

The image displays a large grid of approximately 15 columns and 15 rows of small, illegible data tables or charts. Each cell in the grid contains a small-scale version of a data visualization, possibly a bar chart or a table of values. The text within these cells is too small to be read, but the overall layout suggests a comprehensive set of test results or performance metrics for the TSV05 device. The grid is organized in a regular pattern, with each cell containing a similar but distinct set of data.

B1
D kv w
A [C
1

SEQ 000

USER DOCUMENTATION

MACRO V05.03 Tuesday 28-Apr-87 10:28 Page 2

.REM_
IDENTIFICATION

PRODUCT ID: AC-T093C-MC
PRODUCT TITLE: CVTSACO TSV05 CTRL PART 1
DECO/DEPO: 1.0
DEPARTMENT: COMPUTER SPECIAL SYSTEMS/PGG
DATE: JUNE 4, 1987

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) 1983, 1987 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 LSI-11 RESIDENT DIAGNOSTIC WHICH CHECKS THE FUNCTIONALITY OF A TSV05 MAGTAPE SUBSYSTEM WHILE CONNECTED TO A LSI-11/23 SYSTEM (QBUS). THE PROGRAM PROVIDES ERROR MESSAGES WHICH IDENTIFY FAILING FUNCTIONS THAT AID IN THE REPAIR OF THE DEVICE. THIS DIAGNOSTIC CONSIST OF ELEVEN TEST WHICH ARE EXECUTED IN SEQUENCE.

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

1.2 SYSTEM REQUIREMENTS

LSI-11 PROCESSOR AND MEMORY
CAUTION:DIAGNOSTIC REQUIRES 32K WORDS OF MEMORY
(28K USEABLE I.E. 4K FOR I/O PAGE)
TSV05 MAGTAPE SUBSYSTEM (DRIVE AND CONTROLLER)
CONSOLE TERMINAL
PDP-11 DIAGNOSTIC SUPERVISOR (HSA44.SYS VERSION 34 OR LATER)
PDP-11 DIAGNOSTIC LOADER/MONITOR (XXDP+)

1.3 RELATED DOCUMENTS AND STANDARDS

DIGITAL EQUIPMENT CORPORATION DOCUMENTS:

1. CHBUS 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 1983
3. TSV05 TRANSPORT SUBSYSTEM TECHNICAL MANUAL; DOCUMENT NUMBER EK-TSV05-TM-001
DATE: AUGUST 1983
4. TSV05 TRANSPORT SUBSYSTEM INSTALLATION MANUAL; DOCUMENT NUMBER EK-TSV05-IN-001
DATE: AUGUST 1983

1.4 DIAGNOSTIC HIERARCY PREREQUISITES

FUNCTIONAL LSI-11 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 TSV05 TRANSPORT IS A KNOWN GOOD REEL OF TAPE.

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 LSI-11 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 VTSA??  
DIAG. RUN-TIME SERVICES REV D. APR 79  
CVTSA-C-0  
****TSV05 LOGIC DIAGNOSTIC****  
UNIT IS TSV05  
>DR
```

2.2 SWITCHES

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

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

EXAMPLE OF SWITCH USAGE:

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

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

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

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

2.3 FLAGS

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

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

*ERROR MESSAGES ARE DESCRIBED IN SECTION 3.1

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

/FLAGS:LOE:IER:BOE

2.4 HARDWARE QUESTIONS

WHEN A DIAGNOSTIC IS STARTED, THE RUNTIME SERVICES WILL PROMPT THE USER FOR HARDWARE INFORMATION BY TYPING "CHANGE HW (L) ?" YOU MUST ANSWER "Y" AFTER A START COMMAND UNLESS THE HARDWARE INFORMATION HAS BEEN "PRELOADED" USING THE SETUP UTILITY (SEE CHAPTER 14 OF THE XXDP USER'S MANUAL). WHEN YOU ANSWER THIS QUESTION WITH A "Y", THE RUNTIME SERVICES WILL ASK FOR THE NUMBER OF UNITS (IN DECIMAL).

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

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

TSBA/TSDB = 172520, VECTOR = 224

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

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

UNIT 0

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

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

THE ADDRESS AND VECTOR QUESTIONS WILL BE ASKED FOR EACH OF THE NUMBER OF UNITS (CONTROLLERS) SPECIFIED IN THE "# UNITS?" QUESTION. LOGICAL UNIT NUMBERS ARE ASSIGNED IN ORDER, BEGINNING AT 0. UP TO FOUR UNITS CAN BE SELECTED FOR TESTING AS FOLLOWS:
UP TO 4 TSV05 CONTROLLERS PER LSI-11 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 (0) ? 160000<CR>
SUB-DEVICE # (0) ? 0<CR>
Q-FACTOR (0) 0 ? 1<CR>
UNIT 2
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 1<CR>
Q-FACTOR (0) 1 ? 0<CR>
UNIT 3
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 2<CR>
Q-FACTOR (0) 0 ? <CR>
UNIT 4
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 3<CR>
Q-FACTOR (0) 0 ? <CR>
UNIT 5
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 4<CR>
Q-FACTOR (0) 0 ? <CR>
UNIT 6
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 5<CR>
Q-FACTOR (0) 0 ? <CR>
UNIT 7
CSR ADDRESS (0) ? 160000<CR>
SUB-DEVICE # (0) ? 6<CR>
Q-FACTOR (0) 0 ? 1<CR>
UNIT 8
CSR ADDRESS (0) 160000<CR>
SUB-DEVICE # (0) ? 7<CR>
Q-FACTOR (0) 1 ? <CR>
```

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

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

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

```
# UNITS (D) ? 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 (D) ? 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. TYPE "R NAME", WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM
3. TYPE "START"
4. ANSWER THE "CHANGE HW" QUESTION WITH "Y"
5. ANSWER ALL THE HARDWARE QUESTIONS
6. 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 "IXE" FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL ERROR MESSAGE AND ANY ASSOCIATED BASIC ERROR MESSAGES.

3.2 SPECIFIC ERROR MESSAGES

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
CVTSA 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: 100C20	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 AN UNRECOVERABLE ERROR OCCURED WHICH INDICATES THAT THE CONTROLLER MAY BE DEFECTIVE.

CVTSA 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.

CVTS 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 (LSI-11)

DR>STA/FLA:PNT:HOE

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 TWO SWITCHES ON WHICH ARE "PRINT EACH TEST NBR AS EXECUTED" AND "HALT ON ERROR".

TST: 001 INITIALIZE #1
TST: 002 WRAP DATA HIGH BYTE TEST
TST: 003 WRAP DATA LOW BYTE TEST
TST: 004 RAM TEST
TST: 005 INITIALIZE 2 TEST
TST: 006 COMMAND REJECT TEST
TST: 007 WRITE CHARACTERISTICS TEST
TST: 008 VOLUME CHECK
TST: 009 COMPLETION INTERRUPT TEST
TST: 010 BASIC PACKET PROTOCOL TEST
TST: 011 NON-TAPE-MOTION COMMANDS TEST

0 ERRORS

NOTE: THE DIAGNOSTIC WILL RUN CONTINUOUSLY UNLESS A PASS NUMBER 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 LSI-11 PROCESSOR WITH A LA34 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	1	30	29
2	1	10	9
3	1	8	7
4	25	120	95
5	5	140	135
6	25	475	450
7	20	20	0
8	1	10	9
9	20	20	0
10	1	2	1
11	8	11	3

THE TIMES REQUIRED TO RUN TESTS 1 THROUGH 12 IN ONE COMMAND:

Q.V.	1 MIN 57 SECONDS
DEFAULT	12 MINS

5.0 DEVICE INFORMATION TABLES

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

CHANGE HW (L) ?

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

UNIT 0

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

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

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

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

CHANGE SW (L) ?

INHIBIT ITERATIONS (L) N ?

6.0 TEST SUMMARIES

TEST 1: BUS RESET TEST

THIS TEST VERIFIES THAT THE M7196 MODULE'S DEVICE REGISTERS ARE ACCESSIBLE ON THE BUS (SUBTEST 1) AND THEN CHECKS THAT THE BUILT-IN INITIALIZATION SELF-TEST MICRODIAGNOSTIC DID NOT FIND ANY BASIC PROBLEMS IN THE MODULE. AREAS OF LOGIC TESTED BY THE SELF-TEST SEQUENCE ARE AS FOLLOWS: ROM AND PIPELINE REGISTER, SEQUENCER, INTERNAL BUSES, 2901 MICROPROCESSOR, AND RAM. THIS TEST INITIALIZES THE CONTROLLER BY ISSUING THE BUS INIT SIGNAL VIA A RESET INSTRUCTION, OR BY WRITING INTO THE TSSR REGISTER, WAITS A PERIOD OF TIME (TO ALLOW THE CONTROLLER'S INITIALIZATION MICRODIAGNOSTIC SEQUENCE TO BE COMPLETED), AND THEN CHECKS THE CONTENTS OF THE TSSR REGISTER. SUCCESSFUL INITIALIZATION IS INDICATED BY SUBSYSTEM READY (SSR) AND NEED BUFFER ADDRESS (NBA) BITS BEING SET (1) AND ALL OTHER BITS (EXCEPT A17 AND A16 AND OFL, WHICH ARE IGNORED FOR THIS TEST) BEING CLEAR (0). IF THE CONTENTS OF TSSR ARE NOT AS EXPECTED, AN ERROR REPORT IS ISSUED LISTING THE EXPECTED DATA, ACTUAL DATA, AND THE DISCREPANCIES. THE ERROR REPORT ANALYZES THE TSSR CONTENTS AND DISCERNS AND REPORTS ONE OF THREE POSSIBILITIES:

1. TSSR CONTENTS ARE AMBIGUOUS (ANY OF BITS 11-14 ARE SET, OR STATES OF SSR AND SC BITS DO NOT CORRESPOND TO THE APPARENT ERROR CODE IN BITS 0-5): INDICATES THAT THE TSSR CONTENT CANNOT BE TRUSTED. INDICATES A CATASTROPHIC CONTROLLER MALFUNCTION. THIS IS A FATAL ERROR (EXECUTION IS ABORTED). FIELD ACTION WOULD BE TO REPLACE THE M7196. IF THE M7196 ITSELF IS BEING DEBUGGED, THE PROGRAM SHOULD BE RESTARTED WITH LOOP ON ERROR ENABLED IN ORDER TO PROBE FOR THE PROBLEM.
2. SSR = 0, SC = 0 AND THE ERROR CODE IN BITS 0-5 IS IN THE RANGE 17-13: THIS IS A FATAL ERROR. THE ERROR CODE IS DECODED AND THE APPROPRIATE DESCRIPTION GIVEN. INDICATES THAT A SERIOUS PROBLEM EXISTS.

TEST 2: WRAP DATA - HIGH BYTE

THIS TEST VERIFIES OPERATION OF:

1. PART OF THE LSI-11 BUS INTERFACE SECTION OF THE M7196 MODULE: PART OF THE INPUT FILE (TSDB HIGH BYTE), PART OF THE OUTPUT FILE (TSSR HIGH BYTE AND TSBA, BOTH BYTES), PART OF THE DCO05 TRANSCEIVER CIRCUITS (ADDRESS DECODER, BDAL DRIVERS, HIGH BYTE OF INTERNAL DAL BUS DRIVERS), AND BASIC PROGRAMMED I/O CONTROL SEQUENCES AND LOGIC;
2. PART OF 2901 MICROPROCESSOR ELEMENTS (Q-REGISTER, REGISTER 0, ROTATE AND NEGATE FUNCTIONS
3. Y AND SOURCE BUSES;
4. BASIC MICROPROGRAM SEQUENCES.

THE PROGRAM WRITES A TEST DATA BYTE INTO THE HIGH BYTE OF TSDB, WAITS FOR THE SSR BIT IN TSSR TO SET, THEN CHECKS THE CONTENTS OF BOTH TSBA AND TSSR. THE MODULE IS FUNCTIONING CORRECTLY IF DATA WRITTEN APPEARS IN BOTH BYTES OF TSBA AND THE FINAL CONTENT OF TSSR IS CORRECT (SAME AS AFTER INITIALIZATION EXCEPT FOR BITS 8 AND 9, WHICH SHOULD CONTAIN BITS 8 AND 9 OF THE DATA PATTERN WRITTEN. AN ERROR IS REPORTED AND A DESCRIPTIVE ANALYSIS GIVEN IF A DISCREPANCY IN TSBA OR TSSR IS DETECTED. THE ANALYSIS LISTS LIKELY FAULTY CANDIDATES FROM THE LOGIC ELEMENTS LISTED ABOVE. THE TEST IS REPEATED FOR ALL COMBINATIONS OF TEST DATA BYTES (0-377 OCTAL).

TEST 3: WRAP DATA - LOW BYTE

THIS TEST FURTHER VERIFIES OPERATION OF MANY OF THE SAME ELEMENTS TESTED IN TEST 2, AND ADDITIONALLY VERIFIES:

1. LOW BYTE OF THE TSDB INPUT FILE REGISTER,
2. LOW BYTE OF INTERNAL DAL BUS DRIVERS ON THE DC005 TRANSCEIVER CIRCUITS,
3. BASIC FUNCTIONING OF PARTS OF THE RAM.

THE PROGRAM WRITES A TEST DATA BYTE INTO THE LOW BYTE OF TSDB, WAITS FOR THE SSR BIT IN TSSR TO SET, THEN CHECKS THE CONTENTS OF BOTH TSBA AND TSSR. THE MODULE IS FUNCTIONING CORRECTLY IF DATA WRITTEN APPEARS IN BOTH BYTES OF TSBA AND THE FINAL CONTENT OF TSSR IS CORRECT (SAME AS AFTER INITIALIZATION EXCEPT FOR BITS 8 AND 9, WHICH SHOULD CONTAIN BITS 8 AND 9 OF THE DATA PATTERN WRITTEN. AN ERROR IS REPORTED AND A DESCRIPTIVE ANALYSIS GIVEN IF A DISCREPANCY IN TSBA OR TSSR IS DETECTED. THE ANALYSIS LISTS LIKELY FAULTY CANDIDATES FROM THE LOGIC ELEMENTS LISTED ABOVE. THE TEST IS REPEATED FOR ALL COMBINATIONS OF TEST DATA BYTES (0-377 OCTAL).

TEST 4: RAM TEST

THIS TEST VERIFIES THAT ALL LOCATIONS OF THE RAM ON THE M7196 CAN PROPERLY STORE AND READ BACK ALL DATA PATTERNS, AND THAT EACH RAM LOCATION IS UNIQUELY ADDRESSED (I.E., THAT ONE AND ONLY ONE LOCATION IS ACCESSED BY ANY PARTICULAR ADDRESS). THE BYPRODUCT OF THESE TESTS IS A VERIFICATION OF TWO REGISTERS IN THE 2901 AND THE CAPABILITY OF THE 2901 TO CORRECTLY PERFORM AN ADD.

TEST 5: SECOND INITIALIZATION TEST

THIS TEST VERIFIES THE SAME ELEMENTS AS DID INITIALIZATION TEST #1 AND ALSO CHECKS THAT CERTAIN PARTS OF RAM IS CLEARED TO ZERO AND THAT 2901 REGISTERS 10 AND 11 ARE ALSO CLEARED TO ZERO. THIS IS A CONFIDENCE CHECK OF A PART OF THE SELF-TEST SEQUENCE (I.E., THAT IT IS REALLY BEING EXECUTED). FOR EACH OF TWO SUBTESTS (ONE FOR INITIALIZING VIA A BUS INIT, THE OTHER FOR INITIALIZING BY WRITING INTO THE TSSR), THE FOLLOWING SEQUENCE IS PERFORMED:

1. EACH RAM LOCATION AND 2901 REGISTERS 10 AND 11 ARE SET TO ALL 1'S BY USING WRITES INTO THE TSDB REGISTER (LOW BYTE AND MAINTENANCE MODE WORD WRITES).
2. THE CONTROLLER IS INITIALIZED AND THE VARIOUS CHECKS ON THE TSSR DESCRIBED IN INITIALIZATION TEST #1 ARE PERFORMED.
3. #1'S (377 OCTAL) ARE WRITTEN INTO THE LOW BYTE OF TSDB, WHICH SHOULD CAUSE RAM LOCATION 0 TO BE WRITTEN TO ALL 1'S SINCE 2901 REGISTERS 10 AND 11, SPECIFYING THE RAM ADDRESS, SHOULD BE 0. RAM LOCATION 0 IS VERIFIED BY WRITING A WORD OF ZEROS INTO THE TSDB. THE RESULTING LOW BYTE OF TSBA SHOULD CONTAIN ALL 1'S.
4. THE ENTIRE RAM IS SCANNED. LOCATION 0 SHOULD CONTAIN ALL 1'S AND THE REMAINING LOCATIONS, EXCEPT FOR THE MESSAGE BUFFER IMAGE AREA, SHOULD CONTAIN 0. DISCREPANCIES ARE REPORTED. AN ERROR AT THIS POINT IS MOST LIKELY DUE TO A ROM, PIPELINE OR SEQUENCER PROBLEM OR A TIMING PROBLEM.

TEST 6: COMMAND REJECT

THIS TEST VERIFIES THAT ALL COMMANDS OTHER THAN WRITE CHARACTERISTICS ARE REJECTED DUE TO THE NEED BUFFER ADDRESS (NBA) BIT BEING SET IN TSSR, AND THAT THE TSBA AND TSSR REGISTERS ARE LEFT IN THE PROPER STATE AFTER EACH COMMAND IS REJECTED. THIS TEST CHECKS MICROPROCESSOR SEQUENCING, BASIC COMMAND DECODING AND DATA DMA HANDLING. THIS TEST CONTAINS TWO SUBTESTS: SUBTEST 1 SEQUENCES THROUGH ALL COMMAND WORDS (OTHER THAN WRITE CHARACTERISTICS) WITH THE INTERRUPT ENABLE (IE) BIT CLEAR AND VERIFIES THAT AN INTERRUPT IS NOT GENERATED BY THE REJECTED COMMAND; SUBTEST 2 PERFORMS SIMILARLY TO SUBTEST 1 BUT SETS THE IE BIT IN EACH COMMAND WORD AND VERIFIES THAT AN INTERRUPT IS GENERATED WHEN THE COMMAND IS REJECTED.

TEST 7: WRITE CHARACTERISTICS

THIS TEST VERIFIES BASIC OPERATION OF THE WRITE CHARACTERISTICS COMMAND. IT VERIFIES THAT THE COMMAND BLOCK AND CHARACTERISTICS DATA BLOCK ARE FETCHED PROPERLY FROM CPU MEMORY, THE NEED BUFFER ADDRESS (NBA) BIT IN TSSR IS HANDLED PROPERLY, AND THAT A PROPER MESSAGE PACKET IS STORED, WHERE APPROPRIATE. THIS TEST DOES NOT CHECK THAT THE VARIOUS FUNCTIONS ENABLED BY CHARACTERISTIC MODE DATA BITS OPERATE PROPERLY; THE FUNCTIONING OF THESE BITS IS VERIFIED IN SUBSEQUENT TESTS. ALL COMMANDS EXECUTED IN THIS TEST HAVE THE INTERRUPT ENABLE (IE) BIT CLEARED TO ZERO, SO NO INTERRUPTS SHOULD BE GENERATED. HOWEVER, THE PROGRAM RUNS AT PROCESSOR PRIORITY 0, WITH THE INTERRUPT SERVICE ROUTINE SET UP TO FLAG UNEXPECTED INTERRUPTS. IF AN INTERRUPT OCCURS, A PROBLEM EXISTS IN EITHER THE LSI-11 BUS INTERFACE SECTION OR IN THE ROM OR PIPELINE.

TEST 8: VOLUME CHECK

THIS TEST VERIFIES THAT THE VOLUME CHECK (VCK) BIT, A FLAG HELD WITHIN THE M7196 AND APPEARING IN XST0, IS SET BY INITIALIZE AND CLEARED BY EXECUTING A WRITE CHARACTERISTICS COMMAND WITH THE CVC BIT SET. IT IS ALSO VERIFIED THAT A WRITE CHARACTERISTICS COMMAND WITH THE CVC BIT CLEAR DOES NOT AFFECT THE STATE OF THE VOLUME CHECK BIT. THE ACTUAL FUNCTION OF VOLUME CHECK, THAT OF PREVENTING OR ALLOWING A TAPE MOTION COMMAND DEPENDING UPON WHETHER VOLUME CHECK IS SET OR CLEAR, IS NOT CHECKED BY THIS TEST; THIS FUNCTIONALITY IS CHECKED IN THE INDIVIDUAL TESTS OF TAPE MOTION COMMANDS.

TEST 9: COMPLETION INTERRUPT

THIS TEST VERIFIES THAT AN INTERRUPT IS GENERATED AT THE COMPLETION OF THE WRITE CHARACTERISTICS COMMAND IF THE INTERRUPT ENABLE (IE) BIT IN THE COMMAND HEADER WORD IS SET. THIS TEST CHECKS THE FUNCTIONING OF THE INTERRUPT LOGIC AND BASIC PROCESSING OF THE IE BIT.

THE SEQUENCES OF TEST 7 ARE REPEATED, EXCEPT THAT THE INTERRUPT SERVICE ROUTINE IS SET UP TO EXPECT INTERRUPTS AND EACH WRITE CHARACTERISTICS COMMAND IS ISSUED WITH THE IE BIT SET (1). IT IS VERIFIED, WHERE APPROPRIATE, THAT THE IE STATUS BIT IN XSTO OF ANY MESSAGE PACKET IS SET AND THAT A COMPLETION INTERRUPT IS GENERATED. FINALLY, A SEQUENCE OF TWO COMMANDS ARE ISSUED, THE FIRST WITH IE=1 AND THE SECOND WITH IE=0. IT IS VERIFIED THAT NO INTERRUPT IS GENERATED AFTER THE SECOND COMMAND AND THAT THE IE BIT IN XSTO IS 0.

TEST 10: BASIC PACKET PROTOCOL

THIS TEST VERIFIES BASIC OPERATION OF THE MESSAGE BUFFER RELEASE COMMAND, THE FUNCTION OF THE ACK BIT IN THE COMMAND HEADER WORD, AND THE REGISTER MODIFICATION REFUSED (RMR) LOGIC.

TEST 11: NON-TAPE MOTION COMMANDS

THIS TEST VERIFIES PROPER OPERATION OF THE INITIALIZE COMMAND. TWO SUBTESTS ARE USED. THE FIRST VERIFIES THAT THE COMMAND RUNS TO COMPLETION AND STORES A VALID MESSAGE PACKET. THE SECOND VERIFIES THAT NON-ZERO VALUES IN THE COMMAND MODE FIELD CAUSES COMMAND REJECT.

7.0 MAINTENANCE HISTORY

REVISION A - MARCH 1982

REVISION B - JUNE 1984

MINOR CHANGES FOR THE ORION CPU (11/??).
ELIMINATED THE MESSAGE DESCRIBING THE CPU TYPE.

REVISION C - APRIL 1987

CHANGES MADE TO ALLOW DIAGNOSTICS TO WORK WITH
THE NEW TSV05 MICROCODE (REVISION 2). THE NEW
TSV05 MICROCODE ALWAYS IN EXTENDED FEATURE MODE.

J2

3 000000
 4 000000
 5 000000
 11 000000
 12 000000
 13 000000
 14 000000
 20 000000
 21 002000
 22 002000
 23 002000
 24
 25
 26
 27
 28
 29
 30 002000
 31 002000
 002000
 002000 103
 002001 126
 002002 124
 002003 123
 002004 101
 002005 000
 002006 000
 002007 000
 002010
 002010 103
 002011
 002011 060
 002012
 002012 000000
 002014
 002014 001217
 002016
 002016 046004
 002020
 002020 046136
 002022
 002022 002154
 002024
 002024 002164
 002026
 002026 047004
 002030
 002030 000000
 002032
 002032 000000
 002034
 002034 000000
 002036
 002036 000000
 002040

```

.TITLE TSV2 - PROGRAM HEADER
.SBTTL PROGRAM HEADER
.PSECT ABS

.MCALL SVC
SVC ; INITIALIZE SUPERVISOR MACROS
.ENABLE LC
.NLIST BEX,CND
.ENABL ABS,AMA
.=2000
BGNMOD TSV2

TSV2::

;+
; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
;--

POINTER BGNSW,BGNSFT,BGNAU,BGNDU,BGNRPT
HEADER CVTSA,C,0,655,0
L#NAME:: ;DIAGNOSTIC NAME
.ASCII /C/
.ASCII /V/
.ASCII /T/
.ASCII /S/
.ASCII /A/
.BYTE 0
.BYTE 0
.BYTE 0

L#REV:: ;REVISION LEVEL
.ASCII /C/

L#DEPO:: ;0
.ASCII /0/

L#UNIT:: ;NUMBER OF UNITS
.WORD 0

L#TIML:: ;LONGEST TEST TIME
.WORD 655.

L#HPCP:: ;POINTER TO H.W. QUES.
.WORD L#HARD

L#SPCP:: ;POINTER TO S.W. QUES.
.WORD L#SOFT

L#HPTP:: ;PTR. TO DEF. H.W. PTABLE
.WORD L#HW

L#SPTP:: ;PTR. TO S.W. PTABLE
.WORD L#SW

L#LADP:: ;DIAG. END ADDRESS
.WORD L#LAST

L#STA:: ;RESERVED FOR APT STATS
.WORD 0

L#CO::
.WORD 0

L#DTYP:: ;DIAGNOSTIC TYPE
.WORD 0

L#APT:: ;APT EXPANSION
.WORD 0

L#DTP:: ;PTR. TO DISPATCH TABLE
.WORD 0
  
```

PROGRAM HEADER

002040	002124			L#DISPATCH	
002042		L#PRIO::	.WORD	0	;DIAGNOSTIC RUN PRIORITY
002042	000000		.WORD	0	
002044		L#ENVI::	.WORD	0	;FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000		.WORD	0	
002046		L#EXP1::	.WORD	0	;EXPANSION WORD
002046	000000		.WORD	0	
002050		L#MREV::	.WORD	0	;SVC REV AND EDIT #
002050	003		.BYTE	C#REVISION	
002051	003		.BYTE	C#EDIT	
002052		L#EF::	.WORD	0	;DIAG. EVENT FLAGS
002052	000000		.WORD	0	
002054	000000		.WORD	0	
002056		L#SPC::	.WORD	0	
002056	000000		.WORD	0	
002060		L#DEVP::	.WORD	0	; POINTER TO DEVICE TYPE LIST
002060	003402		.WORD	L#DVTYP	
002062		L#REPP::	.WORD	0	;PTR. TO REPORT CODE
002062	022700		.WORD	L#RPT	
002064		L#EXP4::	.WORD	0	
002064	000000		.WORD	0	
002066		L#EXP5::	.WORD	0	
002066	000000		.WORD	0	
002070		L#AUT::	.WORD	0	;PTR. TO ADD UNIT CODE
002070	022366		.WORD	L#AU	
002072		L#DUT::	.WORD	0	;PTR. TO DROP UNIT CODE
002072	022464		.WORD	L#DU	
002074		L#LUN::	.WORD	0	;LUN FOR EXERCISERS TO FILL
002074	000000		.WORD	0	
002076		L#DESP::	.WORD	0	;POINTER TO DIAG. DESCRIPTION
002076	003410		.WORD	L#DESC	
002100		L#LOAD::	EMT	E#LOAD	;GENERATE SPECIAL AUTOLOAD EMT
002100	104035		EMT	E#LOAD	
002102		L#ETP::	.WORD	0	;POINTER TO ERR_TBL
002102	000000		.WORD	0	
002104		L#ICP::	.WORD	0	;PTR. TO INIT CODE
002104	021572		.WORD	L#INIT	
002106		L#CCP::	.WORD	0	;PTR. TO CLEAN-UP CODE
002106	022652		.WORD	L#CLEAN	
002110		L#ACP::	.WORD	0	;PTR. TO AUTO CODE
002110	022572		.WORD	L#AUTO	
002112		L#PRT::	.WORD	0	;PTR. TO PROTECT TABLE
002112	021562		.WORD	L#PROT	
002114		L#TEST::	.WORD	0	;TEST NUMBER
002114	000000		.WORD	0	
002116		L#DLY::	.WORD	0	;DELAY COUNT
002116	000000		.WORD	0	
002120		L#HIME::	.WORD	0	;PTR. TO HIGH MEM
002120	000000		.WORD	0	

PROGRAM HEADER

35
36
37
38
39
40
41
42
43002122
002122 000013
002124
002124 023462
002126 023702
002130 024400
002132 025072
002134 026426
002136 027532
002140 031004
002142 034462
002144 035366
002146 040522
002150 043634

.SBTTL DISPATCH TABLE

```

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

```

```

DISPATCH 11
.WORD 11
L#DISPATCH::
.WORD T1
.WORD T2
.WORD T3
.WORD T4
.WORD T5
.WORD T6
.WORD T7
.WORD T8
.WORD T9
.WORD T10
.WORD T11

```

44
45
46
47
48
49
50
51
52
5354 002152
002152 000003
002154
002154

.SBTTL DEFAULT HARDWARE P-TABLE

```

; **
; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
; THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
; IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
; --

```

```

BGNHM DFPTBL ;DEFAULT HARD-P-TABLE
.WORD L10000-L#HM/2
L#HM::
DFPTBL::
.WORD 172520 ; 1ST (OF 2) REGISTERS.
.WORD 224 ; INTERRUPT VECTOR
.WORD PRI04 ; INTERRUPT PRIORITY.
ENDHM
L10000:

```

55 002154 172520
56 002156 000224
57 002160 000200
58 002162
59 002162
60

M2

SOFTWARE P-TABLE

```

62          .SBTTL  SOFTWARE P-TABLE
63
64          ;**
65          ; THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
66          ; PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
67          ;--
68          002162      BGNSW   SFPTBL
                   002162      .WORD  L10001-L$SW/2
                   002164
                   002164      L$SW::
                   002164      SFPTBL::
69
70          002164      000000      TRANSTST::      .WORD  0      ; ENABLE TEST OF TRANSPORT(S) IF =1
71          002166      000000      NOITS::        .WORD  0      ; INHIBIT ITERATION OPTION.
72
73
74          002170      000017      LERRMAX::      .WORD  15.    ; ... 0 = ITERATE.
75          002172      000310      GERRMAX::      .WORD  200.  ; ...NZ = INHIBIT ITERATE.
76          002174      000000      ; LOCAL (PER TEST) ERROR LIMIT
                   002174      ; GLOBAL (PER UNIT) ERROR LIMIT
77          002174      ENDSW
78          002174      L10001:
79          002174      ENDMOD
80
81
82
83
84

```

N2

SOFTWARE P-TABLE

7
8
13
19
20 002174
002174
21
22
23
24
25
26
27
28
32 002174

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

000040
000037
000036
000035
000034

```

.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
;
BIT15== 100000
BIT14== 40000
BIT13== 20000
BIT12== 10000
BIT11== 4000
BIT10== 2000
BIT09== 1000
BIT08== 400
BIT07== 200
BIT06== 100
BIT05== 40
BIT04== 20
BIT03== 10
BIT02== 4
BIT01== 2
BIT00== 1

;
BIT9== BIT09
BIT8== BIT08
BIT7== BIT07
BIT6== BIT06
BIT5== BIT05
BIT4== BIT04
BIT3== BIT03
BIT2== BIT02
BIT1== BIT01
BIT0== BIT00

; EVENT FLAG DEFINITIONS
; EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION
;
EF.START== 32.          ; START COMMAND WAS ISSUED
EF.RESTART== 31.       ; RESTART COMMAND WAS ISSUED
EF.CONTINUE== 30.      ; CONTINUE COMMAND WAS ISSUED
EF.NEW== 29.           ; A NEW PASS HAS BEEN STARTED
EF.PWR== 28.           ; A POWER-FAIL/POWER-UP OCCURRED

;
; PRIORITY LEVEL DEFINITIONS

```

B3

GLOBAL EQUATES SECTION

000340
000300
000240
000200
000140
000100
000040
000000

!
PRI07== 340
PRI06== 300
PRI05== 240
PRI04== 200
PRI03== 140
PRI02== 100
PRI01== 40
PRI00== 0

!
;OPERATOR FLAG BITS

000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

!
EVL== 4
LOT== 10
ADR== 20
IDU== 40
ISR== 100
UAM== 200
BOE== 400
PNT== 1000
PRI== 2000
IXE== 4000
IBE== 10000
IER== 20000
LOE== 40000
HOE== 100000

33
34 002174

000250
177572
177574
177576
172516

KT11
.SBTTL MEMORY MANAGEMENT DEFINITIONS
;*KT11 VECTOR ADDRESS
MMVEC= 250
;*KT11 STATUS REGISTER ADDRESSES
SR0= 177572
SR1= 177574
SR2= 177576
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 REGISTORS
UDPDR0= 177620
UDPDR1= 177622
UDPDR2= 177624
UDPDR3= 177626
UDPDR4= 177630
UDPDR5= 177632
UDPDR6= 177634
UDPDR7= 177636
.ENDC

;DEFINE MEMORY MANAGEMENT REGISTERS

MEMORY MANAGEMENT DEFINITIONS

```
;*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
```

D3

MEMORY MANAGEMENT DEFINITIONS

```
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
; *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
; *KERNEL "D" PAGE ADDRESS REGISTERS
KDPAR0= 172360
KDPAR1= 172362
KDPAR2= 172364
KDPAR3= 172366
KDPAR4= 172370
KDPAR5= 172372
KDPAR6= 172374
KDPAR7= 172376
.ENDC
```

TSV05 REGISTER AND PACKET DEFINITIONS

```

39          .SBTTL  TSV05 REGISTER AND PACKET DEFINITIONS
40
41          ;
42          ; SOME GENERAL EQUATES.
43          ;
44
45          000004  ERRVEC==      4          ; POINTER TO ERROR VECTOR FOR BUS TIME OUT.
46          000060  TTIVEC==     60          ; INTERRUPT VECTOR FOR CONSOLE INPUT
47          177560  TTICSR==    177560       ; BUS ADDRESS OF CONSOLE INPUT
48          177562  TTIBFR==    177562       ; CONSOLE INPUT DATA BUFFER
49          177520  BDVPCR==    177520       ; BDV11 PAGE CONTROL REGISTER
50
51          ;*
52          ;BIT DEFINITIONS FOR TSSR REGISTER
53          ;-
54
55          100000  SC=          BIT15       ;SPECIAL CONDITION
56          040000  BIE=        BIT14       ;BUS INTERFACE ERROR
57          020000  SCE=        BIT13       ;SANITY CHECK ERROR
58          010000  RMR=        BIT12       ;MODIFICATION REFUSED
59          004000  NXM=        BIT11       ;NONEXISTANT MEMORY ERROR
60          002000  NBA=        BIT10       ;NEED BUFFER ADDRESS
61          001400  HIAADR=     BIT9:BIT8   ;EXTENDED ADDRESS BITS
62          000200  SSR=        BIT7        ;SUB SYSTEM READY
63          000100  OFL=        BIT6        ;OFF LINE BIT
64          000060  FATERR=     BIT4:BITS   ;FATAL TERMINATION ERROR CODES
65          000016  TERCLS=     BIT3:BIT2:BIT1 ;TERMINATION CODES
66
67          ;*
68          ;
69          ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 0
70          ;(XST0)
71          ;
72          ;-
73
74          100000  XSOTMK=     BIT15       ;TAPE MARK DETECTED
75          040000  XSORLS=     BIT14       ;RECORD LENGTH SHORT
76          020000  XSOLET=     BIT13       ;LOGICAL END OF TAPE
77          010000  XSORLL=     BIT12       ;RECORD LENGTH LONG
78          004000  XSOMLE=     BIT11       ;WRITE LOCK ERROR
79          002000  XSONEF=     BIT10       ;NON EXECUTABLE FUNCTION
80          001000  XSOILC=     BIT9        ;ILLEGAL COMMAND
81          000400  XSOILA=     BIT8        ;ILLEGAL ADDRESS
82          000200  XSOMOT=     BIT7        ;TAPE IN MOTION
83          000100  XSOONL=     BIT6        ;TRANSPORT ON LINE
84          000040  XSOIE=      BITS       ;INTERRUPT ENABLE
85          000020  XSOVCK=     BIT4        ;VOLUME CHECK BIT
86          000010  XSOPED=     BIT3        ;PHASE ENCODED DRIVE
87          000004  XSOMLK=     BIT2        ;WRITE LOCKED
88          000002  XSOMOT=     BIT1        ;BEGINNING OF TAPE
89          000001  XS0EOT=     BIT0        ;END OF TAPE

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

91
92      ;*
93      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 1
94      ;(XST1)
95      ;-
96      100000 X1.DLT = BIT15      ;DATA LATE
97      040000 X1.SPARE= BIT14      ;NOT USED
98      020000 X1.COR = BIT13      ;CORRECTABLE DATA ERROR
99      017375 X1.MBZ = BIT12·BIT11·BIT10·BIT9·BIT7·BIT6·BIT5·BIT4·BIT3·BIT2·BIT0 ;ALWAYS 0
100     000400 X1.RBP = BIT8       ;READ BUS PARITY ERROR
101     000002 X1.UNC = BIT1       ;UNCORRECTABLE DATA OR HARD ERROR
102
103     ;*
104     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 2
105     ;(XST2)
106     ;-
107     100000 X2.OPM = BIT15      ;OPERATION IN PROGRESS (TAPE MOVING)
108     040000 X2.RCE = BIT14      ;RAM CHECKSUM ERROR
109     035400 X2.SPARE= BIT13·BIT12·BIT11·BIT9·BIT8 ;NOT USED BY TSV05 (ALWAYS=0)
110     002000 X2.WCF = BIT10      ;WRITE CLOCK FAILURE (FIFO NOT EMPTIED BY TRANSPORT)
111     000200 X2.EXTF = BIT7      ;IF WRITE CHAR CMD THEN = EXTENDED FEATURES ENABLED
112     000100 X2.BUFE = BIT6      ;IF WRITE CHAR CMD THEN = BUFFERING ENABLED
113     000077 X2.REV = 000077    ;IF WRITE CHAR CMD THEN = MICROCODE REVISION LEVEL
114     000007 X2.UNIT = BIT2·BIT1·BIT0 ;IF GET STATUS THEN = CURRENTLY SELECTED UNIT NO.
115
116     ;*
117     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 3
118     ;(XST3)
119     ;-
120     177400 X3.MDE = 177400     ;MICRO-DIAGNOSTIC ERROR CODE
121     000200 X3.SPARE= BIT7      ;NOT USED BY TSV05
122     000100 X3.OPI = BIT6      ;OPERATION INCOMPLETE
123     000040 X3.REV = BIT5      ;REVERSE
124     000020 X3.TRF = BIT4      ;TRANSPORT RESPONSE FAILURE
125     000010 X3.DCK = BIT3      ;DENSITY CHECK
126     000006 X3.MBZ =BIT2·BIT1  ;NOT USED ALWAYS 0
127     000001 X3.RIB = BIT0      ;REVERSE INTO BOT
128
129     ;*
130     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 4
131     ;(XST4)
132     ;-
133     100000 X4.HSP = BIT15      ;HIGH SPEED
134     040000 X4.RCE = BIT14      ;RETRY COUNT EXCEEDED
135     020000 X4.TSM = BIT13      ;TRANSPORT SPECIAL MODE
136     017400 X4.MBZ = BIT12·BIT11·BIT10·BIT9·BIT8 ;NOT USED ALWAYS 0
137     000377 X4.WRC = 000377    ;WRITE RETRY COUNT FIELD
138
139     ;*
140     ;TSSR TERMINATION CODES (BIT 0-2)
141     ;-
142
143
144     000006 TSREJ= 3*2          ;COMMAND REJECTED
145     000006 UNREC= 6           ;UNRECOVERABLE ERROR

```


TSV05 REGISTER AND PACKET DEFINITIONS

