005037 003134
 6354 071414 004737 074034
 6355 071420 000137 066702
 6356

MOV #T38BFR+20,R0
 CLR R1
 CLR KTENABLE
 JSR PC,T38MBP
 JMP 2\$

TRAP C\$CLP1
 ;MESSAGE BUFFER ADDRESS
 ;NO HIGH ORDER ADDRESS BITS
 ;NO KT11 STUFF EITHER
 ;GO PRINT MESSAGE BUFFER CONTENTS
 ;GO BACK TO MENU

6357 071424 000137 000200
 6358 071430
 071430 104432
 071432 003054

63\$: JMP 200
 64\$: EXIT TST

;REALLY RETURN TO THE SUPERVISOR
 ;LEAVE TEST

TRAP C\$EXIT
 .WORD L10075-

6359
 6360
 6361
 6362
 6363
 6364
 6365
 6366
 6367
 6368
 6369

;+
 ;LOCAL TEXT MESSAGES FOR TEST
 ;-

;LOCAL STORAGE FOR THIS TEST

;+
 ;LOCAL STORAGE FOR THIS TEST
 ;-

6370 071434 000000
 6371 071436 000000
 6372 071440 000000
 6373
 6374 071442 000000
 6376 071450
 6378 071450
 6379 071450 140006
 6380 071452 071460
 6381 071454 000000
 6382 071456 000012
 6383 071460
 6384 071460 000
 6385 071461 000
 6386 071462 000000
 6387 071464 000000
 6388 071466
 6389 072142 000000

TTION2: .WORD 0
 TVSAV2: .WORD 0
 TPSAV2: .WORD 0

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

T38DLY: .WORD 0
 ;=<.+10>8177770

;DELAY COUNTER FOR TEST

T38PACKET:
 .WORD 140006
 .WORD T38TAD
 .WORD 0
 .WORD 10.

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

T38TAD:
 T38BS0: .BYTE 0
 T38BS1: .BYTE 0
 T38BS2: .WORD 0
 .WORD 0
 T38BFR: .BLKW 150.
 T38EB: .WORD

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

6390
 6391
 6393 072150
 6395 072150
 6396 072150 140004
 6397 072152 072160
 6398 072154 000000
 6399 072156 000012
 6400
 6401

T38PK2: .=<.+10>8177770
 .WORD 140004
 .WORD T38DTA
 .WORD 0
 .WORD 10.

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

;STARTING VALUE OF BLOCK SIZE

6402 072160
 6403 072160 071466
 6404 072162 000000
 6405 072164 000400
 6406 072166 000000
 6407 072170 000000

T38DTA:
 .WORD T38BFR
 .WORD 0
 .WORD 256.
 T38EAI: .WORD 0
 T38DSW: .WORD 0

;SELECT DATA BLOCK
 ;ADDRESS OF MESSAGE BUFFER
 ;LENGTH OF MESSAGE BUFFER
 ;EAI BIT WORD
 ;DRIVE SELECT WORD ETC

6409		072200				.=<.+10>&177770	
6411	072200	140212			T38PK3:	.WORD 140212	:MESSAGE BUFFER RELEASE COMMAND
6412	072202	000000				.WORD 0	:NOT USED
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			.=<.+10>&177770		
6422	072220	140005			T38PK4:	.WORD 140005	:WRITE, ACK, CVC=1 COMMAND
6423	072222	000000			T38WR:	.WORD 0	:ADDRESS OF WRITE BUFFER
6424	072224	000000				.WORD 0	:MORE ADDRESS OF WRITE BUFFER
6425	072226	000400			T38SIZ:	.WORD 256.	:SIZE OF RECORD
6426							
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%A 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%A Interrupt Received'
6446	072766	045	116	045	T38ONL:	.ASCIZ	'%N%A Drive Is Now ON-LINE'
6447	073022	045	116	045	T38OFL:	.ASCIZ	'%N%A 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%AToggle ON-LINE Switch to Generate ATTENTION Interrupts'
6450	073241	045	116	045	T38MS2:	.ASCIZ	'%N%AType RETURN To Return To Menu%
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			10\$:	.ASCIZ	' '
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      T38REST: 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 BSEL0 AND BSEL1
6482 074030 005011          CLR (R1)+      ;CLEAR SEL2
6483 074032 000207          CLR (R1)      ;CLEAR DATA AREA
6484          RTS PC      ;RETURN
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          ;RC      LOW ORDER ADDRESS OF MESSAGE BUFFER
6494          ;R1      HIGH ORDER ADDRESS OF MESSAGE BUFFER
6495          ;NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
6496
6497          ;--
6498
6499
6500
6501 074034      T38MBP: SAVREG          ;SAVE THE REGISTERS
6502 074034      MOV R0,R5      ;SAVE LOW ORDER ADDRESS
6503 074040 010005 003134      TST KTENABLE      ;ADDRESS ABOVE 28K?
6504 074042 005737          BNE 910$      ;BR IF YES
6505 074046 001001          CLR R1      ;SET HIGH ORDER ADDRESS TO 0
6506 074050 005001          MOV R1,R3      ;SAVE HIGH ORDER ADDRESS
6507 074052 010103          ROL R0      ;SHIFT BIT15 TO C BIT
6508 074054 006100          ROL R1      ;SHIFT TO HIGH ORDER FOR PRINTOUT
6509 074056 006101          PRINTX #T38AS0,R1,R5      ;PRINT MESSAGE BUFFER ADDRESS
6510 074060
6511          074060 010546          MOV R5,-(SP)
6512          074062 010146          MOV R1,-(SP)
6513          074064 012746 074336      MOV #T38AS0,-(SP)
6514          074070 012746 000003      MOV #3,-(SP)
6515          074074 010600          MOV SP,R0
6516          074076 104415          TRAP C$PNTX
6517          074100 062706 000010      ADD #10,SP
6518
6519          PRINTX #T38AS1      ;PRINT HEADER FOR CONTENTS
6520
6521          MOV #T38AS1,-(SP)
6522          MOV #1,-(SP)
6523          MOV SP,R0
6524          TRAP C$PNTX
6525          ADD #4,SP
6526
6527          MOV R5,R1      ;COPY LOW ORDER ADDRESS
6528          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:	.ASCIIZ	'%N% Message Buffer Address = %01%05'	
6551	074403	045	116	045	T38AS1:	.ASCIIZ	'%N% Message Buffer Contents:'	
6552	074441	045	101	040	T38ASC:	.ASCIIZ	'%A %D4%A: %03'	
6553	074460	045	116	045	T38ASN:	.ASCIIZ	'%N% Byte%D4%A: %03'	
6554						.EVEN		
6555	074504	000000			T38CNT:	.WORD		:COUNTER FOR PRINT
6556	074506				ENDTST			

TSV5 - HARDWARE TESTS MACRO M1113 25-MAY-82 09:19 PAGE 122-15^I 6
TEST 10: MANUAL INTERVENTION

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 074510
      074510
6601 074510
      074510 104421
6602 074512 001403
6603 074514 012700 076503
6604 074520 000402
6605 074522 012700 077632
6606 074526 004737 016510
6607 074532 004737 020500
6608 074536 103402
6609 074540 000137 075720
6610 074544
6611 074544 004737 015774
6612 074550 103405
6616 074552 010001
6617 074554
      074554 104455
      074556 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

      T11::
      :GET OPERATOR FLAGS
      TRAP CSRFLA

      BEQ 10$
      MOV #T39NE,R0
      BR 11$
      10$: MOV #TST39ID,R0
      11$: 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

```

	074560	003652					.WORD	SFIERR
	074562	012034					.WORD	SFIMSG
6618	074564			25\$:	CKLOOP	;LOOP IF SELECTED	TRAP	C\$CLP1
	074564	104406						
6619	074566	013737	002202	076450	MOV UNITN,T39DSW	:SET UNIT NUMBER		
6620	074574	012704	076430		MOV #T39PK2,R4	:SUBROUTINE NEEDS PACKET ADDRESS		
6621	074600	004737	010662		JSR PC,WRTCHR	:ISSUE WRITE CHARACTERISTICS		
6622	074604	103405			BCS 50\$	:BR, IF COMMAND ISSUED OK		
6626	074606	010001			MOV R0,R1	:SAVE CONTENTS OF TSSR		
6627	074610				ERRHRD ERNO,WRTMSG,SFIMSG	:WRITE CHARACTERISTICSC FAILED		
	074610	104456					TRAP	C\$ERHRD
	074612	002116					.WORD	1102
	074614	005056					.WORD	WRTMSG
	074616	012034					.WORD	SFIMSG
6628	074620			50\$:	CKLOOP	:LOOP IF SELECTED		
	074620	104406					TRAP	C\$CLP1
6629	074622	013701	075760		MOV T39BFR+12,R1	:GET XST2 STATUS FROM MESSAGE BUFFER		
6630	074626				PRINTX #T39SFS	:STATE OF EXTENDED FEATURES SW ='		
	074626	012746	077351				MOV	#T39SFS,-(SP)
	074632	012746	000001				MOV	#1,-(SP)
	074636	010600					MOV	SP,R0
	074640	104415					TRAP	C\$PNTX
	074642	062706	000004				ADD	#4,SP
6631	074646	032701	000200		BIT #BIT7,R1	:CHECK STATE OF E.F.S.		
6632	074652	001011			BNE 100\$	:BR, IF EXT. FEA. SW. IS ON		
6633	074654				PRINTX #T39OFF	:OFF"		
	074654	012746	077475				MOV	#T39OFF,-(SP)
	074660	012746	000001				MOV	#1,-(SP)
	074664	010600					MOV	SP,R0
	074666	104415					TRAP	C\$PNTX
	074670	062706	000004				ADD	#4,SP
6634	074674	000410			BR 110\$	:SKIP OTHER PRINT STATEMENT		
6635	074676			100\$:	PRINTX #T39ON	:ON "		
	074676	012746	077504				MOV	#T39ON,-(SP)
	074702	012746	000001				MOV	#1,-(SP)
	074706	010600					MOV	SP,R0
	074710	104415					TRAP	C\$PNTX
	074712	062706	000004				ADD	#4,SP
6636	074716			110\$:	PRINTX #T39SBS	:STATE OF BUFFERING SWITCH ='		
	074716	012746	077423				MOV	#T39SBS,-(SP)
	074722	012746	000001				MOV	#1,-(SP)
	074726	010600					MOV	SP,R0
	074730	104415					TRAP	C\$PNTX
	074732	062706	000004				ADD	#4,SP
6637	074736	032701	000100		BIT #BIT6,R1	:CHECK STATE OF BUFFERING SW		
6638	074742	001011			BNE 120\$	:BR, IF BUFFERING IS ON		
6639	074744				PRINTX #T39OFF	:OFF"		
	074744	012746	077475				MOV	#T39OFF,-(SP)
	074750	012746	000001				MOV	#1,-(SP)
	074754	010600					MOV	SP,R0
	074756	104415					TRAP	C\$PNTX
	074760	062706	000004				ADD	#4,SP
6640	074764	000410			BR 130\$	:SKIP OTHER PRINT STATEMENT		
6641	074766			120\$:	PRINTX #T39ON	:ON "		
	074766	012746	077504				MOV	#T39ON,-(SP)
	074772	012746	000001				MOV	#1,-(SP)
	074776	010600					MOV	SP,R0

Address	Offset	Hex	Label	Instruction	Comment
075000	104415				
075002	062706	000004			
6642 075006	042701	177700			
6643 075012	010137	077570	130\$:	BIC #177700,R1	:ONLY LEAVE MICROCODE REV LEVEL
6644 075016				MOV R1,T39RL	:LOAD UP REV LEVEL
				PRINTX #T39MCL,T39RL	:MICROCODE REVISION LEVEL =000xxx''
	013746	077570			MOV T39RL,-(SP)
	012746	077513			MOV #T39MCL,-(SP)
	012746	000002			MOV #2,-(SP)
	010600				MOV SP,R0
	104415				TRAP C\$PNTX
	062706	000006			ADD #6,SP
6645 075042	004737	015774		JSR PC,SOFINIT	:DO SOFT INIT OF CONTROLLER
6646 075046	103405			BCS 140\$	:BR IF SOFT INIT = OK
6650 075050	010001			MOV R0,R1	:SAVE CONTENTS OF TSSR
6651 075052				ERRDF ERRNO,SFIERR,SFIMSG	:DEVICE FATAL ERROR DURING INIT
	104455				TRAP C\$ERDF
	002117				.WORD 1103
	003652				.WORD SFIERR
	012034				.WORD SFIMSG
6652 075062			140\$:	CKLOOP	:LOOP IF SELECTED
	104406				TRAP C\$CLP1
6653 075064	013737	002202	076450	MOV UNITN,T39DSW	:SET UNIT NUMBER
6654 075072	012704	076430		MOV #T39PK2,R4	:SUBROUTINE NEEDS PACKET ADDRESS
6655 075076	004737	010662		JSR PC,WRTCHR	:ISSUE WRITE CHARACTERISTICS
6656 075102	103405			BCS 150\$	:BR, IF COMMAND ISSUED OK
6660 075104	010001			MOV R0,R1	:SAVE CONTENTS OF TSSR
6661 075106				ERRHRD ERRNO,WRTMSG,SFIMSG	:WRITE CHARACTERISTICS FAILED
	104456				TRAP C\$ERHRD
	002120				.WORD 1104
	005056				.WORD WRTMSG
	012034				.WORD SFIMSG
6662 075116			150\$:	CKLOOP	:LOOP IF SELECTED
	104406				TRAP C\$CLP1
6663 075120	005737	002226		TST EXTFEA	:CHECK FOR EXTENDED FEATURES SW SWITCH
6664 075124	001036			BNE 174\$	:BR IF SWITCH IS ON
6665 075126	112737	000200	075741	MOVB #200,T39BS1	:WRITE MISCELLANEOUS CONT/READ STATUS
6666 075134	112737	000010	075740	MOVB #10,T39BS0	:FUNCTION SELECTION BIT (TURN ON EXTFEA HW SWITCH)
6667 075142	012704	075730		MOV #T39PACKET,R4	:WRITE SUBSYS MEM PACKET
6668 075146	010465	000000		MOV R4,TSDB(R5)	:ISSUE COMMAND
6669 075152	004737	016336		JSR PC,CHKTSSR	:WAIT FOR SSR
6670 075156	103405			BCS 160\$	:BR, IF NO ERROR
6671 075160	010001			MOV R0,R1	:ERROR, SAVE TSSR
6675 075162				ERRHRD ERRNO,T39NBA,PKTSSR	:TSSR NOT CORRECT AFTER WRT. MISCELLANEOUS
	104456				TRAP C\$ERHRD
	002121				.WORD 1105
	077205				.WORD T39NBA
	012046				.WORD PKTSSR
6676 075172			160\$:	CKLOOP	:LOOP IF SELECTED
	104406				TRAP C\$CLP1
6677 075174	012704	076430		MOV #T39PK2,R4	:SUBROUTINE NEEDS PACKET ADDRESS
6678				*****	
6679				:WRITE CHARACTERISTICS COMMAND (CALL TO WRTCHR)	
6680				*****	
6681					
6682					
6683					
6684 075200	004737	010662		JSR PC,WRTCHR	:ISSUE WRITE CHARACTERISTICS

Address	Offset	Hex	Label	Operation	Comments	Register/Value	Trap	Characteristics
6685	075204	103405		BCS	170\$			
6689	075206	010001		MOV	R0,R1			
6690	075210			ERRHRD	ERRNO,WRTMSG,SFIMSG			
	075210	104456						
	075212	002122						
	075214	005056						
	075216	012034						
6691	075220		170\$:	CKLOOP				
	075220	104406						
6692	075222	005037	002202					
6693	075226	013737	002202	174\$:	CLR	UNITN		
6694	075234	012704	076430	175\$:	MOV	UNITN,T39DSW		
6695	075240	004737	010662		MOV	#T39PK2,R4		
6696	075244	103405			JSR	PC,WRTCHR		
6700	075246	010001			BCS	180\$		
6701	075250				MOV	R0,R1		
	075250	104456			ERRHRD	ERRNO,WRTMSG,SFIMSG		
	075252	002123						
	075254	005056						
	075256	012034						
6702	075260		180\$:	CKLOOP				
	075260	104406						
6703								
6704	075262	016501	000002	190\$:	MOV	TSSR(R5),R1		
6705	075266	032701	000100		BIT	#OFL,R1		
6706	075272	001414			BEQ	200\$		
6707	075274				PRINTX	#T39OF2,UNITN		
	075274	013746	002202					
	075300	012746	076744					
	075304	012746	000002					
	075310	010600						
	075312	104415						
	075314	062706	000006					
6708	075320	000137	075654		JMP	250\$		
6709	075324			200\$:	PRINTX	#T39ON2,UNITN		
	075324	013746	002202					
	075330	012746	077010					
	075334	012746	000002					
	075340	010600						
	075342	104415						
	075344	062706	000006					
6710	075350	013701	075754		MOV	T39BFR+6,R1		
6711	075354	032701	000004		BIT	#BIT2,R1		
6712	075360	001013			BNE	210\$		
6713	075362				PRINTX	#T39WPN,UNITN		
	075362	013746	002202					
	075366	012746	077126					
	075372	012746	000002					
	075376	010600						
	075400	104415						
	075402	062706	000006					
6714	075406	000412			BR	220\$		
6715	075410			210\$:	PRINTX	#T39WRT,UNITN		
	075410	013746	002202					
	075414	012746	077053					
	075420	012746	000002					
	075424	010600						

Line	Address	Offset	Value	Label	Operation	Comment	Register/Value
	075426	104415					
	075430	062706	000006				
6716	075434	012737	125252	076046	220\$:	MOV	#125252,T39BFR+100
6717	075442	112737	000000	075741		MOVB	#0,T39BS1
6718	075450	112737	000024	075740		MOVB	#24,T39BS0
6719	075456	012704	075730			MOV	#T39PACKET,R4
6720	075462	010465	000000			MOV	R4,TSD8(R5)
6721	075466	012737	000144	075724		MOV	#100.,T39DLY
6722	075474	004737	016250		222\$:	JSR	PC,WAITF
6723	075500	016501	000002			MOV	TSSR(R5),R1
6724	075504	032701	000200			BIT	#SSR,R1
6725	075510	001017				BNE	225\$
6726	075512					DELAY	250
	075512	012727	000250				
	075516	000000					
	075520	013727	002116				
	075524	000000					
	075526	005367	177772				
	075532	001375					
	075534	005367	177756				
	075540	001367					
6727	075542	005337	075724			DEC	T39DLY
6728	075546	001352				BNE	222\$
6729	075550	004737	016336		225\$:	JSR	PC,CHKTSSR
6730	075554	103405				BCS	230\$
6731	075556	010001				MOV	R0,R1
6735	075560					ERRHRD	ERRNO,T39NBA,PKTSSR
	075560	104456					
	075562	002124					
	075564	077205					
	075566	012046					
6736	075570				230\$:	CKLOOP	
	075570	104406					
6737	075572	023727	076046	125252		CMP	T39BFR+100,#125252
6738	075600	001013				BNE	240\$
6739	075602					PRINTX	#T39ETN,UNITN
	075602	013746	002202				
	075606	012746	076651				
	075612	012746	000002				
	075616	010600					
	075620	104415					
	075622	062706	000006				
6740	075626	000412				BR	250\$
6741	075630				240\$:	PRINTX	#T39ETS,UNITN
	075630	013746	002202				
	075634	012746	076562				
	075640	012746	000002				
	075644	010600					
	075646	104415					
	075650	062706	000006				
6742	075654	005237	002202		250\$:	INC	UNITN
6743	075660	023727	002202	000003		CMP	UNITN,#3
6744	075666	001402				BEQ	63\$
6745	075670	000137	075226			JMP	175\$
6746	075674				63\$:	PRINTX	#T39NFL
	075674	012746	076500				
	075700	012746	000001				