```

147
148
149
150
151
152
153      000000      TSBA== 0
154      000000      TSDB== 0          ;TSDB/TSBA REGISTER
155      000001      TSBAH== 1
156      000001      TSDBH== 1          ;TSDB/TSBA REGISTER HIGH BYTE
157      000002      TSSR== 2          ;TSSR REGISTER
158      000003      TSSRH== 3         ;TSSR REGISTER HIGH BYTE
159
160
161
162
163      000003      A1716 = BIT17:BIT0 ;ADDRESS BITS 17:16 ARE IN 1:0
164
165
166
167
168      000017      P.GETSTAT = 17    ;GET STATUS
169      000013      P.INIT = 13      ;INITIALIZE
170      000012      P.CONTROL = 12   ;CONTROL COMMANDS
171      000011      P.FORMAT = 11    ;FORMAT
172      000010      P.POSITION = 10  ;POSITION
173      000006      P.WRTSUB = 6     ;SUBSYSTEM WRITE
174      000005      P.WRITE = 5     ;WRITE
175      000004      P.WRTCHAR = 4   ;WRITE CHARACTERISTICS
176      000001      P.READ = 1      ;READ
177
178
179
180
181      100000      P.ACK = BIT15    ;BUFFER AVAIL FOR CONTROLLER
182      040000      P.CVC = BIT14   ;CLEAR VOLUME CHECK
183      020000      P.OPP = BIT13    ;REVERSE SEQUENCE OF DATA BITS
184      010000      P.SWB = BIT12    ;SWAP BYTES IN MEMORY
185      007400      P.MODE = BIT11:BIT10:BIT9:BIT8 ;EXTENDED COMMAND MODE FIELD
186      000200      P.IE = BIT7     ;INTERRUPT ENABLE
187      000140      P.FMT = BIT6:BIT5 ;PACKET HEADER TYPE (ALWAYS=0)
188      000037      P.CMD = 37      ;MAJOR COMMAND FIELD
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
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
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
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
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
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

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

198
199
200
201      000167
202      000200
203      000201
204      000210
205      000215
206      000234
207
208
209
210
211
212
213      000006
214      000010
215      000012
216      000014
217      000016
218
219
220
221
222
223
224
225      000002
226      000004
227      000006
228
229      000010
230
231
232
233
234      000000
235      000001
236      000002
237      000004

; *
; CONTROLLER RAM DEFINITIONS
; -
RMCHBEG = 167      ; CHARACTERISTICS IO DATA BEGIN RAM ADDRESS
RMCHEND = 200      ; CHARACTERISTICS IO DATA END RAM ADDRESS
RMPKTBE = 201      ; COMMAND PACKET BEGIN RAM ADDRESS
RMPKTEN = 210      ; COMMAND PACKET END RAM ADDRESS
RMMSGBE = 215      ; MESSAGE BUFFER BEGIN RAM ADDRESS
RMMSGEN = 234      ; MESSAGE BUFFER END RAM ADDRESS
; *
; REGISTER DEFINITIONS IN THE MESSAGE BUFFER
; -
XST0 = 6           ; EXTENDED STATUS REGISTER 0 (WORD 4)
XST1 = 8           ; EXTENDED STATUS REGISTER 1 (WORD 5)
XST2 = 10          ; EXTENDED STATUS REGISTER 2 (WORD 6)
XST3 = 12          ; EXTENDED STATUS REGISTER 3 (WORD 7)
XST4 = 14          ; EXTENDED STATUS REGISTER 4 (WORD 8)
; *
; OFFSETS TO WORD LOCATIONS IN PACKET DEFINITIONS
; -
PKLOW = 2          ; LOW ORDER CHARACTERISTIC DATA POINTER
PKHI  = 4          ; HIGH ORDER CHARACTERISTIC DATA POINTER
PKBCN = 6          ; NUMBER OF BYTES IN DATA PACKET
EXBCN = 10         ; NUMBER OF BYTES IN EXTENDED DATA PACKET
; *
; DATA PACKET OFFSETS FOR WRITE SUBSYSTEM COMMAND
; -
BSEL0 = 0          ; BYTE 0
BSEL1 = 1          ; BYTE 1
SEL2  = 2          ; WORD 2
SELDA = 4          ; WORD 3

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

239
240      ;*
241      ;BSEL0 SELECT CODES FOR WRITE SUBSYSTEM COMMAND
242      ;-
243      000000      PW.NOP          = 0          ;NO-OP
244      000001      PW.RDRAM       = 1          ;READ RAM
245      000002      PW.WTRAM       = 2          ;WRITE RAM
246      000003      PW.RFIFO       = 3          ;READ FIFO
247      000004      PW.WFIFO       = 4          ;WRITE FIFO
248      000005      PW.RDSTAT      = 5          ;READ STATUS
249      000006      PW.WCTL        = 6          ;WRITE TAPE CONTROL
250      000007      PW.WFMT        = 7          ;WRITE TAPE FORMAT
251      000010      PW.WMISC       = 10         ;WRITE MISCELLANEOUS
252      000011      PW.WNPR        = 11         ;WRITE NPR CONTROL
253      000020      PW.D22         = 20         ;DO MICROTEST 22
254      000021      PW.D11         = 21         ;DO MICROTEST 11
255      000022      PW.D13         = 22         ;DO MICROTEST 13
256      000023      PW.NO1311      = 23         ;DISABLE MICROTEST 11 AND 13
257      000024      PW.RDXT        = 24         ;READ EXT. TAPE STATUS (NOT SUPPORTED BY ALL TRANSPORTS)
258
259      ;*
260      ;BSEL1 CODES FOR WRITE TAPE CONTROL
261      ;-
262      000200      WC.IFAD         = BIT7       ;IFAD - FORMATTER ADDRESS
263      000100      WC.IOTAD        = BIT6       ;ITADO - TRANSPORT ADDRESS BIT 0
264      000040      WC.I1TAD        = BIT5       ;ITAD1 - TRANSPORT ADDRESS BIT 1
265      000020      WC.ISRESV       = BIT4       ;IRESV5 - RESERVED #5
266      000010      WC.IREW         = BIT3       ;IREW - REWIND
267      000004      WC.IRWU         = BIT2       ;IRWU - REWIND AND UNLOAD
268      000002      WC.IFEN         = BIT1       ;IFEN - FORMATTER ENABLE
269      000001      WC.IGO          = BIT0
270
271      ;*
272      ;BSEL1 CODES FOR WRITE FORMAT
273      ;-
274      000200      WF.IHISP        = BIT7       ;IHISP - HIGH SPEED
275      000100      WF.IWRT         = BIT6       ;IWRT - WRITE
276      000040      WF.IREV         = BIT5       ;IREV - REVERSE
277      000020      WF.IWFM         = BIT4       ;IWFM - WRITE FILE MARK
278      000010      WF.IEDIT        = BIT3       ;IEDIT - EDIT
279      000004      WF.IERASE       = BIT2       ;IERASE - ERASE
280      000002      WF.I3RESV       = BIT1       ;IRESV3 - RESERVED #3
281      000001      WF.I4RESV       = BIT0       ;IRESV4 - RESERVED #4
282
283      ;*
284      ;BSEL1 CODES FOR WRITE MISCELLANEOUS SUBCOMMAND
285      ;-
286      000200      MS.EXT          = BIT7       ;INVERT SENSE OF EXTENDED FEATURES SWITCH
287      000020      MS.RSFIFO       = BIT4       ;RESET FIFO AND INPUT PARITY ERRORR
288      000010      MS.RSTAPE       = BIT3       ;RESET TAPE STATUS IN 2 FLIP-FLOPS
289      000006      MS.ATTN         = BIT2:BIT1  ;ATTENTION TRIGGER FIELD
290      000001      MS.RSD          = BIT0       ;RESET TIMER A,B THEN DELAY TIMES IN SEL2

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

291
292      ;*
293      ; MS.ATTN SUBCODES
294      ; -
294      000000      MSA.NOP = 0*2      ;NO-OP (NOTHING TRIGGERED)
295      000002      MSA.VOL = 1*2      ;SIMULATE ON-LINE/OFF-LINE TRANSITION
296      000004      MSA.NRAM= 2*2      ;FORCE NON-FATAL RAM ERROR (FORCES ERRCODE 54)
297      000006      MSA.FRAME= 3*2     ;FORCE FATAL RAM ERROR (CAUSES SCE TO SET)
298
299      ;*
300      ; WRITE SUBSYSTEM WRITE NPR BSEL1 BIT DEFINITIONS
301      ; -
301      000200      NP.IR      = BIT7      ;INTERRUPT REQUEST (0-1 TRANSITION)
302      000100      NP.OUT     = BIT6      ;TAPE DATA DIRECTION OUT (0= IN)
303      000040      NP.LOOP    = BIT5      ;ENABLE TRANSPORT LOOPBACK
304      000020      NP.WRP     = BIT4      ;WRITE CORRECT PARITY (SET=0 TO WRITE WRONG)
305
306      ;*
307      ; READ STATUS MESSAGE BUFFER BIT DEFINITIONS
308      ; -
309      000200      S2.DIM      = BIT7      ;WORD #9 BYTE 2 DATA IN MISS
310      000100      S2.ILW     = BIT6      ;ILW H
311      000040      S2.OUTRDY   = BIT5      ;OUT RDY H
312      000020      S2.INRDY   = BIT4      ;IN RDY H
313      000010      S2.ATIMR   = BIT3      ;TIMER A FLAG H
314      000004      S2.BTIMR   = BIT2      ;TIMER B FLAG H
315      000003      S2.UNDEF    = BIT1-BIT0 ;(UNDEFINED)
316      100000      S1.PARIN    = BIT15     ;WORD #8 BYTE 1 PARIN H
317      040000      S1.I2RESV   = BIT14     ;I2RESV2
318      020000      S1.I1RESV   = BIT13     ;I1RESV1
319      010000      S1.IEOT     = BIT12     ;IEOT L
320      004000      S1.IIDENT   = BIT11     ;IIDENT H
321      002000      S1.ICER     = BIT10     ;ICER H
322      001000      S1.IFMK     = BIT9      ;IFMK H
323      000400      S1.IHER     = BIT8      ;IHER H
324      000200      S0.ISPEED   = BIT7      ;WORD #8 BYTE 0 ISPEED H
325      000100      S0.IRDY     = BIT6      ;IRDY L
326      000040      S0.IONL     = BIT5      ;IONL L
327      000020      S0.ILDP     = BIT4      ;ILDP L
328      000010      S0.IDBY     = BIT3      ;IDBY L
329      000004      S0.IRWD     = BIT2      ;IRWD L
330      000002      S0.IFBY     = BIT1      ;IFBY L
331      000001      S0.IFPT     = BIT0      ;IFPT L
    
```

SPECIAL MACROS AND OPDEFS.

```

333 .SBTTL SPECIAL MACROS AND OPDEFS.
334
335 ;*
336 ;SAVE GENERAL REGS 1 TO 5
337 ;-
338
339 .MACRO SAVREG
340 JSR R5,REGSAV
341 .ENDM
342
343 ;*
344 ; MACRO TO FORCE AN ERROR
345 ;-
346 .MACRO FORCERROR TAG,NOTSSR
347 .NLIST
348 .IIF NDF LISTALL, .NLIST
349 .LIST
350 .IF B NOTSSR
351 MOV TSSR(R5),R1 ;READ TSSR
352 .ENDC
353 MOV FORCER,FORCER ;IS FORCER SET? (LEAVE C BIT ALONE)
354 BNE TAG ;BR IF YES
355 .NLIST
356 .IIF NDF LISTALL, .LIST
357 .LIST
358 .ENDM
359
360 ;*
361 ; MACRO TO FORCE AN EXIT TO AVOID SECTION ITERATIONS
362 ; WILL EXIT TO A LABEL IF FORCER IS NEGATIVE
363 ; SO TO FORCE ERRORS AND EXIT ON 1 ERROR SET
364 ; FORCER TO 177777
365 ; TO FORCE ERRORS AND ITERATIONS SET FORCER TO 1.
366 ;-
367 .MACRO FORCEEXIT TAG
368 .NLIST
369 .IIF NDF LISTALL, .NLIST
370 .LIST
371 MOV FORCER,FORCER ;IS FORCER NEGATIVE?
372 BMI TAG ;BR IF YES
373 .NLIST
374 .IIF NDF LISTALL, .LIST
375 .LIST
376 .ENDM
377 ;*
378 ; MACRO TO INCREMENT ERROR COUNTS
379 ;-
380 .MACRO NEXT.ERRNO
381 .NLIST
382 ;;;.IIF NDF LISTALL, .NLIST
383 ERRNO=ERRNO+1
384 ;;;.IIF NDF LISTALL, .LIST
385 .LIST
386 .ENDM

```

L3

SPECIAL MACROS AND OPDEFS.

```

388
389
390
391
392
393
394
395
396
397
398
399          000000
400
401
402
403
404
405
406
407 002174  000000
408
409

```

```

;+
;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.

```

GLOBAL DATA SECTION

411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451

.SBTTL GLOBAL DATA SECTION

```

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

;
; THE FOLLOWING DATA ARE SET FOR EACH UNIT AT INIT TIME.
; SINGLE UNIT DEFAULTS (LISTED) ARE IN THE DEFAULT P-TABLE.
;
EPRTSW::      .WORD 0      ;PRINT SWITCH
UNITN::       .WORD 0      ;UNIT # UNDER TEST.
QVP::         .WORD 0      ;QUICK VERIFY FLAG.
CSRADDR::     .WORD 0      ;ADDRESS OF CSR FOR CURRENT DEVICE
IVEC::        .WORD 224    ;INTERRUPT VECTOR
IPRI::        .WORD PRI04  ;INTERRUPT PRIORITY.
TSTCNT::     .WORD 0      ;NUMBER OF TESTS RUN IN THIS PASS
LOOPCNT::    .WORD 0      ;REMAINING ITERATION COUNT FOR TEST
DEVCNT::     .WORD 0      ;NUMBER OF DEVICE UNDER TEST
FATFLG::     .WORD 0      ;SET IF FATAL ERROR IS DETECTED IN TEST
INTRECV::    .WORD 0      ;SET IF TAPE INTERRUPT WAS RECEIVED
EXTFEA::     .WORD 0      ;EXTENDED FEATURES SOFTWARE SW 0-OFF,1-ON
REV::        .WORD 0      ;REV LEVEL (old microcode=1,new micro=2)
BENBSW::     .WORD 0      ;BUFFER ENABLE SWITCH SW 0-OFF,1-ON
EXPD::       .WORD 0      ;EXPECTED RAM DATA FOR PRAMPKT ROUTINE
RECV::       .WORD 0      ;RECEIVED RAM DATA FOR PRAMPKT ROUTINE
ERRHI::      .WORD 0      ;HIGH ADDRESS MEMORY ERROR
ERRLO::      .WORD 0      ;LOW ADDRESS MEMORY ERROR
RANDATA::    .BLKW 16.    ;DATA READ FROM RAM PACKET OR MESSAGE BUF AREA
RAMSIZ::     .WORD 0      ;RAM DATA SIZE FOR PRAMPKT ROUTINE
RCVHIADD::   .WORD 0      ;RECEIVED BUFFER HIGH ADDRESS
RCVLOADD::   .WORD 0      ;RECEIVED BUFFER LOW ADDRESS
COUNT::    .WORD 0      ;TEST COUNT PATTERN
DATA::       .WORD 0      ;TEST DATA
TSTFLAG::    .WORD 0      ;TEST FLAG WORD
TSTPTR::     .WORD 0      ;TSTBLK POINTER
PRMNO::      .WORD 0      ;PRINT ROUTINE TEMP
EXPMSG::     .BLKB 100.   ;EXPECTED MESSAGE BUFFER DATA
RECMMSG::    .BLKB 100.   ;RECEIVED MESSAGE BUFFER DATA
TMPBFR::     .BLKB 80.    ;TEMPORARY STORAGE FOR PRINT

```

TSTBLK - TEST DATA TABLE

453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469 002752
 470 002752 000000
 471 002754 177777
 472 002756 000001
 473 002760 000002
 474 002762 000004
 475 002764 000010
 476 002766 000020
 477 002770 000040
 478 002772 000100
 479 002774 000200
 480 002776 000400
 481 003000 001000
 482 003002 002000
 483 003004 004000
 484 003006 010000
 485 003010 020000
 486 003012 040000
 487 003014 100000
 488 003016 177776
 489 003020 177775
 490 003022 177773
 491 003024 177767
 492 003026 177757
 493 003030 177737
 494 003032 177677
 495 003034 177577
 496 003036 177377
 497 003040 176777
 498 003042 175777
 499 003044 175777
 500 003046 167777
 501 003050 157777
 502 003052 137777
 503 003054 077777
 504 003056 125252
 505 003060 052525
 506 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 ; DATA FOR WALKING ZEROS
    .WORD †CBIT0
    .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 ; ALTERNATING ONES, ZEROS
    .WORD 125252 ; ALTERNATING ONES, ZERO OPPOSITE FROM ABOVE
    .WORD 052525
TBLEND==
    
```


GLOBAL ENVIRONMENT STORAGE

```

508 .SBTTL GLOBAL ENVIRONMENT STORAGE
509
510 ; STORAGE FOR DEVICE REGISTERS
511
512 003062 000000 100000 000000 DUMMY: 0,100000,0,0 ; DUMMY DEVICE REGISTERS...
513 003072 000000 000000 000000 0,0,0,0,0,0,0,0 ; ...FOR MULTI-UNIT CHECKOUT.
514
515
516 003112 000000 DUFLG:: .WORD 0 ; "DROPPED UNIT" FLAG.
517 ; INHIBITS CODE IN "CLEAN-UP".
518 003114 000000 NODEV:: .WORD 0 ; FLAG TO SAY NO DEVICE.
519
520 003116 000000 TEMP1:: .WORD 0 ; SOME TEMP LOCATIONS.
521 003120 000000 TEMP2:: .WORD 0
522 003122 000000 XXCOMM:: .WORD 0 ; XXDP* (COMM BLOCK POINTER.
523 003124 000000 FREE:: .WORD 0 ; 1ST FREE MEMORY ADDRESS...
524 003126 000000 FRESIZ:: .WORD 0 ; ...AND SIZE (IN WORDS).
525 003130 000000 FREEHI: .WORD 0 ; LAST WORD IN FREE SPACE
526 003132 000000 KTFLG:: .WORD 0 ; KT11, MEM AVAIL FLAG -
527 ; - .WORD 0 = <24K OR NO KT -
528 ; - NZ = >24K AND KT.
529 003134 000000 KTENABLE:: .WORD 0 ; SET BY TEST ROUTINES TO FLAG >28K UNDER TEST
530 003136 000000 NXMFLG:: .WORD 0 ; SET IF WE CAN TEST CLEARED OTHERWISE
531 003140 000000 NXML0:: .WORD 0 ; NXM LO ADDRESS BITS
532 003142 000000 NXMHI:: .WORD 0 ; NXM HI ADDRESS BITS FOR DAL'S 16-21
533 003144 000000 T23A:: .WORD 0 ; 11/23A FLAG
534 003146 000000 T23B:: .WORD 0 ; 11/23B FLAG
535 003150 000000 T3BFLG:: .WORD 0 ; TEST 3B FLAG +0
536 003152 002000 PST32W:: .WORD 2000 ; 32W BLOCK ADDRESS FOR 32K START
537 003154 000000 SIFLAG:: .WORD 0
538 003156 000000 BADDAT:: .WORD 0 ; ACTUAL DATA
539 003160 000000 GDDAT:: .WORD 0 ; EXPECTED DATA
540 003162 000000 LOOPFL:: .WORD 0
541 003164 CTAB:: ; CONFIGURATION TABLES.
542 003164 000000 CTABM:: .WORD 0 ; CONFIG WORK.
543 003166 000000 .WORD 0
544 003170 000000 .WORD 0
545 003172 000000 .WORD 0
546 003174 177777 .WORD -1 ; END OF MEM TABLE.
547 003176
548 CTABE::
549 ; ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:
550 ;
551 ; 0 = UNIT NOT TESTED
552 ; 100000 = UNIT ONLINE, NO ERRORS
553 ; 10XXXX = UNIT ONLINE, ENCOUNTERED XXXX ERRORS
554 ; 160000 = UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
555 ; 160001 = UNIT DROPPED, NOT IDLE AT START
556 ; 14XXXX = UNIT DROPPED, ENCOUNTERED XXXX ERRORS
557 003176 ERTABL: .BLKW 64.
558 003376 000000 ERTABE: .WORD 0
559
560 003400 000000 SKIPT: .WORD 0 ; 1=SKIP SUBTEST 0=NO SKIP OF SUBTEST

```

GLOBAL TEXT MESSAGES

```

562
563
564
565
566
567
568
569
570
571 003402
    003402
    003402      124      123      126
572
574
575
576
577 003410
    003410
    003410      052      052      052
598
599
600
601
602 003502 003542 003545 003551
603 003522 003603 003607 003613
604 003542      123      103      000
605 003545      102      111      105
606 003551      123      103      105
607 003555      122      115      122
608 003561      116      130      115
609 003565      116      102      101
610 003571      102      111      124
611 003576      102      111      124
612 003603      123      123      122
613 003607      117      106      114
614 003613      102      111      124
615 003620      102      111      124
616 003625      102      111      124
617 003632      102      111      124
618 003637      102      111      124
619 003644      102      111      124
620
621 003652      124      123      123
622 003705      124      123      123
623 003740      040      040      116
624 003777      045      101      040
625 004020      045      101      040
626 004060      045      101      040
627 004117      045      116      045
628 004123      040      040      125
629 004152      040      040      111
630 004215      045      116      045
631 004221      040      040      116
632 004256      040      040      111
633 004300      045      101      040

```

```

.SBTTL GLOBAL TEXT MESSAGES
; **
; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
; MORE THAN ONE TEST.
; --
; *
; NAMES OF DEVICES SUPPORTED
; -
L#DVTYP: .DEV TYP <TSV05>
        .ASCIZ /TSV05/
        .EVEN
; *
; TEST DESCRIPTION
; -
L#DESC: .DESCRIP <**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****>
        .ASCIZ /**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****/
        .EVEN
; *
; BIT TO ASCII CONVERSION FOR TSSR REGISTER
; -
L#SSRBIT: .WORD 1#,2#,3#,4#,5#,6#,7#,8#
          .WORD 9#,10#,11#,12#,13#,14#,15#,16#
          1#: .ASCIZ 'SC'
          2#: .ASCIZ 'BIE'
          3#: .ASCIZ 'SCE'
          4#: .ASCIZ 'RMR'
          5#: .ASCIZ 'NXM'
          6#: .ASCIZ 'NBA'
          7#: .ASCIZ 'BIT9'
          8#: .ASCIZ 'BIT8'
          9#: .ASCIZ 'SSR'
          10#: .ASCIZ 'OFL'
          11#: .ASCIZ 'BIT5'
          12#: .ASCIZ 'BIT4'
          13#: .ASCIZ 'BIT3'
          14#: .ASCIZ 'BIT2'
          15#: .ASCIZ 'BIT1'
          16#: .ASCIZ 'BIT0'
          .EVEN
SFIERR: .ASCIZ 'TSSR ERROR AFTER SOFT INIT'
SFHERR: .ASCIZ 'TSSR ERROR AFTER BUS RESET'
NXR: .ASCIZ / NON-EXISTANT DEVICE REGISTER/
NXRX: .ASCIZ /#A ADDRESS: #06/
TSSX: .ASCIZ /#A TSBA,TSSR EXP'D: #06#A,#06#N/
      .ASCIZ /#A TSBA,TSSR REC'D: #06#A,#06/
FUSI: .ASCII /#N#A/
USI: .ASCIZ / UNEXPECTED INTERRUPT/
NSI: .ASCIZ / INTERRUPT EXPECTED, NOT RECEIVED/
FNOINTR: .ASCII /#N#A/
NOINTR: .ASCIZ / NO INTERRUPT WAS GENERATED/
IFALT: .ASCIZ / INTERRUPT FAULT/
INTX: .ASCIZ /#A CPU PC: #06#A TSBA: #06/

```

GLOBAL TEXT MESSAGES

```

634 004335      040      040      042 NOINIT: .ASCIZ / "BUS-INIT" DIDN'T INITIALIZE CONTROLLER/
635 004407      040      040      042 NSINIT: .ASCIZ / "SOFT-INIT" DIDN'T INITIALIZE THE DPU/
636 004457      040      040      042 BRINIT: .ASCIZ / "BUS-RESET" DIDN'T INITIALIZE THE DPU/
637 004527      000
638 004530      045      116      000 NULCR: .ASCIZ //
639 004533      045      101      040 EXPGOT: .ASCIZ /#A EXP'D: #06#A, REC'D: #06/
640 004567      045      116      045 EXPGT2: .ASCIZ /#N#A EXP'D: #06#A, #06#N#A REC'D: #0#A, #06/
641 004643      045      101      040 DUAD12: .ASCIZ /#A REG(W) WRITTEN TO: #06#A REG(R) READ; EXP'D: #06#A, REC'D: #06/
642 004745      122      101      115 PKTRAM: .ASCIZ .ASCIZ 'RAM Contents Do Not Match Packet Sent'
643 005013      040      040      103 SCME: .ASCIZ / CONFIG DOESN'T MATCH MFG. MASTER/
644 005056      127      122      111 WRTMSG: .ASCIZ 'WRITE CHARACTERISTICS Failed'
645 005113      124      123      123 WRTERR: .ASCIZ 'TSSR Incorrect After WRITE Command, More Bits Set Than SSR'
646 005206      124      123      123 RDERR: .ASCIZ 'TSSR Incorrect After READ Command, More Bits Set Than SSR'
647 005300      106      101      124 SCHERR: .ASCIZ 'FATAL ERROR IN SUBTEST - CHECK TAPE,CABLES,TRANSPORT etc.'
648 005372      105      122      122 RETERR: .ASCIZ 'ERROR IN SUBTEST - WRITE DATA RETRY FIVE TIMES FAILED'
649 005460      045      116      045 NOMEM: .ASCIZ '#N#A ***** NO NXM ADDRESS--CANNOT TEST NXM TIMEOUT. *****#N'
650 005554      045      116      045 M8186: .ASCIZ '#N#A ***** 11/23A SYSTEM *****#N'
651 005645      045      116      045 M8189: .ASCIZ '#N#A ***** 11/23B SYSTEM *****#N'
652
653
654
655
656
657
658
659
660 005736
661 005736
662 005762
663 005766
664
665
666
667
668 005770
669 005772
670 005774
671 005776
672 006002
673 006022

```

GLOBAL ERROR REPORT SECTION

```

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

```

```

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

```

```

L10002: TRAP C#MSG
;
; THIS ROUTINE APPENDS A UNIQUE EXTENSION (IF REQUIRED)
; TO ANY OF THE ABOVE ERROR SIGNATURES.

```

```

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

```

PRITSSR - PRINT TSSR CONTENTS

```

675                                     .SBTTL PRITSSR - PRINT TSSR CONTENTS
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693 006024
694 006024
695 006030 010104
696 006032
   006032 010446
   006034 012746 006415
   006040 012746 000002
   006044 010600
   006046 104414
   006050 062706 000006
697 006054 010400
698 006056 004737 016124
699 006062 103410
700 006064
   006064 012746 006635
   006070 012746 000001
   006074 010600
   006076 104415
   006100 062706 000004
701 006104 010403
702 006106 042703 001476
703 006112 001434
704 006114 012702 002632
705 006120 012701 003502
706 006124 005703
707 006126 001413
708 006130 000241
709 006132 006103
710 006134 103006
711 006136 011100
712 006140 112022
713 006142 001376
714 006144 112762 000054 177777
715 006152 005721
716 006154 000763
717 006156 105042
718 006160
   006160 012746 002632
   006164 012746 006606

;
;
; ROUTINE TO DISPLAY THE CONTENTS, AND BIT DEFINITIONS, OF
; THE TSSR REGISTER. THIS ROUTINE IS NORMALLY CALLED ONLY
; BY A MESSAGE PRINTING ROUTINE
;
; INPUTS:
;
;     R1     CONTENTS OF TSSR
;
; SUBORDINATE ROUTINES:
;
;     CHKAMB CHECK FOR AMBIGUOUS CONTENTS
;
;
;
PRITSSR:
  SAVREG                                ;SAVE GENERAL REGISTERS
  MOV R1,R4                              ;SAVE THE TSSR CONTENTS
  PRINTB @TSSRFOR,R4                    ;PRINT THE CONTENTS OF TSSR
  MOV R4,-(SP)
  MOV @TSSRFOR,-(SP)
  MOV @2,-(SP)
  MOV SP,R0
  TRAP C#PNTB
  ADD #6,SP
  MOV R4,R0                              ;GET TSSR BACK FOR CHKAMB
  JSR PC,CHKAMB                          ;ARE CONTENTS AMBIGUOUS ?
  BCS 5#                                  ;BRANCH IF NOT
  PRINTX @AMBTSSR                         ;SHOW CONTENTS ARE AMBIGUOUS
  MOV @AMBTSSR,-(SP)
  MOV @1,-(SP)
  MOV SP,R0
  TRAP C#PNTX
  ADD #4,SP
5#:  MOV R4,R3                              ;CONTENTS OF TSSR
  BIC #MIADDR!FATERR!TERCLS,R3          ;CLEAR ALL MULTIPLE BIT FIELDS
  BEQ 20#                                  ;NO BITS ARE SET
  MOV @TMPBFR,R2                          ;TEMPORARY ASCII BUFFER
  MOV @TSSRBIT,R1                          ;ASCII EQUIVALENT OF BITS
10#: TST R3                                ;REMAINING BITS TO CONVERT
  BEQ 15#                                  ;BRANCH WHEN ALL ARE DONE
  CLC                                       ;CLEAR CARRY FOR SHIFT
  ROL R3                                    ;SHIFT NEXT BIT TO CARRY
  BCC 13#                                  ;BRANCH IF BIT NOT SET
  MOV (R1),R0                              ;POINTER TO BIT DEFINITION
11#: MOVB (R0),-(R2)                        ;MOVE ASCII TO BUFFER
  BNE 11#                                  ;MOVE ALL BITS
  MOVB @'-1(R2)                            ;INSERT A COMMA TO TERMINATE
13#: TST (R1)                              ;POINT TO NEXT DESCRIPTION
  BR 10#                                   ;GET THE REMAINING BITS
15#: CLRB -(R2)                             ;TERMINATE THE LINE
  PRINTX @TSSDEF,@TMPBFR                 ;PRINT THE BIT DEFINITIONS
  MOV @TMPBFR,-(SP)
  MOV @TSSDEF,-(SP)

```

PRITSSR - PRINT TSSR CONTENTS

006170	012746	000002		MOV	#2,-(SP)	
006174	010600			MOV	SP,R0	
006176	104415			TRAP	C#PNTX	
006200	062706	000006		ADD	#6,SP	
719						
720	006204	010403	204:	MOV	R4,R3	;GET THE TSSR CONTENTS
721	006206	042703		BIC	#1CTERCLS,R3	;CLEAR ALL BUT TERMINATION
722	006212	016303		MOV	TCOCOD(R3),R3	;GET THE TERMINATION CODE MEANING
723	006216			PRINTX	#TCOASC,R3	;PRINT THE TERMINATION CODE
	006216	010346		MOV	R3,-(SP)	
	006220	012746		MOV	#TCOASC,-(SP)	
	006224	012746		MOV	#2,-(SP)	
	006230	010600		MOV	SP,R0	
	006232	104415		TRAP	C#PNTX	
	006234	062706		ADD	#6,SP	
724	006240	010403		MOV	R4,R3	;TSSR CONTENTS AGAIN
725	006242	042703		BIC	#1CFATERR,R3	;CLEAR ALL BUT FATAL TERMINATION
726	006246	001416		BEQ	254	;DON'T PRINT IF ZERO
727	006250	006203		ASR	R3	
728	006252	006203		ASR	R3	
729	006254	006203		ASR	R3	;ALINE TERMINATION CODE FOR INDEX
730	006256	016303		MOV	TSFCOD(R3),R3	;GET THE FATAL TERMINATION CODE
731	006262			PRINTX	#TFCASC,R3	;PRINT THE FATAL TERMINATION CODE
	006262	010346		MOV	R3,-(SP)	
	006264	012746		MOV	#TFCASC,-(SP)	
	006270	012746		MOV	#2,-(SP)	
	006274	010600		MOV	SP,R0	
	006276	104415		TRAP	C#PNTX	
	006300	062706		ADD	#6,SP	
732	006304	042704	254:	BIC	#1CHIADDR,R4	;CLEAR ALL BUT EXTENDED ADDRESS
733	006310	001411		BEQ	304	;DON'T PRINT IF ZERO
734	006312			PRINTX	#TEXASC,R4	;PRINT THE EXTENDED ADDRESS BITS
	006312	010446		MOV	R4,-(SP)	
	006314	012746		MOV	#TEXASC,-(SP)	
	006320	012746		MOV	#2,-(SP)	
	006324	010600		MOV	SP,R0	
	006326	104415		TRAP	C#PNTX	
	006330	062706		ADD	#6,SP	
735	006334	013703	304:	MOV	EPRTSW,R3	;PRINT MEASGE BUFFER ADDRESS
736	006340			PRINTX	R3	;PRINT PROPER MESSAGE
	006340	010346		MOV	R3,-(SP)	
	006342	012746		MOV	#1,-(SP)	
	006346	010600		MOV	SP,R0	
	006350	104415		TRAP	C#PNTX	
	006352	062706		ADD	#4,SP	
737	006356	000207		RTS	PC	;RETURN TO CALLER

G4

PRITSSR - PRINT TSSR CONTENTS

740	006360				EPRT2:				
741	006360	045	116	045	EPRT1:	.ASCIZ	'#N#A *****REPLACE M7196*****'		
756	006415	045	116	045	TSSRFOR:	.ASCIZ	'#N#A TSSR = #06'		
757	006435	045	116	045	TEXASC:	.ASCIZ	'#N#A Extended Address Bits = #06'		
758	006476	045	116	045	TCOASC:	.ASCIZ	'#N#A Termination Class Code = #T'		
759	006537	045	116	045	TFCASC:	.ASCIZ	'#N#A Fatal Termination Class Code = #T'		
760	006606	045	116	045	TSSDEF:	.ASCIZ	'#N#A TSSR Bits Set: #T'		
761	006635	045	116	045	AMBTSSR:	.ASCIZ	'#N#A TSSR Contents Are Ambiguous'		
762						.EVEN			
763	006676	006716	006741	006767	TCOCOD:	.WORD	1#,2#,3#,4#,5#,6#,7#,8#		
764	006716	116	157	162	1#:	.ASCIZ	'Normal Termination'		
765	006741	124	145	162	2#:	.ASCIZ	'Termination Condition'		
766	006767	124	141	160	3#:	.ASCIZ	'Tape Status Alert'		
767	007011	106	165	156	4#:	.ASCIZ	'Function Reject'		
768	007031	122	145	143	5#:	.ASCIZ	'Recoverable Error - Tape Position One Record Down'		
769	007113	122	145	143	6#:	.ASCIZ	'Recoverable Error - Tape Was Not Moved'		
770	007162	125	156	162	7#:	.ASCIZ	'Unrecoverable Error'		
771	007206	106	141	164	8#:	.ASCIZ	'Fatal Controller Error'		
772						.EVEN			
773									
774	007236	007246	007302	007313	TSFCOD:	.WORD	1#,2#,3#,4#		
775	007246	111	156	164	1#:	.ASCIZ	'Internal Diagnostic Failure'		
776	007302	122	145	163	2#:	.ASCIZ	'Reserved'		
777	007313	102	165	163	3#:	.ASCIZ	'Bus Interface or Sanity Check Error'		
778	007357	122	145	163	4#:	.ASCIZ	'Reserved'		
779						.EVEN			

H4

PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET

.SBTTL PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET

781
782
783
784
785
786
787
788
789
790
791
792
793
794
795

;*
;THIS ROUTINE PRINTS THE ADDRESS AND CONTENTS OF A COMMAND PACKET.
;THIS ROUTINE IS NORMALLY ONLY CALLED FROM A PRINT ROUTINE.
;INPUT:
;
; R0 NUMBER OF WORDS IN PACKET
; R3 HIGH ORDER COMMAND PACKET ADDRESS
; R4 ADDRESS OF COMMAND PACKET
;
; NOTE: R3 IS IGNORED IF THE KTENABLE FLAG IS CLEAR.
;-

796 007370
797 007370
798 007374 010005
799 007376 005737 003134
800 007402 001001
801 007404 005003
802 007406 010301
803 007410 010400
804 007412 006100
805 007414 006101
806 007416
007416 010446
007420 010146
007422 012746 007554
007426 012746 000003
007432 010600
007434 104414
007436 062706 000010
807 007442 010300
808 007444 001404
809 007446 010401
810 007450 004737 017376
811 007454 010004
812 007456 005001
813 007460 012402
814 007462
007462 010246
007464 010146
007466 012746 007516
007472 012746 000003
007476 010600
007500 104414
007502 062706 000010
815 007506 005201
816 007510 020105
817 007512 002762
818 007514 000207

PRIPKT: :
SAVREG ;SAVE THE REGISTERS
MOV R0,R5 ;SAVE NO. OF WORDS IN PACKET
TST KTENABLE ;ABOVE 28K UNDER TEST?
BNE 10\$;BR IF YES
CLR R3 ;SET HIGH ORDER ADDRESS TO 0
10\$: MOV R3,R1 ;COPY HIGH ORDER ADDRESS
MOV R4,R0 ;GET LOWER ADDRESS
ROL R0 ;SHIFT BIT 15 INTO C BIT
ROL R1 ;AND INTO HIGH ORDER.
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
15\$: ADD @10,SP
MOV R3,R0 ;GET HIGH ORDER ADDRESS
BEQ 20\$;BR IF NOT ABOVE 28K.
MOV R4,R1 ;GET LOW ORDER ADDRESS
JSR PC,SETMAP ;SETUP PAR6 MAPPING FOR 18 BIT ADDRESS
MOV R0,R4 ;GET RETURNED PAR6 ADDRESS BIAS
20\$: CLR R1 ;SAVE WORD NUMBER
25\$: MOV (R4),R2 ;GET PACKET CONTENTS
PRINTB @PKTFRM,R1,R2 ;PRINT THE DATA
MOV R2,-(SP)
MOV R1,-(SP)
MOV @PKTFRM,-(SP)
MOV @3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD @10,SP
INC R1 ;NEXT WORD NUMBER
CMP R1,R5 ;DONE ALL PACKET WORDS?
BLT 25\$;LOOP TILL ALL DONE
RTS PC ;RETURN

819
820 007516 045 116 045 PKTFRM: .ASCIZ '#N#A Packet Word #D1#A = #06'
821 007554 045 116 045 PKTADD: .ASCIZ '#N#A Packet Address = #01#05'
822 .EVEN

PRIBXOR - PRINT EXPD, RECV AND XOR BYTE

```

824                                     .SBTTL PRIBXOR - PRINT EXPD, RECV AND XOR BYTE
825
826
827 ;*
828 ;PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE DATA BYTE
829 ;THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
830
831 ;INPUTS:
832 ;
833 ;   R1   RECEIVED DATA
834 ;   R2   EXPECTED DATA
835
836 ;OUTPUT:
837 ;
838 ;   R0   XOR OF EXPECTED/RECEIVED DATA
839
840 PRIBXOR::
841     SAVREG                               ;SAVE THE REGISTERS
842     MOV     R2,R3                         ;EXPECTED DATA
843     XOR     R1,R3                         ;FORM THE EXCLUSIVE OR
844     MOV     #C<377>,R0                    ;BYTE MASK
845     BIC     R0,R1                         ;SAVE LOW BYTE RECV
846     BIC     R0,R2                         ;SAVE LOW BYTE EXPD
847     BIC     R0,R3                         ;SAVE LOW BYTE XOR
848     PRINTB @XORBFOR,R2,R1,R3 ;PRINT THE MESSAGE
849     MOV     R3,-(SP)
850     MOV     R1,-(SP)
851     MOV     R2,-(SP)
852     MOV     @XORBFOR,-(SP)
853     MOV     #4,-(SP)
854     MOV     SP,R0
855     TRAP   C#PNTB
856     ADD     #12,SP
857     MOV     R3,R0                         ;R0 HAS XOR ON RETURN
858     RTS    PC                             ;RETURN TO CALLER
859
860
861
862
863
864
865
866
867
868
869
870
871
872