```
TEST 11: CONFIGURATION TYPEOUT

075704 010600
075706 104415
075710 062706 000004
6747 075714 000137 000200
6748 075720 104432
075722 001736

6749
6750
6751
6752
6753
6754
6755
6756 075724 000000
6758 075730
6760 075730 140006
6761 075730 075740
6762 075732 000000
6763 075734 000012
6764 075736 000
6765 075740 000
6766 075741 000000
6767 075742 000000
6768 075744 000000
6769 075746
6770
6771
6772
6774 076430
6776 076430 140004
6777 076432 076440
6778 076434 000000
6779 076436 000012
6780
6781
6782
6783 076440
6784 076440 075746
6785 076442 000000
6786 076444 000400
6787 076446 000000
6788 076450 000000
6789 076460 076460
6790 076460 140012
6791 076462 000000
6792
6793
6794
6795
6796
6798 076470 076470
6800 076472 140005
6801 076474 000000
6802 076476 000000
6803 076478 000400
6804
6805
6806

64$: JMP 200
EXIT TST

;+
;LOCAL TEXT MESSAGES FOR TEST
;-

;LOCAL STORAGE FOR THIS TEST
;-

T39DLY: .WORD 0
        .=<.+10>8177770
T39PACKET:
        .WORD 140006
        .WORD T39TAD
        .WORD 0
        .WORD 10.
T39TAD:
T39BS0: .BYTE 0
T39BS1: .BYTE 0
T39BS2: .WORD 0
        .WORD 0
T39BFR: .BLKW 150.

        .=<.+10>8177770
T39PK2:
        .WORD 140004
        .WORD T39DTA
        .WORD 0
        .WORD 10.

T39DTA:
        .WORD T39BFR
        .WORD 0
        .WORD 256.
T39EAI: .WORD 0
T39DSW: .WORD 0
        .=<.+10>8177770
T39PK3: .WORD 140012
        .WORD 0
;
;WRITE TAPE PACKET
;
        .=<.+10>8177770
T39PK4: .WORD 140005
T39WR:  .WORD 0
        .WORD 0
T39SI2: .WORD 256.

;RETURN TO SUPERVISOR
;EXIT THIS SECTION

MOV TRAP SP,RO
ADD C$PNTX #4,SP

TRAP C$EXIT
.WORD L10076-.
```

```

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%A Extended Tape Status Available, Drive Number %D2'
6818 076651      045      116      045  T39ETN:  .ASCIZ  '%N%A Extended Tape Status NOT Available, Drive Number %D2'
6819 076744      045      116      045  T39OF2:  .ASCIZ  '%N%A Drive Number %D2%A Is Off-Line'
6820 077010      045      116      045  T39ON2:  .ASCIZ  '%N%A Drive Number %D2%A Is On-Line'
6821 077053      045      116      045  T39WRT:  .ASCIZ  '%N%A Drive Number %D2%A Is Write Protected'
6822 077126      045      116      045  T39WPN:  .ASCIZ  '%N%A Drive Number %D2%A 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%A State Of Extended Features Switch  ='
6827 077423      045      116      045  T39SBS:  .ASCIZ  '%N%A 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%A M7196 Microcode Revision Level      =%02'
6831
6832 077570      000000      T39RL:  .WORD    0
6833      .EVEN
6834      .EVEN
6835      .EVEN
6836
6837      ;+
6838      ;LOCAL STORAGE FOR THIS TEST
6839      ;+
6840 077572      000000      T39DAT:  .WORD    0
6841 077574
6842 077574      T39REST:
6843 077600      012701      075730      SAVREG
6844 077604      012721      140006      MOV      #T39PACKET,R1
6845 077610      012721      075740      MOV      #140006,(R1)+
6846 077614      005021      MOV      #T39TAD,(R1)+
6847 077616      012721      000006      CLR      (R1)+
6848 077622      005021      MOV      #6,(R1)+
6849 077624      005021      CLR      (R1)+
6850 077626      005011      CLR      (R1)+
6851 077630      000207      CLR      (R1)
6852      RTS      PC
6853
6854      ;+
6855      ;LOCAL TEXT MESSAGES FOR TEST
6856      ;+
6857 077632      103      157      156  TST39ID: .ASCIZ  'Configuration Typeout'
6858      .EVEN
6859 077660      ENDTST
6860      077660
6860      077660      104401

```

L10076: TRAP C\$ETST

6862
6863
6864
6865
6866
6867
6868
6869
6870
6871
6872
6873
6874
6875
6876
6877
6878
6879
6880
6881
6882
6883
6884
6885
6886
6887
6888
6889
6890
6891
6892
6893
6894
6895
6896
6897
6898
6899
6900

.SBTTL 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. TYPING <RETURN> CAUSES AN EXIT FROM THE SCOPE LOOP BACK TO MENU LEVEL.

6901 077662
077662
6906 077662
077662 104421
6907 077664 001403
6908 077666 012700 101255
6909 077672 000402
6910 077674 012700 101322
6911 077700 004737 016510
6912 077704 004737 020500
6913 077710 103402
6914 077712 000137 100376
6915 077716 004737 015774
6916 077722 103405
6917 077724 010001
6921 077726
077726 104455
077730 002261

BGNTST

RFLAGS R0

BEQ 1\$
MOV #T4ONE,R0
BR 100\$
MOV #TST40ID,R0
JSR PC,TSTSETUP
JSR PC,CHKMAN
BCS 2\$
JMP 64\$
JSR PC,SOFINIT
BCS 5\$
MOV R0,R1
ERRDF ERRNO,SFIERR,SFIMSG

T12::
:GET OPERATOR FLAGS
TRAP CSRFLA
:BR, IF OK TO RUN
: 'TEST NOT EXECUTED'
: JUST EXIT IF NOT
: TEST ID MESSAGE
: DO THE COMMON SETUP
: SEE IF MANUAL INTERVENTION ALLOWED
: CARRY SET IF INTERVENTION ALLOWED
: EXIT IF NO MANUAL INTERVENTION
: DO A SOFT INIT
: BRANCH IF OK
: CONTENTS OF TSSR REGISTER
: REPORT FATAL ERROR
TRAP CSERDF
.WORD 1201

```

        .WORD      SF1ERR
        .WORD      SF1MSG
;MENU OF SCOPE LOOP SELECTIONS
;MAXIMUM ALLOWED SELECTION
;GO GET THE OPERATORS SELECTION
;WAS ZERO SPECIFIED ?
;REPEAT MENU IF YES.
;EXTENDED TSSR ?
;BRANCH IF NOT
        ;CHECK FOR EXTENDED FEATURES SET
;BR, IF IT IS ON
;WARN OPERATOR EXTENDED FEATURES CLEAR
                                MOV      #EXFMSG,-(SP)
                                MOV      #1,-(SP)
                                MOV      SP,RO
                                TRAP     C$PNTF
                                ADD      #4,SP
;GO BACK TO BASIC MENU
;SAVE THE MENU SELECTION
;RAISE THE PRIORITY
                                MOV      #PRI07,RO
                                TRAP     C$SPRI
;ASSUME INTERRUPTS ARE ENABLED
;ARE TTI INTERRUPTS ON ?
;BRANCH IF YES
;FLAG SET IF INTERRUPTS OFF
;ENABLE INTERRUPTS
;START OF TTI VECTORS
;SAVE THE CURRENT TTI VECTOR
;SET NEW INTERRUPT ROUTINE
;SAVE THE VECTOR PRIORITY
;USE PRIORITY SEVEN
;LOWER INTERRUPT BR LEVEL
                                MOV      #PRI00,RO
                                TRAP     C$SPRI
;CONVERT TO WORD OFFSET
;JUMP TO PROPER LOOP
;RETYPE THE MENU
;TSBA READ ACCESS
;TSSR READ ACCESS
;TSSR WRITE ACCESS
;TSDB HIGH BYTE WRITE ACCESS
;TSDB LOW BYTE WRITE ACCESS
;TSDB MAINTENANCE MODE
;TSDBX WRITE ACCESS
;LEAVE THE TEST