```

```

045 XORBFOR: .ASCIZ '#N#A EXPD: #03#A RECV: #03#A XOR: #03'
.EVEN
.SBTTL PRIBXOR - PRINT EXPD, RECV AND XOR

```

```

;*
;PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE TWO
;THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
;INPUTS:
;
;   R1   RECEIVED DATA
;   R2   EXPECTED DATA
;OUTPUT:
;
;   R0   XOR OF EXPECTED/RECEIVED DATA
;
;PRIBXOR::
;   SAVREG                               ;SAVE THE REGISTERS
;   MOV     R2,R3                         ;EXPECTED DATA
;   XOR     R1,R3                         ;FORM THE EXCLUSIVE OR
;   PRINTB @XORFOR,R2,R1,R3 ;PRINT THE MESSAGE

```


J4

PRIXOR - PRINT EXPD, RECV AND XOR

007760	010346				MOV	R3,-(SP)		
007762	010146				MOV	R1,-(SP)		
007764	010246				MOV	R2,-(SP)		
007766	012746	010012			MOV	#XORFOR,-(SP)		
007772	012746	000004			MOV	#4,-(SP)		
007776	010600				MOV	SP,R0		
010000	104414				TRAP	C#PNTB		
010002	062706	000012			ADD	#12,SP		
873	010006	010300			MOV	R3,R0		;R0 HAS XOR ON RETURN
874	010010	000207			RTS	PC		;RETURN TO CALLER
875								
876	010012	045	116	045	XORFOR:	.ASCIZ	'#N#A EXPD: #06#A RECV: #06#A XOR: #06'	
877						.EVEN		

PRIEQU - PRINT BIT NUMBERS AS ASCII EQUIVALENT

```

879 .SBTTL PRIEQU - PRINT BIT NUMBERS AS ASCII EQUIVALENT
880
881 ;*
882 ;
883 ;ROUTINE TO CONVERT BIT VALUES TO ASCII AND PRINT THE STRING
884 ;THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
885 ;
886 ;INPUTS:
887 ;
888 ; R0 OCTAL VALUE TO CONVERT
889 ; R1 TABLE OF POINTERS TO ASCII EQUIVALENT
890 ;
891 ;-
892
893 010060 PRIEQU: SAVREG ;SAVE THE REGISTERS
894 010060 RTS PC ;RETURN TO CALLER
895 010064 000207
896
897 .SBTTL PRIRAM - PRINT RAM ADDRESS
898
899 ;*
900 ;PRINT CONTROLLER RAM ADDRESS.
901 ;THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
902 ;
903 ;INPUTS:
904 ;
905 ; R4 RAM ADDRESS
906 ;
907 ;-
908 010066 PRIRAM: SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
909 010066 PRINTB #RAMFOR,R4 ;PRINT RAM ADDRESS IN ERROR
910 010072 010446 MOV R4,-(SP)
911 010074 012746 010116 MOV #RAMFOR,-(SP)
912 010100 012746 000002 MOV #2,-(SP)
913 010104 010600 MOV SP,R0
914 010106 104414 TRAP C#PNTB
915 010110 062706 000006 ADD #6,SP
916 010114 000207 RTS PC ;RETURN
917
918 010116 045 116 045 RAMFOR: .ASCIZ '#N#A CONTROLLER RAM ADDRESS = #06'
919 .EVEN

```


M4

PRITADD - PRINT MEMORY TEST ADDRESS

```

961 010326 062706 000006      ADD    #6,SP
    010332      PRINTB  #PRIT1,R2      ;PRINT MEMORY ADDRESS HIGH IN ERROR
    010332 010246      MOV    R2,-(SP)
    010334 012746 010421      MOV    #PRIT1,-(SP)
    010340 012746 000002      MOV    #2,-(SP)
    010344 010600      MOV    SP,R0
    010346 104414      TRAP  C#PNTB
962 010350 062706 000006      ADD    #6,SP
    010354 000207      RTS    PC      ;RETURN
963
964 010356      045      116      045 PRIT0: .ASCIZ 'MEMA MEMORY TEST ADDRESS LOW = #06'
965 010421      045      116      045 PRIT1: .ASCIZ 'MEMA MEMORY TEST ADDRESS HIGH = #06'
966      .EVEN

```

SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND

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

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

```

;+
;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:
;-
    
```

1000
1001
1002
1003 010466
1004 010466
1005 010472 012737 000764 010660
1006 010500 012737 140010 010650
1007 010506 005703
1008 010510 100403
1009 010512 010337 010652
1010 010516 000407
1011 010520 042703 100000
1012 010524 010337 010652
1013 010530 052737 000400 010650
1014 010536 012704 010650
1015 010542 010465 000000
1016 010546 004737 016330
1017 010552 103420
1018 010554
010554 012727 000250
010560 000000
010562 013727 002116
010566 000000
010570 005367 177772
010574 001375

```

SPACE::
    SAVREG
    MOV     #500.,SDELAY
    MOV     #140010,804
    TST     R3
    BMI    54
    MOV     R3,904
    BR     104
54:      BIC     #BIT15,R3
    MOV     R3,904
    BIS     #BIT8,804
    MOV     #804,R4
    MOV     R4,TSDB(R5)
    JSR     PC,WAITF
    BCS    204
    DELAY   250
    MOV     #250,(PC)-
    .WORD  0
    MOV     L#DLY,(PC)-
    .WORD  0
    DEC     -6(PC)
    BNE    .-4
;SAVE THE GENERAL REGISTERS
;SET UP DELAY
;SET UP COMMAND, SPACE FORWARD
;CHECK FOR DIRECTION
;BR IF REVERSE INDICATED
;LOAD UP NUMBER OF RECORDS TO SPACE
;GO DO COMMAND
;CLEAR DIRECTION BIT
;LOAD UP NUMBER OF RECORDS TO SPACE
;SET REVERSE BIT IN COMMAND PACKET
;SET UP R4 WITH PACKET ADDRESS
;SEND OUT COMMAND
;WAIT FOR SSR
;BR IF SSR IS SET AND OK
;DELAY ABOUT .25 SECONDS
    
```

SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND

```

010576 005367 177756          DEC    -22(PC)
010602 001367                BNE    .-20
1019 010604 005337 010660    DEC    SDELAY          ;BUMP DELAY COUNTER DOWN
1020 010610 001356                BNE    15#           ;BR, IF MORE DELAY
1021 010612 000411                BR     60#           ;BR IF TROUBLE CARRY = CLEAR
1022 010614 016501 000002    20#:  MOV    TSSR(R5),R1 ;READ TSSR
1023 010620 012702 000200    MOV    #SSR,R2       ;SET UP EXPECTED
1024 010624 020201    25#:  CMP    R2,R1       ;ARE THEY OK
1025 010626 001401                BEQ    40#           ;BR, IF EQUAL = OK
1026 010630 000402                BR     60#           ;TROUBLE EXIT
1027 010632 000261    40#:  SEC                    ;SET CARRY NO TROUBLE
1028 010634 000401                BR     70#           ;EXIT
1029 010636 000241    60#:  CLC                    ;CARRY CLEAR = ERROR
1030 010640    70#:
1031 010640 010400                MOV    R4,R0
1032 010642 000207                RTS    PC            ;PASS PACKET ADDRESS
1033                                ;RETURN
1034                                ;
1035                                ;
1036                                ;PACKET FOR SPACE COMMAND
1037                                ;
1039                                ;      .=<.*10>&177770
1041                                ;
1042                                ;COMMAND WORD
1043 010650 000000    80#:  .WORD
1044                                ;NUMBER OF RECORDS TO BE SPACED OVER WORD
1045 010652 000000    90#:  .WORD
1046 010654 000000                .WORD
1047 010656 000000                .WORD
1048 010660 000000    SDELAY: .WORD    0          ;DELAY COUNTER
1049                                .EVEN
1050                                .SBTTL WRTCHR - WRITE CHARACTERISTICS COMMAND

```

WRTCHR - WRITE CHARACTERISTICS COMMAND

```

1052
1053 ;ROUTINE TO ISSUE A WRITE CHARACTERISTICS
1054 ;COMMAND SO THAT OTHER COMMANDS WILL BE ACCEPTED
1055
1056 ;INPUT:
1057 ; R4 ADDRESS OF PACKET FROM TEST
1058 ; R5 FIRST DEVICE UNIBUS ADDRESS
1059 ; REQUIRES A CALL TO SOFINIT BE DONE PREVIOUSLY
1060
1061 ;OUTPUT:
1062 ; R0 TSSR CONTENTS
1063 ; CARRY SET - WRITE CHARACTERISTICS COMMAND OK
1064 ; CLR - WRITE CHARACTERISTICS FAILED
1065
1066 ;IMPLICIT OUTPUT:
1067 ;
1068 ; MESSAGE BUFFER AND OTHER BUFFERS ALL SET UP
1069 ; SOFTWARE SWITCHES SET AS FOLLOWS:
1070 ; EXTFEA = EXTENDED FEATURES PRESENT
1071 ; BENBSW = BUFFER ENABLE SWITCH ON OR OFF
1072
1073 ;SIDE EFFECTS:
1074 ;
1075 ;- WRTCHR::
1076 ; SAVREG ;SAVE THE GENERAL REGISTERS
1077 ; CLR BENBSW ;CLEAR BUFFER ENABLE SWITCH
1078 ; CLR EXTFEA ;CLEAR EXTENDED FEATURES SW SWITCH
1079 ; MOV R4, TSD8(R5) ;SEND OUT COMMAND
1080 ; JSR PC, CHKTSSR ;WAIT FOR SSR
1081 ; BCS 20$ ;BR, IF SSR IS SET AND OK
1082 ; BR 60$ ;BR, IF TROUBLE CARRY = CLEAR
1083 ; MOV TSSR(R5), R1 ;READ TSSR
1084 ; MOV #SSR, R2 ;SET UP EXPECTED
1085 ; BIT #OFL, R1 ;WAS OFF LINE SET IN TSSR
1086 ; BEQ 25$ ;BR, IF NO OFL SET
1087 ; BIS #OFL, R2 ;MAKE THEM LOOK ALIKE
1088 ; CMP R2, R1 ;ARE THEY OK
1089 ; BEQ 40$ ;BR, IF EQUAL = OK
1090 ; BR 60$ ;TROUBLE EXIT
1091 ; ADD #8, R4 ;POINT TO WRT CHARA DATA PACKET
1092 ; MOV (R4), R3 ;GET ADDRESS OF MESSAGE BUFFER
1093 ; BIT #X2.EXTF, XST2(R3) ;EXTENDED FEATURES BIT SET?
1094 ; BEQ 45$ ;BR IF NO
1095 ; INC EXTFEA ;SET EXTENDED FEATURES SW SWITCH
1096
1097 ; BIT #X2.BUFE, XST2(R3) ;BUFFER ENABLE SWITCH SET
1098 ; BEQ 50$ ;BR, IF SWITCH NOT SET
1099 ; INC BENBSW ;SET SOFTWARE SWITCH FOR ENABLED
1100
1101 ; MOV XST2(R3), REV ;MICROCODE REV LEVEL
1102 ; BIC #17700, REV ;CLEAR UNWANTED BITS
1103 ; CMP #1, REV ;IS IT A NEW MICROCODE
1104 ; BEQ 55$ ;NO BR
1105 ; MOV #1, EXTFEA ;ALWAY EXTENDED FEATURE FOR NEW
1106 ; ;MICROCODE
1107 ; BIS #X2.EXTF, XST2(R3) ;EXTENDED FEATURE ALWAYS SET IN
1108 ; ;MICROCODE

```

D5

WRTCHR - WRITE CHARACTERISTICS COMMAND

1109 011040 000261
1110 011042 000401
1111 011044 000241
1112 011046 016500 000002
1113 011052 000207

554: SEC
 BR 704
604: CLC
704: MOV TSSR(R5),R0
 RTS PC

;SET CARRY NO TROUBLE
;EXIT
;CARRY CLEAR = ERROR
;RETURN TSSR CONTENTS
;RETURN

REWIND - POSITION TAPE (REWIND) COMMAND

```

1115 .SBTTL REWIND - POSITION TAPE (REWIND) COMMAND
1116
1117 ;*
1118 ; THIS ROUTINE WILL REWIND THE SELECTED TAPE.
1119
1120 ; CAUTION: THE ROUTINE DOES NOT WAIT FOR BOT
1121 ; TO ARRIVE. ALSO THE CALLER MUST CHECK FOR
1122 ; SSR TO SET IN THE TSSR
1123
1124 ;
1125 ; CALLING SEQUENCE:
1126
1127 ; DO A SOFT INIT
1128 ; DO A WRITE CHARACTERISTICS
1129 ; JSR PC,REWIND
1130
1131 ; INPUT:
1132
1133 ; R5 FIRST DEVICE UNIBUS ADDRESS
1134
1135 ;
1136 ; OUTPUT
1137
1138 ; R0 THE CONTENTS OF R4 IS PASSED TO R0
1139
1140 ;
1141 ;
1142 REWIND:
1143 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1144 MOV #RMPACK,R4 ;GET PACKET ADDRESS
1145 MOV R4,TSDB(R5) ;SEND PACKET ADDRESS TO EXECUTE
1146 MOV #360,R3 ;ENOUGH TIME FOR 2400' REEL TO REWIND
1147 JSR PC,WAITF ;WAIT FOR SSR TO SET
1148 BCS 20$ ;LEAVE WHEN SSR IS SET
1149 DELAY 250 ;WAIT FOR .25 SECONDS
1150 MOV #250,.(PC)-
1151 .WORD 0
1152 MOV L$DLY,(PC)-
1153 .WORD 0
1154 DEC -6(PC)
1155 BNE -4
1156 DEC -22(PC)
1157 BNE -20
1158 DEC R3 ;BUMP COUNTER DOWN
1159 BNE 10$ ;KEEP GOING
1160 CLC ;CLEAR CARRY TO SET ERROR
1161 MOV R4,R0 ;PASS THE PACKET ADDRESS
1162 RTS PC ;RETURN
1163
1164 RMPACK: .=<.>10>&177770
1165 .WORD 102010 ;POSTION COMMAND (REWIND)
1166 .WORD 0 ;NOT USED

```

CKRAM - COMPARE RAM TO I/O PACKET

```

1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191 011154
1192 011154
1193 011160 012701 002242
1194 011164 012702 000201
1195 011170 005003
1196 011172 004737 016416
1197 011176 112765 000000 000000
1198 011204 004737 016416 10#:
1199 011210 010265 000000
1200 011214 004737 016416
1201 011220 116511 000000
1202 011224 122124
1203 011226 001401
1204 011230 005203
1205 011232 005202 20#:
1206 011234 020227 000210
1207 011240 003761
1208 011242 005703
1209 011244 001402
1210 011246 000241
1211 011250 000401
1212 011252 000261 30#:
1213 011254 012737 000010 002302 50#:
1214 011262 000207
    
```

```

.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
  CLR    R3              ;CLEAR THE ERROR FLAG
  JSR    PC,CHKTSSR      ;WAIT FOR SSR
  MOVB   #0,TSDB(R5)     ;SET MAINTENANCE MODE
  JSR    PC,CHKTSSR      ;WAIT FOR SSR TO SET
  MOV    R2,TSDB(R5)     ;SELECT NEXT RAM ADDRESS
  JSR    PC,CHKTSSR      ;WAIT FOR SSR TO SET
  MOVB   TSBA(R5),(R1)   ;READ THE RAM DATA
  CMPB   (R1),.(R4)     ;COMPARE TO EXPECTED
  BEQ    20#            ;BRANCH IF OK
  INC    R3              ;SET ERROR FLAG
  INC    R2              ;ADDRESS OF NEXT RAM LOCATION
  CMP    R2,#RMPKTEND   ;REACHED END YET ?
  BLE    10#           ;BRANCH TILL ALL READ
  TST    R3              ;WAS AN ERROR FOUND ?
  BEQ    30#           ;BRANCH IF NOT
  CLC
  BR     50#           ;CLEAR CARRY TO SHOW ERROR
  SEC
  MOV    #8.,RAMSIZ     ;AND EXIT
  RTS   PC              ;SHOW GOOD COMPARE
                          ;SETUP RAMSIZ FOR PRAMPKT ROUTINE
                          ;RETURN
    
```

CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA

```

1216 .SBTTL CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA
1217 ;*
1218 ;
1219 ;ROUTINE TO READ THE FIRST 8 OR 10 BYTES FROM RAM
1220 ;MEMORY AND COMPARE THIS DATA TO A CHARACTERISTICS DATA BLOCK.
1221 ;
1222 ;INPUT:
1223 ;
1224 ; R4 ADDRESS OF THE CHARACTERISTICS DATA
1225 ; R5 FIRST DEVICE UNIBUS ADDRESS
1226 ;
1227 ;OUTPUT:
1228 ;
1229 ; CARRY SET - RAM MATCHES PACKET
1230 ; CLR - RAM DOES NOT MATCH PACKET
1231 ;
1232 ;IMPLICIT OUTPUT:
1233 ;
1234 ; THE TABLE RAMDATA IS FILLED WITH THE
1235 ; DATA HELD IN RAM.
1236 ; RAMSIZ IS SET TO 8. OR 10. FOR PRAMPKT ROUTINE
1237 ;
1238 ;SIDE EFFECTS:
1239 ;
1240 ; THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
1241 ;-
1242 CKRAM2::
1243 SAVREG
1244 MOV #RAMDATA,R1 ;SAVE THE GENERAL REGISTERS
1245 MOV #RMCHBEG,R2 ;ADDRESS TO SAVE THE RAM DATA
1246 CLR R3 ;BYTE ADDRESS OF FIRST RAM DATA
1247 JSR PC,CHKTSSR ;CLEAR THE ERROR FLAG
1248 MOV #0,TSDB(R5) ;WAIT FOR SSR
1249 JSR PC,CHKTSSR ;SET MAINTENANCE MODE
1250 MOV R2,TSDB(R5) ;WAIT FOR SSR TO SET
1251 JSR PC,CHKTSSR ;SELECT NEXT RAM ADDRESS
1252 MOV# TSBA(R5),(R1) ;WAIT FOR SSR TO SET
1253 CMPB (R1),.(R4). ;READ THE RAM DATA
1254 BEQ 20$ ;COMPARE TO EXPECTED
1255 INC R3 ;BRANCH IF OK
1256 INC R2 ;SET ERROR FLAG
1257 MOV #8, RAMSIZ ;ADDRESS OF NEXT RAM LOCATION
1258 TST EXTFEA ;ASSUME EXTFEA NOT SET
1259 BEQ 25$ ;IS THE SOFTWARE EXTENDED FEATURES SET
1260 MOV #10, RAMSIZ ;BR, IF NOT SET
1261 CMP R2, #RMCHEND ;SET RAMSIZ FOR EXTEND FEATURES
1262 BLE 10$ ;AT END OF EXTENDED BUFFER
1263 BR 27$ ;BR, IF NOT AT END YET
1264 CMP R2, #RMCHEND-2 ;AT END BRANCH
1265 BLE 10$ ;REACHED END YET ?
1266 TST R3 ;BRANCH TILL ALL READ
1267 BEQ 30$ ;WAS AN ERROR FOUND ?
1268 CLC ;BRANCH IF NOT
1269 BR 50$ ;CLEAR CARRY TO SHOW ERROR
1270 SEC ;AND EXIT
1271 RTS PC ;SHOW GOOD COMPARE
;RETURN

```

CKMSG - COMPARE WRITE CHAR. MESSAGE BUFFERS

```

1273 .SBTTL CKMSG - COMPARE WRITE CHAR. MESSAGE BUFFERS
1274
1275 ;*
1276 ;ROUTINE TO COMPARE A WRITE CHARACTERISTICS EXPD AND RECV
1277 ;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
1278 ;ERROR PRINT ROUTINES.
1279
1280 ;INPUT:
1281
1282 ; R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS
1283 ; R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS
1284 ; R2 EXPD MESSAGE BUFFER ADDRESS
1285 ;OUTPUT:
1286
1287 ; CARRY SET - MESSAGE BUFFERS MATCH
1288 ; CLR -MESSAGE BUFFERS DON'T MATCH
1289
1290 ;IMPLICIT OUTPUT:
1291
1292 ; EXPMSG BUFFER IS SET TO EXPD DATA
1293 ; RECMMSG BUFFER IS SET TO RECV DATA
1294 ; RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
1295 ; RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
1296
1297 ;-
1298 CKMSG::
1299 SAVREG
1300 MOV R0,RCVHIADD ;SAVE R1-R5 UNTIL NEXT RETURN
1301 MOV R1,RCVLOAD ;SAVE RECV HIGH ADDRESS
1302 TST KTENABLE ;SAVE RECV LOW ADDRESS
1303 BEQ 10$ ;TESTING ABOVE 28K?
1304 JSR PC,SETMAP ;BR IF NO
1305 MOV R0,R1 ;RETURN ADDRESS BIASED TO PAR6 IN R0
1306 10$: CLR R4 ;GET RETURNED ADDRESS BIASED TO PAR6
1307 CLR R3 ;WORD IN BUFFER
1308 MOV R2,R5 ;CLEAR ERROR SEEN FLAG
1309 15$: MOV (R2),EXPMSG(R4) ;GET EXPD BUFFER ADDRESS
1310 MOV (R1),RCMSG(R4) ;SAVE EXPD FOR ERROR REPORT
1311 CMP (R2), (R1) ;SAVE RECV FOR ERROR REPORT
1312 BEQ 25$ ;EXPD EQUAL RECV?
1313 INC R3 ;BR IF YES
1314 25$: ADD #2,R4 ;SET ERROR SEEN FLAG
1315 CMP R4,#14 ;POINT TO NEXT WORD ADDRESS
1316 BLE 15$ ;DONE FIRST 7 WORDS?
1317 CMP #1,REV ;BR IF NO
1318 BNE 30$ ;IS IT A NEW MICROCODE
1319 BIT #X2.EXTF,XST2(R5) ;NO BR
1320 BEQ 50$ ;IS EXTENDED FEATURES SET IN EXPD?
1321 30$: CMP R4,#16 ;BR IF NO
1322 BLE 15$ ;DONE EXTENDED FEATURES WORD?
1323 50$: TST R3 ;BR IF NO
1324 BEQ 55$ ;ANY ERRORS SEEN?
1325 CLC ;BR IF NO
1326 BR 60$ ;SET FAILURE
1327 55$: SEC ;SET SUCCESS
1328 60$: RTS PC ;RETURN

```

CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

```

1330 .SBTTL CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS
1331 ;*
1332 ;ROUTINE TO COMPARE AN EXPECTED AND RECEIVED MESSAGE
1333 ;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
1334 ;ERROR PRINT ROUTINES.
1335 ;
1336 ;INPUT:
1337 ; R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS
1338 ; R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS
1339 ; R2 EXPD MESSAGE BUFFER ADDRESS
1340 ; R3 NUMBER OF BYTES TO COMPARE
1341 ;
1342 ;OUTPUT:
1343 ; CARRY SET - MESSAGE BUFFERS MATCH
1344 ; CLR - MESSAGE BUFFERS DON'T MATCH
1345 ;
1346 ;IMPLICIT OUTPUT:
1347 ; EXPMSG BUFFER IS SET TO EXPD DATA
1348 ; RECVMSG BUFFER IS SET TO RECV DATA
1349 ; RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
1350 ; RCVLOAD SET TO LOW ORDER ADDRESS OF RECV
1351 ;-
1352 CKMSG2::
1353 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1354 CMP R3,#RECVMSG-EXPMSG ;@AD IS COUNT ABOVE MAX ALLOWED?
1355 BLE 5# ;@AD BR IF NO
1356 MOV #RECVMSG-EXPMSG,R3 ;@AD
1357 PRINTF #DEBUGMSG ;@AD
1358 MOV #DEBUGMSG,-(SP)
1359 MOV #1,-(SP)
1360 MOV SP,R0
1361 TRAP C#PNTF
1362 ADD #4,SP
1363 MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
1364 MOV R1,RCVLOAD ;SAVE RECV LOW ADDRESS
1365 TST KTNABLE ;TESTING ABOVE 28K?
1366 BEQ 10# ;BR IF NO
1367 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
1368 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
1369 CLR R4 ;WORD IN BUFFER
1370 CLR R5 ;CLEAR ERROR SEEN FLAG
1371 MOVB (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
1372 MOVB (R1),RECVMSG(R4) ;SAVE RECV FOR ERROR REPORT
1373 CMPB (R2),.(R1) ;EXPD EQUAL RECV?
1374 BEQ 25# ;BR IF YES
1375 INC R5 ;SET ERROR SEEN FLAG
1376 ADD #1,R4 ;POINT TO NEXT BYTE
1377 CMP R4,R3 ;DONE ALL BYTES?
1378 BGE 50# ;BR IF YES
1379 BR 15# ;DO NEXT BYTE
1380 TST R5 ;ANY ERRORS SEEN?
1381 BEQ 55# ;BR IF NO
1382 CLC ;SET FAILURE
1383 BR 60#
1384 SEC ;SET SUCCESS
1385 RTS PC ;RETURN

```

J5

CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

```

1382 011702      120      122      117  DEBUGMSG: .ASCIZ 'PROGRAM INTERNAL ERROR -CKMSG2 MESSAGE BUFFER EXCEEDED-';@@D
1383 011772      045      116      045  FERCM:  .ASCII /#N#A ***/
1384 012003      040      040      124  ERCM:   .ASCIZ / TSSR ERROR CODE REC'D = /
1385 012036      056      056      056  SIMSG:  .ASCIZ /... AFTER DOING SOFT INIT/
1386 012071      124      105      123  TINERR: .ASCIZ /TEST: .../
1387                                     .EVEN
1388                                     ;*
1389                                     ;
1390                                     ;PRINT ROUTINE TO FATAL SOFT INIT ERRORS
1391                                     ;
1392                                     ;INPUT:
1393                                     ;
1394                                     ;      R1      CONTENTS OF TSSR AT ERROR
1395                                     ;
1396                                     ;SIDE EFFECTS:
1397                                     ;
1398                                     ;      EXECUTES DROP UNIT TO CEASE TESTING
1399                                     ;
1400                                     ;
1401                                     ;-
1402 012104      012104      BGNMSG  SFIMSG
1403 012104      004737  006024  SFIMSG:  JSR      PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
1404 012110      004737  017262  JSR      PC,CKDROP      ;DROP UNIT, IF ALLOWED
1405 012114      012114      ENDMSG
1406 012114      104423  L10003:  TRAP    C#MSG
1407                                     ;*
1408                                     ;PRINT ROUTINE TO PRINT THE CONTENTS OF
1409                                     ;TSSR AND A COMMAND PACKET OTHER THAN GET STATUS COMMAND PACKET.
1410                                     ;
1411                                     ;INPUTS:
1412                                     ;
1413                                     ;      R1      TSSR CONTENTS
1414                                     ;      R4      ADDRESS OF COMMAND PACKET
1415                                     ;
1416                                     ;-
1417                                     ;
1418 012116      012116      BGNMSG  PKTSSR
1419 012116      004737  006024  PKTSSR:  JSR      PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
1420 012122      012700  000004  MOV      #4,R0          ;NO. OF WORDS IN PACKET
1421 012126      004737  007370  JSR      PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
1422 012132      012132      ENDMSG
1423 012132      104423  L10004:  TRAP    C#MSG

```

K5

CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

1424
1425
1426
1427
1428
1429
1430
1431
1432
1433 012134
012134
1434 012134 004737 006024
1435 012140 012700 000002
1436 012144 004737 007370
1437 012150
012150
012150 104423
1438
1439
1440
1441
1442
1443
1444
1445 012152
012152
1446 012152 004737 006024
1447 012156
012156
012156 104423
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460 012160
012160
1461 012160 004737 006024
1462 012164 010200
1463 012166 010301
1464 012170 004737 014312
1465 012174
012174
012174 104423

```

;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:
JSR    PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
MOV    #2,R0           ;NO. OF WORDS IN GET STATUS PACKET
JSR    PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
ENDMSG
L10005:
TRAP   C#MSG
;
;PRINT TSSR ERRORS FOR INITIALIZATION TESTS
;
;INPUTS:
;
;   R1   TSSR CONTENTS
;   R4   ADDRESS OF COMMAND PACKET
;-
BGNMSG  SFFMSG
SFFMSG:
JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
ENDMSG
L10006:
TRAP   C#MSG
.SBTTL  PKTMES - PRINT TSSR AND MESSAGE BUFFER
;
;PRINT ROUTINE TO PRINT THE CONTENTS OF TSSR AND MESSAGE
;BUFFER FOR ERROR REPORTS
;
;INPUTS:
;
;   R1   CONTENTS OF TSSR
;   R2   LOW ORDER MESSAGE BUFFER
;   R3   HIGH ORDER MESSAGE BUFFER ADDRESS
;   NOTE: R3 IS IGNORED IF KTENABLE FLAG IS CLEAR
;-
BGNMSG  PKTMES
PKTMES:
JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR
MOV    R2,R0           ;LOW ORDER ADDRESS
MOV    R3,R1           ;HIGH ORDER ADDRESS
JSR    PC,PRMESS      ;PRINT THE MESSAGE BUFFER
ENDMSG
L10007:
TRAP   C#MSG

```

L5

ADDSSR - PRINT TEST ADDRESS AND TSSR

```

1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479 012176
      012176
1480 012176 004737 010274
1481 012202 016501 000002
1482 012206 004737 006024
1483 012212
      012212
      012212 104423
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497 012214
      012214
1498 012214 012700 000007
1499 012220 005737 002224
1500 012224 001402
1501 012226 012700 000010
1502 012232 004737 014632
1503 012236
      012236
      012236 104423

```

```

.SBTTL ADDSSR - PRINT TEST ADDRESS AND TSSR
;*
;PRINT ROUTINE TO PRINT THE CONTENTS OF
;TSSR AND A MEMORY TEST ADDRESS
;
;INPUTS:
;
;      R5      FIRST DEVICE UNIBUS ADDRESS
;      ERRHI   HIGH ORDER MEMORY TEST ADDRESS
;      ERRLO   LOW ORDER MEMORY TEST ADDRESS
;-
      BGNMSG  ADDSSR
ADDSSR:  JSR    PC,PRITADD      ;PRINT MEMORY TEST ADDRESS
        MOV    TSSR(R5),R1    ;GET CURRENT TSSR
        JSR    PC,PRITSSR     ;PRINT THE CONTENTS OF TSSR REGISTER
        ENDMSG
L10010: TRAP   C#MSG
.SBTTL MSGEXP - PRINT WRITE CHAR. EXPD-RCV MESSAGE BUFFERS
;*
;PRINT ROUTINE TO PRINT WRITE CHARACTERISTIC MESSAGE BUFFER
;
;IMPLICIT INPUTS:
;
;      EXPMSG  - EXPECTED MESSAGE BUFFER
;      RECMMSG - RECEIVED MESSAGE BUFFER
;      RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
;      RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
;-
      BGNMSG  MSGEXP
MSGEXP:  MOV    #7,R0           ;ASSUME NO EXT FEATURES
        TST    EXTFEA         ;EXT FEATURES SET?
        BEQ   S#,            ;BR IF NO
        MOV    #8,R0         ;EXT FEATURE BUFFER IS 8 WORDS
        JSR    PC,PRMSGEXP    ;PRINT EXPD/RCV MESSAGE BUFFERS
        ENDMSG
L10011: TRAP   C#MSG

```


FIFEXP - PRINT FIFO EXP/RECV DATA

```

1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517 012240
      012240
1518 012240
      012240 010146
      012242 012746 012312
      012246 012746 000002
      012252 010600
      012254 104415
      012256 062706 000006
1519 012262
      012262 012746 012361
      012266 012746 000001
      012272 010600
      012274 104415
      012276 062706 000004
1520 012302 010100
1521 012304 004737 015202
1522 012310
      012310
      012310 104423
1523 012312 045 116
1524 012361 045 116
1525

```

```

.SBTTL FIFEXP - PRINT FIFO EXP/REC DATA
;
;PRINT ROUTINE TO PRINT FIFO EXP/REC DATA
;
; R1 - BYTE COUNT
;
;IMPLICIT INPUTS:
;
; EXPMSG - EXPECTED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
; RECVMSG - 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
;ASCIZ '#N#A NUMBER OF BYTES TRANSFERRED = #D2'
;ASCIZ '#N#A FIFO DATA BYTES IN ERROR:'
.EVEN

```

MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS

```

1527          .SBTTL MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS
1528          ;*
1529          ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
1530          ;
1531          ;
1532          ;IMPLICIT INPUTS:
1533          ;
1534          ;
1535          ;   EXPMSG - EXPECTED MESSAGE BUFFER
1536          ;   RECMSG - RECEIVED MESSAGE BUFFER
1537          ;   RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1538          ;   RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1539          ;
1540          BGNMSG MSGSTAT
1541          MSGSTAT:
1542          MOV     #STATCOD,R1      ;ASCII ADDRESS TABLE
1543          10$:  MOV     (R1)+,R0    ;DONE ALL MSG LINES?
1544          BEQ    20$              ;BR IF YES
1545          PRINTX R0              ;PRINT STATUS BIT NAMES
1546          MOV     R0,-(SP)
1547          MOV     #1,-(SP)
1548          MOV     SP,R0
1549          TRAP   C#PNTX
1550          ADD    #4,SP
1551          BR     10$              ;DO ANOTHER MSG LINE
1552          20$:  MOV     #10,R0     ;NUMBER OF WORDS IN A READ STATUS BUFFER
1553          JSR   PC,PRMSGEXP      ;PRINT EXPD/RCV MESSAGE BUFFERS
1554          ENDMSG
1555          L10013: TRAP   C#MSG
1556          STATCOD: .WORD 1$,2$,3$,4$,5$,6$,0
1557          1$: .ASCIZ 'MMA Tape Bus Signals in Word #8:'
1558          2$: .ASCIZ 'MMA PARENR<15> IEOT <12> IFMK <9> IRDY<6> IRND<2>'
1559          3$: .ASCIZ 'MMA IRESV2<14> IIDENT<11> IHER <8> IONL<5> IFBY<1>'
1560          4$: .ASCIZ 'MMA IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>'
1561          5$: .ASCIZ 'MMA Tape Bus Signals in Word #9:'
1562          6$: .ASCIZ 'MMA DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>'
1563          .EVEN

```

MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS

```

1560                                     .SBTTL MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS
1561                                     ;*
1562                                     ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RECV
1563                                     ;
1564                                     ;IMPLICIT INPUTS:
1565                                     ;
1566                                     ;     EXPMSG - EXPECTED MESSAGE BUFFER
1567                                     ;     RECMSG - RECEIVED MESSAGE BUFFER
1568                                     ;     RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1569                                     ;     RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1570                                     ;
1571                                     ;-
1572 013134 BGNMSG MSGLOOP
1573 013134 MSGLOOP::
1574 013134 012701 013176 MOV #LOOPCOD,R1 ;ASCII ADDRESS TABLE
1575 013140 012100 10#: MOV (R1)+,RO ;DONE ALL MSG LINES?
1576 013142 001410 BEQ 20# ;BR IF YES
1577 013144 PRINTX RO ;PRINT STATUS BIT NAMES
1578 013144 010046 MOV RO,-(SP)
1579 013146 012746 000001 MOV #1,-(SP)
1580 013152 010600 MOV SP,RO
1581 013154 104415 TRAP C#PNTX
1582 013156 062706 000004 ADD #4,SP
1583 013162 000766 BR 10# ;DO ANOTHER MSG LINE
1584 013164 012700 000012 20#: MOV #10,RO ;NUMBER OF WORDS IN A READ STATUS BUFFER
1585 013170 004737 014632 JSR PC,PRMSGEXP ;PRINT EXPD/RECV MESSAGE BUFFERS
1586 013174 ENDMSG
1587 013174 L10014: TRAP C#MSG
1588 013174 104423
1589 013174
1590 1581
1582 013176 013216 013271 013370 LOOPCOD: .WORD 1#,2#,3#,4#,5#,6#,7#,0
1583 013216 045 116 045 1#: .ASCIZ '#N#A Tape Bus Loopback Signals in Word #8:'
1584 013271 045 116 045 2#: .ASCIZ '#N#A PARERR<15> IRESV2<14> IRESV1<13>'
1585 013370 045 116 045 3#: .ASCIZ '#N#A IHISP=>IEOT<12> IWRT=>IIDENT<11> IREV =>ICER <10>'
1586 013467 045 116 045 4#: .ASCIZ '#N#A IWFM =>IFMK<09> IEDIT=>IHER <08> IFAD =>ISPEED<07>'
1587 013566 045 116 045 5#: .ASCIZ '#N#A ITADO=>IRDY<06> ITAD1=>IONL <05> IERASE=>ILDPA <04>'
1588 013665 045 116 045 6#: .ASCIZ '#N#A IREW =>IDBY<03> IRWU =>IRWD <02> IFEN =>IFBY <01>'
1589 013764 045 116 045 7#: .ASCIZ '#N#A IGO =>IFPT<00>'
1590 .EVEN

```

C6

MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER

```

1592                                     .SBTTL MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER
1593                                     ;*
1594                                     ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
1595                                     ;
1596                                     ;
1597                                     ;IMPLICIT INPUTS:
1598                                     ;
1599                                     ;
1600                                     ;     EXPMSG - EXPECTED MESSAGE BUFFER
1601                                     ;     RECMSG - RECEIVED MESSAGE BUFFER
1602                                     ;     RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1603                                     ;     RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1604                                     ;
1605                                     ;-
1605 014012 BGNMSG MSGSUB
1606 014012 MSGSUB:: MOV #10,R0 ;SIZE OF WRITE SUBSYSTEM BUFFER
1607 014016 012700 000012 JSR PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
1608 014022 ENDMSG
1609 014022 L10015: TRAP C#MSG
1610                                     ;
1611                                     ;
1612                                     .SBTTL MEMADD - PRINT MEMORY ADDRESS DATA ERROR
1613                                     ;*
1614                                     ;PRINT ROUTINE TO PRINT MEMORY ADDRESS DATA COMPARE ERROR
1615                                     ;
1616                                     ;IMPLICIT INPUTS:
1617                                     ;
1618                                     ;     ERRHI - MEMORY ERROR HIGH ORDER ADDRESS
1619                                     ;     ERRLO - MEMORY ERROR LOW ORDER ADDRESS
1620                                     ;     EXP - EXPECTED DATA
1621                                     ;     RECV - RECEIVED DATA
1622                                     ;
1622 014024 BGNMSG MEMADD
1623 014024 MEMADD:: JSR PC,PRMADD ;PRINT MEMORY ADDRESS IN ERROR
1624 014030 004737 010160 MOV EXPD,R1 ;GET EXPD DATA
1625 014034 013701 002232 MOV RECV,R2 ;GET RECEIVED DATA
1626 014040 004737 007742 JSR PC,PRIXOR ;PRINT EXPD/RCV
1627 014044 ENDMSG
1627 014044 L10016: TRAP C#MSG
1627 014044 104423

```

PRAMPKT - PRINT RAM AND PACKET DATA

```

1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645 014046
1646 014046
1647 014052 012701 002242
1648 014056 005002
1649 014060 122124
1650 014062 001005
1651 014064
1652 014074 000436
1653 014076 116105 177777
1654 014102 116403 177777
1655 014106
1656 014116 042703 177400
1657 014122 116137 177777 002234
1658 014130 116437 177777 002232
1659 014136
    014136 010346
    014140 013746 002232
    014144 013746 002234
    014150 010246
    014152 012746 014226
    014156 012746 000005
    014162 010600
    014164 104414
    014166 062706 000014
1660 014172 005202
1661 014174 005737 002302
1662 014200 001404
1663 014202 020237 002302
1664 014206 003724
1665 014210 000403
1666 014212 020227 000010
1667 014216 002720
1668 014220 005037 002302
1669 014224 000207
1670
1671 014226 045 116 045 RAMASC: .ASCIZ 'N#A BYTE: #D2#A RAM: #03#A Packet: #03#A XOR:#03'
1672