;ENTER MAINTENANCE MODE
;READ TSBA REGISTER
;LOOP UNTIL HALTED

;ADDRESS OF TSSR REGISTER
;POINT TO TSV05'S REGISTERS
;READ TSSR REGISTER
;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$: CLRB   TSDB(R5)    ;ENTER MAINTENANCE MODE
6978 100204 004737 020174      JSR    PC,GETPAT    ;READ THE DATA PATTERN
6979 100210 010001              MOV    R0,R1      ;DATA PATTERN FOR LOOP
6980 100212 012703 000001      MOV    #TSDBH,R3    ;ADDRESS OF HIGH BYTE OF TSDB
6981 100216 060503              ADD    R5,R3      ;POINT TO TSV05'S REGISTERS
6982 100220 110113      27$: MOVB   R1,(R3)    ;WRITE THE DATA TO TSDB, HIGH BYTE
6983 100222 000776              BR     27$      ;LOOP UNTIL STOPPED
6984
6985
6986 100224 105065 000000      30$: CLRB   TSDB(R5)    ;ENTER MAINTENANCE MODE
6987 100230 004737 020174      JSR    PC,GETPAT    ;READ THE DATA PATTERN
6988 100234 010001              MOV    R0,R1      ;DATA PATTERN FOR LOOP
6989 100236 012703 000000      MOV    #TSDB,R3     ;ADDRESS OF TSSR
6990 100242 060503              ADD    R5,R3      ;POINT TO TSV05'S REGISTERS
6991 100244 110113      32$: MOVB   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$: JSR    PC,GETPAT    ;READ THE DATA PATTERN
7003 100274 010001              MOV    R0,R1      ;SAVE THE DATA PATTERN
7004 100276 012703 000003      MOV    #TSSRH,R3    ;BYTE ADDRESS OF TSSR, HIGH BYTE
7005 100302 060503              ADD    R5,R3      ;POINT TO TSV05'S REGISTERS
7006 100304 110113      42$: MOVB   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$: MOV    R0,-(SP)    ;SAVE WORK REGISTER
7015 100312 113700 177562      MOVB   @#TTIBFR,R0  ;GET THE OPERATOR INPUT
7016 100316 042700 000200      BIC    #200,R0    ;STRIP OFF PARITY BIT
7017 100322 122700 000015      CMPB   #15,R0     ;IS IT A CARRIAGE RETURN ?
7018 100326 001021              BNE    61$      ;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
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$, 6$
7049      .WORD      7$, 8$, 9$, 10$, 11$, 12$, 0
7050
7051 100446      012      123      105      1$:      .ASCIIZ <12>'SELECT SCOPE LOOP FROM FOLLOWING OPTIONS:'
7052 100521      012      011      060      2$:      .ASCIIZ <12>' 0      Display This Menu'
7053 100547      011      061      011      3$:      .ASCIIZ ' 1      TSBA Read Access'
7054 100573      011      062      011      4$:      .ASCIIZ ' 2      TSSR Read Access'
7055 100617      011      063      011      5$:      .ASCIIZ ' 3      Initialize (TSSR Write Access,'
7056 100661      011      064      011      6$:      .ASCIIZ ' 4      TSDB High Byte Write Access'
7057 100720      011      065      011      7$:      .ASCIIZ ' 5      TSDB Low Byte Write Access'
7058 100756      011      066      011      8$:      .ASCIIZ ' 6      TSDB Maintenance Mode Write Access'
7059 101024      011      067      011      9$:      .ASCIIZ ' 7      TSDBX (TSSR High Byte) Write Access'
7060 101073      011      070      011     10$:      .ASCIIZ ' 8      Return to Diagnostic Supervisor'
7061 101136      000      11$:      .ASCIIZ ''
7062 101137      124      171      160     12$:      .ASCIIZ 'Type RETURN To Stop Scope Loops'
7063 101177      045      116      045     EXFMSG: .ASCIIZ '%NZA *** Extended Features Switch Not On *** '
7064 101255      123      164      141     T4ONE:  .ASCIIZ 'Stand-alone Scope Loops Not Executed'
7065 101322      123      143      157     TST40ID: .ASCIIZ 'Scope Loops'
7066
7067 101336      .EVEN
      101336      ENDTST
      101336
7068 101340      104401
      ENDMOD
      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$CODE
101344   .WORD     HPM1
101346   .WORD     T$LOLIM
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$LOLIM
101360   .WORD     T$HILIM
36          :GPRMD   HPM3,4,0,340,0,7,YES          ;GET INTERRUPT PRIORITY.
37 101362   ENDHRD
          .EVEN
          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      :++
48      : THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
49      : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
50      : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
51      : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
52      : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
53      : WITH THE OPERATOR.
54      :--
55      101472      BGNSFT
56      101472      .WORD L10101-L$SOFT/2
57      101474      000003
58      L$SOFT::
59      :          GPRML  SPM1,0,-1,YES          ; GET TRANSPORT TEST FLAG.
60      :          GPRML  SPM4,2,-1,YES          ; GET ITERATION CONTROL.
61      :          .WORD  T$CODE
62      :          .WORD  SPM4
63      :          .WORD  -1
64      :          GPRMD  SPM6,4,D,7777,0,7777,YES ; GET LOCAL ERROR LIMIT
65      :          GPRMD  SPM7,6,D,7777,0,7777,YES ; GET GLOBAL ERROR LIMIT
66      :          ENDSFT
67      :          .EVEN
68      101502      L10101:
69
70      60
71      61
72      62 101502      105      116      101  SPM1:  .ASCIZ  'ENABLE TRANSPORT TESTS '
73      63 101532      111      116      110  SPM4:  .ASCIZ  'INHIBIT ITERATIONS '