```

```

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

```

PRMESS - PRINT CONTENTS OF MESSAGE BUFFER

```

1674                                     .SBTTL PRMESS - PRINT CONTENTS OF MESSAGE BUFFER
1675                                     ;*
1676                                     ;THIS ROUTINE PRINTS THE CONTENTS OF
1677                                     ;THE 7 OR 8 WORD MESSAGE BUFFER RETURNED BY THE TSV-05.
1678                                     ;
1679                                     ;INPUT:
1680                                     ;      R0      LOW ORDER ADDRESS OF MESSAGE BUFFER
1681                                     ;      R1      HIGH ORDER ADDRESS OF MESSAGE BUFFER
1682                                     ;      NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
1683                                     ;THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
1684                                     ;-
1685 014312 PRMESS: SAVREG                                     ;SAVE THE REGISTERS
1686 014316 010005 MOV      R0,R5                                     ;SAVE LOW ORDER ADDRESS
1687 014320 005737 003134 TST      KTENABLE                                     ;ADDRESS ABOVE 28K?
1688 014324 001001 BNE      10$                                           ;BR IF YES
1689 014326 005001 CLR      R1                                           ;SET HIGH ORDER ADDRESS TO 0
1690 014330 010103 10$: MOV     R1,R3                                     ;SAVE HIGH ORDER ADDRESS
1691 014332 006100 ROL     R0                                           ;SHIFT BIT15 TO C BIT
1692 014334 006101 ROL     R1                                           ;SHIFT TO HIGH ORDER FOR PRINTOUT
1693 014336 PRINTX  #PROASC,R1,R5                               ;PRINT MESSAGE BUFFER ADDRESS
1694 014336 010546 MOV     R5,-(SP)
1695 014340 010146 MOV     R1,-(SP)
1696 014342 012746 014500 MOV     #PROASC,-(SP)
1697 014346 012746 000003 MOV     #3,-(SP)
1698 014352 010600 MOV     SP,R0
1699 014354 104415 TRAP   C#PNTX
1700 014356 062706 000010 ADD     #10,SP
1701 014362 PRINTX  #PRIASC                                     ;PRINT HEADER FOR CONTENTS
1702 014362 012746 014545 MOV     #PRIASC,-(SP)
1703 014366 012746 000001 MOV     #1,-(SP)
1704 014372 010600 MOV     SP,R0
1705 014374 104415 TRAP   C#PNTX
1706 014376 062706 000004 ADD     #4,SP
1707 014402 005004 CLR     R4                                           ;NUMBER OF THE NEXT WORD
1708 014404 010501 MOV     R5,R1                                     ;COPY LOW ORDER ADDRESS
1709 014406 010300 MOV     R3,R0                                     ;COPY HIGH ORDER ADDRESS
1710 014410 001403 BEQ    20$                                           ;BR IF NOT ABOVE 28K
1711 014412 004737 017376 JSR    PC,SETMAP                                     ;SETUP PAR ADDRESS IN R0
1712 014416 010005 MOV     R0,R5                                     ;GET PAR FORMAT ADDRESS ABOVE 28K
1713 014420 20$: PRINTX #PRASC,R4,(R5)-                     ;PRINT THE CONTENTS OF MEMORY BUFFER
1714 014420 012546 MOV     (R5)-,-(SP)
1715 014422 010446 MOV     R4,-(SP)
1716 014424 012746 014603 MOV     #PRASC,-(SP)
1717 014430 012746 000003 MOV     #3,-(SP)
1718 014434 010600 MOV     SP,R0
1719 014436 104415 TRAP   C#PNTX
1720 014440 062706 000010 ADD     #10,SP
1721 014444 005204 INC     R4                                           ;NUMBER OF THE NEXT
1722 014446 020427 000007 CMP     R4,#7                                     ;DONE ALL YET ?
1723 014452 003011 BGT    50$                                           ;BRANCH IF ALL DONE
1724 014454 002761 BLT    20$                                           ;PRINT FIRST 7 WORDS
1725 014456 022737 000001 002226 CMP     #1,REV                                     ;IS IT A NEW MICROCODE
1726 014464 001355 BNE    20$                                           ;NO BR
1727 014466 032763 000200 000012 BIT     #X2.EXTF,XST2(R3) ;EXTENDED FEATUTES ON ?
1728 014474 001351 BNE    20$                                           ;PRINT EXTENDED STATUS WORD
1729 014476 000207 50$: RTS    PC                                           ;RETURN
1730 014500 045 116 045 PROASC: .ASCIZ '#N$A Message Buffer Address = #01#05'

```

F6

PRMESS - PRINT CONTENTS OF MESSAGE BUFFER

1712	014545	045	116	045	PR1ASC: .ASCIZ	'#N#A Message Buffer Contents:'
1713	014603	045	116	045	PRASC: .ASCIZ	'#N#A Word#D1#A: #0'

PRMESS - PRINT CONTENTS OF MESSAGE BUFFER

```

1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726 014632
1727 014632
1728 014636 010005
1729 014640 013700 002306
1730 014644 010004
1731 014646 013701 002304
1732 014652 006100
1733 014654 006101
1734 014656
    014656 010446
    014660 010146
    014662 012746 015012
    014666 012746 000003
    014672 010600
    014674 104415
    014676 062706 000010
1735 014702
    014702 012746 015057
    014706 012746 000001
    014712 010600
    014714 104415
    014716 062706 000004
1736 014722 005004
1737 014724 012701 002322
1738 014730 012702 002466
1739 014734 011100
1740 014736 011203
1741 014740
1742 014750
    014750 010346
    014752 012246
    014754 012146
    014756 010446
    014760 012746 015115
    014764 012746 000005
    014770 010600
    014772 104415
    014774 062706 000014
1743 015000 005204
1744 015002 020405
1745 015004 002001
1746 015006 000752
1747 015010 000207
1748 015012 045 116
1749 015057 045 116
1750 015115 045 116

    .EVEN
    .SBTTL PRMSGEXP - PRINT EXPD/RCV MESSAGE BUFFERS
; *
; ROUTINE TO PRINT EXPECTED AND RECEIVED MESSAGE BUFFERS
; RO - NUMBER OF WORDS IN BUFFER
; IMPLICIT INPUTS:
; EXPMSG - EXPECTED MESSAGE BUFFER
; RECMMSG - RECEIVED MESSAGE BUFFER
; RCVHIADD - RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
; RCVLOADD - RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
; -
PRMSGEXP::
    SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
    MOV RO,R5 ;SAVE NUMBER OF WORDS
    MOV RCVLOADD,RO ;GET RECV LOW ADDRESS
    MOV RO,R4 ;COPY LOW ADDRESS
    MOV RCVHIADD,R1 ;GET RECV HIGH ADDRESS
    ROL RO ;SHIFT BIT15 TO C BIT
    ROL R1 ;SHIFT TO HIGH ORDER FOR PRINTOUT
    PRINTX #PRMSGO,R1,R4 ;PRINT MESSAGE BUFFER ADDRESS
    MOV R4,-(SP)
    MOV R1,-(SP)
    MOV #PRMSGO,-(SP)
    MOV #3,-(SP)
    MOV SP,RO
    TRAP C#PNTX
    ADD #10,SP
    PRINTX #PRMSG1 ;PRINT HEADER FOR CONTENTS
    MOV #PRMSG1,-(SP)
    MOV #1,-(SP)
    MOV SP,RO
    TRAP C#PNTX
    ADD #4,SP
    CLR R4 ;NUMBER OF THE CURRENT WORD
    MOV #EXPMSG,R1 ;GET EXPD BUFFER ADDRESS
    MOV #RECMMSG,R2 ;GET RECV BUFFER ADDRESS
    20$: MOV (R1),RO ;GET EXPD
    MOV (R2),R3 ;GET RECV
    XOR RO,R3 ;XOR EXPD/RCV
    PRINTX #PRMSG2,R4,(R1)*,(R2)*,R3
    MOV R3,-(SP)
    MOV (R2)*,-(SP)
    MOV (R1)*,-(SP)
    MOV R4,-(SP)
    MOV #PRMSG2,-(SP)
    MOV #5,-(SP)
    MOV SP,RO
    TRAP C#PNTX
    ADD #14,SP
    INC R4 ;NUMBER OF THE NEXT
    CMP R4,R5 ;DONE ALL YET?
    BGE 50$ ;BR IF YES
    BR 20$ ;DO ANOTHER
    50$: RTS PC ;RETURN
    PRMSG0: .ASCIZ 'N#A Message Buffer Address = #01#05'
    PRMSG1: .ASCIZ 'N#A Message Buffer Contents:'
    PRMSG2: .ASCIZ 'N#A WORD #D2#A EXPD: #06#A RECV: #06#A XOR: #06'

```


PRMSGEXP - PRINT EXPD/RECV MESSAGE BUFFERS

```

1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766 015202
1767 015202
1768 015206 010005
1769 015210 005037 002320
1770 015214 005004
1771 015216 012701 002322
1772 015222 012702 002466
1773 015226 111100
1774 015230 042700 177400
1775 015234 110037 015550
1776 015240 111203
1777 015242 042703 177400
1778 015246 110337 015552
1779 015252
1780 015262 122122
1781 015264 001431
1782 015266 005237 002320
1783 015272 023727 002320 000010
1784 015300 101023
1785 015302
    015302 010346
    015304 013746 015552
    015310 013746 015550
    015314 010446
    015316 012746 015416
    015322 012746 000005
    015326 010600
    015330 104415
    015332 062706 000014
1786 015336
1787 015346 000404
1788 015350
1789 015350
1790 015360
1791 015360 005204
1792 015362 020405
1793 015364 002001
1794 015366 000717
1795 015370
    015370 013746 002320
    015374 012746 015503
    015400 012746 000002
    015404 010600

```

```

      .EVEN
      .SBTTL PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER
;
;
; ROUTINE TO PRINT ERROR BYTES IN MESSAGE BUFFERS
; ONLY THE FIRST 8 ERRORS ENCOUNTERED ARE PRINTED DUE TO SCREEN SPACE
;
; RO - NUMBER OF BYTES IN BUFFER
;
; IMPLICIT INPUTS:
;
; EXPMSG - EXPECTED MESSAGE BUFFER
; RECMMSG - RECEIVED MESSAGE BUFFER
;
PRBYTEXP::
      SAVREG
      MOV R0,R5 ;SAVE R1-R5 UNTIL NEXT RETURN
      CLR PRMNO ;SAVE NUMBER OF BYTES
      CLR R4 ;INIT ERROR COUNT
      MOV #EXPMSG,R1 ;NUMBER OF THE CURRENT BYTE
      MOV #RECMMSG,R2 ;GET EXPD BUFFER ADDRESS
20$: MOVB (R1),R0 ;GET RECV BUFFER ADDRESS
      BIC #C<377>,R0 ;GET EXPD BYTE
      MOVB R0,PRBEXP ;CLEAR UPPER BYTE
      MOVB (R2),R3 ;SAVE FOR ERROR REPORT
      BIC #C<377>,R3 ;GET RECV BYTE
      MOVB R3,PRBREC ;CLEAR UPPER BYTE
      XOR R0,R3 ;FOR ERROR REPORT
      CMPB (R1)+,(R2)+ ;XOR EXPD/RECV
      BEQ 30$ ;EXPD = RECV?
      INC PRMNO ;BR IF YES
      CMP PRMNO,#8 ;UPDATE ERROR COUNT
      BHI 30$ ;PRINTED 8?
27$: PRINTX #PRBMSG,R4,PRBEXP,PRBREC,R3 ;BR IF YES
      MOV R3,-(SP)
      MOV PRBREC,-(SP)
      MOV PRBEXP,-(SP)
      MOV R4,-(SP)
      MOV #PRBMSG,-(SP)
      MOV #5,-(SP)
      MOV SP,R0
      TRAP C#PNTX
      ADD #14,SP
      FORCEEXIT 50$ ;a00
      BR 35$ ;a00
30$: FORCERROR 27$,NOTSSR ;a00
35$: INC R4 ;a00
      CMP R4,R5 ;NUMBER OF THE NEXT
      BGE 50$ ;DONE ALL YET?
      BR 20$ ;BR IF YES
      BR 20$ ;DO ANOTHER
50$: PRINTX #PRBTOT,PRMNO ;PRINT TOTAL ERROR COUNT
      MOV PRMNO,-(SP)
      MOV #PRBTOT,-(SP)
      MOV #2,-(SP)
      MOV SP,R0

```

PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER

```

015406 104415
015410 062706 000006
1796 015414 000207 TRAP C#PNTX
1797 ADD #6,SP
1798 015416 045 116 045 PRBMSG: .ASCIZ 'N#A BYTE #D2#A EXPD: #03#A RECV: #03#A XOR: #03'
1799 015503 045 116 045 PRBTOT: .ASCIZ 'N#A NUMBER OF BYTES IN ERROR = #D2'
1800 .EVEN
1801 015550 000000 PRBEXP: .WORD 0 ;EXPD
1802 015552 000000 PRBREC: .WORD 0 ;RECV
1803 .SBTTL EXPREC - PRINT EXPD/RECV WORD DATA
1804 ;*
1805 ;PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
1806 ;INPUTS:
1807 ;
1808 ; R1 RECEIVED DATA
1809 ; R2 EXPECTED DATA
1810 ;
1811 ;-
1812
1813
1814
1815 015554 BGNMSG EXPREC
015554 EXPREC:: JSR PC,PRIXOR ;PRINT THE DATA
1816 015554 004737 007742 ENDMSG
1817 015560 L10017: TRAP C#MSG
015560 104423 .SBTTL EXPBREC - PRINT EXPD/RECV BYTE DATA
1818 ;*
1819 ;PRINT ROUTINE TO DISPLAY BYTE EXPD/RECV DATA
1820 ;INPUTS:
1821 ;
1822 ; R1 RECEIVED DATA BYTE
1823 ; R2 EXPECTED DATA BYTE
1824 ;
1825 ;-
1826
1827
1828
1829
1830
1831 015562 BGNMSG EXPBREC
015562 EXPBREC:: JSR PC,PRIBXOR ;PRINT THE DATA
1832 015562 004737 007612 ENDMSG
1833 015566 L10020: TRAP C#MSG
015566 104423 .SBTTL RAMERR - PRINT RAM AND PACKET DATA
1834 ;*
1835 ;PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
1836 ;INPUTS:
1837 ;
1838 ; R4 POINTER TO COMMAND PACKET
1839 ;
1840 ;
1841 ;
1842 ;
1843 ;
1844 ;

```

J6

RAMERR - PRINT RAM AND PACKET DATA

```

1845 ;IMPLICIT INPUTS:
1846 ;
1847 ;     RAMDATA     DATA AS READ FROM THE RAM
1848 ;     RAMSIZ     NUMBER OF BYTES IN PACKET
1849 ;                 IF RAMSIZ=0 THEN DEFAULT TO 8.
1850 ;
1851 ;IMPLICIT OUTPUTS:
1852 ;
1853 ;     RAMSIZ     SET TO 0
1854 ;
1855 ;-
1856 015570 BGNMSG RAMERR
1857 015570 004737 014046 RAMERR:: JSR PC,PRAMPKT ;PRINT RAM/PACKET DATA
1858 015574 ENDMSG
1859 015574 104423 L10021: TRAP C#MSG
1860 ;
1861 ;     .SBTTL RAMTADD - PRINT TEST ADDRESS, RAM AND PACKET DATA
1862 ;*
1863 ;PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
1864 ;
1865 ;INPUTS:
1866 ;
1867 ;     R4     POINTER TO COMMAND PACKET
1868 ;
1869 ;IMPLICIT INPUTS:
1870 ;
1871 ;     RAMDATA     DATA AS READ FROM THE RAM
1872 ;     RAMSIZ     NUMBER OF BYTES IN PACKET
1873 ;                 IF RAMSIZ=0 THEN DEFAULT TO 8.
1874 ;     ERRHI     HIGH ORDER TEST ADDRESS
1875 ;     ERRLO     LOW ORDER TEST ADDRESS
1876 ;
1877 ;IMPLICIT OUTPUTS:
1878 ;
1879 ;     RAMSIZ     SET TO 0
1880 ;
1881 ;-
1882 015576 BGNMSG RAMTADD
1883 015576 004737 010274 RAMTADD:: JSR PC,PRITADD ;PRINT TEST ADDRESS
1884 015602 004737 014046 JSR PC,PRAMPKT ;PRINT RAM/PACKET DATA
1885 015606 ENDMSG
1886 015606 104423 L10022: TRAP C#MSG
1887 ;
1888 ;     .SBTTL RAMEXP - PRINT RAM EXPD/RECV DATA
1889 ;*
1890 ;PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
1891 ;
1892 ;INPUTS:
1893 ;
1894 ;     R1     RECEIVED DATA
1895 ;     R2     EXPECTED DATA

```

K6

RAMEXP - PRINT RAM EXPD/RECV DATA

```

1896          ;          R4          CONTROLLER RAM ADDRESS
1897          ; -
1898
1899 015610    BGNMSG  RAMEXP
          015610
1900 015610    042701  177400  RAMEXP::
1901 015614    042702  177400    BIC      #1C<377>,R1          ;SAVE EXPD RAM DATA BYTE
1902 015620    004737  010066    BIC      #1C<377>,R2          ;SAVE EXPD RAM DATA BYTE
1903 015624    004737  007742    JSR      PC,PRIRAM          ;PRINT THE RAM ADDRESS
1904 015630    JSR      PC,PRIXOR          ;PRINT THE DATA
          015630
          015630  104423
1905
1906          ;          R4          CONTROLLER RAM ADDRESS
1907          ; -
1908
1909          BGNMSG  TIMEXP
1910          .SBTTL  TIMEXP - PRINT TIMER A,B AND EXP/REC
1911          ; *
1912          ; PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
1913          ; AND TIMER A,B HEADER MESSAGE
1914          ;
1915          ; INPUTS:
1916          ;
1917          ;          R1          RECEIVED DATA
1918          ;          R2          EXPECTED DATA
1919          ; -
1920          BGNMSG  TIMEXP
          015632
          015632
1919 015632    PRINTX  #TIMSGO          ;PRINT HEADER
          015632    012746  015660    MOV      #TIMSGO,-(SP)
          015636    012746  000001    MOV      #1,-(SP)
          015642    010600    MOV      SP,R0
          015644    104415    TRAP    C#PNTX
1920 015646    062706  000004    ADD      #4,SP
1921 015652    004737  007742    JSR      PC,PRIXOR          ;PRINT THE DATA
          015656
          015656  104423
1922
1923 015660    045      116      045  TIMSGO: .ASCIZ  '##N#A TIMER A STATUS IS IN BIT 3##N#A TIMER B STATUS IS IN BIT 2'
1924          .EVEN
1925          .SBTTL  BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS
1926
1927          ; *
1928          ; PRINT ROUTINE FOR TSSR ERRORS ON DATA TRANSFERS
1929          ;
1930          ; INPUTS:
1931          ;
1932          ;          R1          CONTENTS OF TSSR
1933          ;          R2          DATA WRITTEN (8 BITS)
1934          ; -
1935          BGNMSG  BADSSR
1936          015760
          015760
1938 015760    BADSSR:
          015760    MOV      R2,-(SP)          ;SAVE DATA TRANSFERRED
1939 015760    010246  177400    BIC      #177400,R2          ;GET JUST ONE BYTE
1940 015762    042702  177400

```

L6

BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS

1941	015766				PRINTB	@XFERASC,R2		
	015766	010246			MOV	R2,-(SP)		
	015770	012746	016020		MOV	@XFERASC,-(SP)		
	015774	012746	000002		MOV	@2,-(SP)		
	016000	010600			MOV	SP,R0		
	016002	104414			TRAP	C#PNTB		
	016004	062706	000006		ADD	#6,SP		
1942	016010	012602			MOV	(SP)+,R2		;RESTORE R2
1943	016012	004737	006024		JSR	PC,PRITSSR		;DECODE TSSR CONTENTS
1944	016016				ENDMSG			
	016016			L10025:				
	016016	104423			TRAP	C#MSG		
1945	016020	045	116	045	XFERASC:	.ASCIZ	'#N#A Data Transferred = #03'	

GLOBAL SUBROUTINES SECTION

1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994

016054
016054
016060 012765 000000 000002
016066 004737 016330
016072 016500 000002
016076 010004
016100 042704 176277
016104 052704 002200
016110 020400
016112 001402
016114 000241
016116 000401
016120 000261
016122 000207

```
.SBTTL GLOBAL SUBROUTINES SECTION
; **
; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
; THAT ARE USED IN MORE THAN ONE TEST.
; --
.SBTTL SOFINIT - SOFT INITIALIZE OF CONTROLLER
; *
; ROUTINE TO DO A SOFT INITIALIZE OF THE CONTROLLER
; BY WRITING INTO THE TSSR REGISTER. AFTER THE INIT,
; THE TSSR REGISTER IS TESTED FOR ERRORS. ANY ERRORS
; DETECTED SHOULD BE TREATED AS DEVICE FATAL ERRORS.
; INPUTS:
; R5 ADDRESS OF FIRST REGISTER
; OUTPUTS:
; R0 CONTENTS OF TSSR, IF ERROR
; CARRY SET IF INIT WAS OKAY
; CLEAR IF FATAL ERROR
; CALLING SEQUENCE:
; MOV #ADDRESS,R5
; JSR PC,SOFINIT
; BCS CONTINUE
; ERRDF ;REPORT FATAL ERROR
; -
SOFINIT::
SAVREG ; SAVE THE REGISTERS
MOV #0,TSSR(R5) ; DO THE INIT.
JSR PC,WAITF ; WAIT FOR SSR
MOV TSSR(R5),R0 ; GET THE TSSR REGISTER
MOV R0,R4 ; TSSR CONTENTS
BIC #1<HIADDR!OFL>,R4 ; R4 HAS EXPECTED CONTENTS
BIS #SSR!NBA,R4 ; ONLY EXPECTED BITS SET ?
CMP R4,R0 ; BRANCH IF OKAY
BEQ 54 ; CLEAR THE CARRY FOR ERROR
CLC ; GO TO EXIT
BR 104 ; SET THE CARRY BIT
54: SEC ; RETURN TO CALLER
104: RTS PC
```

CHKAMB - CHECK TSSR FOR AMBIGUITY

```

1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016 016124
2017 016124
2018 016130 010004
2019 016132 032700 100000
2020 016136 001004
2021 016140 032700 174077
2022 016144 001023
2023 016146 000424
2024 016150 032700 000200
2025 016154 001011
2026 016156 032700 000040
2027 016162 001414
2028 016164 042704 177761
2029 016170 020427 000016
2030 016174 001007
2031 016176 000410
2032 016200 032700 000040
2033 016204 001405
2034 016206 032700 000006
2035 016212 001002
2036 016214 000241
2037 016216 000401
2038 016220 000261
2039 016222 000207

```

```

.SBTTL  CHKAMB - CHECK TSSR FOR AMBIGUITY
;*
;THIS ROUTINE TESTS THE CONTENTS OF THE TSSR REGISTER
;FOR AMBIGUITY
;INPUT:
;      RO      CONTENTS OF TSSR
;OUTPUT:
;      RO      CONTENTS OF TSSR
;      CARRY   SET - NO AMBIGUITY
;             CLR - AMBIGUOUS CONTENTS
;-
CHKAMB:
  SAVREG          ;SAVE THE GENERAL REGISTERS
  MOV            RO,R4      ;CONTENTS OF TSSR
  BIT            #SC,RO     ;IS BIT 15 SET ?
  BNE            54        ;BRANCH IF YES
  BIT            #1C<NBA!OFL!SSR!HIADDR>,RO ;ANY OTHER BITS SET ?
  BNE            40        ;MUST BE AN ERROR
  BR             45        ;RETURN WITH SUCCESS
54:  BIT            #SSR,RO   ;IS READY BIT SET ?
  BNE            104       ;BRANCH IF READY BIT IS SET.
  BIT            #BITS,RO  ;IS FATAL ERROR BIT SET ?
  BEQ            40        ;ERROR IF NOT
  BIC            #1CTERCLS,R4 ;CLEAR ALL BUT TERMINATION CODE
  CMP            R4,#16    ;ALL THREE BITS MUST BE SET
  BNE            40        ;ERROR IF NOT SET
  BR             45        ;OK IF ALL ARE SET
104: BIT            #BITS,RO ;IS FATAL ERROR BIT SET ?
  BEQ            45        ;ERROR IF BIT IS SET WITH SSR
  BIT            #BIT2!BIT1,RO ;IS THIS A FUNCTION REJECT
  BNE            45        ;BR, IF TSSR IS OK
40:  CLC              ;AMBIGUOUS CONTENTS
  BR             50
45:  SEC              ;SHOW SUCCESS - NO AMBIGUITY
50:  RTS              ;RETURN TO CALLER
PC

```

ENAIN,DSBINT - ENABLE/DISABLE INTERRUPTS

```

2041      .SBTTL ENAIN,DSBINT - ENABLE/DISABLE INTERRUPTS
2042      ;
2043      ; DEFAULT DISPLAY INTERRUPT HANDLERS.
2044      ; IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
2045      ; OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
2046      ;
2047      ;
2048      ; BIT DEFINITIONS FOR "INTMASK" AND "INTFLAG" BYTES:
2049      ;
2050      000200      IOKCKIN=BIT7      ; DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
2051      000001      IOKSTP=BIT0      ; EXPECT "STOP" INTERRUPT.
2052      ;
2053      ; INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
2054      016224      000      INTMASK:      .BYTE      0
2055      ; INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
2056      016225      000      INTFLAG:      .BYTE      0
2057      ;
2058      ; SAVED INTERRUPT VECTOR:
2059      016226      000000      INTVEC:      .WORD      0
2060      ; SAVE CPU PC
2061      016230      000000      INTCPC:      .WORD      0
2062      ;
2063      ; SUBROUTINE TO ENABLE INTERRUPTS:
2064      016232      010046      ENAIN:      MOV      RO,-(SP)      ;SAVE RO
2065      016234      013700      002206      MOV      IVEC,RO      ;GET POINTER TO VECTORS
2066      016240      012720      016276      MOV      @INTR,(RO)+      ;SET UP INTERRUPT VECTOR
2067      016244      012720      000340      MOV      @PRI07,(RO)+
2068      016250      012600      MOV      (SP)+,RO      ;RESTORE RO
2069      016252      011646      MOV      (SP)-,(SP)
2070      016254      012766      000000      000002      MOV      @0,2(SP)      ;SET CPU TO LEVEL 0
2071      016262      000002      RTI
2072      ;
2073      ; SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 7)
2074      016264      011646      DSBINT:      MOV      (SP)-,(SP)
2075      016266      012766      000340      000002      MOV      @PRI07,2(SP)
2076      016274      000002      RTI
2077      .SBTTL INTR - INTERRUPT HANDLERS
2078      ;
2079      016276      016276      BGNSRV INTR      ;DEFINE INTERRUPT ENTRY
2080      016276      012737      000001      002222      INTR::      MOV      #1,INTRECV      ;SET FLAG TO SHOW INTERRUPT RECEIVED
2081      016304      105037      016225      CLR      INTFLAG      ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2082      016310      132737      000001      016224      BIT      @IOKSTP,INTMASK      ;EXPECTING STOP INTERRUPT?
2083      016316      001003      BNE      1$      ;BR IF YES
2084      016320      152737      000001      016225      BIS      @IOKSTP,INTFLAG      ;NO. SET THE ERROR FLAG.
2085      ;
2086      ; SAVE REGISTERS, MSG BUFFER, ETC.
2087      016326      1$:
2088      016326      016326      ENDSRV
2089      016326      000002      L10026:      RTI

```


WAITF - WAIT FOR SUBSYSTEM READY

```

2090          .SBTTL WAITF - WAIT FOR SUBSYSTEM READY
2091          ;
2092          ; SUBROUTINE TO WAIT FOR THE SUBSYSTEM READY FLAG
2093          ;
2094          ; INPUTS:
2095          ;
2096          ; R5      ADDRESS OF FIRST DEVICE REGISTER
2097          ;
2098          ; OUTPUTS:
2099          ;
2100          ; R0      CONTENTS OF LAST TSSR READ
2101          ; CARRY   SET - READY BIT SET
2102          ;          CLR - TIMEOUT WAITING FOR READY
2103          ;
2104          ; WAITF:: BR      1#          ;NOP WHEN SUPER FIXED
2105          ;          BREAK    1#          ; DO A SUPVSR BREAK FIRST.
2106          ;          TRAP     C#BRK
2107          ;          MOV      #11000,-(SP) ;25-APRIL-83 REV B - 1100 MSEC TIMER
2108          ;          MOV      TSSR(R5),R0 ;READ THE TSSR REGISTER
2109          ;          TSTB     R0          ;TEST FOR READY BIT SET
2110          ;          BMI      3#          ; EXIT ON STOP FLAG.
2111          ;          DELAY    1          ; WAIT 100 USEC
2112          ;          MOV      #1,(PC)+
2113          ;          .WORD    0
2114          ;          MOV      L#DLY,(PC)+
2115          ;          .WORD    0
2116          ;          DEC      -6(PC)
2117          ;          BNE     --4
2118          ;          DEC      -22(PC)
2119          ;          BNE     -20
2120          ;          DEC      (SP)        ;REDUCE DELAY COUNT
2121          ;          BNE     2#          ;RETRY UNTIL TIMER EXPIRES
2122          ;          CLC
2123          ;          BR      4#          ; C = 0, CONTROLLER STILL RUNNING...
2124          ;          SEC
2125          ;          DEC      (SP)+      ;...OR HUNG-UP AFTER 300 MSEC.
2126          ;          PC
2127          ;          RTS

```

CHKTSSR - CHECK TSSR FOR READY

```

2120      .SBTTL  CHKTSSR - CHECK TSSR FOR READY
2121      ;*
2122      ;THIS ROUTINE WAITS FOR READY IN THE TSSR
2123      ;AND TESTS FOR AMBIGUOUS BIT SETTINGS IN TSSR.
2124      ;
2125      ;INPUT:
2126      ;      RS      ADDRESS OF CSR REGISTERS
2127      ;
2128      ;OUTPUT:
2129      ;      RO      CONTENTS OF TSSR
2130      ;      CARRY   SET - OKAY
2131      ;              CLR - NOT READY AMBIGUOUS, OR SC SET
2132      ;
2133      ;-
2133      CHKTSSR:
2134      JSR      PC, WAITF      ;WAIT FOR READY
2135      BCC      20$           ;BRANCH IF TIME OUT
2136      JSR      PC, CHKAMB    ;TSSR AMBIGUOUS?
2137      BCC      10$           ;BR IF YES
2138      BIT      @SC, RO      ;SPECIAL CONDITION SET?
2139      BEQ      15$           ;BR IF NO
2140      BIT      @<SC!BIE!RMR!NXM>, RO ;ANY ERROR BITS SET?
2141      BEQ      15$           ;BR IF NO
2142      10$:    CLC              ;SET FAILURE
2143      BR       20$           ;
2144      15$:    SEC              ;SET SUCCESS
2145      20$:    RTS      PC      ;RETURN TO CALLER
2146      .SBTTL  XNXM      - CHECK FOR NONEXISTENT MEMORY
2147      ;*
2148      ; ROUTINE TO TEST FOR A NEXM IN THE RANGE (R1) THRU (R2).
2149      ; ON RETURN, IF "C" = 1, (R1) = NEXM ADDRESS.
2150      ; "C" = 0, ALL ADDRESSES OK.
2151      ;
2152      ;CALL:  MOV  ADR1, R1
2153      ;        MOV  ADR2, R2
2154      ;        JSR  PC, NXM
2155      ;        RETURN
2156      XNXM:  MOV      @2$, @#4 ;TEST "C" AND PROCEED.
2157      MOV      @PRI04, @#6 ; SET BUSERR VECTOR.
2158      CLR      R3
2159      1$:    TST      (R1) ;FLAG.
2160      ;TEST THE ADDRESS(ES).
2161      CMP      R1, R2 ;IF ANY TRAP, CONTINUE AT 2$.
2162      BEQ      3$ ;OTHERWISE, CONTINUE HERE.
2163      ADD      @2, R1 ;BR IF FINISHED (NO NEXM'S).
2164      BR       1$ ;SET NEXT ADDRESS...
2165      2$:    COM      R3 ;... AND CONTINUE.
2166      MOV      @3$, (SP) ;GOT ONE, SET FLAG...
2167      RTI
2168      3$:    CLRVEC @4 ;... AND DISMISS INTERRUPT...
2169      MOV      @4, RO ;... AND GIVE BACK THE VECTOR.
2170      TRAP   C!CVEC
2171      TST      R3
2172      BEQ      .+4 ;DID WE CATCH ONE ??
2173      SEC
2174      RTS      PC ;NO, "C" = 0, SKIP NEXT.
2175      ;YES, "C" = 1, (R1) = NEXM ADDR.

```

E7

TSTLOOP - CHECK ITERATION COUNT

2174
2175
2176
2177
2178
2179
2180
2181
2182 016536
2183 016536 005737 002166
2184 016542 001006
2185 016544 005737 002202
2186 016550 100403
2187 016552 005337 002214
2188 016556 001002
2189 016560 000241
2190 016562 000401
2191 016564 000261
2192 016566 000207
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220 016570
2221 016570 010046
2222 016572 005037 003154
2223 016576 005037 017036
2224 016602 005037 005772
2225 016606 105037 016224
2226 016612 013700 002200
2227 016616 006300
2228 016620 005737 003114
2229 016624 001430
2230 016626 100010

```

.SBTTL TSTLOOP - CHECK ITERATION COUNT
;*
; SUBROUTINE TO EXECUTE TEST ITERATIONS.
; EXIT WITH "C" SET IF LOOPS ALLOWED AND LOOP COUNT NON-ZERO.
; LOOP COUNTER IS SET BY "BEGIN.TEST" MACRO.
;
; CALL: LOOPTO ARG
;
TSTLOOP::
    TST     NOITS           ; ITERATIONS INHIBITED?
    BNE     1$             ; YES.
    TST     QVP            ; NO.
    BMI     1$             ; LOOPS DISALLOWED IN QUICK PASS.
    DEC     LOOPCNT        ; BUMP LOOP COUNTER.
    BNE     2$
1$:      CLC              ; LOOP DISALLOWED, OR DONE.
    BR     3$
2$:      SEC              ; LOOP ENABLED.
3$:      RTS     PC

.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.
    CLR     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

```

TSTSETUP - PRINT TEST NAME AND INIT ERROR COUNTS

```

2231 016630 052760 160000 003176      BIS      #160000,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
2232 016636      ERRDF    1,NXR,NXRERR ; NO DEVICE HERE -- PRINT IT
      016636 104455      TRAP    C#ERDF
      016640 000001      .WORD   1
      016642 003740      .WORD   NXR
      016644 005736      .WORD   NXRERR
2233 016646 000407      BR      2#
2234 016650 052760 160001 003176 3# :  BIS      #160001,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
2235 016656      ERRDF    2,NOINIT ; DEVICE NOT IDLE
      016656 104455      TRAP    C#ERDF
      016660 000002      .WORD   2
      016662 004335      .WORD   NOINIT
      016664 000000      .WORD   0
2236 016666 012737 177777 003112 2# :  MOV      #-1,DUFLG ; DROP THE UNIT
2237 016674      DODU     UNITN
      016674 013700 002200      MOV      UNITN,R0
      016700 104451      TRAP    C#DODU
2238 016702      DOCLN   ; ABORT THE PASS
      016702 104444      TRAP    C#DCLN
2239 016704 000423      BR      5#
2240
2241 016706      RFLAGS   R0 ; GET THE OPERATOR FLAGS.
      016706 104421      TRAP    C#RFLA
2242 016710 032700 001000      BIT      #PNT,R0 ; PRINT THE TEST NUMBERS?
2243 016714 001412      BEQ     1# ; BR IF NO
2244 016716 011600      MOV      (SP),R0 ; GET THE ID MESSAGE
2245 016720      PRINTF   #TNAM,R0 ; DISPLAY THE TEST ID
      016720 010046      MOV      R0,-(SP)
      016722 012746 016764      MOV      #TNAM,-(SP)
      016726 012746 000002      MOV      #2,-(SP)
      016732 010600      MOV      SP,R0
      016734 104417      TRAP    C#PNTF
      016736 062706 000006      ADD      #6,SP
2246 016742 005237 002212      1# :  INC      TSTCNT ; BUMP TEST COUNTER.
2247 016746      SETPRI   IPRI ; PRIORITY THAT OF DEVICE
      016746 013700 002210      MOV      IPRI,R0
      016752 104441      TRAP    C#SPRI
2248 016754 005726      5# :  TST      (SP) ; FIX UP THE STACK
2249 016756 013705 002204      MOV      CSRADDR,R5 ; ADDRESS OF TSV REGISTERS ON UNIBUS
2250 016762 000207      RTS     PC
2251 016764      045     123     045  TNAM:  .ASCIZ  '#S#T#A Test'
2252
2253      .EVEN
2254      .SBTTL  TSTEND - PRINT ERRORS RECEIVED
2255      ; AT END OF EACH TEST, PRINT THE NUMBER OF ERRORS RECEIVED
2256      ; IF NORMAL ERROR REPORTING IS DISABLED (FLA:IER).
2257
2258 017000      TSTEND: RFLAGS   R0
      017000 104421      TRAP    C#RFLA
2259 017002 030027 020000      BIT      R0,#IER
2260 017006 001412      BEQ     1# ; BR IF "IER" NOT SET.
2261 017010      PRINTF   #ESUM,ERRK ; PRINT ERROR COUNT.
      017010 013746 017036      MOV      ERRK,-(SP)
      017014 012746 017040      MOV      #ESUM,-(SP)
      017020 012746 000002      MOV      #2,-(SP)
      017024 010600      MOV      SP,R0
      017026 104417      TRAP    C#PNTF

```


SETMAP - SETUP PAR6 MAPPING

```

2347          .SBTTL  SETMAP  -  SETUP  PAR6  MAPPING
2348
2349          ;*
2350          ;THIS ROUTINE SETS UP KERNEL PAR6 TP HANDLE
2351          ;AN 18 BIT ADDRESS. THE OFFSET INTO THE PAGE
2352          ;IS RETURNED BIASED TO PAR6.
2353          ;
2354          ;INPUTS:
2355          ;
2356          ;      RO      HIGH ORDER ADDRESS BITS
2357          ;      R1      LOW ORDER ADDRESS BITS
2358          ;
2359          ;OUTPUTS:
2360          ;
2361          ;      RO      OFFSET INTO BLOCK WITH PAR6 BIAS (I.E. THE ADDRESS)
2362          ;      CARRY   SET IF SUCCESS
2363          ;              CLR IF ERROR
2364          ;
2365          ;-
2366          SETMAP:
2367          SAVREG          ;SAVE R1-R4 UNTIL NEXT RETURN
2368          TST            KTF LG          ;SYSTEM HAVE ABOVE 28K?
2369          BEQ            10$           ;BR IF NO
2370          MOV            R1,R2         ;SAVE LOW ORDER BITS
2371          .REPT          6
2372          ASR            RO
2373          ROR            R1
2374          .ENDR
2375          BIC            @177,R1       ;ALINE FOR LOWER 4K BOUNDARY
2376          CMP            R1,KTF LG     ;HIGHER THAN EXISTING MEMORY?
2377          BHIS          10$           ;BR IF YES
2378          MOV            R1,@KIPAR6   ;SETUP MAPPING REGISTER PAR6
2379          BIC            @160000,R2    ;SETUP DISPLACEMENT IN PAGE
2380          ADD            @140000,R2    ;ADD IN PAR6 BIAS
2381          MOV            R2,RO        ;RETURN IN RO
2382          SEC
2383          BR             15$           ;SET SUCCESS
2384          CLC            10$           ;SET FAILURE
2385          RTS            PC            ;RETURN
2386          .SBTTL  FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN
2387
2388          ;*
2389          ; FILL MEMORY WITH A BACKGROUND PATTERN
2390          ;
2391          ; INPUTS:
2392          ;
2393          ;      RO = BACKGROUND PATTERN
2394          ;      FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
2395          ;      KTF LG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
2396          ;
2397          ; OUTPUTS:
2398          ;
2399          ;      NONE
2400          ;-
2401          FILLMEM:
2402          SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
2403          JSR            PC,KTOFF      ;DISABLE KT.