74      64 101562      120      105      122  SPM6:  .ASCIZ  'PER TEST ERROR LIMIT '
75      65 101612      120      105      122  SPM7:  .ASCIZ  'PER UNIT ERROR LIMIT '
76      66                                     .SBTTL  PATCH AREA
77      67
78      68
79      69      :
80      70      : FINALLY A GENEROUS PATCH AREA.
81      71
82      72      : AND AN ADJUSTMENT TO ACCOUNT FOR THE 'LASTAD BIT7' HACK
83      73      : DESCRIBED IN 'SUPPRG.MEM' (FOR REV C).
84      74      :
85      75      PATCH::
86      76
87      77      .B' KW  32.
88      78
89      80      .=.!377+1
90      82 102000      LASTAD          ;SET LAST USED ADDRESS.
91      :          .EVEN
92      :          .WORD  0
93      :          .WORD  0
94      L$LAST::
95      :          ENDMOD
96      :          .END
97      102000      000000
98      102002      000000
99      102004
100     83 102004
101     84      000001

```

ADDSR 012126 G	CSAU = 000052	DEVDR0 023376	FREEHI 003130	INTCPC 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	FSBGN = 000040	INTVEC 016146
BADSSR 015700 G	CSCLCK= 000062	DIAGMC= 000000	FSCLEA= 000007	INTX 004300
BDVPCR= 177520 G	CSCLEA= 000012	DICEB = 000001	FSDU = 000016	INVERT 021206 G
BENBSW 002230 G	CSCLPS= 000006	DSBINT 016204	FSEND = 000041	IOKCKI= 000200
BIE = 040000	CSCLP1= 000006	DUAD12 004643	FSHARD= 000004	IOKSTP= 000001
BIT0 = 000001 G	CSVEEC= 000036	DUFLG 003112 G	FSHW = 000013	IPRI 002212 G
BIT00 = 000001 G	CSDCLN= 000044	DUMMY 003062	FSINIT= 000006	ISR = 000100 G
BIT01 = 000002 G	CSDDDU= 000051	EF.CON= 000036 G	FSJMP = 000050	IVEC 002210 G
BIT02 = 000004 G	CSDRPT= 000024	EF.NEW= 000035 G	FSMOD = 000000	IXE = 004000 G
BIT03 = 000010 G	CSDU = 000053	EF.PWR= 000034 G	FSMSG = 000011	ISAU = 000041
BIT04 = 000020 G	CSEDIT= 000003	EF.RES= 000037 G	FSPROT= 000021	ISAUTO= 000041
BIT05 = 000040 G	CSERDF= 000055	EF.STA= 000040 G	FSPWR = 000017	ISCLN = 000041
BIT06 = 000100 G	CSERHR= 000056	EMAXDU 016777	FSRPT = 000012	ISDU = 000041
BIT07 = 000200 G	CSERRO= 000060	EN = 000000	FSSEG = 000003	ISHRD = 000041
BIT08 = 000400 G	CSERSF= 000054	ENAINI 016152	FSOFT= 000005	ISINIT= 000041
BIT09 = 001000 G	CSERSO= 000057	ENVIRN 020630	FSSRV = 000010	ISMOD = 000041
BIT1 = 000002 G	CSESCA= 000010	EPRTSW 002200 G	FSSUB = 000002	ISMSG = 000041
BIT10 = 002000 G	CSSESG= 000005	EPRT1 006360	FSSW = 000014	ISPROT= 000040
BIT11 = 004000 G	CSSESUB= 000003	EPRT2 006360	FSTEST= 000001	ISPTAB= 000041
BIT12 = 010000 G	CSSTST= 000001	ERCM 011733	GDDAT 003160 G	ISPRW = 000041
BIT13 = 020000 G	CSXIT= 000032	ERRHI 002236 G	GERRMA 002174 G	ISRPT = 000041
BIT14 = 040000 G	CSGETB= 000026	ERRK 016756	GETPAT 020174 G	ISSEG = 000041
BIT15 = 100000 G	CSGETW= 000027	ERRLO 002240 G	GETSEL 020256 G	ISSETU= 000041
BIT2 = 000004 G	CSGMAN= 000043	ERRNO = 002261	GSCNTO= 000200	ISSFT = 000041
BIT3 = 000010 G	CSGPHR= 000042	ERRVEC= 000004 G	G\$DELM= 000372	ISSRV = 000041
BIT4 = 000020 G	CSGPLO= 000030	ERTABE 003376	G\$DISP= 000003	ISSUB = 000041
BIT5 = 000040 G	CSGPRI= 000040	ERTABL 003176	G\$EXCP= 000400	ISTST = 000041
BIT6 = 000100 G	CSINIT= 000011	ESUM 016760	G\$HILI= 000002	JSJMP = 000167
BIT7 = 000200 G	CSINLP= 000020	EVL = 000004 G	G\$LOLI= 000001	KIPAR0= 172340
BIT8 = 000400 G	CSMANI= 000050	EXBCNT= 000010	G\$NO = 000000	KIPAR1= 172342
BIT9 = 001000 G	CSMEM = 000031	EXFMSG 101177	G\$OFFS= 000400	KIPAR2= 172344
BOE = 000400 G	CSMSG = 000023	EXPBRE 015502 G	G\$OFSI= 000376	KIPAR3= 172346
BRINIT 004457	CSOPEN= 000034	EXPD 002232 G	G\$PRMA= 000001	KIPAR4= 172350
BSELO = 000000	CSPTB= 000014	EXPGOT 004533	G\$PRMD= 000002	KIPAR5= 172352
BSEL1 = 000001	CSPTF= 000017	EXPGT2 004567	G\$PRML= 000000	KIPAR6= 172354
CHKAMB 016044	CSPTS= 000016	EXPMSG 002322 G	G\$RADA= 000140	KIPAR7= 172356
CHKMAN 020500 G	CSPTX= 000015	EXPREC 015474 G	G\$RADB= 000000	KIPDR0= 172300
CHKTSS 016336	CSQIO = 000377	EXTA 005772	G\$RADD= 000040	KIPDR1= 172302
CKDROP 017202	CSRDBU= 000007	EXTEND 005770	G\$RADL= 000120	KIPDR2= 172304
CKEMAX 017102	CSREFG= 000047	EXTFEA 002226 G	G\$RADO= 000020	KIPDR3= 172306
CKMSG 011360 G	CSRESE= 000033	ESEND = 002100	G\$XFER= 000004	KIPDR4= 172310
CKMSG2 011500 G	CSREVI= 000003	ESLOAD= 000035	G\$YES = 000010	KIPDR5= 172312
CKRAM 011114 G	CSRFLA= 000021	FATERR= 000060	HIADDR= 001400	KIPDR6= 172314
CKRAM2 011224 G	CSRPT = 000025	FATFLG 002222 G	HOE = 100000 G	KIPDR7= 172316
CMDPKT 021260 G	CSSEFG= 000046	FERCM 011722	HPM1 101362	KTENAB 003134 G
CMPMEM 017660	CSSPRI= 000041	FIFEXP 012170 G	HPM2 101416	KTFLG 003132 G
CONFIG 017250	CSVEEC= 000037	FIF1MS 012242	HPM3 101442	KTINIT 021054
COUNT 002310 G	CSTPRI= 000013	FIF2MS 012311	IBE = 010000 G	KTOFF 017274
CSRADD 002206 G	DATA 002312 G	FILLME 017422	IDU = 000040 G	KTON 017256
CTAB 003164 G	DATASC 020232	FNOINT 004215	IER = 020000 G	LERRMA 002172 G
CTABE 003176 G	DEBUGM 011632	FORCER 002176 G	IFALT 004256	LERRNO= 000000
CTABM 003164 G	DEVCNT 002220 G	FREE 003124 G	INCERK 017044	LISTAL= 000001

LOE = 040000 G	LSUNIT 002012 G	L10071 056510	OFL = 000100	PRMNO 002320 G
LOOPCN 002216 G	L10000 002164	L10072 057754	ONEFIL= 000000	PRMSG0 014552 G
LOOPCO 013126	L10001 002176	L10073 066644	OSAPTS= 000000	PRMSG1 014777