```


K7

FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN

```

2404 017512 010003          MOV      R0,R3          ;COPY TEST PATTERN
2405 017514 013701 003124  MOV      FREE,R1        ;GET FIRST FREE LOCATION
2406 017520 013702 003126  MOV      FRESIZ,R2      ;SIZE OF FREE SPACE BELOW 28K.
2407 017524 010321          10$:  MOV      R3,(R1)+       ;STORE A BACKGROUND WORD
2408 017526 005302          DEC      R2             ;DONE ALL MEMORY IN FREE SPACE?
2409 017530 003375          BGT     10$            ;BR IF NO
2410 017532 005737 003132  TST     KTFLG          ; GOT KT?
2411 017536 001477          BEQ     55$            ; NO. GET OUT.
2412 017540 004737 017336  JSR     PC,KTON        ; YES. ENABLE KT.
2413 017544 005000          CLR     R0             ;HIGH ORDER ADDRESS START
2414 017546 013701 003152  MOV     PST32W,R1      ;GET >28K START ADDRESS (IN 32W BLOCKS)
2415          000006          .REPT   6
2416          CLC
2417          ROL     R1          ;CLEAR C BIT
2418          ROL     R0          ;CONVERT BLOCKS TO WORDS
2419          .ENDR          ;MAKE IT DOUBLE PRECISION
2420 017616 004737 017376  JSR     PC,SETMAP      ;SETUP PAR6 MAPPING REGISTER
2421 017622 010320          30$:  MOV     R3,(R0)+       ;STORE TEST PATTERN IN >28K ADDRESS
2422 017624 020027 160000  CMP     R0,#160000     ;END OF PAR6 MAPPING AREA?
2423 017630 103774          BLO    30$            ;BR IF NO
2424 017632 162700 020000  SUB     #20000,R0      ;BACKUP INTO PAR6 MAPPING BEGIN
2425 017636 062737 000200 172354  ADD     #200,@#KIPAR6  ;POINT TO NEXT 4K BLOCK >28K.
2426 017644 023737 172354 003132  CMP     @#KIPAR6,KTFLG ;END OF MEMORY?
2427 017652 001427          BEQ     50$            ;BR IF YES
2428 017654 005737 003144  TST     T23A          ;11/23A?
2429 017660 001407          BEQ     35$            ;NO KEEP GOING
2430 017662 013704 177572  MOV     SRO,R4         ;GET SRO CONTENTS
2431 017666 042704 177761  BIC     #177761,R4     ;CLEAR ALL BUT PAGE NUMBER
2432 017672 022704 000016  CMP     #16,R4        ;SEE IF PAGE 7
2433 017676 001415          BEQ     50$            ;EXIT IF THERE
2434 017700 005737 003146          35$:  TST     T23B          ;11/23B?
2435 017704 001410          BEQ     45$            ;NO KEEP GOING
2436 017706 023727 172354 007600  CMP     @#KIPAR6,#7600 ;REACHED 18 BITS?
2437 017714 103001          BHIS   40$            ;YES
2438 017716 000403          BR     45$            ;NO KEEP GOING
2439 017720 012737 000020 172516 40$:  MOV     #20,SR3       ;SET 22 BIT RELOCATION
2440 017726 000137 017622          45$:  JMP     30$            ;KEEP GOING ON ETC.
2441 017732 004737 017354          50$:  JSR     PC,KTOFF      ; DISABLE KT.
2442 017736 000207          55$:  RTS     PC

```


M7

CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN

```

2501 020120 010137 002240      MOV      R1,ERRLO      ;SAVE LOW ORDER IN ERROR
2502 020124 010337 002232      MOV      R3,EXPD      ;SAVE EXPD FOR ERROR REPORT
2503 020130 011437 002234      MOV      (R4),RECV    ;SAVE RECV FOR ERROR REPORT
2504 020134 000421              BR       50$          ;
2505 020136 062701 000002      32$:    ADD      #2,R1    ;UPDATE NON PAR6 ADDRESS
2506 020142 005500              ADC      R0           ;MAKE IT DOUBLE PRECISION ADD
2507 020144 062704 000002      ADD      #2,R4        ;UPDATE PAR FORMAT ADDRESS
2508 020150 020427 160000      CMP      R4,#160000   ;END OF PAR6 MAPPING AREA?
2509 020154 103755              BLO     30$          ;BR IF NO
2510 020156 162704 020000      SUB      #20000,R4    ;BACKUP INTO PAR6 MAPPING BEGIN
2511 020162 062737 000200 172354  ADD      #200,@#KIPAR6 ;POINT TO NEXT 4K BLOCK >28K.
2512 020170 023737 172354 003132  CMP      @#KIPAR6,KTFLG ;END OF MEMORY?
2513 020176 101744              BLOS    30$          ;BR IF NO
2514 020200 004737 017354      50$:    JSR      PC,KTOFF   ;TURN OFF MEMORY MAPPING
2515 020204 000241              CLC                    ;SET FAILURE
2516 020206 000403              BR       60$          ;
2517 020210 004737 017354      55$:    JSR      PC,KTOFF   ;TURN OFF MEMORY MAPPING
2518 020214 000261              SEC                    ;SET SUCCESS
2519 020216 000207      60$:    RTS      PC      ;
2520              .SBTTL  REGSAV - SAVE R1-R5 ON STACK
2521              ;*
2522              ;
2523              ;ROUTINE TO
2524              ;SAVE R1 THROUGH R5 ON THE STACK
2525              ;
2526              ;CALLING SEQUENCE:
2527              ;
2528              ;      JSR      R5,REGSAV
2529              ;
2530              ;THIS IS A COOROUTINE WHICH TRANSFER CONTROL BACK TO
2531              ;THE CALLING ROUTINE. AT THE END OF THE CALLING ROUTINE,
2532              ;THE RTS PC RETURNS CONTROL TO THIS ROUTINE TO RESTORE
2533              ;REGISTERS.
2534              ;
2535              ;THIS ROUTINE SHOULD ONLY BE CALLED FROM ROUTINES WHICH ARE
2536              ;CALLED VIA A JSR PC INSTRUCTION
2537              ;
2538              ;-
2539              ;
2540              REGSAV:
2541 020220 010446      MOV      R4,-(SP)
2542 020222 010346      MOV      R3,-(SP)
2543 020224 010246      MOV      R2,-(SP)
2544 020226 010146      MOV      R1,-(SP)
2545 020230 010546      MOV      R5,-(SP)
2546 020232 016605 000012      MOV      10.(SP),R5
2547 020236 004736      JSR      PC,@(SP)+
2548 020240 012601      MOV      (SP)+,R1
2549 020242 012602      MOV      (SP)+,R2
2550 020244 012603      MOV      (SP)+,R3
2551 020246 012604      MOV      (SP)+,R4
2552 020250 012605      MOV      (SP)+,R5
2553 020252 000207      RTS      PC

```

GETPAT - GET 8 BIT PATTERN FROM OPERATOR

```

2555          .SBTTL  GETPAT - GET 8 BIT PATTERN FROM OPERATOR
2556          ;*
2557          ;ROUTINE TO REQUEST AN 8 BIT DATA PATTERN FROM THE OPERATOR
2558          ;
2559          ;INPUTS:      NONE.
2560          ;
2561          ;OUTPUTS:
2562          ;          RO      OCTAL NUMBER FROM THE OPERATOR
2563          ;
2564          ;CALLING SEQUENCE:
2565          ;          JSR      PC,GETPAT
2566          ;-
2567          GETPAT::
2568          SAVREG          ;SAVE THE GENERAL REGISTERS
2569          1$:      GMANID  DATASC,PATDAT,0,377,0,377,NO
                   TRAP     C#GMAN
                   BR       10000#
                   .WORD    PATDAT
                   .WORD    T#CODE
                   .WORD    DATASC
                   .WORD    377
                   .WORD    T#LOLIM
                   .WORD    T#HILIM
2570          10000#:      BNCOMPLETE      1#      ;RETRY IF ERROR
                   BCC      1#
                   MOV     PATDAT,RO      ;DATA PATTERN FROM OPERATOR
                   RTS      PC           ;RETURN TO CALLER
2571          ;
2572          ;*
2573          ;LOCAL DATA AREA
2574          ;-
2575          ;
2576          ;
2577          PATDAT: .WORD    0          ;TEMPORARY STORAGE FOR DATA
2578          DATASC: .ASCIZ  'ENTER DATA PATTERN'
2579          .EVEN
2580

```

GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE

```

2582 .SBTTL GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE
2583 ;
2584 ;ROUTINE TO ISSUE A MENU AND GET THE OPERATOR'S RESPONSE.
2585 ;
2586 ;INPUTS:
2587 ; R0 ADDRESS OF ASCIZ STRING OF MENU
2588 ; R1 MAXIMUM ALLOWABLE OPERATOR RESPONSE
2589 ;
2590 ;OUTPUTS:
2591 ; R0 NUMBER OF THE OPERATOR'S SELECTION
2592 ;
2593 GETSEL::
2594 SAVREG ;SAVE GENERAL REGISTERS
2595 MOV R0,R2 ;SAVE THE MENU ADDRESS
2596 MOV R2,R3 ;START OF MENU STRING
2597 TST (R3) ;END OF ASCII ?
2598 BEQ 3$ ;BRANCH IF ALL LINES DISPLAYED
2599 PRINTF #SELASC,(R3)+ ;DISPLAY THE MENU
      MOV (R3)+,-(SP)
      MOV #SELASC,-(SP)
      MOV #2,-(SP)
      MOV SP,R0
      TRAP C#PNTF
      ADD #6,SP
      BR 2$
2600 2$:
2601 3$: GMANID MENASC,MENRES,D,-1,0,-1,NO
      TRAP C#GMAN
      BR 10001$
      .WORD MENRES
      .WORD T#CODE
      .WORD MENASC
      .WORD -1
      .WORD T#LOLIM
      .WORD T#HILIM
2602 10001$: BNCOMPLETE 1$ ;RETRY IF ERROR
      BCC 1$
      MOV MENRES,R0 ;GET THE OPERATOR'S REPLY
      CMP R0,R1 ;COMPARE TO MAXIMUM ALLOWED
      BLOS 5$ ;BRANCH IF OK
      PRINTF #MENERR ;DISPLAY ERROR MESSAGE
      MOV #MENERR,-(SP)
      MOV #1,-(SP)
      MOV SP,R0
      TRAP C#PNTF
      ADD #4,SP
      BR 1$
2603 1$: ;RETRY
2604 5$: RTS PC ;RETURN TO CALLER
2605 045 MENERR: .ASCIZ '#N#A *** Menu Selection Too Large ***'
2606 045 SELASC: .ASCIZ '#N#T'
2607 164 MENASC: .ASCIZ 'Enter Menu Selection: '
2608 .EVEN
2609 MENRES: .WORD 0
2610
2611
2612
2613

```

```

2593 020336
2594 020336
2595 020342 010002
2596 020344 010203
2597 020346 005713
2598 020350 001412
2599 020352
      020352 012346
      020354 012746 020522
      020360 012746 000002
      020364 010600
      020366 104417
      020370 062706 000006
2600 020374 000764
2601 020376
      020376 104443
      020400 000406
      020402 020556
      020404 000042
      020406 020527
      020410 177777
      020412 000000
      020414 177777
      020416
2602 020416
      020416 103352
2603 020420 013700 020556
2604 020424 020001
2605 020426 101411
2606 020430
      020430 012746 020454
      020434 012746 000001
      020440 010600
      020442 104417
      020444 062706 000004
2607 020450 000735
2608 020452 000207
2609 020454 045 116
2610 020522 045 116
2611 020527 105 156
2612
2613 020556 000000

```

CHKMAN - CHECK MANUAL INTERVENTION LEGALITY

```

2615           .SBTTL  CHKMAN  - CHECK MANUAL INTERVENTION LEGALITY
2616           ;*
2617           ;ROUTINE TO TEST FOR MANUAL INTERVENTION LEGALITY.
2618           ;
2619           ;INPUT:
2620           ;
2621           ;     NONE.
2622           ;
2623           ;OUTPUT:
2624           ;
2625           ;     CARRY  0      MANUAL INTERVENTION NOT ALLOWED
2626           ;           1      MANUAL INTERVENTION IS OK
2627           ;
2628           ;SIDE EFFECTS:
2629           ;
2630           ;     A MESSAGE IS DISPLAYED WARNING THAT TEST IS
2631           ;     NOT EXECUTED IF MANUAL INTERVENTION IS NOT
2632           ;     ALLOWED.
2633           ;
2634           ;
2635           ;-
2636
2637 CHKMAN::
2638           SAVREG           ;SAVE THE REGISTERS
2639           MANUAL         ;SEE IF MANUAL INTERVENTION OK
2640           TRAP          C#MANI
2641           BCOMPLETE 1#   ;BRANCH IF ALLOWED
2642           BCS          1#
2643           PRINTF #NOMAN ;PRINT THE WARNING MESSAGE
2644           MOV          #NOMAN,-(SP)
2645           MOV          #1,-(SP)
2646           MOV          SP,RO
2647           TRAP          C#PNTF
2648           ADD          #4,SP
2649           CLC           ;CLEAR CARRY FOR ERROR
2650           RTS          PC ;RETURN
2651
2652           1#:
2653           NOMAN: .ASCIZ 'N#A *** Manual Intervention not Allowed - Test Aborted ***'
2654           .even

```

ENVIRN - SETUP FREE DIAGNOSTIC SPACE

```

2648                                     .SBTTL ENVIRN - SETUP FREE DIAGNOSTIC SPACE
2649                                     ;
2650                                     ; SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
2651                                     ;
2652 ENVIRN: MEMORY R0
      020710 104431 TRAP C#MEM
2653 020712 010037 003124 MOV R0,FREE ; GET 1ST FREE ADDRESS...
2654 020716 062737 000002 003124 ADD #2,FREE ; ...AND WORD COUNT.
2655 020724 011037 003126 MOV (R0),FRESIZ
2656 020730 162737 000004 003126 SUB #4,FRESIZ
2657 020736 013702 002012 MOV L#UNIT,R2 ; GET NUMBER OF UNITS
2658 020742 162737 000007 003126 10$: SUB #7,FRESIZ ; TAKE AWAY 7 WORDS PER UNIT
2659 020750 005302 DEC R2
2660 020752 001373 BNE 10$
2661 020754 013700 003124 MOV FREE,R0 ; GET FIRST FREE ADDRESS
2662 020760 063700 003126 ADD FRESIZ,R0 ; POINT TO LAST FREE ADDRESS
2663 020764 162700 000002 SUB #2,R0 ; BACKUP 1 WORD
2664 020770 010037 003130 MOV R0,FREEHI ; STORE LAST FREE ADDRESS
2665 020774 000240 NOP ; *****
2666 020776 012701 177520 MOV #BDVPCR,R1 ; GET BDV11 PCR ADDRESS
2667 021002 010102 MOV R1,R2 ; COPY TO R2
2668 021004 062702 000002 ADD #2,R2 ; SET THE RANGE
2669 021010 004737 016456 JSR PC,XNXM ; SEE IF WE HAVE ONE
2670 021014 103001 BCC 15$ ; OK TO SET FLAGS
2671 021016 000423 BR 40$ ; RETURN WITH FLAGS CLEAR
2672 021020 013701 177520 15$: MOV BDVPCR,R1 ; SAVE PCR CONTENTS
2673 021024 062701 000001 ADD #1,R1 ; ADD ONE TO IT
2674 021030 012702 177520 MOV #BDVPCR,R2 ; GET BDV11 PCR ADDRESS
2675 021034 005212 INC (R2) ; TRY TO WRITE TO IT
2676 021036 013703 177520 MOV BDVPCR,R3 ; GET RESULTS
2677 021042 020103 CMP R1,R3 ; DID IT CHANGE?
2678 021044 001006 BNE 20$ ; NO, MUST BE 11/238
2679 021046 005237 003144 INC T23A ; SET THE FLAG
2680 021052 042737 170000 002120 BIC #170000,L#HIME ; SUPERVISOR COULD BE WRONG
2681 ; NOP ; BR 40$ FOR RELEASE
2682 ; PRINTF #M8186 ; TELL THE SYSTEM TYPE
2683 021060 000402 BR 40$ ; RETURN
2684 021062 005237 003146 20$: INC T238 ; SET THE FLAG
2685 ; NOP ; BR 40$ FOR RELEASE
2686 ; PRINTF #M8189 ; TELL THE SYSTEM TYPE
2687 021066 40$: ;
2688 021066 000207 RTS PC ; RETURN

```

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```

2690                                     .SBTTL  KTINIT  - SETUP KT11 MEMORY MANAGEMENT REGISTERS
2691                                     ;*
2692                                     ;
2693                                     ;ROUTINE TO INIT KT-11
2694                                     ;
2695                                     ;-
2696
2697 021070                               KTINIT:
2698 021070 005037 003132                 CLR      KTFLG           ; INIT >28K MEMORY FLAG
2699 021074 005037 003134                 CLR      KTENABLE      ; INIT TEST >28K FLAG
2700 021100 023727 002120 001577         CMP      L#HIME,#1577   ; GOT ENOUGH MEMORY (>28K)?
2701 021106 101444                         BLOS     9#            ; NO.
2702 021110 013700 000004                 MOV      @#ERRVEC,RO   ; SAVE OLD ERR VEC PTR.
2703 021114 012737 021206 000004         MOV      #2#,@#ERRVEC ; SET ERR VEC PTR.
2704 021122 005737 177572                 TST      @#SRO         ; GOT KT11?
2705 021126 000240                         NOP                       ; (TRAP IF NO).
2706 021130 013737 002120 003132         MOV      L#HIME,KTFLG  ; YES. SET KT FLAG.
2707 021136 042737 000177 003132         BIC      #177,KTFLG   ;
2708 021144 010037 000004                 MOV      RO,@#ERRVEC  ; RESTORE OLD FRR VEC PTR.
2709 021150 005000                         CLR      RO            ; RO = AR DATA.
2710 021152 012701 172340                 MOV      #KIPAR0,R1   ; R1 = KI REGS PTR.
2711 021156 012761 077406 177740 1#:#   MOV      #77406,-40(R1); SET DESCRIPTOR REG.
2712 021164 010021                         MOV      RO,(R1)      ; SET KIPAR REG.
2713 021166 062700 000200                 ADD      #200,RO      ; BUMP AR DATA BY "4K".
2714 021172 020027 002000                 CMP      RO,#2000     ; AT "I/O"?
2715 021176 001367                         BNE     1#            ; NO.
2716 021200 012741 177600                 MOV      #177600,-(R1); YES. SET KTPAR7 FOR I/O.
2717 021204 000405                         BR      9#            ;
2718
2719 021206 012716 021214 2#:#           MOV      #6#,(SP)     ; SET UP RETURN
2720 021212 000002                         RTI                       ; RTI TO NEXT LOCATION
2721
2722 021214 010037 000004 6#:#           MOV      RO,@#ERRVEC  ; RESTORE OLD ERR VEC PTR.
2723
2724 021220 000207 9#:#                 RTS      PC

```


KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```

2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739 021222
2740
2741 021222 005737 002224
2742 021226 001020
2743 021230 012737 100206 021274
2744 021236 012737 021304 021276
2745 021244 012737 000006 021302
2746 021252 012737 100010 021304
2747 021260 012704 021274
2748 021264 004737 010662
2749 021270 000207
2750
2751
2752
2753 021274
2754
2755 021274 000000
2756 021276 000000
2757 021300 000000
2758 021302 000000
2759
2760
2761
2762 021304 000000
2763 021306 000000
2764 021310 000000
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776 021312
2777
2778 021312
2779 021316 005037 003136
2780 021322 005037 003140
2781 021326 005037 003142
2782 021332 005737 003146

```

```

;
; * 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
1$: RTS PC ; RETURN
;
; COMMAND PACKET.
;
; = <..3>&177774 ;MUST BE ON MOD 4 BOUNDRY.
CMDPKT:: 0 ;1ST WORD IS TS05 COMMAND.
          0 ;2ND WORD IS THE BUFFER LOW ADDRESS.
          0 ;3RD WORD IS THE BUFFER HIGH ADDRESS.
          0 ;4TH WORD IS THE BYTE/RECORD/FILE COUNT.
;
; WRITE SUB-SYSTEM MEMORY CHARACTERISTIC BLOCK.
WSMBK:: 0 ;1ST WORD:: SEL 0
        0 ;2ND WORD:: SEL 2
        0 ;3RD WORD:: SEL 4
        .EVEN
;
; * 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 ;SAVE THE REGISTERS
    CLR NXMFLG ;CLEAR THE FLAG
    CLR NXMLO ;CLEAR THE TEST ADDRESS LO
    CLR NXMHI ;CLEAR THE TEST ADDRESS HI
    TST T238 ;IS IT A 11/238?

```

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```

2783 021336 001407          BEQ      1#          ;NO
2784 021340 023727 002120 007777  CMP      L#HIME,#7777 ; GREATER THAN 128K
2785 021346 103406          BLO      2#          ; NO
2786 021350 004737 021466  JSR      PC,NXMTST  ;SETUP THE ADDRESS
2787 021354 000427          BR       13#         ;SET THE FLAG AND EXIT
2788 021356 005737 003144 1# :    TST      T23A    ;IS IT A 11/23A?
2789 021362 001413          BEQ      4#          ;NO
2790 021364 023727 002120 005777 2# :    CMP      L#HIME,#5777 ;GREATER THAN 96K
2791 021372 101023          BHI      14#         ;YES, 23A/23B WITH 128K MEMORY
2792 021374 023727 002120 003777  CMP      L#HIME,#3777 ;GREATER THAN 64K BUT LESS THAN 92K?
2793 021402 103403          BLO      4#          ;NO, CHECK 24K
2794 021404 004737 021466  JSR      PC,NXMTST  ;SETUP THE ADDRESS
2795 021410 000411          BR       13#         ;SET THE FLAG AND EXIT
2796 021412 023727 002120 001577 4# :    CMP      L#HIME,#1577 ;GREATER THAN 24K BUT LESS THAN 64K?
2797 021420 103410          BLO      14#         ;NO, TELL THEM AND EXIT WITH FLAG CLEAR
2798 021422 004737 021466  JSR      PC,NXMTST  ;SETUP THE ADDRESS
2799 021426 062737 000077 003142  ADD      #77,NXMHI  ;FOOL THE 11/02 & 11/03
2800 021434 005237 003136 13# :    INC      NXMFLG   ;SET THE FLAG
2801 021440 000411          BR       15#         ;EXIT
2802 021442 000410 14# :    BR       15#         ;NOP FOR PRINTOUT
2803 021444          PRINTF  #NOMEM   ;TELL THEM & EXIT ***NO PRINT*****
      021444 012746 005460  MOV      #NOMEM,-(SP)
      021450 012746 000001  MOV      #1,-(SP)
      021454 010600          MOV      SP,R0
      021456 104417          TRAP    C#PNTF
      021460 062706 000004  ADD      #4,SP
2804 021464 000207 15# :    RTS      PC          ;RETURN
2805
2806
2807 ;*
2808 ; SUBROUTINE TO SETUP THE NXM ADDRESS FOR TESTING
2809 ;
2810 ; OUTPUTS: NXMLO, NXMHI ; SETUP WITH NXM ADDRESS
2811 ;
2812 ;-
2813 021466 013701 002120  NXMTST: MOV      L#HIME,R1 ;GET TOP OF MEMORY
2814 021472 062701 000200  ADD      #200,R1   ;MAKE IT I/O BLOCK OR OTHER NXM
2815 021476 042701 000177  BIC      #177,R1
2816 021502 010102          MOV      R1,R2    ;RESAVE RESULTS
2817          000006          .REPT    6
2818          ASL      R1    ;PUT IN PLACE FOR XFER
2819          .ENDR
2820 021520 010137 003140  MOV      R1,NXMLO ;SAVE TEST ADDRESS LOW
2821          000012          .REPT    10
2822          ASR      R2    ;PUT IN PLACE FOR XFER
2823          .ENDR
2824 021550 042702 177700  BIC      #177700,R2 ;DON'T WANT ILA!
2825 021554 010237 003142  MOV      R2,NXMHI ;SAVE TEST ADDRESS HIGH
2826 021560 000207          RTS      PC          ;RETURN
2827
2828 021562          ENDMOD

```

H8

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```
7  
8  
9 021562 .TITLE TSV4 - MISCELLANEOUS SECTIONS  
021562 BGNMOD TSV4  
10 TSV4::  
16  
17  
18  
19 .SBTTL PROTECTION TABLE  
20 021562 BGNPROT  
021562 L#PROT::  
21 021562 177777 177777 177777 .WORD -1, -1, -1, -1 ;NO DEVICE PROTECTION REQUIRED.  
22 021572 ENDPROT
```

INITIALIZE SECTION

```

24                                     .SBTTL INITIALIZE SECTION
25
26                                     ;**
27                                     ;THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
28                                     ;AT THE BEGINNING OF EACH PASS.
29
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 021572                                BGNINIT
39 021572                                L#INIT::
40 021572 005037 002224                    40$: CLR     EXTFEA
41 021576 005037 003136                    CLR     NXMFLG
42 021602 012737 006360 002176            MOV     #EPT1,EPTSW
43 021610 005037 003154                    CLR     SIFLAG
44 021614 005037 003134                    CLR     KTENABLE
45 021620 005037 002302                    CLR     RAMSIZ
46 021624 012700 000036                    READEF  #EF.CONTINUE
47 021632 104447                            MOV     #EF.CONTINUE,R0
48 021632 103023                            TRAP   C#REFG
49 021634 023737 002200 002012            BNCOMPLET 1$
50 021642 103070                            BCC     1$
51 021644 005737 003112                    CMP     UNITN,L#UNIT
52 021650 100472                            BHIS   4$
53 021652 013701 002200                    TST    DUFLG
54 021656 006301                            BMI    NXTU
55 021660 005761 003176                    MOV    UNITN,R1
56 021664 001516                            ASL    R1
57 021666 032761 040000 003176            TST    ERTABL(R1)
58 021674 001060                            BEQ    SETU
59 021676 104432                            BIT    #BIT14,ERTABL(R1)
60 021700 000416                            BNE    NXTU
61 021702 012700 000035                    EXIT   INIT
62 021706 104447                            TRAP   C#EXIT
63 021710 103052                            .WORD  L10030-.
64 021712 012700 000040                    1$: READEF #EF.NEW
65 021716 104447                            MOV    #EF.NEW,R0
66 021720 103404                            TRAP   C#REFG
67 021722 012700 000037                    BNCOMPLET NXTU
68 021726 104447                            BCC    NXTU
69 021730 103031                            READEF #EF.START
70 021732 012700 000037                    MOV    #EF.START,R0
71 021732 104433                            TRAP   C#REFG
72 021732 104433                            BCOMPLET 2$
73 021732 104433                            BCS    2$
74 021732 104433                            READEF #EF.RESTART
75 021732 104433                            MOV    #EF.RESTART,R0
76 021732 104433                            TRAP   C#REFG
77 021732 104433                            BNCOMPLET 31$
78 021732 104433                            BCC    31$
79 021732 104433                            BRESET
80 021732 104433                            TRAP   C#RESET

```

```

;SET UP PRIMARY MESSAGE FOR REPLACEMENT
;CLEAR "SOFT INIT" FLAG
;CLEAR TEST ABOVE 28K FLAG
;CLEAR RAM SIZE FOR RAMERR ROUTINE

```

```

;UNIT IN RANGE?
;BR IF NO.
;DROPPED UNIT?
;BR IF YES

```

```

;DROPPED?
;DO NOTHING IF "CONTINUE".

```

```

;TAKE NEXT UNIT IF NOT NEW PASS.

```

```

;1ST PASS, BUS-INIT...
;BUS RESET.

```

INITIALIZE SECTION

```

65 021734 005037 002212          CLR      TSTCNT          ;NUMBER OF TESTS RUN IN PASS
66 021740 005037 002220          CLR      FATFLG         ;CLEAR FATAL ERROR COUNT
67 021744 005037 003144          CLR      T23A          ;CLEAR 11/23A FLAG
68 021750 005037 003146          CLR      T23B          ;CLEAR 11/23B FLAG
69                               ;      MOV      #340,-(SP)
70                               ;      MOV      #20,-(SP)          ;RETURN TO DEBUGGER
71                               ;      JMP      0,ODT          ;ENTER THE DEBUGGER
72 021754 005037 003400          CLR      SKIPT          ;CLEAR THE SUBTEST "SKIPPER"
73 021760                               20$:
74 021760 012737 177777 002202    MOV      #-1,QVP        ;...QUICK VERIFY...
75 021766 004737 020710          JSR      PC,ENVIRN      ;SET ENVIRONMENT.
76 021772 004737 021070          JSR      PC,KTINIT     ;INITIALIZE KT MEMORY MANAGEMENT
77 021776 012700 003176          MOV      #ERTABL,RO
78 022002 005020 003376          30$:  CLR      (RO)+        ;CLEAR THE ERROR TABLE
79 022004 020027 003376          CMP      RO,#ERTABE
80 022010 103774          BLO     30$
81 022012 000404          BR      4$
82 022014 005037 002202          31$:  CLR      QVP
83 022020 000137 022070          JMP      PASRPT        ;GO REPORT THE STATUS
84
85 022024                               4$:
86 022024 012737 177777 002200    NEWPAS: MOV      #-1,UNITN ;INIT UNIT NUMBER...
87 022032 005037 002216          CLR      DEVCNT        ;CLEAR COUNT OF DEVICES RUNNING
88 022036                               NXTU:
89 022036 104422          TRAP    C#BRK
90 022040 005237 002200          INC     UNITN          ;...AND SET NEXT UNIT NUMBER.
91 022044 023737 002200 002012    CMP      UNITN,L#UNIT
92 022052 103423          BLO     SETU
93 022054 012737 177777 003112    MOV      #-1,DUFLG
94 022062 000401          BR      11$
95 022064                               DOCLN
96 022064 104444          TRAP    C#DCLN        ;ABORT, NO MORE UNITS.
97 022066 000240          NOP
98 022070                               11$:
99 022070 023727 002012 000001    PASRPT: CMP      L#UNIT,#1    ;HOW MANY UNITS SELECTED?
100 022100 005737 002216          BLOS   NEWPAS         ;BR IF ONLY 1
101 022104 001747          TST    DEVCNT         ;ARE ANY STILL RUNNING?
102 022106 104421          BEQ    NEWPAS         ;BR IF NO
103 022110 032700 000100          RFLAGS RO
104 022114 001343          TRAP   C#RFLA
105 022116          BIT   #ISR,RO        ;SHOULD WE PRINT STATISTICS
106 022116 104424          BNE   NEWPAS         ;BR IF NO
107 022120 000741          DORPT
108 022122          TRAP   C#DRPT
109 022122          BR    NEWPAS
110 022122          SETU: GPHARD UNITN,RO ;GET UNIT N P-TABLE POINTER.
111 022122 013700 002200          MOV    UNITN,RO
112 022126 104442          TRAP   C#GPHRD
113 022130          BNCOMPLETE NXTU     ;BR IF UNIT NOT AVAILABLE.
114 022132 005037 003112          BCC    NXTU
115 022136 005237 002216          CLR    DUFLG         ;CLEAR "DROPPED" FLAG.
116 022142 012001          INC    DEVCNT
117 022144 010137 002204          MOV    (RO)+,R1      ;GET 1ST REGISTER ADDRESS.
118                               MOV    R1,CSRADDR     ;ADDRESS OF REGISTERS OF UNIT UNDER TEST

```

INITIALIZE SECTION

```

115
116 022150 012001          MOV      (R0),R1          ;GET VECTOR ADDRESS.
117                      ;MOV      (R0),R2          ;GET INTERRUPT PRIORITY
118                      ;MOV      R2,IPRI        ;SET INTERRUPT PRIORITY.
119 022152 010137 002206    MOV      R1,IVEC         ;SET INTERRUPT VECTOR POINTER...
120 022156 012721 016276    MOV      #INTR,(R1)+     ;...VECTOR...
121 022162 013721 002210    MOV      IPRI,(R1)+     ;...AND PRIORITY.
122
123 022166                14:
124                      ;      TST      QVP          ;1ST PASS ??
125                      ;      BEQ      54          ;NO, SKIP THE PASS 1 STUFF.
126
127
128                      ;
129                      ;1ST PASS, CHECK THAT DEVICE ADDRESSES ARE VALID, AND
130                      ;THAT THE DISPLAY STATUS IS PROPERLY INITIALIZED.
131
131 022166 013701 002200    MOV      UNITN,R1
132 022172 006301          ASL      R1
133 022174 052761 100000 003176  BIS      #BIT15,ERTABL(R1) ;SAY DEVICE RUNNING
134 022202 005037 005772    CLR      EXTA           ;CLEAR ERROR EXTENSION FLAG.
135 022206 023727 002012 000001  CMP      L#UNIT,#1      ;ARE WE TESTING MULTIPLE UNITS?
136 022214 101416          BLOS    104            ;BR IF NO.
137 022216                RFLAGS   RO           ;YES -- GET OPERATOR FLAGS.
138 022220 032700 001000    TRAP    C#RFLA
139 022224 001412          BIT      #PNT,RO       ;SHOULD WE PRINT UNIT #?
140 022226                BEQ      104            ;BR IF NOT.
141 022226 013746 002200    PRINTF  #PUNIT,UNITN   ;PRINT THE UNIT #
142 022232 012746 022320    MOV      UNITN,-(SP)
143 022236 012746 000002    MOV      #PUNIT,-(SP)
144 022242 010600          MOV      #2,-(SP)
145 022244 104417          MOV      SP,RO
146 022246 062706 000006    TRAP    C#PNTF
147 022252                ADD      #6,SP
148 022252 005037 003114    104:   CLR      NODEV
149 022256 013701 002204    MOV      CSRADDR,R1    ;ADDRESS OF FIRST REGISTER
150 022262 010102          MOV      R1,R2         ;START OF REGISTERS
151 022264 062702 000002    ADD      #TSSR,R2      ;ADDRESS OF TSSR REGISTER
152 022270 004737 016456    JSR      PC,XNXM       ;TEST BOTH CONTROLLER REGISTERS...
153 022274 103005          BCC     24             ;...AND BR IF ALL OK.
154 022276 010137 003114    MOV      R1,NODEV      ;FLAG DEVICE AS NON-EXISTENT
155 022302 012737 177777 003112  MOV      #-1,DUFLG     ;DROP THIS UNIT.
156 022310
157
158                      ;
159                      ;FINALLY, SET CPU PRIORITY AND WE'RE DONE.
160
161 022310                54:
162 022310 012700 000000    SETPRI  #PRIO0         ;ENABLE INTERRUPTS.
163 022314 104441          MOV      #PRIO0,RO
164 022316                TRAP    C#SPRI
165 022316                ENDINIT
166 022316 104411          L10030: TRAP    C#INIT
167 022320 045 116 045 PUNIT: .ASCIZ  /#N#N#A***** TESTING UNIT #D2#A *****/
168                      .EVEN

```

ADD AND DROP UNITS SECTIONS

```

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                                     ;--
167 022366                                     BGNAU
168 022366 010001                               L#AU::
169 022370 006301                               MOV     RO,R1                ; GET UNIT TO BE ADDED (RO)
170 022372 052761 100000 003176                ASL     R1                    ; MAKE IT A WORD INDEX
171 022400 042761 040000 003176                BIS     #100000,ERTABL(R1)   ; SET THE "ACTIVE" BIT
172 022406                                     BIC     #40000,ERTABL(R1)   ; CLEAR THE "DROPPED" BIT
173 022406 010046                               PRINTF  #1#,RO
174 022410 012746 022434                               MOV     RO,-(SP)
175 022414 012746 000002                               MOV     #1,-(SP)
176 022420 010600                               MOV     #2,-(SP)
177 022422 104417                               MOV     SP,RO
178 022424 062706 000006                               TRAP   C#PNTF
179 022430                                     ADD     #6,SP
180 022430 000167                               EXIT   AU
181 022432 000026                               .WORD  J#JMP
182 022434 045 116 045 1# :                     .WORD  L10031-2-
183                                     .ASCIZ /#N#A UNIT #D#A ADDED/
184                                     .EVEN
185
186                                     ENDAU                ; UNUSED.
187
188 L10031:                                     TRAP   C#AU
189
190                                     ;**
191                                     ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
192                                     ; TO BE REMOVED FROM THE TEST LIST.
193
194                                     ; SUPVSR DOES THE "DROPPING". THIS IS JUST TO TELL THE MAN.
195                                     ; "DROPPED" UNITS ARE RE-SELECTED ON OPERATOR "STA" OR "ADD"
196                                     ; COMMAND, OTHERWISE REMAIN INACTIVE. THE "DISPLAY" COMMAND
197                                     ; WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
198                                     ; WHICH ARE STILL ACTIVE.
199                                     ; UPON ENTRY, RO CONTAINS THE UNIT TO BE DROPPED.
200
201                                     BGNDU
202 L#DU::
203 022464 012737 177777 003112                MOV     #-1,DUFLG
204 022472 010001                               MOV     RO,R1
205 022474 006301                               ASL     R1
206 022476 052761 140000 003176                BIS     #140000,ERTABL(R1)   ; SAY DROPPED
207 022504 000240 000240 000240                240,240,240                ; ??????????
208 022512                                     PRINTF  #1#,RO
209 022512 010046                               MOV     RO,-(SP)
210 022514 012746 022540                               MOV     #1,-(SP)
211 022520 012746 000002                               MOV     #2,-(SP)
212 022524 010600                               MOV     SP,RO
213 022526 104417                               TRAP   C#PNTF
214 022530 062706 000006                               ADD     #6,SP
215 022534                                     EXIT   DU
216 022534 000167                               .WORD  J#JMP
217 022536 000030                               .WORD  L10032-2-

```

M8

SEQ 0103

ADD AND DROP UNITS SECTIONS

```

197 022540 045 116 045 14: .ASCIZ /#N#A UNIT #D#A DROPPED/
198 .EVEN
199 022570 ENDDU
022570 L10032: TRAP C4DU
022570 104453
200
201 ;**
202 ; AUTO-DROP CODE SECTION.
203 ;--
022572 BGNAUTO
022572 L4AUTO:
204 022572 013705 002204 MOV CSRADDR,R5 ;POINT TO DEVICE REGISTER
205 022576 012703 000550 MOV #360.,R3 ;ENOUGH TIME FOR 2400' REEL TO REWIND
206 022602 004737 016330 104: JSR PC,WAITF ;WAIT FOR SSR TO SET
207 022606 103420 BCS 204 ;LEAVE WHEN SSR IS SET
208 022610 DELAY 250. ;WAIT FOR .25 SECONDS
022610 012727 000372 MOV #250.,(PC)
022614 000000 .WORD 0
022616 013727 002116 MOV L4DLY,(PC)
022622 000000 .WORD 0
022624 005367 177772 DEC -6(PC)
022630 001375 BNE -.4
022632 005367 177756 DEC -22(PC)
022636 001367 BNE -.20
209 022640 005303 DEC R3 ;BUMP COUNTER DOWN
210 022642 001357 BNE 104 ;KEEP GOING
211 022644 004737 017262 JSR PC,CKDROP ;TRY AND DROP UNIT
212 022650
213 022650 204: ENDAUTO ; UNUSED.
022650 L10033: TRAP C4AUTO
022650 104461