LOOPFL 003162 G	L10002 005766	L10074 065004	OSAU = 000001	PRMSG2 015035
LOT = 000010 G	L10003 012044	L10075 074506	OSBGNR= 000001	PROASC 014420
LSACP 002110 G	L10004 012062	L10076 077660	OSBGNS= 000001	PR1ASC 014465
LSAPT 002036 G	L10005 012100	L10077 101336	OSDU = 000001	PST32W 003152 G
LSAU 022352 G	L10006 012106	L10100 101362	OSERRT= 000000	PUNIT 022304
LSAUT 002070 G	L10007 012124	L10101 101502	OSGNSW= 000001	PW.D11= 000021
LSAUTO 022556 G	L10010 012142	MEMADD 013754 G	OSPOIN= 000001	PW.D13= 000022
LSCCP 002106 G	L10011 012166	MEMCK 021276 G	OSSETU= 000000	PW.D22= 000020
LSCLEA 022636 G	L10012 012240	MENASC 020447	PASRPT 022054	PW.NOP= 000000
LSCO 002032 G	L10013 012410	MENERR 020374	PATCH 101642 G	PW.NO1= 000023
LSDEPO 002011 G	L10014 013124	MENRES 020476	PATDAT 020230	PW.RDE= 000024
LSDESC 003410 G	L10015 013752	MIMENU 073372	PC.ERA= 002400	PW.RDR= 000001
LSDESP 002076 G	L10016 013774	MMVEC = 000250	PC.IER= 002000	PW.RDS= 000005
LSDEVP 002060 G	L10017 015500	MSA.FR= 000006	PC.NO0= 001000	PW.RFI= 000003
LSDISP 002124 G	L10020 015506	MSA.NO= 000000	PC.REL= 000000	PW.WCT= 000006
LSDLY 002116 G	L10021 015514	MSA.NR= 000004	PC.REW= 000400	PW.WFI= 000004
LSDTP 002040 G	L10022 015526	MSA.VO= 000002	PKBCNT= 000006	PW.WFM= 000007
LSDTYP 002034 G	L10023 015550	MSGEXP 012144 G	PKHI = 000004	PW.WMI= 000010
LSDU 022450 G	L10024 015576	MSGLOO 013064 G	PKLOW = 000002	PW.WNP= 000011
LSDUT 002072 G	L10025 015736	MSGSTA 012350 G	PKTADD 007554	PW.WTR= 000002
LSDVTY 003402 G	L10026 016246	MSGSUB 013742 G	PKTFRM 007516	P.ACK = 100000
LSEF 002052 G	L10030 022302	MS.ATT= 000006	PKTGET 012064 G	P.CMD = 000037
LSENV1 002044 G	L10031 022446	MS.EXT= 000200	PKTMES 012110 G	P.CONT= 000012
LSETP 002102 G	L10032 022554	MS.RSD= 000001	PKTRAM 004745 G	P.CVC = 040000
LSEXP1 002046 G	L10033 022634	MS.RSF= 000020	PKTSSR 012046 G	P.FMT = 000140
LSEXP4 002064 G	L10034 022662	MS.RST= 000010	PNT = 001000 G	P.FORM= 000011
LSEXP5 002066 G	L10035 023124	M8186 005554	PRAMPK 013776	P.GETS= 000017
LSHARD 101342 G	L10036 024424	M8189 005645	PRASC 014523	P.IE = 000200
LSHIME 002120 G	L10037 026416	NBA = 002000	PRBEXP 015470	P.INIi= 000013
LSHPCP 002016 G	L10040 024700	NEWPAS 022010	PRBMSG 015336	P.MODE= 007400
LSHPTP 002022 G	L10041 025144	NODEV 003114 G	PRBREC 015472	P.OPP = 020000
LSHW 002156 G	L10042 031772	NOEXTF 030164	PRBTOT 015423	P.POSI= 000010
LSICP 002104 G	L10043 026772	NOINIT 004335	PRBYTE 015122 G	P.READ= 000001
LSINIT 021556 G	L10044 027262	NOINTR 004221	PRI = 002000 G	P.SWB = 010000
LSLADP 002026 G	L10045 027560	NOITS 002170 G	PRIADD 010160	P.WRIT= 000005
LSLAST 102004 G	L10046 030170	NOMAN 020534	PRIA0 010230	P.WRTC= 000004
LSLOAD 002100 G	L10047 034562	NOMEM 005460	PRIBX0 007612 G	P.WRTS= 000006
LSLUN 002074 G	L10050 032326	NP.IR = 000200	PRIEQU 010060	QVP 002204 G
LSMREV 002050 G	L10051 032720	NP.LOO= 000040	PRIPKT 007370 G	RAMASC 014156
LSNAME 002000 G	L10052 033326	NP.OUT= 000100	PRIRAM 010066	RAMDAT 002242 G
LSPRIO 002042 G	L10053 040354	NP.WRP= 000020	PRITAD 010274	RAMERR 015510 G
LSPROT 021546 G	L10054 035646	NSI 004152	PRITSS 006024	RAMEXP 015530 G
LSPRT 002112 G	L10055 036600	NSINIT 004407	PRIT0 010356	RAMFOR 010116
LSREPP 002062 G	L10056 050466	NUL 004527	PRIT1 010421	RAMSIZ 002302 G
LSREV 002010 G	L10057 040660	NULCR 004530	PRI XOR 007742 G	RAMTAD 015516 G
LSRPT 022664 G	L10060 042070	NXM = 004000	PRI00 = 000000 G	RCVHIA 002304 G
LSOFT 101474 G	L10061 043430	NXMFLG 003136 G	PRI01 = 000040 G	RCVL0A 002306 G
LSGPC 002056 G	L10062 044006	NXMHI 003142 G	PRI02 = 000100 G	RDERR 005206
LSGPCP 002020 G	L10063 045260	NXMLO 003140 G	PRI03 = 000140 G	RECMSG 002466 G
LSPTP 002024 G	L10064 046324	NXMTST 021452	PRI04 = 000200 G	RECV 002234 G
LSSTA 002030 G	L10065 051746	NXR 003740	PRI05 = 000240 G	REGSAV 020140
LSW 002166 G	L10066 062574	NXRERR 005736 G	PRI06 = 000300 G	RETErr 005372
LSTEST 002114 G	L10067 053752	NXR 003777	PRI07 = 000340 G	REWIND 011014 G
LSIML 002014 G	L10070 055240	NXTU 022022	PRMESS 014242	

RMCHBE= 000167	S1.IID= 004000	TST40I 101322	T10 066646 G	T15BFR 033372
RMCHEN= 000200	S1.IIR= 020000	TSV2 002000 G	T11 074510 G	T15BF2 034060
RMMSGB= 000215	S1.I2R= 040000	TSV3 002176 G	T12 077662 G	T15BS0 034060
RMMSGC= 000234	S1.PAR= 100000	TSV4 021546 G	T12BFR 030262	T15BS1 034061
RMPKTB= 000201	S2.ATI= 000010	TSV5 023446 G	T12BKG 030737	T15DAT 033360
RMPKTE= 000210	S2.BTI= 000004	TSV6 101340 G	T12BLK 030314	T15LOO 032024
RMR = 010000	S2.DIM= 000200	TT1BFR= 177562 G	T12CHA 031730	T15PAC 033350
RWPACK 011110	S2.ILW= 000100	TT1CSR= 177560 G	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 G	T12DPR 031116	T15SSR 034066
SCME 005013	TBEND= 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	TFLAG= 000040	T12NIN 031025	T16DAT 040200
SETU 022106	TEXASC 006435	TSGMAN= 000000	T12NXM 031207	T16DT2 040250
SFFMSG 012102 G	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 G	TINERR 012021	TSLSYM= 010000	T12SWR 031620	T16D78 036640
SFPTBL 002166 G	TMPBFR 002632 G	TSLTNO= 000014	T12TBE 030446	T16LEN 037162
SIFLAG 003154 G	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 G	TRANST 002166 G	TSNS2 = 000002	T123LO 027334	T16REJ 037274