```


CLEAN-UP AND REPORT CODING SECTIONS

```

215          .SBTTL  CLEAN-UP AND REPORT CODING SECTIONS
216
217          ;**
218          ; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS
219          ; EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).
220          ; USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.
221          ;--
222          BGNCLN
223          L$CLEAN::
224          MOV      CSRADDR,R5          ;POINT TO DEVICE REGISTER
225          TST      DUFLG              ;"DROPPED" FLAG IS SET ON...
226          BMI      1$                ;...AND GROSS CONTROLLER FAULT...
227          ;...DON'T TRY TO XCT CLEANUP CODE.
228          MOV      #0,TSSR(R5)        ;DO SOFT INIT
229          JSR      PC,WAITF
230          1$:
231          ENDCLN
232          L10034: TRAP  C$CLEAN
233
234          ;**
235          ; THE REPORT CODING SECTION CONTAINS THE
236          ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
237          ;--
238          BGNRPT
239          L$RPT::
240          PRINTS  #DEVSUM
241          MOV      #DEVSUM,-(SP)
242          MOV      #1,-(SP)
243          MOV      SP,R0
244          TRAP    C$PNTS
245          ADD      #4,SP
246          MOV      R2,-(SP)
247          MOV      R3,-(SP)
248          MOV      R4,-(SP)
249          MOV      #ERTABL,R4          ; GET START OF ERROR TABLE.
250          CLR      R3                  ; CLEAR UNIT NUMBER
251          1$: MOV      (R4),R2          ; GET ERROR TABLE ENTRY & TEST IT.
252          BEQ      4$                  ; ZERO IF UNIT NOT RUN
253          BPL      4$
254          BIT      #BIT14,R2          ; WAS UNIT DROPPED?
255          BNE      2$                  ; BR IF YES
256          BIC      #1C7777,R2         ; GET ERROR COUNT FIELD
257          PRINTS  #DEVONL,R3,R2      ; PRINT
258          MOV      R2,-(SP)
259          MOV      R3,-(SP)
260          MOV      #DEVONL,-(SP)
261          MOV      #3,-(SP)
262          MOV      SP,R0
263          TRAP    C$PNTS
264          ADD      #10,SP
265          BR      4$
266          2$: CMP      R2,#160000      ; WAS UNIT NON-EXISTENT?
267          BNE      3$                  ; BR IF NO
268          PRINTS  #DEVNKR,R3
269          MOV      R3,-(SP)
270          MOV      #DEVNKR,-(SP)

```

CLEAN-UP AND REPORT CODING SECTIONS

```

023016 012746 000002      MOV      #2,-(SP)
023022 010600      MOV      SP,R0
023024 104416      TRAP     C#PNTS
023026 062706 000006      ADD      #6,SP
254 023032 000431      BR       4#
255 023034 020227 160001      3# :    CMP      R2,#160001      ; WAS UNIT NOT READY AT STARTUP?
256 023040 001012      BNE      30#                ; BR IF NO.
257 023042      PRINTS  #DEVNRD,R3
023042 010346      MOV      R3,-(SP)
023044 012746 023331      MOV      #DEVNRD,-(SP)
023050 012746 000002      MOV      #2,-(SP)
023054 010600      MOV      SP,R0
023056 104416      TRAP     C#PNTS
023060 062706 000006      ADD      #6,SP
258 023064 000414      BR       4#
259 023066 042702 170000      30# :   BIC      #1C7777,R2
260 023072      PRINTS  #DEVDR0,R3,R2
023072 010246      MOV      R2,-(SP)
023074 010346      MOV      R3,-(SP)
023076 012746 023412      MOV      #DEVDR0,-(SP)
023102 012746 000003      MOV      #3,-(SP)
023106 010600      MOV      SP,R0
023110 104416      TRAP     C#PNTS
023112 062706 000010      ADD      #10,SP
261 023116 062704 000002      4# :    ADD      #2,R4
262 023122 005203      INC      R3
263 023124 020427 003376      CMP      R4,#ERTABE
264 023130 103701      BLO     1#
265 023132 012604      MOV      (SP)+,R4
266 023134 012603      MOV      (SP)+,R3
267 023136 012602      MOV      (SP)+,R2
268 023140      ENDRPT      ; UNUSED.
023140      L10035:
023140 104425      TRAP     C#RPT
269
270 023142      045      116      045  DEVSUM: .ASCIZ  /#N#ADEVICE STATUS SUMMARY:#N/
271 023177      045      101      040  DEVONL: .ASCIZ  /#A UNIT #D3#A ONLINE, ERRORS = #D#N/
272 023247      045      101      040  DEVNXR: .ASCIZ  /#A UNIT #D3#A DROPPED, NON-EXISTENT REGISTER#N/
273 023331      045      101      040  DEVNRD: .ASCIZ  /#A UNIT #D3#A DROPPED, NOT READY AT STARTUP#N/
274 023412      045      101      040  DEVDR0: .ASCIZ  /#A UNIT #D3#A DROPPED, ERRORS = #D#N/
275      .EVEN
276
277 023462      ENDMOD
278

```

CLEAN-UP AND REPORT CODING SECTIONS

1
2
3
9
10
16
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63

.TITLE TSVSA - HARDWARE TESTS

023462
023462

TSV5:: BGNMOD TSV5

.SBTTL TEST 1: BUS RESET TEST

: THIS TEST VERIFIES THAT THE M7196 MODULE'S DEVICE REGISTERS ARE
: ACCESSIBLE ON THE BUS (SUBTEST 1) AND THEN CHECKS THAT THE
: BUILT-IN INITIALIZATION SELF-TEST MICRODIAGNOSTIC DID NOT FIND
: ANY BASIC PROBLEMS IN THE MODULE. AREAS OF LOGIC TESTED BY THE
: SELF-TEST SEQUENCE ARE AS FOLLOWS: ROM AND PIPELINE REGISTER,
: SEQUENCER, INTERNAL BUSES, 2901 MICROPROCESSOR, AND, RAM. THIS
: TEST INITIALIZES THE CONTROLLER BY ISSUING THE BUS INIT SIGNAL
: VIA A RESET INSTRUCTION, OR BY WRITING INTO THE TSSR REGISTER,
: WAITS A PERIOD OF TIME (TO ALLOW THE CONTROLLER'S INITIALIZATION
: MICRODIAGNOSTIC SEQUENCE TO BE COMPLETED), AND THEN CHECKS THE
: CONTENTS OF THE TSSR REGISTER. SUCCESSFUL INITIALIZATION IS
: INDICATED BY SUBSYSTEM READY (SSR) AND NEED BUFFER ADDRESS (NBA)
: BITS BEING SET (1) AND ALL OTHER BITS (EXCEPT A17 AND A16 AND
: OFL, WHICH ARE IGNORED FOR THIS TEST) BEING CLEAR (0). IF THE
: CONTENTS OF TSSR ARE NOT AS EXPECTED, AN ERROR REPORT IS ISSUED
: LISTING THE EXPECTED DATA, ACTUAL DATA, AND THE DISCREPANCIES.
: THE ERROR REPORT ANALYZES THE TSSR CONTENTS AND DISCERNS AND
: REPORTS ONE OF THREE POSSIBILITIES:

- 1. TSSR CONTENTS ARE AMBIGUOUS (ANY OF BITS 11-14 ARE SET, OR STATES OF SSR AND SC BITS DO NOT CORRESPOND TO THE APPARENT ERROR CODE IN BITS 0-5): INDICATES THAT THE TSSR CONTENT CANNOT BE TRUSTED. INDICATES A CATASTROPHIC CONTROLLER MALFUNCTION. THIS IS A FATAL ERROR (EXECUTION IS ABORTED). FIELD ACTION WOULD BE TO REPLACE THE M7196. IF THE M7196 ITSELF IS BEING DEBUGGED, THE PROGRAM SHOULD BE RESTARTED WITH LOOP ON ERROR ENABLED IN ORDER TO PROBE FOR THE PROBLEM.
- 2. SSR = 0, SC = 0 AND THE ERROR CODE IN BITS 0-5 IS IN THE RANGE 17-13: THIS IS A FATAL ERROR. THE ERROR CODE IS DECODED AND THE APPROPRIATE DESCRIPTION GIVEN. INDICATES THAT A SERIOUS PROBLEM EXISTS.

023462

BGNTST

023462
69 023462 012700 023660
70 023466 004737 016570
71 023472 012737 000024 002214
72 023500
73 023500 005003
74
75 023502
023502
023502 104402

T1LOOP:

MOV #TST1ID,RO
JSR PC,TSTSETUP
MOV #20.,LOOPCNT
CLR R3

BGN SUB

T1::
;ASCII MESSAGE TO IDENTIFY TEST
;DO INITIAL TEST SETUP
;PERFORM 20 ITERATIONS
;USE R3 AS FATAL ERROR FLAG
;////////// BEGIN SUBTEST ///////////
T1.1: TRAP C#BSUB

E9

TEST 1: BUS RESET TEST

```

124 023654          404:  EXIT  TST          ;ALL DONE THIS TEST          TRAP  C4EXIT
      023654 104432          .WORD  L10036-.
      023656 000022

```

```

125
126
127      ;+
128      ;LOCAL TEXT MESSAGES FOR TEST
129      ;-

```

```

130 023660      111      156      151  TST1ID: .ASCIZ 'Initialization'
131
132 023700          .EVEN
      023700          .ENDTST

```

```

L10036:  TRAP  C4ETST

```

```

133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168

```

.SBTTL TEST 2: WRAP DATA - HIGH BYTE

THIS TEST VERIFIES OPERATION OF:

1. PART OF THE LSI-11 BUS INTERFACE SECTION OF THE M7196 MODULE: PART OF THE INPUT FILE (TSDB HIGH BYTE), PART OF THE OUTPUT FILE (TSSR HIGH BYTE AND TSBA, BOTH BYTES), PART OF THE DCO05 TRANSCEIVER CIRCUITS (ADDRESS DECODER, BDAL DRIVERS, HIGH BYTE OF INTERNAL DAL BUS DRIVERS), AND BASIC PROGRAMMED I/O CONTROL SEQUENCES AND LOGIC;
2. PART OF 2901 MICROPROCESSOR ELEMENTS (Q-REGISTER, REGISTER 0, ROTATE AND NEGATE FUNCTIONS);
3. Y AND SOURCE BUSES;
4. BASIC MICROPROGRAM SEQUENCES.

THE PROGRAM WRITES A TEST DATA BYTE INTO THE HIGH BYTE OF TSDB. WAITS FOR THE SSR BIT IN TSSR TO SET, THEN CHECKS THE CONTENTS OF BOTH TSBA AND TSSR. THE MODULE IS FUNCTIONING CORRECTLY IF DATA WRITTEN APPEARS IN BOTH BYTES OF TSBA AND THE FINAL CONTENT OF TSSR IS CORRECT (SAME AS AFTER INITIALIZATION EXCEPT FOR BITS 8 AND 9, WHICH SHOULD CONTAIN BITS 8 AND 9 OF THE DATA PATTERN WRITTEN. AN ERROR IS REPORTED AND A DESCRIPTIVE ANALYSIS GIVEN IF A DISCREPANCY IN TSBA OR TSSR IS DETECTED. THE ANALYSIS LISTS LIKELY FAULTY CANDIDATES FROM THE LOGIC ELEMENTS LISTED ABOVE. THE TEST IS REPEATED FOR ALL COMBINATIONS OF TEST DATA BYTES (0-377 OCTAL).

```

169 023702          BGNTST
      023702
174 023702 012700 024350
175 023706 004737 016570
176 023712 012737 000024 002214
177 023720 005004
178 023722 012703 177777
179 023726 005703

```

```

T2LOOP:
54:  TST

```

```

MOV  #TST2ID,R0
JSR  PC,TSTSETUP
MOV  #20,LOOPCNT
CLR  R4
MOV  #-1,R3
TST  R3

```

```

T2::
;ASCII MESSAGE TO IDENTIFY TEST
;DO INITIAL TEST SETUP
;PERFORM 20 ITERATIONS
;STARTING DATA PATTERN
;DO INIT ON FIRST TIME THROUGH
;DO WE NEED SOFT INIT

```


TEST 4: RAM TEST

```

385
386
387
388
389
390
391
392
393
394
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

```

TEST 4 , SUBTEST 1: -

THIS SUBTEST VERIFIES EACH RAM LOCATION BY FIRST PLACING THE M7196 INTO MAINTENANCE MODE BY WRITING INTO THE LOW BYTE OF TSDB AND THEN PERFORMING THE FOLLOWING SEQUENCE FOR EACH ADDRESS 0-7777 (OCTAL):

1. THE ADDRESS TO BE TESTED IS LOADED INTO THE TSDB (VIA A WORD WRITE).
2. THE ADDRESSED RAM LOCATION IS WRITTEN, THEN READ INTO THE LOW BYTE OF TSBA, BY WRITING A DATA BYTE INTO THE LOW BYTE OF TSDB.
3. THE LOW BYTE OF TSBA IS CHECKED TO SEE IF IT CONTAINS THE DATA PATTERN ORIGINALLY WRITTEN; A DISCREPANCY IS REPORTED AS AN ERROR.
4. THE ADDRESS OF THE LOCATION BEING TESTED IS AGAIN WRITTEN INTO TSDB (WORD WRITE), TO CAUSE THE LOCATION UNDER TEST TO AGAIN BE READ INTO THE LOW BYTE OF TSBA. THE LOW BYTE OF TSBA IS AGAIN CHECKED AND DISCREPANCIES REPORTED.
5. THE HIGH BYTE OF TSBA IS CHECKED; IT SHOULD CONTAIN THE SUM OF THE HIGH AND LOW BYTES LAST WRITTEN INTO TSDB AS A WORD. A DISCREPANCY IS REPORTED AS A 2901 PROBLEM.
6. THE CONTENT OF TSSR IS CHECKED; SETTING OF THE SC BIT IS IGNORED. OTHER DISCREPANCIES IN TSSR ARE REPORTED.

```

421 025072
      025072

```

BGNTST

T4::

```

422
423 025072
      025072
      025072 104402

```

BGNSUB

////////// BEGIN SUBTEST //////////

T4.1:

TRAP C#BSUB

```

424
429 025074 012700 026402
430 025100 004737 016570
431 025104 012737 000005 002214
432 025112
433 025112 004737 016054
434 025116 103405
438 025120 010001
439 025122

```

T4LOOP:

```

      MOV #TST4ID,R0
      JSR PC,TSTSETUP
      MOV #5,LOOPCNT
      JSR PC,SOFINIT
      BCS 20#
      MOV R0,R1
      ERROF ERANO,SFIERR,SFIMSG

```

```

;ASCII MESSAGE TO IDENTIFY TEST
;DO INITIAL TEST SETUP
;PERFORM 5 ITERATIONS

```

```

;DO INITIALIZE ON CONTROLLER
;BR IF INIT WAS OK
;CONTENTS OF TSSR REGISTER
;FATAL ERROR TSSR WAS NOT OK

```

```

      TRAP C#ERDF
      .WORD 401
      .WORD SFIERR
      .WORD SFIMSG

```

```

      025122 104455
      025124 000621
      025126 003652
      025130 012104
440 025132 005004
441 025134 004737 016416

```

20#:

```

      CLR R4
      JSR PC,CHKTSSR

```

```

;SET RAM ADDRESS AT ZERO
;WAIT FOR READY, NON-AMBIGUOUS

```


TEST 4: RAM TEST

```

537
538 025452 005304          35$: DEC    R4          ;SET BACK TO 7777
539 025454 005002          CLR    R2          ;SET TO ALL ZEROS
540 025456 004737 016416  40$: JSR    PC,CHKTSSR ;WAIT FOR READY (SSR) TO SET
541 025462 010465 000000  MOV    R4,TSDB(R5) ;LOAD UP THE ADDRESS FOR RAM
542 025466 004737 016416  JSR    PC,CHKTSSR ;WAIT FOR READY (SSR) TO SET
543 025472 016501 000000  MOV    TSBA(R5),R1 ;READ THE RAM CONTENTS BACK
544 025476 005002          CLR    R2          ;LOOKING FOR 000000 (EXPECTED)
545 025500 120102          CMPB   R1,R2       ;BOTH SHOULD BE 00000000 BINARY
546 025502 001404          BEQ    43$        ;BR, IF DATA IS GOOD
550 025504          ERRHRD ERRNO,TSBAM3,EXPREC ;CHARACTERISTICS DATA NOT CORRECT
      025504 104456          TRAP   C$ERHRD
      025506 000627          .WORD  407
      025510 026322          .WORD  TSBAM3
      025512 015554          .WORD  EXPREC
551 025514 012702 000377  43$: MOV    #000377,R2 ;SET ALL ONES WORD
552 025520 010465 000000  MOV    R4,TSDB(R5) ;LOAD UP RAM ADDRESS POINTER
553 025524 004737 016416  JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
554 025530 110265 000000  MOVB   R2,TSDB(R5) ;WRITE DATA INTO RAM
555 025534 004737 016416  JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
556 025540 016501 000000  MOV    TSBA(R5),R1 ;READ RAM CONTENTS BACK
557 025544 120102          CMPB   R1,R2       ;CHECK WITH DATA WRITTEN
558 025546 001404          BEQ    45$        ;BR IF OK, DATA IN = DATA OUT
562 025550          ERRHRD ERRNO,TSBAM2,EXPREC ;WRITTEN DATA NOT = TO READ
      025550 104456          TRAP   C$ERHRD
      025552 000630          .WORD  408
      025554 026240          .WORD  TSBAM2
      025556 015554          .WORD  EXPREC
563 025560          45$: CKLOOP ;SCOPE LOOP
      025560 104406          TRAP   C$CLP1
564 025562 004737 016416  JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
565 025566 010465 000000  MOV    R4,TSDB(R5) ;WORD WRITE TO SET UP ADDRESS
566 025572 004737 016416  JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
567 025576 116501 000001  MOVB   TSBAH(R5),R1 ;HIGH BYTE READ OF TSBA
568 025602 010403          MOV    R4,R3      ;DATA PATTERN WRITTEN
569 025604 000303          SWAB   R3         ;HIGH TO LOW
570 025606 060403          ADD    R4,R3      ;TOTAL OF BYTES IN LOW BYTE
571 025610 120103          CMPB   R1,R3      ;SUM OF BYTES WRITTEN TO TSDB = TSBAH
572 025612 001404          BEQ    50$        ;BR IF OK, THEY SHOULD BE
576 025614          ERRHRD ERRNO,M2901,EXPREC ;2901 PROBLEM ADDER
      025614 104456          TRAP   C$ERHRD
      025616 000631          .WORD  409
      025620 026150          .WORD  M2901
      025622 015554          .WORD  EXPREC
577 025624          50$: CKLOOP ;SCOPE LOOP
      025624 104406          TRAP   C$CLP1
578 025626 005304          DEC    R4         ;DROP RAM ADDRESS POINTER
579 025630 002312          BGE    40$        ;NOT AT LOC. ZERO YET
580
581 025632          ENDSUB ;////////////////////// END SUBTEST ////////////////////////
      025632          L10045:
      025632 104403          TRAP   C$ESUB
582
583
584 025634          BGNSUB ;////////////////////// BEGIN SUBTEST ////////////////////////
      025634          T4.3:
      025634 104402          TRAP   C$BSUB

```

TEST 4: RAM TEST

```

585
586      |      TEST 4, SUBTEST 3
587      |      |
588      |      |      THIS SUBTEST WRITES RAM WITH ALL ONES
589      |      |      |      THEN WALKS A ZERO WORD DOWN THROUGH MEMORY
590      |      |
591 025636 004737 016054      JSR      PC,SOFINIT      ;DO INITIALIZE ON CONTROLLER
592 025642 103405      BCS      20$            ;BR IF INIT WAS OK
596 025644 010001      MOV      R0,R1           ;CONTENTS OF TSSR REGISTER
597 025646      ERRDF   ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
                    |      TRAP      C4ERDF
                    |      .WORD   410
                    |      .WORD   SFIERR
                    |      .WORD   SFIMSG
598 025656 012702 177777      20$:   MOV      #177777,R2 ;SET DATA AT ALL ONES
599 025662 005004      CLR      R4             ;SET RAM ADDRESS AT ZERO
600 025664 004737 016416      JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
601 025670 105065 000000      CLR     TSDB(R5)      ;SET INTO MAINTENANCE MODE
602 025674      25$:
603 025674      BGNSEG                                          ;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                    |      TRAP      C4BSEG
604
605 025676 004737 016416      JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
606 025702 010465 000000      MOV     R4,TSDB(R5)  ;LOAD ADDRESS INTO TSDB
607 025706 004737 016416      JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
608 025712 110265 000000      MOV     R2,TSDB(R5)  ;LOADS DATA INTO RAM LOCATION
609 025716 004737 016416      JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
610 025722 016501 000000      MOV     TSBA(R5),R1  ;READS WRAP DATA
611 025726 120102      CMP     R1,R2        ;DOES WRITTEN(WRAP) = READ ?
612 025730 001404      BEQ     30$          ;BR IF OK, THEY ARE EQUAL
616 025732      ERRHRD  ERRNO,TSBAM2,EXPREC ;DATA NOT WRAPPED CORRECTLY
                    |      TRAP      C4ERHRD
                    |      .WORD   411
                    |      .WORD   TSBAM2
                    |      .WORD   EXPREC
617 025742      30$:
618 025742      ENDSEG                                          ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
                    |      10000$: TRAP C4ESEG
619
620 025744 005204      INC     R4            ;NEXT ADDRESS
621 025746 020427 010000      CMP     R4,#10000   ;END OF RAM MEMORY CHECK
622 025752 001350      BNE     25$          ;BR, MORE RAM TO GO
623
624 025754      ESCAPE  SUB                                          ;NO CHECK IF WRITTEN INCORRECTLY
                    |      TRAP      C4ESCAPE
                    |      .WORD   L10046-.
625 025760 005304      35$:   DEC     R4            ;SET BACK TO 7777
626 025762 004737 016416      40$:   JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
627 025766 012702 000377      MOV     #000377,R2 ;SET UP EXPECTED DATA REGISTER
628 025772 005001      CLR     R1            ;CLEAN OUT REGISTER
629 025774 010465 000000      MOV     R4,TSDB(R5)  ;SELECT ADDRESS IN RAM
630 026000 004737 016416      JSR      PC,CHKTSSR    ;WAIT FOR READY (SSR)
631 026004 016501 000000      MOV     TSBA(R5),R1  ;PICK UP RAM CONTENTS
632 026010 120102      CMP     R1,R2        ;IS MEMORY STILL ALL ONES
633 026012 001404      BEQ     43$          ;BR, IF OK (ALL ONES)
637 026014      ERRHRD  ERRNO,TSBAM3,EXPREC ;MEMORY CHANGED AFTER ALL ONES WRITE

```

TEST 4: RAM TEST

```

026014 104456                                TRAP      C4ERHRD
026016 000634                                .WORD    412
026020 026322                                .WORD    TSBAM3
026022 015554                                .WORD    EXPREC
638 026024 005002          43$: CLR      R2                                ;SET UP NEW EXPECTED
639 026026 010465 000000    MOV      R4,TSDB(R5)                        ;LOAD UP RAM ADDRESS POINTER
640 026032 004737 016416    JSR      PC,CHKTSSR                          ;WAIT FOR READY, NON-AMBIGUOUS
641 026036 110265 000000    MOV      R2,TSDB(R5)                        ;WRITE DATA INTO RAM
642 026042 004737 016416    JSR      PC,CHKTSSR                          ;WAIT FOR READY, NON-AMBIGUOUS
643 026046 016501 000000    MOV      TSBA(R5),R1                        ;READ RAM CONTENTS BACK
644 026052 120102          CMPB     R1,R2                                ;CHECK WITH DATA WRITTEN
645 026054 001404          BEQ      45$                                ;BR IF OK, DATA IN = DATA OUT
649 026056          ERRHRD  ERRNO,TSBAM2,EXPREC                ;WRITTEN DATA NOT = TO READ
                                TRAP      C4ERHRD
                                .WORD    413
                                .WORD    TSBAM2
                                .WORD    EXPREC
026056 104456
026060 000635
026062 026240
026064 015554
650 026066          45$: CKLOOP                                ;SCOPE LOOP
                                TRAP      C4CLP1
026066 104406
651 026070 004737 016416    JSR      PC,CHKTSSR                          ;WAIT FOR READY, NON-AMBIGUOUS
652 026074 116501 000001    MOV      TSBAH(R5),R1                       ;HIGH BYTE READ OF TSBA
653 026100 010203          MOV      R2,R3                               ;DATA PATTERN WRITTEN
654 026102 000303          SWAB     R3                                  ;HIGH TO LOW
655 026104 060203          ADD      R2,R3                               ;TOTAL OF BYTES IN LOW BYTE
656 026106 120103          CMPB     R1,R3                               ;SUM OF BYTES WRITTEN TO TSDB = TSBAH
657 026110 001404          BEQ      50$                                ;BR IF OK, THEY SHOULD BE
661 026112          ERRHRD  ERRNO,M2901,EXPREC                ;2901 PROBLEM ADDER
                                TRAP      C4ERHRD
                                .WORD    414
                                .WORD    M2901
                                .WORD    EXPREC
026112 104456
026114 000636
026116 026150
026120 015554
662 026122          50$: CKLOOP                                ;SCOPE LOOP
                                TRAP      C4CLP1
026122 104406
663 026124 005304          DEC      R4                                  ;DROP RAM ADDRESS POINTER
664 026126 001315          BNE      40$                                ;NOT AT LOC. ZERO YET
665
666 026130          ENDSUB                                ;////////////////// END SUBTEST ////////////////////
                                L10046: TRAP      C4ESUB
026130
026130 104403
667
668 026132 004737 016536    JSR      PC,TSTLOOP                          ;DO WE NEED TO ITERATE TEST ?
669 026136 103002          BCC      63$                                ;BRANCH IF NOT
670 026140 000137 025112    JMP      T4LOOP                              ;EXECUTE AGAIN
671 026144          63$: EXIT  TST                            ;ALL DONE THIS TEST
                                TRAP      C4EXIT
                                .WORD    L10043-.
026144 104432
026146 000256
672
673          ;*
674          ;LOCAL TEXT MESSAGES FOR TEST
675          ;-
676 026150          040 124 123 M2901: .ASCIZ ' TSBA High Byte Not Sum of Last TSDB Write (2901 Error)'
677 026240          040 127 162 TSBAM2: .ASCIZ ' Write to TSDB Not Equal to Read of TSBA Low Byte'
678 026322          127 162 151 TSBAM3: .ASCIZ 'Write To RAM Location Modified Another Location'
679 026402          122 101 115 TST4ID: .ASCIZ 'RAM Verification'
680
681 026424          .EVEN
                                ENDTST
                                L10043:
026424

```

TEST 4: RAM TEST

TRAP C#ETST

026424 104401

682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719

.SBTTL TEST 5: SECOND INITIALIZATION TEST

THIS TEST VERIFIES THE SAME ELEMENTS AS DID INITIALIZATION TEST #1 AND ALSO CHECKS THAT CERTAIN PARTS OF RAM IS CLEARED TO ZERO AND THAT 2901 REGISTERS 10 AND 11 ARE ALSO CLEARED TO ZERO. THIS IS A CONFIDENCE CHECK OF A PART OF THE SELF-TEST SEQUENCE (I.E., THAT IT IS REALLY BEING EXECUTED). FOR EACH OF TWO SUBTESTS (ONE FOR INITIALIZING VIA A BUS INIT, THE OTHER FOR INITIALIZING BY WRITING INTO THE TSSR), THE FOLLOWING SEQUENCE IS PERFORMED:

1. EACH RAM LOCATION AND 2901 REGISTERS 10 AND 11 ARE SET TO ALL 1'S BY USING WRITES INTO THE TSDB REGISTER (LOW BYTE AND MAINTENANCE MODE WORD WRITES).
2. THE CONTROLLER IS INITIALIZED AND THE VARIOUS CHECKS ON THE TSSR DESCRIBED IN INITIALIZATION TEST #1 ARE PERFORMED.
3. #1'S (377 OCTAL) ARE WRITTEN INTO THE LOW BYTE OF TSDB, WHICH SHOULD CAUSE RAM LOCATION 0 TO BE WRITTEN TO ALL 1'S SINCE 2901 REGISTERS 10 AND 11, SPECIFYING THE RAM ADDRESS, SHOULD BE 0. RAM LOCATION 0 IS VERIFIED BY WRITING A WORD OF ZEROS INTO THE TSDB. THE RESULTING LOW BYTE OF TSBA SHOULD CONTAIN ALL 1'S.
4. THE ENTIRE RAM IS SCANNED. LOCATION 0 SHOULD CONTAIN ALL 1'S AND THE REMAINING LOCATIONS, EXCEPT FOR THE MESSAGE BUFFER IMAGE AREA, SHOULD CONTAIN 0. DISCREPANCIES ARE REPORTED. AN ERROR AT THIS POINT IS MOST LIKELY DUE TO A ROM, PIPELINE OR SEQUENCER PROBLEM OR A TIMING PROBLEM.

720 026426
026426
725 026426 012700 027400
726 026432 004737 016570
727 026436 012737 000024 002214
728 026444
729 026444 005037 002220
730
731 026450
026450
026450 104402
732
733 026452 004737 016054
734 026456 103404
738 026460
026460 104455
026462 000765
026464 003652

TSLOOP:

```

BGNTST
MOV    #TSTSID,RO      ;ASCII MESSAGE TO IDENTIFY TEST
JSR    PC,TSTSETUP    ;DO INITIAL TEST SETUP
MOV    #20,LOOPCNT    ;PERFORM 20 ITERATIONS
CLR    FATFLG         ;CLEAR THE FATAL ERROR FLAG
BGNSUB                ;////////// BEGIN SUBTEST ////////////
TS.1:
TRAP   C#BSUB
JSR    PC,SOFINIT     ;DO A SOFT TO START
BCS    10$            ;BRANCH IF O.K.
ERRDF  ERRNO,SFIERR,SFIMSG ;REPORT ERROR AND DROP DRIVE
TRAP   C#ERDF
.WORD  501
.WORD  SFIERR
    
```


TEST 5: SECOND INITIALIZATION TEST

```

026466 012104
739 026470 012702 177777      10#:  MOV      # -1,R2      ;ALL ONE DATA PATTERN
740 026474 005004              CLR      R4              ;STARTING RAM ADDRESS
741 026476 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
742 026502 105065 000000      CLRB    TSDB(R5)        ;SET MAINTENANCE MODE
743 026506 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
744 026512 010465 000000      MOV      R4,TSDB(R5)    ;SET THE NEXT RAM ADDRESS
745 026516 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
746 026522 110265 000000      MOVB    R2,TSDB(R5)    ;LOAD TEST DATA
747 026526 005204              INC      R4              ;NEXT ADDRESS TO TEST
748 026530 020427 007777      CMP      R4,#7777       ;COMPARE TO LAST ADDRESS
749 026534 003762              BLE     15#              ;BRANCH TILL ALL DATA WRITTEN
750 026536              BRESET                       ;ISSUE A BUS RESET
026536 104433              TRAP    C#RESET
751 026540 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
752 026544 016501 000002      MOV      TSSR(R5),R1    ;GET THE CONTENTS OF TSSR
753 026550 010102              MOV      R1,R2          ;CONTENTS OF TSSR
754 026552 042702 176277      BIC     #1<HIADDR!OFL>,R2 ;THESE BITS MAY BE SET
755 026556 052702 002200      BIS     #SSR!NBA,R2    ;READY AND NEW DATA SHOULD BE SET
756 026562 020102              CMP      R1,R2          ;COMPARE EXPECTED TO RECEIVED
757 026564 001406              BEQ     20#              ;BRANCH IF COMPARE
761 026566              ERDF    ERRNO,SFHERR,SFFMSG ;REPORT A FATAL ERROR
026566 104455              TRAP    C#ERDF
026570 000766              .WORD  502
026572 003705              .WORD  SFHERR
026574 012152              .WORD  SFFMSG
762 026576 005237 002220      20#:  INC      FATFLG      ;SET FATAL ERROR FLAG
763 026602              CKLOOP                       ;LOOP ON ERROR IF FLAG SET
026602 104406              ESCAPE  SUB                  ;EXIT IF FATAL ERROR DETECTED
764 026604              TRAP    C#CLP1
026604 104410              TRAP    C#ESCAPE
026606 000170              .WORD  L10050-.
765 026610 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
766 026614 105065 000000      CLRB    TSDB(R5)        ;PUT BACK INTO MAINTENANCE MODE
767 026620 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
768 026624 005065 000000      CLR     TSDB(R5)        ;SET ADDRESS BACK TO 0000
769 026630 012702 000377      MOV      #377,R2
770 026634 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
771 026640 110265 000000      MOVB    R2,TSDB(R5)    ;SHOULD POINT TO RAM 0
772 026644 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
773 026650 005065 000000      CLR     TSDB(R5)        ;SELECT LOCATION 0
774 026654 004737 016416      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
775 026660 116501 000000      MOVB    TSBA(R5),R1    ;READ RAM LOCATION SPECIFIED
776 026664 120102              CMPB    R1,R2          ;LOCATION SHOULD BE 377 OCTAL
777 026666 001406              BEQ     25#              ;BR IF OK
778 026670              ERDF    ERRNO,TSADDR,EXPREC ;WASN'T POINTING TO CORRECT LOC.
026670 104455              TRAP    C#ERDF
026672 000766              .WORD  502
026674 027466              .WORD  TSADDR
026676 015554              .WORD  EXPREC
779 026700 005237 002220      25#:  INC      FATFLG      ;SET THE FATAL ERROR FLAG
780 026704              CKLOOP                       ;SCOPE LOOP
026704 104406              ESCAPE  SUB                  ;NO MORE CHECKS IF FATAL ERROR
781 026706              TRAP    C#CLP1
026706 104410              TRAP    C#ESCAPE
026710 000066              .WORD  L10050-.
782 026712 012704 000310      MOV      #310,R4        ;START WITH LOC 310

```

TEST 5: SECOND INITIALIZATION TEST

```

783 026716 005002          CLR    R2          ;MEMORY EXPECTED SHOULD BE 000000
784 026720 004737 016416 304:   JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
785 026724 010465 000000          MOV    R4,TSDB(R5) ;SELECT LOCATION SPECIFIED
786 026730 004737 016416          JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
787 026734 116501 000000          MOVB  TSBA(R5),R1 ;READ LOC CONTENTS
788 026740 120102          CMPB  R1,R2       ;CHECK MEMORY FOR 000000
789 026742 001406          BEQ   404        ;BRANCH IF DATA OKAY
790 026744          ERRDF  ERRNO,TSMEM,SFFMSG ;MEMORY NOT ZERO AFTER INIT.
      026744 104455          TRAP  C#ERDF
      026746 000766          .WORD 502
      026750 027430          .WORD TSMEM
      026752 012152          .WORD SFFMSG
791 026754 005237 002220          INC    FATFLG      ;SET THE FATAL ERROR FLAG
792 026760          CKLOOP
793 026762          ESCAPE SUB      ;EXIT ON FATAL ERROR
      026762 104410          TRAP  C#CLP1
      026764 000012          .WORD C#ESCAPE
794 026766 005204          INC    R4          ;LOOK AT NEXT RAM LOC.
795 026770 020427 000400          CMP   R4,#400    ;AT TOP OF RAM ADDRESS SPACE
796 026774 001353          BNE   304        ;BRANCH TILL ALL MEMORY TESTED
797
798 026776          ENDSUB      ;//////////////////// END SUBTEST //////////////////////
      026776          L10050:
      026776 104403          TRAP  C#ESUB
799
800 027000 005737 002220          TST   FATFLG      ;IS FATAL ERROR FLAG SET ?
801 027004 001404          BEQ   504        ;BRANCH IF NOT
802 027006 004737 017262          JSR   PC,CKDROP  ;NO LOOP, TRY TO DROP DEVICE
803 027012 005037 002220          CLR   FATFLG     ;CLEAR THE FATAL ERROR FLAG
804 027016
805
806 027016          BGNSUB      ;//////////////////// BEGIN SUBTEST //////////////////////
      027016          T5.2:
      027016 104402          TRAP  C#BSUB
807
808 027020 004737 016054          JSR   PC,SOFINIT ;DO A SOFT TO START
809 027024 103404          BCS  104        ;BRANCH IF O.K.
813 027026          ERRDF  ERRNO,SFIERR,SFIMSG ;REPORT ERROR AND DROP DRIVE
      027026 104455          TRAP  C#ERDF
      027030 000767          .WORD 503
      027032 003652          .WORD SFIERR
      027034 012104          .WORD SFIMSG
814 027036 012702 177777          104:   MOV    #-1,R2      ;ALL ONE DATA PATTERN
815 027042 005004          CLR   R4         ;STARTING RAM ADDRESS
816 027044 004737 016416          JSR   PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
817 027050 105065 000000          CLRB TSDB(R5)    ;SET MAINTENANCE MODE
818 027054 004737 016416          JSR   PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
819 027060 010465 000000          MOV   R4,TSDB(R5) ;SET THE NEXT RAM ADDRESS
820 027064 004737 016416          JSR   PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
821 027070 110265 000000          MOVB  R2,TSDB(R5) ;LOAD TEST DATA
822 027074 005204          INC  R4         ;NEXT ADDRESS TO TEST
823 027076 020427 007777          CMP  R4,#7777   ;COMPARE TO LAST ADDRESS
824 027102 003762          BLE  154        ;BRANCH TILL ALL DATA WRITTEN
825 027104 005065 000002          CLR  TSSR(R5)   ;ISSUE A SOFT RESET
826 027110 004737 016416          JSR   PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
827 027114 016501 000002          MOV  TSSR(R5),R1 ;GET THE CONTENTS OF TSSR

```

TEST 5: SECOND INITIALIZATION TEST

828	027120	010102		MOV	R1,R2		;CONTENTS OF TSSR		
829	027122	042702	176277	BIC	#1C<HIADDR!OFL>,R2		;THESE BITS MAY BE SET		
830	027126	052702	002200	BIS	#SSR!NBA,R2		;READY AND NEW DATA SHOULD BE SET		
831	027132	020102		CMP	R1,R2		;COMPARE EXPECTED TO RECEIVED		
832	027134	001406		BEQ	20:		;BRANCH IF COMPARE		
836	027136			ERRDF	ERRNO,SFHERR,SFFMSG		;REPORT A FATAL ERROR		
	027136	104455						TRAP	C#ERDF
	027140	000770						.WORD	504
	027142	003705						.WORD	SFHERR
	027144	012152						.WORD	SFFMSG
837	027146	005237	002220	INC	FATFLG		;SET FATAL ERROR FLAG		
838	027152			20:	CKLOOP		;LOOP ON ERROR IF FLAG SET		
839	027152	104406						TRAP	C#CLP1
	027154			ESCAPE	SUB		;EXIT IF FATAL ERROR DETECTED		
	027154	104410						TRAP	C#ESCAPE
	027156	000170						.WORD	L10051-.
840	027160	004737	016416	JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET		
841	027164	105065	000000	CLRB	TSDB(R5)		;PUT BACK INTO MAINTENANCE MODE		
842	027170	004737	016416	JSR	PC,CHKTSSR		;WAIT FOR READY, NON-AMBIGUOUS		
843	027174	005065	000000	CLR	TSDB(R5)		;SET ADDRESS BACK TO 0000		
844	027200	012702	000377	MOV	#377,R2				
845	027204	004737	016416	JSR	PC,CHKTSSR		;WAIT FOR READY, NON-AMBIGUOUS		
846	027210	110265	000000	MOVB	R2,TSDB(R5)		;SHOULD POINT TO RAM 0		
847	027214	004737	016416	JSR	PC,CHKTSSR		;WAIT FOR READY, NON-AMBIGUOUS		
848	027220	005065	000000	CLR	TSDB(R5)		;SELECT LOCATION 0		
849	027224	004737	016416	JSR	PC,CHKTSSR		;WAIT FOR READY, NON-AMBIGUOUS		
850	027230	116501	000000	MOVB	TSBA(R5),R1		;READ RAM LOCATION SPECIFIED		
851	027234	120102		CMPB	R1,R2		;LOCATION SHOULD BE 377 OCTAL		
852	027236	001406		BEQ	25:		;BR IF OK		
853	027240			ERRDF	ERRNO,TSADDR,EXPREC		;WASN'T POINTING TO CORRECT LOC.		
	027240	104455						TRAP	C#ERDF
	027242	000770						.WORD	504
	027244	027466						.WORD	TSADDR
	027246	015554						.WORD	EXPREC
854	027250	005237	002220	INC	FATFLG		;SET THE FATAL ERROR FLAG		
855	027254			25:	CKLOOP		;SCOPE LOOP		
856	027254	104406						TRAP	C#CLP1
	027256			ESCAPE	SUB		;NO MORE CHECKS IF FATAL ERROR		
	027256	104410						TRAP	C#ESCAPE
	027260	000066						.WORD	L10051-.
857	027262	012704	000310	MOV	#310,R4		;START WITH LOC 310		
858	027266	005002		CLR	R2		;MEMORY EXPECTED SHOULD BE 000000		
859	027270	004737	016416	JSR	PC,CHKTSSR		;WAIT FOR READY, NON-AMBIGUOUS		
860	027274	010465	000000	MOV	R4,TSDB(R5)		;SELECT LOCATION SPECIFIED		
861	027300	004737	016416	JSR	PC,CHKTSSR		;WAIT FOR READY, NON-AMBIGUOUS		
862	027304	116501	000000	MOVB	TSBA(R5),R1		;READ LOC CONTENTS		
863	027310	120102		CMPB	R1,R2		;CHECK MEMORY FOR 000000		
864	027312	001406		BEQ	40:		;BRANCH IF DATA OKAY		
865	027314			ERRDF	ERRNO,TSMEM,SFFMSG		;MEMORY NOT ZERO AFTER INIT.		
	027314	104455						TRAP	C#ERDF
	027316	000770						.WORD	504
	027320	027430						.WORD	TSMEM
	027322	012152						.WORD	SFFMSG
866	027324	005237	002220	INC	FATFLG		;SET THE FATAL ERROR FLAG		
867	027330			40:	CKLOOP				
	027330	104406						TRAP	C#CLP1
868	027332			ESCAPE	SUB		;EXIT ON FATAL ERROR		

TEST 5: SECOND INITIALIZATION TEST

```

027332 104410
027334 000012
869 027336 005204
870 027340 020427 000400
871 027344 001353
872
873 027346
027346
027346 104403
874
875 027350 005737 002220
876 027354 001402
877 027356 004737 017262
878 027362 004737 016536
879 027366 103002
880 027370 000137 026444
881 027374
027374 104432
027376 000132
882
883
884
885
886
887 027400 105 170 164
888 027430 111 156 143
889 027466 111 156 143
890
891 027530
027530
027530 104401
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

```

```

INC R4
CMP R4,#400
BNE 30$
ENDSUB
TST FATFLG
BEQ 50$
JSR PC,CKDROP
JSR PC,TSTLOOP
BCC 60$
JMP T$LOOP
EXIT T$T

```

```

TRAP C$ESCAPE
.LWORD L10051-.
;LOOK AT NEXT RAM LOC.
;AT TOP OF RAM ADDRESS SPACE
;BRANCH TILL ALL MEMORY TESTED
;//////////////////////////////// END SUBTEST //////////////////////////////////
L10051: TRAP C$ESUB
;IS FATAL ERROR FLAG SET ?
;BRANCH IF NOT
;NO LOOP, TRY TO DROP DEVICE
;SHOULD WE DO ITERATIONS ?
;BRANCH IF NOT
;LOOP UNTIL COUNT EXPIRED
;ALL DONE THIS TEST
TRAP C$EXIT
.LWORD L10047-.

```

```

;*
;LOCAL TEXT MESSAGES FOR TEST
;-

```

```

T$T5ID: .ASCIZ 'Extended Initialization'
T$MEM: .ASCIZ 'Incorrect RAM Data After Init'
T$ADDR: .ASCIZ 'Incorrect RAM Address After Init'
.EVEN
ENDT$T

```

```

L10047: TRAP C$ET$T

```

.SBTTL TEST 6: COMMAND REJECT

```

;
; THIS TEST VERIFIES THAT ALL COMMANDS OTHER THAN WRITE
; CHARACTERISTICS ARE REJECTED DUE TO THE NEED BUFFER ADDRESS
; (NBA) BIT BEING SET IN TSSR, AND THAT THE TSBA AND TSSR
; REGISTERS ARE LEFT IN THE PROPER STATE AFTER EACH COMMAND IS
; REJECTED. THIS TEST CHECKS MICROPROCESSOR SEQUENCING, BASIC
; COMMAND DECODING AND DATI DMA HANDLING. THIS TEST CONTAINS TWO
; SUBTESTS: SUBTEST 1 SEQUENCES THROUGH ALL COMMAND WORDS (OTHER
; THAN WRITE CHARACTERISTICS) WITH THE INTERRUPT ENABLE (IE) BIT
; CLEAR AND VERIFIES THAT AN INTERRUPT IS NOT GENERATED BY THE
; REJECTED COMMAND; SUBTEST 2 PERFORMS SIMILARLY TO SUBTEST 1 BUT
; SETS THE IE BIT IN EACH COMMAND WORD AND VERIFIES THAT AN
; INTERRUPT IS GENERATED WHEN THE COMMAND IS REJECTED. SUBTEST 1
; SETS UP THE INTERRUPT SERVICE ROUTINE TO FLAG UNEXPECTED
; INTERRUPTS. THE COMMAND WORD IN THE COMMAND BUFFER IS
; INITIALIZED TO 100000 (OCTAL) AND THE REMAINING THREE WORDS IN
; THE COMMAND BUFFER ARE SET TO KNOWN UNIQUE PATTERNS. THEN THE
; FOLLOWING SEQUENCE IS PERFORMED:

```

1. INITIALIZE THE CONTROLLER BY WRITING INTO THE TSSR; PROPER INITIAL CONDITIONS ARE VERIFIED.

TEST 6: COMMAND REJECT

976	027604		ERRDF	ERRNO,SFIERR,SFIMSG	;DEVICE FATAL ERROR DURING INIT	TRAP	C#ERDF
	027604	104455				.WORD	601
	027606	001131				.WORD	SFIERR
	027610	003652				.WORD	SFIMSG
	027612	012104					
977	027614	005037	002220	10#: CLR	FATFLG	;CLEAR FATAL ERROR FLAG	
978	027620	005037	002222	CLR	INTRECV	;CLEAR INTERRUPT RECEIVED FLAG	
979	027624	004737	016416	JSR	PC,CHKTSSR	;WAIT FOR READY, NON-AMBIGUOUS	
980	027630	042714	000200	BIC	#BIT7,(R4)	;DISABLE INTERRUPTS	
981	027634	010465	000000	MOV	R4,TSDB(R5)	;SET THE PACKET ADDRESS	
982	027640	004737	016330	JSR	PC,WAITF	;WAIT FOR SSR TO SET	
983	027644	103407		BCS	15#	;BR IF CARRY SET (GOOD RETURN)	
984	027646	010001		MOV	R0,R1	;SAVE CONTENTS OF TSSR	
988	027650			ERRDF	ERRNO,T6SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	027650	104455				TRAP	C#ERDF
	027652	001132				.WORD	602
	027654	030475				.WORD	T6SSR
	027656	012116				.WORD	PKTSSR
989	027660	005237	002220	15#: INC	FATFLG	;SET FATAL ERROR FLAG	
990	027664			CKLOOP		;LOOP ON ERROR, IF FLAG SET	
	027664	104406				TRAP	C#CLP1
991	027666			ESCAPE	SUB	;BY-PASS SUBTEST IF FATAL ERROR	
	027666	104410				TRAP	C#ESCAPE
	027670	000170				.WORD	L10053-
992	027672	005737	002222	TST	INTRECV	;DID AN INTERRUPT OCCUR ?	
993	027676	001404		BEQ	22#	;BRANCH IF NOT	
997	027700			ERRHRD	ERRNO,T6INT,PKTSSR		
	027700	104456				TRAP	C#ERHRD
	027702	001133				.WORD	603
	027704	030553				.WORD	T6INT
	027706	012116				.WORD	PKTSSR
998	027710	012702	102206	22#: MOV	#SC!NBA!SSR!TSREJ,R2	;EXPECTED CONTENTS OF TSSR	
999	027714	004737	016416	JSR	PC,CHKTSSR	;WAIT FOR READY, NON-AMBIGUOUS	
1000	027720	016501	000002	MOV	TSSR(R5),R1	;GET THE CONTENTS OF TSSR	
1001	027724	032701	000100	BIT	#OFL,R1	;IS OFF-LINE BIT SET ?	
1002	027730	001402		BEQ	25#	;BRANCH IF NOT OFF-LINE	
1003	027732	052702	000100	BIS	#OFL,R2	;SET OFF-LINE IN EXPECTED DATA	
1004	027736	020201		25#: CMP	R2,R1	;DOES EXPECTED MATCH RECEIVED ?	
1005	027740	001404		BEQ	30#	;OKAY IF MATCH	
1009	027742			ERRHRD	ERRNO,T6NBA,PKTSSR	;NBA NOT SET TO REJECT	
	027742	104456				TRAP	C#ERHRD
	027744	001134				.WORD	604
	027746	030450				.WORD	T6NBA
	027750	012116				.WORD	PKTSSR
1010	027752			30#: CKLOOP		;LOOP ON ERROR ?	
	027752	104406				TRAP	C#CLP1
1011	027754	004737	016416	JSR	PC,CHKTSSR	;WAIT FOR READY, NON-AMBIGUOUS	
1012	027760	016501	000000	MOV	TSBA(R5),R1	;GET TSBA REGISTER CONTENTS	
1013	027764	010402		MOV	R4,R2	;START OF THE PACKET	
1014	027766	062702	000010	ADD	#10,R2	;EXPECT TSDA TO PACKET + 10	
1015	027772	020102		CMP	R1,R2	;COMPARE EXPECTED TO RECEIVED	
1016	027774	001404		BEQ	35#	;ERROR IF NOT EQUAL	
1020	027776			ERRHRD	ERRNO,T6TSBA,EXPREC	;PRINT THE ERROR & EXPD/RECV	
	027776	104456				TRAP	C#ERHRD
	030000	001135				.WORD	605
	030002	030711				.WORD	T6TSBA
	030004	015554				.WORD	EXPREC

TEST 6: COMMAND REJECT

```

1021
1022
1023 030006 004737 011154      35$: JSR      PC,CKRAM          ;SEE IF DATA IN RAM IS CORRECT
1024 030012 103404             BCS      40$               ;BRANCH IF PACKET IN RAM IS CORRECT
1028 030014             ERRHRD   ERRNO,PKTRAM,AMERR ;REPORT THE RAM ERROR(S)
        030014 104456             TRAP     C$ERHRD
        030016 001136             .WORD   606
        030020 004745             .WORD   PKTRAM
        030022 015570             .WORD   RAMERR
1029 030024             40$: ENDSEG          ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
        030024             10000$: TRAP     C$ESEG
1030 030026 104405             MOV      (R3),R0          ;PACKET COMMAND WORD
1031 030030 042700 177740       BIC      #177740,R0       ;GET BITS 0-4
1032 030034 020027 000004       CMP      R0,#4           ;DON'T TEST WRITE CHARACTERISTICS
1033 030040 001002             BNE      45$             ;BRANCH IF OK
1034 030042 062703 000002       ADD      #2,R3          ;GET NEXT WORD FROM DATA TABLE
1035 030046 020327 003062       45$:   CMP      R3,#TBLEND ;REACHED END OF TABLE ?
1036 030052 103002             BHS      50$             ;BRANCH IF END OF TABLE
1037 030054 000137 027570       JMP      5$              ;CONTINUE TEST WITH NEW DATA
1038
1039 030060             50$: ENDSUB          ;///////////////// END SUBTEST ///////////////////////////////////
        030060             L10053: TRAP     C$ESUB
        030060 104403
1040
1041 030062 005737 002220       TST      FATFLG          ;ANY FATAL ERRORS ?
1042 030066 001402             BEQ      60$             ;BRANCH IF NOT
1043 030070 004737 017262       JSR      PC,CKDROP      ;TRY TO DROP THE UNIT
1044
1045 030074             60$: BGNSUB          ;///////////////// BEGIN SUBTEST ///////////////////
        030074             T6.2: TRAP     C$BSUB
        030074 104402
1046
1047 030076             SETPRI   #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
        030076 012700 000000       MOV      #PRI00,R0       TRAP     C$SPRI
        030102 104441
1048 030104 012704 030440       MOV      #T6PACKET,R4   ;GET THE ADDRESS OF COMMAND PACKET
1049 030110 012703 002752       MOV      #TSTBLK,R3     ;START OF TEST DATA
1050 030114 012314             MOV      (R3)+,(R4)     ;PLACE NEXT DATA WORD IN PACKET
1051 030116             5$:   BGNSEGE          ;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        030116 104404             TRAP     C$BSEG
1052 030120 004737 016054       JSR      PC,SOFINIT     ;DO SOFT INIT OF CONTROLLER
1053 030124 103405             BCS      10$             ;BR IF SOFT INIT = OK
1057 030126 010001             MOV      R0,R1          ;SAVE CONTENTS OF TSSR
1058 030130             ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
        030130 104455             TRAP     C$ERDF
        030132 001137             .WORD   607
        030134 003652             .WORD   SFIERR
        030136 012104             .WORD   SFIMSG
1059 030140 005037 002220       10$:  CLR      FATFLG     ;CLEAR FATAL ERROR FLAG
1060 030144 005037 002222       CLR      INTRECV        ;CLEAR INTERRUPT RECEIVED FLAG
1061 030150 004737 016416       JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
1062 030154 052714 000200       BIS      #BIT7,(R4)     ;ENABLE INTERRUPTS
1063 030160 010465 000000       MOV      R4,TSDB(R5)   ;SET THE PACKET ADDRESS
1064 030164 004737 016330       JSR      PC,WAITF       ;WAIT FOR SSR TO SET
1065 030170 103407             BCS      15$             ;BR IF CARRY SET (GOOD RETURN)
1066 030172 010001             MOV      R0,R1          ;SAVE CONTENTS OF TSSR

```


TEST 6: COMMAND REJECT

```

1113 030354 042700 177740      BIC      #177740,R0      ;GET BITS 0-4
1114 030360 020027 000004      CMP      R0,#4          ;DON'T TEST WRITE CHARACTERISTICS
1115 030364 001002              BNE      45$           ;BRANCH IF NOT WRITE CHARACTERISTICS
1116 030366 062703 000002      ADD      #2,R3         ;BY-PASS WRITE CHARACTERISTICS
1117 030372 020327 003062      45$:    CMP      R3,#TBLEND ;HAVE WE COMPLETED DATA TABLE ?
1118 030376 103002              BHS      50$           ;BRANCH IF ALL TESTED
1119 030400 000137 030114      JMP      5$            ;TEST WITH NEXT DATA
1120
1121 030404              50$:    ENDSUB          ;////////////////// END SUBTEST ////////////////////
      030404              L10054:              TRAP    C$ESUB
      030404 104403
1122 030406 005737 002220      TST      FATFLG        ;ANY FATAL ERRORS ?
1123 030412 001402              BEQ      60$           ;BRANCH IF NOT
1124 030414 004737 017262      JSR      PC,CKDROP     ;TRY TO DROP THE UNIT
1125 030420 004737 016536      60$:    JSR      PC,TSTLOOP ;SHOULD WE DO ITERATIONS ?
1126 030424 103002              BCC      62$           ;BRANCH IF NOT
1127 030426 000137 027550      JMP      T6LOOP        ;LOOP UNTIL COUNT EXPIRED
1128 030432              62$:    EXIT      TST    ;ALL DONE THIS TEST
      030432 104432              TRAP    C$EXIT
      030434 000346              .WORD   L10052-.
1129
1130
1131      ;*
1132      ;LOCAL STORAGE FOR THIS TEST
1133      ;-
1135 030436              .BLKB   10-<.-TSV2&7>
1137 030440      T6PACKET:
1138 030440 000000              .WORD   0              ;COMMAND PACKET FOR TEST
1139 030442 052525              .WORD   052525        ;WILL CONTAIN VARIABLE COMMANDS
1140 030444 125252              .WORD   125252
1141 030446 052525              .WORD   052525
1142
1143
1144
1145      ;*
1146      ;LOCAL TEXT MESSAGES FOR TEST
1147      ;-
1148 030450 103 157 155 T6NBA: .ASCIZ 'Command Not Rejected'
1149 030475 103 157 156 T6SSR: .ASCIZ 'Contents of TSSR Incorrect After Write Packet'
1150 030553 125 156 145 T6INT: .ASCIZ 'Unexpected Interrupt Received On Write Packet'
1151 030631 105 170 160 T6NINT: .ASCIZ 'Expected Interrupt Not Received On Write Packet'
1152 030711 111 156 143 T6TSBA: .ASCIZ 'Incorrect TSBA Address After Packet Write'
1153 030763 103 157 155 TST6ID: .ASCIZ 'Command Reject'
1154
1155 031002              .EVEN
      031002              ENDTST
      031002 104401              L10052:              TRAP    C$ETST
1156
1157      .SBTTL TEST 7: WRITE CHARACTERISTICS
1158
1159
1160      ;
1161      ; THIS TEST VERIFIES BASIC OPERATION OF THE WRITE CHARACTERISTICS
1162      ; COMMAND. IT VERIFIES THAT THE COMMAND BLOCK AND CHARACTERISTICS
1163      ; DATA BLOCK ARE FETCHED PROPERLY FROM CPU MEMORY, THE NEED BUFFER
1164      ; ADDRESS (NBA) BIT IN TSSR IS HANDLED PROPERLY, AND THAT A PROPER
1165      ; MESSAGE PACKET IS STORED, WHERE APPROPRIATE. THIS TEST DOES NOT
      ; CHECK THAT THE VARIOUS FUNCTIONS ENABLED BY CHARACTERISTIC MODE

```

TEST 7: WRITE CHARACTERISTICS

1166	:	:	:	:	:	:						
1167	:	:	:	:	:	:						
1168	:	:	:	:	:	:						
1169	:	:	:	:	:	:						
1170	:	:	:	:	:	:						
1171	:	:	:	:	:	:						
1172	:	:	:	:	:	:						
1173	:	:	:	:	:	:						
1174	:	:	:	:	:	:						
1175	:	:	:	:	:	:						
1176	:	:	:	:	:	:						
1177	:	:	:	:	:	:						
1178	:	:	:	:	:	:						
1179	:	:	:	:	:	:						
1180	:	:	:	:	:	:						
1181	:	:	:	:	:	:						
1182	:	:	:	:	:	:						
1183	:	:	:	:	:	:						
1184	031004	:	:	:	:	:	BGNTST					
	031004	:	:	:	:	:						
1189	031004	012700	034363	:	:	:	MOV	#TST7ID,R0		T7::		
1190	031010	004737	016570	:	:	:	JSR	PC,TSTSETUP			;ASCII MESSAGE TO IDENTIFY TEST	
1191	031014	012737	000024	002214	:	:	MOV	#20.,LOOPCNT			;DO INITIAL TEST SETUP	
1192	031022	:	:	:	:	:					;PERFORM 20 ITERATIONS	
1193	031022	:	:	:	:	:	T7LOOP:	BGNSUB			;;;;;;;;;;;;; BEGIN SUBTEST ;;;;;;;;;;;;;;	
	031022	:	:	:	:	:					T7.1:	
	031022	104402	:	:	:	:					TRAP	C#BSUB
1194	031024	004737	016054	:	:	:	JSR	PC,SOFINIT				
1195	031030	103405	:	:	:	:	BCS	1#				
1199	031032	010001	:	:	:	:	MOV	R0,R1				
1200	031034	:	:	:	:	:	ERRDF	ERRNO,SFIERR,SFIMSG				
	031034	104455	:	:	:	:					TRAP	C#ERDF
	031036	001275	:	:	:	:					.WORD	701
	031040	003652	:	:	:	:					.WORD	SFIERR
	031042	012104	:	:	:	:					.WORD	SFIMSG
1201	031044	:	:	:	:	:	1#:					
1202	031044	012704	045070	:	:	:	MOV	#T11PK2,R4				
1203	031050	004737	010662	:	:	:	JSR	PC,WRTCHR				
1204	031054	103404	:	:	:	:	BCS	2#				
1208	031056	:	:	:	:	:	ERRHRD	ERRNO,WRTMSG,SFIMSG				
	031056	104456	:	:	:	:					TRAP	C#ERHRD
	031060	001276	:	:	:	:					.WORD	702
	031062	005056	:	:	:	:					.WORD	WRTMSG
	031064	012104	:	:	:	:					.WORD	SFIMSG
1209	031066	:	:	:	:	:	2#:					
1210	031066	004737	034412	:	:	:	JSR	PC,T7REST				
1211	:	:	:	:	:	:						
1212	031072	:	:	:	:	:	SETPRI	#PRIO0				
	031072	012700	000000	:	:	:						
	031076	104441	:	:	:	:					MOV	#PRIO0,R0
				:	:	:					TRAP	C#SPRI
1213	031100	012703	002764	:	:	:	MOV	#TSTBLK*10.,R3				
1214	031104	012704	033230	:	:	:	MOV	#T7PACKET,R4				
1215	031110	012764	000010	000006	:	:	MOV	#8.,PKBCNT(R4)				
1216	031116	:	:	:	:	:						
1217	031116	:	:	:	:	:	5#:					
	031116	104404	:	:	:	:	BGNSEG					
				:	:	:						
1218	031120	004737	016054	:	:	:	JSR	PC,SOFINIT				
				:	:	:					TRAP	C#BSEG
				:	:	:						

DATA BITS OPERATE PROPERLY, THE FUNCTIONING OF THESE BITS IS VERIFIED IN SUBSEQUENT TESTS. ALL COMMANDS EXECUTED IN THIS TEST HAVE THE INTERRUPT ENABLE (IE) BIT CLEARED TO ZERO, SO NO INTERRUPTS SHOULD BE GENERATED. HOWEVER, THE PROGRAM RUNS AT PROCESSOR PRIORITY 0, WITH THE INTERRUPT SERVICE ROUTINE SET UP TO FLAG UNEXPECTED INTERRUPTS. IF AN INTERRUPT OCCURS, A PROBLEM EXISTS IN EITHER THE LSI-11 BUS INTERFACE SECTION OR IN THE ROM OR PIPELINE.

THIS TEST CHECKS VARIOUS MICROPROGRAM SEQUENCES, COMMAND DECODING, DMA LOGIC, AND BASIC PACKET PROTOCOL HANDLING. THIS IS THE FIRST TEST IN WHICH DATO DMA CYCLES (FOR STORING THE MESSAGE PACKET) ARE PERFORMED. ANY ERRORS IN THE BODY OF THE TEST (I.E. ERRORS OTHER THAN INITIALIZATION ERRORS RELATED TO THE TRANSPORT BUS) DEFINITELY INDICATE A BAD M7196 MODULE.

TEST 7: WRITE CHARACTERISTICS

Address	Op/Label	Op/Label	Op/Label	Op/Label	Op/Label	Op/Label	Op/Label	Op/Label	Op/Label	Op/Label	
1219	031124	103405		BCS	104						
1223	031126	010001		MOV	RO,R1						
1224	031130			ERRDF	ERRNO,SFIERR,SFIMSG						
	031130	104455							TRAP	C4ERDF	
	031132	001277							.WORD	703	
	031134	003652							.WORD	SFIERR	
	031136	012104							.WORD	SFIMSG	
1225											
1226	031140	005037	002220	104:	CLR	FATFLG					
1227	031144	005037	002222		CLR	INTRECV					
1228	031150	010465	000000		MOV	R4,TSDB(R5)					
1229	031154	004737	016416		JSR	PC,CHKTSSR					
1230	031160	103407			BCS	154					
1231	031162	010001			MOV	RO,R1					
1235	031164				ERRDF	ERRNO,T7SSR,PKTSSR					
	031164	104455							TRAP	C4ERDF	
	031166	001300							.WORD	704	
	031170	034031							.WORD	T7SSR	
	031172	012116							.WORD	PKTSSR	
1236	031174	005237	002220		INC	FATFLG					
1237	031200			154:	CKLOOP						
	031200	104406			ESCAPE	SEG					
1238	031202								TRAP	C4CLP1	
	031202	104410							TRAP	C4ESCAPE	
	031204	000164							.WORD	100004-	
1239	031206	005737	002222		TST	INTRECV					
1240	031212	001404			BEQ	224					
1244	031214				ERRHRD	ERRNO,T7INT,PKTSSR					
	031214	104456							TRAP	C4ERHRD	
	031216	001301							.WORD	705	
	031220	034211							.WORD	T7INT	
	031222	012116							.WORD	PKTSSR	
1245	031224	016501	000002	224:	MOV	TSSR(R5),R1					
1246	031230	012702	000200		MOV	#SSR,R2					
1247	031234	032701	000100		BIT	#OFL,R1					
1248	031240	001402			BEQ	254					
1249	031242	052702	000100		BIS	#OFL,R2					
1250	031246	020201		254:	CMF	R2,R1					
1251	031250	001404			BEQ	304					
1255	031252				ERRHRD	ERRNO,T7NBA,PKTSSR					
	031252	104456							TRAP	C4ERHRD	
	031254	001302							.WORD	706	
	031256	033370							.WORD	T7NBA	
	031260	012116							.WORD	PKTSSR	
1256	031262			304:	CKLOOP						
	031262	104406							TRAP	C4CLP1	
1257	031264	004737	016416		JSR	PC,CHKTSSR					
1258	031270	016501	000000		MOV	TSBA(R5),R1					
1259	031274	012702	033256		MOV	#T7BFR,R2					
1260	031300	022737	000002	002226	CMF	#2,REV					
1261	031306	001404			BEQ	314					
1262	031310	032762	000200	000012	BIT	#BIT7,XST2(R2)					
1263	031316	001405			BEQ	324					
1264	031320	012737	000001	002224	314:	MOV	#1,EXTFEA				
1265	031326	062702	000002		ADD	#2,R2					
1266	031332	062702	000016	324:	ADD	#14,R2					
1267	031336	020102			CMF	R1,R2					

TEST 7: WRITE CHARACTERISTICS

```

1314 031452 004737 016054      JSR      PC,SOFINIT      ;DO SOFT INIT OF CONTROLLER
1315 031456 103405              BCS      10$             ;BR IF SOFT INIT = OK
1319 031460 010001              MOV      R0,R1           ;SAVE CONTENTS OF TSSR
1320 031462              ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
                                TRAP      C$ERDF
                                .WORD    709
                                .WORD    SFIERR
                                .WORD    SFIMSG
                                031462 104455
                                031464 001305
                                031466 003652
                                031470 012104
1321 031472 005037 002222      10$:   CLR      INTRECV       ;CLEAR INTERRUPT RECEIVED FLAG
1322 031476 010400              MOV      R4,R0           ;START OF THE COMMAND PACKET
1323 031500 061300              ADD      (R3),R0        ;OFFSET TO THE DATA WORD TO TEST
1324 031502 056310 000002      BIS      2(R3),(R0)     ;SET THE DATA BITS TO BE TESTED
1325 031506 010465 000000      MOV      R4,TSD8(R5)   ;SET THE PACKET ADDRESS
1326 031512 004737 016330      JSR      PC,WAITF       ;WAIT FOR SSR TO SET
1327 031516 103405              BCS      15$             ;BR IF CARRY SET (GOOD RETURN)
1328 031520 010001              MOV      R0,R1           ;SAVE CONTENTS OF TSSR
1332 031522              ERRDF   ERRNO,T7SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD    710
                                .WORD    T7SSR
                                .WORD    PKTSSR
                                031522 104455
                                031524 001306
                                031526 034031
                                031530 012116
1333 031532              CKLOOP   ;LOOP ON ERROR, IF FLAG SET
1334 031534              ESCAPE  SEG             ;BY-PASS CHECKS IF FATAL ERROR
                                TRAP      C$CLP1
                                .WORD    10000$-
                                031534 104410
                                031536 000116
1335 031540 005737 002222      TST      INTRECV       ;DID AN INTERRUPT OCCUR ?
1336 031544 001404              BEQ     22$             ;BRANCH IF NOT
1340 031546              ERRHRD  ERRNO,T7INT,PKTSSR
                                TRAP      C$ERHRD
                                .WORD    711
                                .WORD    T7INT
                                .WORD    PKTSSR
                                031546 104456
                                031550 001307
                                031552 034211
                                031554 012116
1341 031556 016501 000002      22$:   MOV      TSSR(R5),R1   ;GET THE CONTENTS OF TSSR
1342 031562 012702 102206      MOV      #SC!SSR!TSREJ!NBA,R2 ;EXPECTED CONTENTS OF TSSR
1343 031566 032701 000100      BIT     #OFL,R1        ;IS OFF-LINE BIT SET ?
1344 031572 001402              BEQ     25$             ;BRANCH IF NOT OFF-LINE
1345 031574 052702 000100      BIS     #OFL,R2        ;SET OFF-LINE IN EXPECTED DATA
1346 031600 020201 000100      25$:   CMP      R2,R1        ;DOES EXPECTED MATCH RECEIVED ?
1347 031602 001414              BEQ     30$             ;OKAY IF MATCH
1348 031604 010100              MOV     R1,R0          ;DATA FROM TSSR
1349 031606              XOR     R2,R0          ;FIND BITS IN ERROR
1350 031616 020027 002000      CMP     R0,#NBA       ;IS NBA ONLY BIT IN ERROR ?
1351 031622 001404              BEQ     30$             ;DON'T PRINT ERROR IF NBA ONLY BAD BIT
1355 031624              ERRHRD  ERRNO,T72REJ,PKTSSR ;COMMAND NOT REJECTED
                                TRAP      C$ERHRD
                                .WORD    712
                                .WORD    T72REJ
                                .WORD    PKTSSR
                                031624 104456
                                031626 001310
                                031630 033443
                                031632 012116
1356 031634              CKLOOP   ;LOOP ON ERROR ?
                                TRAP      C$CLP1
                                031634 104406
1357 031636 032701 002000      BIT     #NBA,R1        ;IS NBA BIT SET ?
1358 031642 001004              BNE     35$             ;OKAY IF NBA SET
1362 031644              ERRHRD  ERRNO,T72NBA,PKTSSR ;NBA NOT SET
                                TRAP      C$ERHRD
                                .WORD    713
                                .WORD    T72NBA
                                031644 104456
                                031646 001311
                                031650 033312

```


E11

TEST 7: WRITE CHARACTERISTICS

```

1408 032000                15$:   CKLOOP           ;LOOP ON ERROR, IF FLAG SET
      032000 104406           ESCAPE  SEG          ;BY-PASS SUBTEST IF FATAL ERROR
1409 032002                TRAP    C4CLP1
      032002 104410           ESCAPE  SEG          ;DID AN INTERRUPT OCCUR ?
      032004 000116           TRAP    C4ESCAPE
1410 032006 005737 002222   TST     INTRECV       ;BRANCH IF NOT
1411 032012 001404       BEQ     22$
1415 032014                ERRHRD  ERRNO,T7INT,PKTSSR
      032014 104456           TRAP    C4ERHRD
      032016 001314           .WORD  716
      032020 034211           .WORD  T7INT
      032022 012116           .WORD  PKTSSR
1416 032024 016501 000002   22$:   MOV     TSSR(R5),R1      ;GET THE CONTENTS OF TSSR
1417 032030 012702 102206   MOV     #SC:SSR:TSREJ:NBA,R2   ;EXPECTED CONTENTS OF TSSR
1418 032034 032701 000100   BIT     #OFL,R1                 ;IS OFF-LINE BIT SET ?
1419 032040 001402           BEQ     25$                       ;BRANCH IF NOT OFF-LINE
1420 032042 052702 000100   BIS     #OFL,R2                 ;SET OFF-LINE IN EXPECTED DATA
1421 032046 020201           CMP     R2,R1                     ;DOES EXPECTED MATCH RECEIVED ?
1422 032050 001414       25$:   BEQ     30$                       ;OKAY IF MATCH
1423 032052 010100       MOV     R1,R0                     ;DATA FROM TSSR
1424 032054           XOR     R2,R0                     ;FIND BITS IN ERROR
1425 032064 020027 002000   CMP     R0,#NBA                 ;IS NBA ONLY BIT IN ERROR ?
1426 032070 001404       BEQ     30$                       ;DON'T PRINT ERROR IF NBA ONLY BAD BIT
1430 032072                ERRHRD  ERRNO,T73REJ,PKTSSR   ;COMMAND NOT REJECTED
      032072 104456           TRAP    C4ERHRD
      032074 001315           .WORD  717
      032076 033542           .WORD  T73REJ
      032100 012116           .WORD  PKTSSR
1431 032102                30$:   CKLOOP           ;LOOP ON ERROR ?
      032102 104406           TRAP    C4CLP1
1432 032104 032701 002000   BIT     #NBA,R1                 ;IS NBA BIT SET ?
1433 032110 001004       BNE     35$                       ;OKAY IF NBA SET
1437 032112                ERRHRD  ERRNO,T72NBA,PKTSSR   ;NBA NOT SET
      032112 104456           TRAP    C4ERHRD
      032114 001316           .WORD  718
      032116 033312           .WORD  T72NBA
      032120 012116           .WORD  PKTSSR
1438 032122                35$:
1439 032122                ENDSEG          ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
      032122 104405           10000$: TRAP    C4ESEG
1440 032124 005203           INC     R3
1441 032126 020327 000006   CMP     R3,#6
1442 032132 002002       BGE     57$
1443 032134 000137 031712   JMP     5$
1444 032140                57$:   ENDSUB          ;////////////////// END SUBTEST ////////////////////
      032140 104403           L10060: TRAP    C4ESUB
1447
1448
1449
1450
1451
1452
1453
;
```

```

;*
;
;TEST 7, SUBTEST 4
;
;SUBTEST TO VERIFY THAT A WRITE CHARACTERISTICS COMMAND IS
;REJECTED IF AN ILLEGAL DATA BLOCK ADDRESS IS ISSUED.
```

TEST 7: WRITE CHARACTERISTICS

```

1454
1455
1456
1457 032142          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
                                T7.4:          TRAP      C#BSUB
1458
1459 032144          SETPRI #PRI00      ;LOWER PRIORITY TO ALLOW INTERRUPTS
                                MOV #PRI00,R0
                                TRAP      C#SPRI
1460 032152 012700 000000
1461 032156 012704 033230      5$: MOV #T72DATA,R3      ;START OF TEST DATA FOR SUBTEST
1462 032162 004737 034412      MOV #T7PACKET,R4    ;GET THE ADDRESS OF COMMAND PACKET
1463
1464
1465 032166 004737 016054      JSR PC,SOFINIT     ;DO SOFT INIT OF CONTROLLER
1466 032172 103405          BCS 10$           ;BR IF SOFT INIT = OK
1470 032174 010001          MOV R0,R1         ;SAVE CONTENTS OF TSSR
1471 032176          ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
                                TRAP      C#ERDF
                                .WORD    719
                                .WORD    SFIERR
                                .WORD    SFIMSG
1472 032206 005037 002222      10$: CLR INTRECV      ;CLEAR INTERRUPT RECEIVED FLAG
1473 032212 052737 000001      BIS #1,T7DATA     ;MAKE ADDRESS ODD
1474 032220 010465 000000      MOV R4,TSDB(R5)   ;SET THE PACKET ADDRESS
1475 032224 004737 016330      JSR PC,WAITF      ;WAIT FOR SSR TO SET
1476 032230 103405          BCS 15$           ;BR IF CARRY SET (GOOD RETURN)
1477 032232 010001          MOV R0,R1         ;SAVE CONTENTS OF TSSR
1481 032234          ERRDF ERRNO,T7SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C#ERDF
                                .WORD    720
                                .WORD    T7SSR
                                .WORD    PKTSSR
1482 032244          15$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
1483 032246          ESCAPE SUB          ;BY-PASS SUBTEST IF FATAL ERROR
                                TRAP      C#CLP1
                                .WORD    C#ESCAPE
                                .WORD    L10061-.
1484 032252 005737 002222      TST INTRECV       ;DID AN INTERRUPT OCCUR ?
1485 032256 001404          BEQ 22$          ;BRANCH IF NOT
1489 032260          ERRHRD ERRNO,T7INT,PKTSSR
                                TRAP      C#ERHRD
                                .WORD    721
                                .WORD    T7INT
                                .WORD    PKTSSR
1490 032270 016501 000002      22$: MOV TSSR(R5),R1   ;GET THE CONTENTS OF TSSR
1491 032274 012702 102206      MOV #SC!SSR!TSREJ!NBA,R2 ;EXPECTED CONTENTS OF TSSR
1492 032300 032701 000100      BIT #OFL,R1       ;IS OFF-LINE BIT SET ?
1493 032304 001402          BEQ 25$          ;BRANCH IF NOT OFF-LINE
1494 032306 052702 000100      BIS #OFL,R2       ;SET OFF-LINE IN EXPECTED DATA
1495 032312 020201          CMP R2,R1        ;DOES EXPECTED MATCH RECEIVED ?
1496 032314 001414          BEQ 30$          ;OKAY IF MATCH
1497 032316 010100          MOV R1,R0        ;DATA FROM TSSR
1498 032320          XOR R2,R0        ;FIND BITS IN ERROR
1499 032330 020027 002000      CMP R0,#NBA      ;IS NBA ONLY BIT IN ERROR ?
1500 032334 001404          BEQ 30$          ;DON'T PRINT ERROR IF NBA ONLY BAD BIT

```


TEST 7: WRITE CHARACTERISTICS

```

1637 032776 005737 002222          TST      INTRECV          ;DID AN INTERRUPT OCCUR ?
1638 033002 001404                  BEQ      22$              ;BRANCH IF NOT
1642 033004                      ERRHRD  ERRNO,T7INT,PKTSSR
                                     TRAP      C$ERHRD
                                     .WORD    731
1643 033014 016501 000002          22$:    MOV      TSSR(R5),R1      ;GET THE CONTENTS OF TSSR
1644 033020 012702 000200          MOV      #SSR,R2          ;EXPECTED CONTENTS OF TSSR
1645 033024 032701 000100          BIT      #OFL,R1         ;IS OFF-LINE BIT SET ?
1646 033030 001402                  BEQ      25$              ;BRANCH IF NOT OFF-LINE
1647 033032 052702 000100          BIS      #OFL,R2         ;SET OFF-LINE IN EXPECTED DATA
1648 033036 020201                  CMP      R2,R1           ;DOES EXPECTED MATCH RECEIVED ?
1649 033040 001404                  BEQ      30$              ;OKAY IF MATCH
1653 033042                      ERRHRD  ERRNO,T7NBA,PKTSSR ;NBA NOT ZERO
                                     TRAP      C$ERHRD
                                     .WORD    732
1654 033052                      CKLOOP
                                     .WORD    T7NBA
1655 033054 004737 016416          30$:    JSR      PC,CHKTSSR      ;LOOP ON ERROR ?
1656 033060 016501 000000          MOV      TSBA(R5),R1      ;WAIT FOR READY, NON-AMBIGUOUS
1657 033064 012702 033256          MOV      #T7BFR,R2      ;GET TSBA REGISTER CONTENTS
1658 033070 022737 000002          CMP      #2,REV          ;START OF THE DATA BUFFER
1659 033076 001404 000002          BEQ      31$              ;IS IT A NEW MICROCODE
1660 033100 032762 000200          BIT      #BIT7,XST2(R2)  ;YES BR
1661 033106 001402 000012          BEQ      32$              ;IS EXTENDED FEATURES BIT SET ?
1662 033110 062702 000002          ADD      #2,R2           ;BRANCH IF EXTENDED FEATURES OFF
1663 033114 062702 000016          ADD      #14,R2          ;EXTRA WORD IF SPECIAL FEATURES
1664 033120 020102                  CMP      R1,R2           ;EXPECTED CONTENTS OF TSBA
1665 033122 001404                  BEQ      35$              ;COMPARE EXPECTED TO RECEIVED
1669 033124                      ERRHRD  ERRNO,T7TSBA,EXPREC ;ERROR IF NOT EQUAL
1670 033124 104456                  TRAP    C$ERHRD          ;PRINT THE ERROR & EXPD/RECV
1671 033126 001335                  .WORD  733
1672 033130 034300                  .WORD  T7TSBA
1673 033132 015554                  .WORD  EXPREC
1672 033134 012704 033240          35$:    MOV      #T7DATA,R4      ;SET POINTER FOR CHECKER
1673 033140 004737 011264          JSR      PC,CKRAM2      ;SEE IF DATA IN RAM IS CORRECT
1674 033144 103404                  BCS     40$              ;BRANCH IF PACKET IN RAM IS CORRECT
1678 033146                      ERRHRD  ERRNO,PKTRAM,RAMERR ;REPORT THE RAM ERROR(S)
1679 033146 104456                  TRAP    C$ERHRD
1680 033150 001336                  .WORD  734
1681 033152 004745                  .WORD  PKTRAM
1682 033154 015570                  .WORD  RAMERR
1680 033156 012704 033230          40$:    MOV      #T7PACKET,R4  ;RESET PACKET POINTER
1681 033162                      ENDSEG
1682 033162 104405                  10000$: TRAP    C$ESEG
1683 033164 012364 000006          MOV      (R3)+,PKBCNT(R4) ;SET THE TEST WORD
1684 033170 020327 003062          CMP      R3,#TBLEND     ;HAS ALL DATA BEEN TESTED ?
1685 033174 103002                  BHIS   55$              ;BRANCH IF ALL DATA DONE
1686 033176 000137 032706          JMP     5$              ;BRANCH TILL BACK TO ZERO