SPACE 010466 G	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
SR0 = 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
SVCSUB= 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\$INI= 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\$RPT= 010035	T14RES 026306	T17EXS 046402
SO.IRD= 000100	TST15I 034433	T\$SEG= 010000	T14RST 026344	T17LOO 040404
SO.IRW= 000004	TST16I 036642	T\$SOF= 010101	T14SSR 025773	T17MSK 046356
SO.ISP= 000200	TST17I 046606	T\$SRV= 010026	T14TSB 026155	T17PAC 050200
S1.ICE= 002000	TST18I 051176	T\$SUB= 010074	T142RE 025676	T17PK2 050350
S1.IEO= 010000	TST19I 060162	T\$SW = 010001	T15AM2 034142	T17RFI 050106
S1.IFM= 001000	TST20I 065162	T\$TES= 010077	T15AM3 034243	T17RSF 050004
S1.IHE= 000400	TST39I 077632	T1 023446 G	T15AM4 034345	T17SET 050146

T17SNP	050026	T19SSR	060223	T3	026420	G	T39NBA	077205	WF.IED=	000010	
T17SRD	047764	T19WCT	062122	T3BFLG	003150	G	T39NE	076503	WF.IER=	000004	
T17SSR	046625	T19WFI	062066	T3.1	026446		T39NFL	076500	WF.IHI=	000200	
T17WFD	046504	T19WFM	062142	T3.2	027006		T39OFF	077475	WF.IRE=	000040	
T17WFI	050052	T19WST	061523	T3.3	027276		T39OF2	076744	WF.IWF=	000020	
T171CM	047145	T191CM	060660	T3.4	027574		T39ON	077504	WF.IWR=	000100	
T172CM	047227	T192CM	060742	T38ASC	074441		T39ON2	077010	WF.I3R=	000002	
T172SS	046662	T192SS	060260	T38ASN	074460		T39PAC	075730	WF.I4R=	000001	
T173CM	047323	T193CM	061036	T38AS0	074336		T39PK2	076430	WRTCHR	010662	G
T173SS	046726	T193SS	060324	T38AS1	074403		T39PK3	076460	WRTERR	005113	
T174CM	047407	T194SS	060371	T38BFR	071466		T39PK4	076470	WRTMSG	005056	
T174SS	046773	T195CM	061124	T38BS0	071460		T39RES	077574	WSMBK	021270	G
T175CM	047472	T195SS	060434	T38BS1	071461		T39RL	077570	XFERAS	015740	
T175SS	047036	T196CM	061200	T38BS2	071462		T39SBS	077423	XNXM	016376	
T176CM	047546	T196SS	060500	T38CNT	074504		T39SFS	077351	XORBFO	007674	
T176SS	047102	T197CM	061263	T38DAT	073774		T39SI2	076476	XORFOR	010012	
T177CM	047654	T197SS	060543	T38DLY	071442		T39SSR	077261	XST0 =	000006	G
T18BFR	051702	T198CM	061351	T38DSW	072170		T39TAD	075740	XST1 =	000010	G
T18CMP	051403	T198SS	060612	T38DTA	072160		T39WPN	077126	XST2 =	000012	G
T18DAT	051670	T199CM	061440	T38EAI	072166		T39WR	076472	XST3 =	000014	G
T18DT2	051740	T2	024426	T38EB	072142		T39WRT	077053	XST4 =	000016	G
T18EXP	051146	T2.1	024444	T38ID	073345		T4	031774	XSOBOT=	000002	
T18LOO	050506	T2.2	024714	T38INT	072736		T4.1	032024	XSOEOT=	000001	
T18MSK	051142	T20BEN	066522	T38MBP	074034		T4.2	032330	XSOIE =	000040	
T18PAC	051660	T20BFR	066402	T38MSG	073305		T4.3	032722	XSOILA=	000400	
T18PK2	051730	T20BFS	066422	T38MS1	073146		T4ONE	101255	XSOILC=	001000	
T18SET	051550	T20CLE	066314	T38MS2	073241		T5	034564	XSOLET=	020000	
T18SMI	051526	T20CLR	066102	T38MS3	072305		T5.1	034602	XSOMOT=	000200	
T18SRD	051506	T20CWP	066007	T38NBA	072572		T5.2	035662	XSONEF=	002000	
T18SSR	051235	T20DAT	066370	T38NE	072230		T6	040356	XSOONL=	000100	
T18XS	051172	T20DT2	066540	T38OFL	073022		T6.1	040404	XSOPEO=	000010	
T182SS	051272	T20EXE	065162	T38ONL	072766		T6.2	040674	XSORLL=	010000	
T183SS	051336	T20EXP	065042	T38PAC	071450		T6.3	042104	XSORLS=	040000	
T19BEN	062452	T20EXS	065062	T38PK2	072150		T6.4	043444	XSOTMK=	100000	
T19BFC	057776	T20LOO	062614	T38PK3	072200		T6.5	044022	XSOVCK=	000020	
T19BFR	062332	T20MSK	065036	T38PK4	072220		T6.6	045274	XSOWLE=	004000	
T19BFS	062352	T20PAC	066360	T38RES	073776		T7	050470	XSOWLK=	000004	
T19CLE	062200	T20PK2	066530	T38SI2	072226		T8	051750	XXCOMM	003122	G
T19CLR	061734	T20RFI	066166	T38SSR	072646		T8.1	051766	XSALWA=	000000	
T19CNV	062244	T20RSF	065706	T38SST	073056		T8.2	053766	XSALS=	000040	
T19DAT	062320	T20SET	066334	T38TAD	071460		T8.3	055254	X\$OFFS=	000400	
T19DT2	062470	T20SEX	066276	T38WLE	072525		T8.4	056524	X\$TRUE=	000020	
T19EXE	060162	T20SRD	066126	T38WR	072222		T9	062576	X1.COR=	020000	
T19EXP	060042	T20SSR	065211	T38WRL	072464		T9.1	062614	X1.DLT=	100000	
T19XS	060062	T20SWP	065577	T38WRT	072400		UAM =	000200	X1.MBZ=	017375	
T19LOO	051766	T20WFI	066206	T39BFR	075746		UNITN =	002202	X1.RBP=	000400	
T19MSK	060036	T20WFM	066242	T39BS0	075740		UNREC =	000006	X1.SPA=	040000	
T19PAC	062310	T20WMI	066262	T39BS1	075741		USI	004123	X1.UNC=	000002	
T19PK2	062460	T20WNP	066146	T39BS2	075742		WAITF	016250	X2.BUF=	000100	
T19PRE	057774	T202SS	065246	T39DAT	077572		WC.IFA=	000200	X2.EXT=	000200	
T19RFI	062046	T203SS	065312	T39DLY	075724		WC.IFE=	000002	X2.OPM=	100000	
T19RSF	062000	T204SS	065357	T39DSW	076450		WC.IGO=	000001	X2.RCE=	040000	
T19RST	061630	T205SS	065422	T39DTA	076440		WC.IRE=	000010	X2.REV=	000077	
T19SET	062220	T206SS	065466	T39EAI	076446		WC.IRW=	000004	X2.SPA=	035400	
T19SEX	062162	T208SS	065531	T39ETN	076651		WC.IOT=	000100	X2.UNI=	000007	
T19SNP	062022	T23A	003144	T39ETS	076562		WC.IIT=	000040	X2.WCF=	002000	
T19SRD	061760	T23B	003146	T39MCI	077513		WC.ISR=	000020	X3.DCK=	000010	

X3.MBZ= 000006	X3.REV= 000040	X3.TRF= 000020	X4.MBZ= 017400	X4.TSM= 020000
X3.MDE= 177400	X3.RIB= 000001	X4.HSP= 100000	X4.RCE= 040000	X4.WRC= 000377
X3.OPI= 000100	X3.SPA= 000200			

. 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