```

TEST 7: WRITE CHARACTERISTICS

```

1687
1688 033202          55$:  ENDSUB          ;////////////////// END SUBTEST ////////////////////
      033202          L10063:          TRAP      C#ESUB
      033202 104403
1689
1690 033204 005737 002220          TST      FATFLG          ;ANY FATAL ERRORS ?
1691 033210 001402          BEQ      60$              ;BRANCH IF NOT
1692 033212 004737 017262          JSR      PC,CKDROP       ;TRY TO DROP THE UNIT
1693 033216
1694 033216          60$:  EXIT      TST          ;ALL DONE THIS TEST
      033216 104432          TRAP      C#EXIT
      033220 001240          .WORD    L10055-.
1695
1696
1697          ;*
1698          ;LOCAL STORAGE FOR THIS TEST
1699          ;-
1701 033222          .BLKB    10-<.-TSV2&7>
1703 033230          T7PACKET: .WORD    100004          ;COMMAND PACKET FOR TEST
1704 033230 100004          .WORD    T7DATA          ;WRITE CHARACTERISTICS COMMAND, WITH ACK
1705 033232 033240          .WORD    0              ;ADDRESS OF CHARACTERISTICS BLOCK
1706 033234 000000          .WORD    8.             ;STARTING VALUE OF BLOCK SIZE
1707 033236 000010
1708
1709 033240          T7DATA:  .WORD    T7BFR          ;CHARACTERISTICS DATA BLOCK
1710 033240 033256          .WORD    0              ;ADDRESS OF MESSAGE BUFFER
1711 033242 000000          .WORD    14.           ;LENGTH OF MESSAGE BUFFER
1712 033244 000016          .WORD    0              ;EXTFEA EXTRA WORD
1713 033246 000000          T7SP:   .WORD    0
1714 033250 000000
1715
1716 033252 000000 000000          T7BFR:  .WORD    0,0          ;SPACE
1717 033256          .BLKW    8.             ;MESSAGE BUFFER
1718
1719          ;*
1720          ;TEST DATA FOR SUBTEST TWO
1721          ;DATA HAS FORMAT:
1722          ;
1723          ;          1ST WORD          OFFSET TO TEST WORD IN PACKET
1724          ;          2ND WORD          BITS TO SET FOR TEST
1725          ;
1726          ;-
1727
1728
1729
1730 033276          T72DATA: .WORD    0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
1731 033276 000000 037140          .WORD    2,BIT0
1732 033302 000002 000001          .WORD    4,BIT6!BIT15
1733 033306 000004 100100          T72DONE=.
1734 033312 033312
1735
1736
1737          ;*
1738          ;LOCAL TEXT MESSAGES FOR TEST
1739          ;-
1740
1741 033312          116      102      101  T72NBA: .ASCIZ 'NBA Not Set On Rejected WRITE CHARACTERISTICS'

```

TEST 7: WRITE CHARACTERISTICS

1742	033370	127	122	111	T7NBA:	.ASCIZ	'WRITE CHARACTERISTICS Command Not Accepted'
1743	033443	127	122	111	T72REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
1744	033542	127	122	111	T73REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
1745	033635	127	122	111	T74REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
1746	033733	127	122	111	T75REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
1747	034031	103	157	156	T7SSR:	.ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
1748	034120	105	170	160	T7NINT:	.ASCIZ	'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
1749	034211	125	156	145	T7INT:	.ASCIZ	'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
1750	034300	111	156	143	T7TSBA:	.ASCIZ	'Incorrect TSBA Address After WRITE CHARACTERISTICS'
1751	034363	127	162	151	TST7ID:	.ASCIZ	'Write Characteristics'
1752						.EVEN	

1753
1754
1755
1756
1757
1758
1759
1760

```

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

```

1761	034412				T7REST:		
1762	034412				SAVREG		
1763	034416	012701	033230		MOV	@T7PACKET,R1	;SAVE THE REGISTERS
1764	034422	012721	100004		MOV	@100004,(R1)+	;START OF THE PACKET
1765	034426	012721	033240		MOV	@T7DATA,(R1)+	;WRITE CHARACTERISTICS WITH ACK
1766	034432	005021			CLR	(R1)+	;ADDRESS OF CHAR DATA BLOCK
1767	034434	012721	000010		MOV	@8,(R1)+	;EXTENDED ADDRESS
1768	034440	012721	033256		MOV	@T7BFR,(R1)+	;SIZE OF DATA BLOCK IN BYTES
1769	034444	005021			CLR	(R1)+	;ADDRESS OF MESSAGE BUFFER
1770	034446	012721	000020		MOV	@16,(R1)+	;LENGTH OF MESSAGE BUFFER
1771	034452	005021			CLR	(R1)+	
1772	034454	005011			CLR	(R1)	
1773	034456	000207			RTS	PC	;RETURN
1774	034460				ENDTST		
	034460						
	034460	104401					

L10055: TRAP C#ETST

1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796

.SBTTL TEST 8: VOLUME CHECK

```

; THIS TEST VERIFIES THAT THE VOLUME CHECK (VCK) BIT, A FLAG HELD
; WITHIN THE M7196 AND APPEARING IN XSTO, IS SET BY INITIALIZE AND
; CLEARED BY EXECUTING A WRITE CHARACTERISTICS COMMAND WITH THE
; CVC BIT SET. IT IS ALSO VERIFIED THAT A WRITE CHARACTERISTICS
; COMMAND WITH THE CVC BIT CLEAR DOES NOT AFFECT THE STATE OF THE
; VOLUME CHECK BIT. THE ACTUAL FUNCTION OF VOLUME CHECK, THAT OF
; PREVENTING OR ALLOWING A TAPE MOTION COMMAND DEPENDING UPON
; WHETHER VOLUME CHECK IS SET OR CLEAR, IS NOT CHECKED BY THIS
; TEST; THIS FUNCTIONALITY IS CHECKED IN THE INDIVIDUAL TESTS OF
; TAPE MOTION COMMANDS.

```

THE TEST PROCEEDS AS FOLLOWS:

1. THE CONTROLLER IS INITIALIZED BY WRITING INTO THE TSSR.
2. A WRITE CHARACTERISTICS COMMAND IS ISSUED (WITH CVC=0) AND XSTO IN THE RETURNED MESSAGE BUFFER IS EXAMINED;

TEST 8: VOLUME CHECK

```

1797          ;           THE VCK BIT SHOULD BE CLEAR (0).
1798          ;
1799          ;           3. THE PREVIOUS STEP IS REPEATED TO VERIFY THAT VCK DOES
1800          ;           NOT CHANGE (REMAINS AT 0).
1801          ;
1802          ;           4. A WRITE CHARACTERISTICS COMMAND IS ISSUED WITH CVC=1
1803          ;           AND THE VCK BIT IN XSTO IN THE MESSAGE BUFFER IS
1804          ;           EXAMINED; THE VCK BIT SHOULD BE CLEAR (0).
1805          ;
1806          ;           5. A WRITE CHARACTERISTICS COMMAND IS ISSUED WITH CVC=0
1807          ;           AND THE VCK BIT IN XSTO IN THE MESSAGE BUFFER IS
1808          ;           EXAMINED; THE VCK BIT SHOULD REMAIN CLEAR (0).
1809          ;
1810          ;
1811 034462      BGNTST
1811 034462
1816 034462    012700 035347      MOV     #TST8ID,R0      ;ASCII MESSAGE TO IDENTIFY TEST
1817 034466    004737 016570      JSR     PC,TSTSETUP    ;DO INITIAL TEST SETUP
1818 034472    012737 000024 002214  MOV     #20.,LOOPCNT   ;PERFORM 20 ITERATIONS
1819 034500      T8LOOP:
1820
1821 034500    012704 035070      MOV     #T8PACKET,R4   ;PACKET FOR WRITE CHARACTERISTICS
1822 034504    004737 016054      JSR     PC,SOFINIT     ;DO SOFT INIT OF CONTROLLER
1823 034510    103405              BCS     10$,           ;BR IF SOFT INIT = OK
1827 034512    010001              MOV     R0,R1          ;SAVE CONTENTS OF TSSR
1828 034514      ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
1828 034514    104455              TRAP   C#ERDF
1828 034516    001441              .WORD 801
1828 034520    003652              .WORD SFIERR
1828 034522    012104              .WORD SFIMSG
1829 034524    042714 040000    10$:   BIC     #BIT14,(R4)   ;CLEAR THE CVC BIT
1830 034530    010465 000000      MOV     R4,TSDB(R5)   ;SET THE PACKET ADDRESS FOR WRITE CHAR
1831 034534    004737 016416      JSR     PC,CHKTSSR    ;WAIT FOR SSR TO SET
1832 034540    103405              BCS     15$,           ;BR IF CARRY SET (GOOD RETURN)
1833 034542    010001              MOV     R0,R1          ;SAVE CONTENTS OF TSSR
1837 034544      ERRDF  ERRNO,T8SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
1837 034544    104455              TRAP   C#ERDF
1837 034546    001442              .WORD 802
1837 034550    035260              .WORD T8SSR
1837 034552    012116              .WORD PKTSSR
1838 034554    104406    15$:   CKLOOP ;LOOP ON ERROR, IF FLAG SET
1839 034556      ESCAPE TST          ;EXIT IF FATAL ERROR
1839 034556    104410              TRAP   C#CLP1
1839 034560    000604              .WORD L10064-.
1840 034562    012702 035112      MOV     #T8BFR,R2     ;ADDRESS OF THE MESSAGE BUFFER
1841 034566    032762 000020 000006  BIT     #XSOVCK,XSTO(R2) ;IS VOLUME CHECK SET IN XSTO ?
1842 034574    001406              BEQ     20$,           ;OKAY IF VOLUME CHECK IS CLEAR
1846 034576    016501 000002      MOV     TSSR(R5),R1   ;CONTENTS OF TSSR FOR ERROR REPORT
1847 034602      ERRHRD ERRNO,T8NVCK,PKTMES ;VOLUME CHECK NOT CLEAR
1847 034602    104456              TRAP   C#ERHRD
1847 034604    001443              .WORD 803
1847 034606    035167              .WORD T8NVCK
1847 034610    012160              .WORD PKTMES
1848 034612    104406    20$:   CKLOOP ;LOOP ON ERROR ?
1849 034614    010465 000000      MOV     R4,TSDB(R5)   ;SET THE PACKET ADDRESS FOR WRITE CHAR

```

N11

TEST 8: VOLUME CHECK

1850	034620	004737	016416		JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET	
1851	034624	103405			BCS	25		;BR IF CARRY SET (GOOD RETURN)	
1852	034626	010001			MOV	R0,R1		;SAVE CONTENTS OF TSSR	
1856	034630				ERRDF	ERRNO,T8SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET	
	034630	104455						TRAP	C4ERDF
	034632	001444						.WORD	804
	034634	035260						.WORD	T8SSR
	034636	012116						.WORD	PKTSSR
1857	034640			25:	CKLOOP			;LOOP ON ERROR, IF FLAG SET	
	034640	104406						TRAP	C4CLP1
1858	034642				ESCAPE	TST		;EXIT IF FATAL ERROR	
	034642	104410						TRAP	C4ESCAPE
	034644	000520						.WORD	L10064--
1859	034646	032762	000020	000006	BIT	#XSOVCK,XSTO(R2)		;IS VOLUME CHECK SET IN XSTO ?	
1860	034654	001406			BEQ	30		;OKAY IF VOLUME CHECK IS SET	
1864	034656	016501	000002		MOV	TSSR(R5),R1		;CONTENTS OF TSSR FOR ERROR REPORT	
1865	034662				ERRHRD	ERRNO,T8NVCK,PKTMES		;VOLUME CHECK NOT SET	
	034662	104456						TRAP	C4ERHRD
	034664	001445						.WORD	805
	034666	035167						.WORD	T8NVCK
	034670	012160						.WORD	PKTMES
1866	034672			30:	CKLOOP			;LOOP ON ERROR ?	
	034672	104406						TRAP	C4CLP1
1867	034674	052714	040000		BIS	#BIT14,(R4)		;SET THE CVC BIT	
1868	034700	010465	000000		MOV	R4,TSDB(R5)		;SET THE PACKET ADDRESS FOR WRITE CHAR	
1869	034704	004737	016416		JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET	
1870	034710	103405			BCS	35		;BR IF CARRY SET (GOOD RETURN)	
1871	034712	010001			MOV	R0,R1		;SAVE CONTENTS OF TSSR	
1875	034714				ERRDF	ERRNO,T8SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET	
	034714	104455						TRAP	C4ERDF
	034716	001446						.WORD	806
	034720	035260						.WORD	T8SSR
	034722	012116						.WORD	PKTSSR
1876	034724			35:	CKLOOP			;LOOP ON ERROR, IF FLAG SET	
	034724	104406						TRAP	C4CLP1
1877	034726				ESCAPE	TST		;EXIT IF FATAL ERROR	
	034726	104410						TRAP	C4ESCAPE
	034730	000434						.WORD	L10064--
1878	034732	032762	000020	000006	BIT	#XSOVCK,XSTO(R2)		;IS VOLUME CHECK CLEAR IN XSTO ?	
1879	034740	001406			BEQ	40		;OKAY IF VOLUME CHECK IS CLEARED	
1883	034742	016501	000002		MOV	TSSR(R5),R1		;CONTENTS OF TSSR FOR ERROR REPORT	
1884	034746				ERRHRD	ERRNO,T8VCK,PKTMES		;VOLUME CHECK NOT CLEARED	
	034746	104456						TRAP	C4ERHRD
	034750	001447						.WORD	807
	034752	035132						.WORD	T8VCK
	034754	012160						.WORD	PKTMES
1885	034756			40:	CKLOOP			;LOOP ON ERROR ?	
	034756	104406						TRAP	C4CLP1
1886	034760	042714	040000		BIC	#BIT14,(R4)		;CLEAR THE CVC BIT	
1887	034764	010465	000000		MOV	R4,TSDB(R5)		;SET THE PACKET ADDRESS FOR WRITE CHAR	
1888	034770	004737	016416		JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET	
1889	034774	103405			BCS	45		;BR IF CARRY SET (GOOD RETURN)	
1890	034776	010001			MOV	R0,R1		;SAVE CONTENTS OF TSSR	
1894	035000				ERRDF	ERRNO,T8SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET	
	035000	104455						TRAP	C4ERDF
	035002	001450						.WORD	808
	035004	035260						.WORD	T8SSR

TEST 8: VOLUME CHECK

```

1895 035006 012116
035010 104406
1896 035012 104410
035014 000350
1897 035016 032762 000020 000006
1898 035024 001406
1902 035026 016501 000002
1903 035032 104456
035034 001451
035036 035132
035040 012160
1904 035042 50$: CKLOOP
035044 104406 50$: CKLOOP
1905 035044 004737 016536 60$: JSR PC,TSTLOOP
1906 035050 103002 60$: BCC 62$
1907 035052 000137 034500 60$: JMP T8LOOP
1908 035056 104432 62$: EXIT TST
035056 000304

1909
1910
1911 ;*
1912 ;LOCAL STORAGE FOR THIS TEST
1913 ;-
1915 035062
1917 035070
1918 035070 100004
1919 035072 035100
1920 035074 000000
1921 035076 000010
1922
1923 035100
1924 035100 035112
1925 035102 000000
1926 035104 000020
1927 035106 000000 000000
1928
1929 035112
1930
1931
1932
1933 ;*
1934 ;LOCAL TEXT MESSAGES FOR TEST
1935 ;-
1936 035132 126 157 154 T8VCK: .ASCIZ 'Volume Check Bit Not Cleared'
1937 035167 126 157 154 T8NVCK: .ASCIZ 'Volume Check Bit (VCK) Not Clear After Initialize (XST0)'
1938 035260 103 157 156 T8SSR: .ASCIZ 'Contents of TSSR Incorrect After Write Characteristics'
1939 035347 126 157 154 T8TID: .ASCIZ 'Volume Check'
1940
1941 035364
035364
035364 104401
1942
1943

```

```

;LOOP ON ERROR, IF FLAG SET .WORD PKTSSR
;EXIT IF FATAL ERROR TRAP C#CLP1
;IS VOLUME CHECK CLEAR IN XST0 ? TRAP C#ESCAPE
;OKAY IF VOLUME CHECK IS CLEARED .WORD L10064-.
;CONTENTS OF TSSR FOR ERROR REPORT
;VOLUME CHECK NOT CLEARED
;LOOP ON ERROR ? TRAP C#ERHRD
;SHOULD WE DO ITERATIONS ? .WORD 809
;BRANCH IF NOT .WORD T8VCK
;LOOP UNTIL COUNT EXPIRED .WORD PKTMES
;ALL DONE THIS TEST TRAP C#EXIT
;WORD L10064-.

```

```

;COMMAND PACKET FOR TEST
;WRITE CHARACTERISTICS COMMAND
;ADDRESS OF CHARACTERISTICS BLOCK
;STARTING VALUE OF COUNTER
;CHARACTERISTICS DATA BLOCK
;ADDRESS OF MESSAGE BUFFER
;LENGTH OF MESSAGE BUFFER
;MESSAGE BUFFER

```

L10064: TRAP C#ETST

.SBTTL TEST 9: COMPLETION INTERRUPT

TEST 9: COMPLETION INTERRUPT

```

036306 012700 000000
036312 104441
2182 036314 012703 037412
2183 036320 012704 037350
2184 036324 004737 040446
2185
2186
2187 036330 004737 016054
2188 036334 103405
2192 036336 010001
2193 036340
036340 104455
036342 001621
036344 003652
036346 012104
2194 036350 005037 002222 037360 10+: CLR INTRECV
2195 036354 052737 000001 BIS #1,T9DATA
2196 036362 010465 000000 MOV R4,TSD8(R5)
2197 036366 004737 016330 JSR PC,WAITF
2198 036372 103405 BCS 15+
2199 036374 010001 MOV R0,R1
2203 036376 ERRDF ERRNO,T9SSR,PKTSSR
036376 104455
036400 001622
036402 040067
036404 012116
2204 036406 15+: CKLOOP
036406 104406 ESCAPE SUB
2205 036410
036410 104410
036412 000056
2206 036414 005737 002222 TST INTRECV
2207 036420 001004 BNE 22+
2211 036422 ERRHRD ERRNO,T9NINT,PKTSSR
036422 104456
036424 001623
036426 040156
036430 012116
2212 036432 016501 000002 22+: MOV TSSR(R5),R1
2213 036436 012702 102206 MOV #SC!SSR!TSREJ!NBA,R2
2214 036442 032701 000100 BIT #OFL,R1
2215 036446 001402 BEQ 25+
2216 036450 052702 000100 BIS #OFL,R2
2217 036454 020201 25+: CMP R2,R1
2218 036456 001404 BEQ 30+
2222 036460 ERRHRD ERRNO,T94REJ,PKTSSR
036460 104456
036462 001624
036464 037673
036466 012116
2223 036470 30+:
2224
2225 036470 ENDSUB
036470
036470 104403
2226
2227

```

```

MOV #PRI00,R0
TRAP C#SPRI
;START OF TEST DATA FOR SUBTEST
;GET THE ADDRESS OF COMMAND PACKET
;RESTORE PACKET TO STARTING VALUES
;DO SOFT INIT OF CONTROLLER
;BR IF SOFT INIT = OK
;SAVE CONTENTS OF TSSR
;DEVICE FATAL ERROR DURING INIT
TRAP C#ERDF
.WORD 913
.WORD SFIERR
.WORD SFIMSG
;CLEAR INTERRUPT RECEIVED FLAG
;MAKE ADDRESS ODD
;SET THE PACKET ADDRESS
;WAIT FOR SSR TO SET
;BR IF CARRY SET (GOOD RETURN)
;SAVE CONTENTS OF TSSR
;DEVICE FATAL SSR FAILED TO SET
TRAP C#ERDF
.WORD 914
.WORD T9SSR
.WORD PKTSSR
;LOOP ON ERROR, IF FLAG SET
;BY-PASS SUBTEST IF FATAL ERROR
TRAP C#CLP1
.WORD C#ESCAPE
.L10071-.
;DID AN INTERRUPT OCCUR ?
;BRANCH IF YES
TRAP C#ERHRD
.WORD 915
.WORD T9NINT
.WORD PKTSSR
;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
;DOES EXPECTED MATCH RECEIVED ?
;OKAY IF MATCH
;COMMAND NOT REJECTED
TRAP C#ERHRD
.WORD 916
.WORD T94REJ
.WORD PKTSSR
;////////// END SUBTEST //////////
L10071: TRAP C#ESUB

```


TEST 9: COMPLETION INTERRUPT

```

2367
2368
2369
2370
2371
2372 037130          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      037130          T9.7:          TRAP      C#BSUB
      037130 104402
2373
2374 037132          SETPRI  #PRI00      ;LOWER PRIORITY TO ALLOW INTERRUPTS
      037132 012700 000000          MOV      #PRI00,R0
      037136 104441          TRAP      C#SPRI
2375 037140 012704 037350          MOV      #T9PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
2376 037144 004737 040446          JSR      PC,T9REST      ;SET UP A VALID PACKET
2377 037150 004737 016054          JSR      PC,SOFINIT     ;DO SOFT INIT OF CONTROLLER
2378 037154 103405          BCS     10#           ;BR IF SOFT INIT = OK
2382 037156 010001          MOV      R0,R1      ;SAVE CONTENTS OF TSSR
2383 037160          ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      037160 104455          TRAP      C#ERDF
      037162 001635          .WORD   925
      037164 003652          .WORD   SFIERR
      037166 012104          .WORD   SFIMSG
2384 037170 005037 002222          10# : CLR      INTRECV      ;CLEAR INTERRUPT RECEIVED FLAG
2385 037174 052714 000200          BIS     #BIT7,(R4)    ;ENABLE INTERRUPTS
2386 037200 010465 000000          MOV     R4,TSDB(R5)   ;SET THE PACKET ADDRESS
2387 037204 004737 016416          JSR     PC,CHKTSSR    ;WAIT FOR SSR TO SET
2388 037210 103405          BCS     15#           ;BR IF CARRY SET (GOOD RETURN)
2389 037212 010001          MOV     R0,R1      ;SAVE CONTENTS OF TSSR
2393 037214          ERRDF  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      037214 104455          TRAP      C#ERDF
      037216 001636          .WORD   926
      037220 040067          .WORD   T9SSR
      037222 012116          .WORD   PKTSSR
2394 037224          15# : CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      037224 104406          TRAP      C#CLP1
2395 037226          ESCAPE  SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      037226 104410          TRAP      C#ESCAPE
      037230 000102          .WORD   L10074-.
2396 037232 005737 002222          TST     INTRECV      ;DID AN INTERRUPT OCCUR ?
2397 037236 001004          BNE     22#           ;BRANCH IF YES
2401 037240          ERRHRD  ERRNO,T9NINT,PKTSSR
      037240 104456          TRAP      C#ERHRD
      037242 001637          .WORD   927
      037244 040156          .WORD   T9NINT
      037246 012116          .WORD   PKTSSR
2402 037250          22# : CKLOOP          ;LOOP ON ERROR ?
      037250 104406          TRAP      C#CLP1
2403
2404 037252 005037 002222          CLR     INTRECV      ;CLEAR INTERRUPT RECEIVED FLAG
2405 037256 042714 000200          BIC     #BIT7,(R4)    ;DISABLE INTERRUPTS
2406 037262 010465 000000          MOV     R4,TSDB(R5)   ;SET THE PACKET ADDRESS
2407 037266 004737 016416          JSR     PC,CHKTSSR    ;WAIT FOR SSR TO SET
2408 037272 103405          BCS     25#           ;BR IF CARRY SET (GOOD RETURN)
2409 037274 010001          MOV     R0,R1      ;SAVE CONTENTS OF TSSR
2413 037276          ERRDF  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      037276 104455          TRAP      C#ERDF
      037300 001640          .WORD   928

```

TEST 9: COMPLETION INTERRUPT

```

037302 040067
037304 012116
2414 037306 25$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
037306 104406 ;BY-PASS SUBTEST IF FATAL ERROR
2415 037310 ESCAPE SUB ;DID AN INTERRUPT OCCUR ?
037310 104410 ;BRANCH IF NOT
037312 000020
2416 037314 005737 002222 TST INTRECV
2417 037320 001404 BEQ 30$
2421 037322 ERRHRD ERRNO,T9INT,PKTSSR
037322 104456
037324 001641
037326 040247
037330 012116
2422 037332 30$:
2423 037332 ENDSUB ;////////////////// END SUBTEST ////////////////////
037332 L10074:
037332 104403 TRAP C#ERHRD
2424 037334 EXIT TST ;ALL DONE THIS TEST
2425 037334 104432 TRAP C#EXIT
037336 001162 .WORD L10065-.

;+
;LOCAL STORAGE FOR THIS TEST
;-
2430
2432 037340 T9PACKET: .BLKB 10-<.-TSV2&7>
2434 037350 .WORD 100204 ;COMMAND PACKET FOR TEST
2435 037350 100204 .WORD T9DATA ;WRITE CHAR COMMAND, WITH IE, ACK
2436 037352 037360 .WORD 0 ;ADDRESS OF CHARACTERISTICS BLOCK
2437 037354 000000 .WORD 8. ;STARTING VALUE OF BLOCK SIZE
2438 037356 000010
2439
2440 037360 T9DATA: ;CHARACTERISTICS DATA BLOCK
2441 037360 037372 .WORD T9BFR ;ADDRESS OF MESSAGE BUFFER
2442 037362 000000 .WORD 0
2443 037364 000016 .WORD 14. ;LENGTH OF MESSAGE BUFFER
2444 037366 000000 000000 .WORD 0,0
2445
2446 037372 T9BFR: .BLKW 8. ;MESSAGE BUFFER
2447
2448
2449
2450 ;+
2451 ;TEST DATA FOR SUBTEST TWO
2452 ;DATA HAS FORMAT:
2453 ;
2454 ; 1ST WORD OFFSET TO TEST WORD IN PACKET
2455 ; 2ND WORD BITS TO SET FOR TEST
2456 ;
2457 ;-
2458
2459 037412 T92DATA:
2460 037412 000000 037140 .WORD 0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
2461 037416 000002 000001 .WORD 2,BIT0
2462 037422 000004 100100 .WORD 4,BIT6!BIT15

```

TEST 9: COMPLETION INTERRUPT

```

2463          037426          T92DONE=.
2464
2465
2466
2467          ;*
2468          ;LOCAL TEXT MESSAGES FOR TEST
2469          ;-
2470 037426      127      122      111 T9NBA: .ASCIZ 'WRITE CHARACTERISTICS Command Not Accepted'
2471 037501      127      122      111 T92REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
2472 037600      127      122      111 T93REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
2473 037673      127      122      111 T94REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
2474 037771      127      122      111 T95REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
2475 040067      103      157      156 T95SR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
2476 040156      105      170      160 T9NINT: .ASCIZ 'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
2477 040247      125      156      145 T9INT: .ASCIZ 'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
2478 040336      111      156      143 T9TSBA: .ASCIZ 'Incorrect TSBA Address After WRITE CHARACTERISTICS'
2479 040421      103      157      155 TST9ID: .ASCIZ 'Completion Interrupt'
2480          .EVEN
2481
2482
2483          ;*
2484          ;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
2485          ;
2486          ;-
2487
2488          T9REST:
2489 040446          SAVREG          ;SAVE THE REGISTERS
2490 040446          MOV          #T9PACKET,R1 ;START OF THE PACKET
2491 040452      012701      037350      MOV          #100204,(R1)- ;WRITE CHARACTERISTICS WITH ACK, IE
2492 040456      012721      100204      MOV          #T9DATA,(R1)- ;ADDRESS OF CHAR DATA BLOCK
2493 040462      012721      037360      CLR          (R1)- ;EXTENDED ADDRESS
2494 040466          005021          MOV          #8,(R1)- ;SIZE OF DATA BLOCK IN BYTES
2495 040470      012721      000010      MOV          #T9BFR,(R1)- ;ADDRESS OF MESSAGE BUFFER
2496 040474      012721      037372      CLR          (R1)-
2497 040500          005021          MOV          #14,(R1)- ;LENGTH OF MESSAGE BUFFER
2498 040502      012721      000016      CLR          (R1)-
2499 040506          005021          CLR          (R1)-
2500 040510          005011          CLR          (R1)-
2501 040512      005037      037372      CLR          T9BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
2502 040516          000207          RTS          PC ;RETURN
2503 040520          040520          ENDTST
2504          104401          L10065: TRAP C#ETST
2505
2506          .SBTTL TEST 10: BASIC PACKET PROTOCOL
2507          ;
2508          ; THIS TEST VERIFIES BASIC OPERATION OF THE MESSAGE BUFFER RELEASE
2509          ; COMMAND, THE FUNCTION OF THE ACK BIT IN THE COMMAND HEADER WORD,
2510          ; AND THE REGISTER MODIFICATION REFUSED (RMR) LOGIC.
2511          ;
2512          ;*
2513          ;TEST 10 SUBTEST 1
2514          ;
2515          ;CHECKS THAT THE MESSAGE BUFFER RELEASE COMMAND WORKS
2516          ;PROPERLY AND THAT NO INTERRUPT IS GENERATED EVEN
2517          ;IF THE "IE" BIT IS SET IN THE COMMAND PACKET
    
```

B13

TEST 10: BASIC PACKET PROTOCOL

```

2518
2519          ;
2520          ;-
2520 040522          BGNTST
2521 040522
2525 040522 012700 043457      MOV      #TST10ID,R0
2526 040526 004737 016570      JSR      PC,TSTSETUP
2527 040532 012737 000024 002214  MOV      #20.,LOOPCNT
2528 040540          T10LOOP:
2529
2530 040540          BGNSUB
2531 040540          104402
2532 040542 004737 043506      JSR      PC,T10RST
2533 040546          SETPRI  #PRI00
2534 040546 012700 000000      MOV      #PRI00,R0
2535 040552 104441          TRAP     C#SPRI
2536 040554 012704 042650      MOV      #T10PACKET,R4
2537 040560 012764 000010 000006  MOV      #8.,PKBCNT(R4)
2538 040566          5#:
2539 040566          BGNSSEG
2540 040570 004737 016054      JSR      PC,SOFINIT
2541 040574 103405          BCS     10#
2542 040576 010001          MOV      R0,R1
2543 040600          ERDF    ERRNO,SFIERR,SFIMSG
2544 040600 104455          TRAP     C#ERDF
2545 040602 001751          .WORD  1001
2546 040604 003652          .WORD  SFIERR
2547 040606 012104          .WORD  SFIMSG
2548 040610 005037 002220      10#: CLR      FATFLG
2549 040614 005037 002222      CLR      INTRECV
2550 040620 010465 000000      MOV      R4,TSDB(R5)
2551 040624 004737 016416      JSR      PC,CHKTSSR
2552 040630 103407          BCS     15#
2553 040632 010001          MOV      R0,R1
2554 040634          ERDF    ERRNO,T10SSR,PKTSSR
2555 040634 104455          TRAP     C#ERDF
2556 040636 001752          .WORD  1002
2557 040640 043210          .WORD  T10SSR
2558 040642 012116          .WORD  PKTSSR
2559 040644 005237 002220      15#: INC      FATFLG
2560 040650          CKLOOP
2561 040650 104406          SET     C#CLP1
2562 040652          ESCAPE  SEG
2563 040652 104410          TRAP     C#ESCAPE
2564 040654 000056          .WORD  10000#-.
2565 040656 005737 002222      22#: TST      INTRECV
2566 040662 001004          BNE     22#
2567 040664          ERHRD   ERRNO,T10NINT,PKTSSR
2568 040664 104456          TRAP     C#ERHRD
2569 040666 001753          .WORD  1003
2570 040670 043277          .WORD  T10NINT
2571 040672 012116          .WORD  PKTSSR
2572 040674 016501 000002      22#: MOV      TSSR(R5),R1
2573 040700 012702 000200      MOV      #SSR,R2
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600

```


G13

TEST 10: BASIC PACKET PROTOCOL

2753	041552	103405			BCS	10:						
2757	041554	010001			MOV		RO,R1					
2758	041556				ERRDF		ERRNO,SFIERR,SFIMSG					
	041556	104455										
	041560	001771								TRAP	C#ERDF	
	041562	003652								.WORD	1017	
	041564	012104								.WORD	SFIERR	
2759	041566	005037	002220		CLR	10:	FATFLG					
2760	041572	005037	002222		CLR		INTRECV					
2761	041576	010465	000000		MOV		R4,TSDB(R5)					
2762	041602	004737	016416		JSR		PC,CHKTSSR					
2763	041606	103407			BCS		15:					
2764	041610	010001			MOV		RO,R1					
2768	041612				ERRDF		ERRNO,T1OSSR,PKTSSR					
	041612	104455										
	041614	001772								TRAP	C#ERDF	
	041616	043210								.WORD	1018	
	041620	012116								.WORD	T1OSSR	
2769	041622	005237	002220		INC		FATFLG			.WORD	PKTSSR	
2770	041626				CKLOOP	15:						
	041626	104406										
2771	041630				ESCAPE		SEG			TRAP	C#CLP1	
	041630	104410										
	041632	000056								TRAP	C#ESCAPE	
2772	041634	005737	002222		TST		INTRECV			.WORD	10000+-	
2773	041640	001004			BNE		22:					
2777	041642				ERRHRD		ERRNO,T1ONINT,PKTSSR					
	041642	104456										
	041644	001773								TRAP	C#ERHRD	
	041646	043277								.WORD	1019	
	041650	012116								.WORD	T1ONINT	
2778	041652	016501	000002		MOV	22:	TSSR(R5),R1			.WORD	PKTSSR	
2779	041656	012702	000200		MOV		#SSR,R2					
2780	041662	032701	000100		BIT		#OFL,R1					
2781	041666	001402			BEQ		25:					
2782	041670	052702	000100		BIS		#OFL,R2					
2783	041674	020201			CMP		R2,R1					
2784	041676	001404			BEQ		30:					
2788	041700				ERRHRD		ERRNO,T1ONBA,PKTSSR					
	041700	104456										
	041702	001774								TRAP	C#ERHRD	
	041704	043051								.WORD	1020	
	041706	012116								.WORD	T1ONBA	
2789	041710									.WORD	PKTSSR	
2790	041710				ENDSEG	30:						
	041710	104405										
2791	041712				BGNSEG							
	041712	104404										
2792	041714	004737	043506		JSR		PC,T1ORST					
2793	041720	005037	002222		CLR		INTRECV					
2794	041724	012737	025252	042672	MOV		#025252,T1OBFR					
2795	041732	012714	100212		MOV		#100212,(R4)					
2796	041736	010465	000000		MOV		R4,TSDB(R5)					
2797	041742	004737	016416		JSR		PC,CHKTSSR					
2798	041746	103407			BCS		45:					
2799	041750	010001			MOV		RO,R1					

```

;BR IF SOFT INIT = OK
;SAVE CONTENTS OF TSSR
;DEVICE FATAL ERROR DURING INIT
TRAP C#ERDF
.WORD 1017
.WORD SFIERR
.WORD SFIMSG
;CLEAR FATAL ERROR FLAG
;CLEAR INTERRUPT RECEIVED FLAG
;SET THE PACKET ADDRESS
;WAIT FOR SSR TO SET
;BR IF CARRY SET (GOOD RETURN)
;SAVE CONTENTS OF TSSR
;DEVICE FATAL SSR FAILED TO SET
TRAP C#ERDF
.WORD 1018
.WORD T1OSSR
.WORD PKTSSR
;SET FATAL ERROR FLAG
;LOOP ON ERROR, IF FLAG SET
TRAP C#CLP1
;BY-PASS SUBTEST IF FATAL ERROR
TRAP C#ESCAPE
.WORD 10000+-
;DID AN INTERRUPT OCCUR ?
;BRANCH IF YES
TRAP C#ERHRD
.WORD 1019
.WORD T1ONINT
.WORD PKTSSR
;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
;DOES EXPECTED MATCH RECEIVED ?
;OKAY IF MATCH
;NBA NOT ZERO
TRAP C#ERHRD
.WORD 1020
.WORD T1ONBA
.WORD PKTSSR
;<<<<<<<<<<<<<<<< END SEGMENT
10000:
TRAP C#ESEG
;>>>>>>>>> BEGIN SEGMENT
TRAP C#BSEG
;RESET PACKETS AND COMMANDS
;CLEAR INTERRUPT RECEIVED FLAG
;WIPE OUT MESSAGE BUFFER AREA
;SET COMMAND PACKET TO MESS BUF REL
;SET THE PACKET ADDRESS
;WAIT FOR SSR TO SET
;BR IF CARRY SET (GOOD RETURN)
;SAVE CONTENTS OF TSSR

```

TEST 10: BASIC PACKET PROTOCOL

```

2803 041752          ERRDF  ERRNO,T10SSR,PKTSSR  ;DEVICE FATAL SSR FAILED TO SET
      041752 104455          TRAP  C#ERDF
      041754 001775          .WORD 1021
      041756 043210          .WORD T10SSR
      041760 012116          .WORD PKTSSR
2804 041762 005237 002220          INC  FATFLG  ;SET FATAL ERROR FLAG
2805 041766          45$: CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      041766 104406          TRAP  C#CLP1
2806 041770 005737 002222          TST  INTRECV  ;DID AN INTERRUPT OCCUR ?
2807 041774 001404          BEQ  52$      ;BRANCH IF NO
2811 041776          ERRHRD  ERRNO,T10INT,PKTSSR
      041776 104456          TRAP  C#ERHRD
      042000 001776          .WORD 1022
      042002 043370          .WORD T10INT
      042004 012116          .WORD PKTSSR
2812 042006 016501 000002          52$: MOV  TSSR(R5),R1  ;GET THE CONTENTS OF TSSR
2813 042012 012702 000200          MOV  #SSR,R2      ;EXPECTED CONTENTS OF TSSR
2814 042016 032701 000100          BIT  #OFL,R1     ;IS OFF-LINE BIT SET ?
2815 042022 001402          BEQ  55$      ;BRANCH IF NOT OFF-LINE
2816 042024 052702 000100          BIS  #OFL,R2     ;SET OFF-LINE IN EXPECTED DATA
2817 042030 020201          55$: CMP  R2,R1    ;DOES EXPECTED MATCH RECEIVED ?
2818 042032 001404          BEQ  60$      ;OKAY IF MATCH
2822 042034          ERRHRD  ERRNO,T10NNBA,PKTSSR ;NBA NOT SET
      042034 104456          TRAP  C#ERHRD
      042036 001777          .WORD 1023
      042040 043133          .WORD T10NNBA
      042042 012116          .WORD PKTSSR
2823 042044          60$:
2824 042044 013701 042672          MOV  T10BFR,R1   ;PICK UP THE 1ST WORD OF MESSAGE BUFFER
2825 042050 012702 025252          MOV  #025252,R2 ;SET UP EXPECTED DATA
2826 042054 020102          CMP  R1,R2      ;WAS ANY MESSAGE REC'D
2827 042056 001404          BEQ  70$      ;BR, IF OK (EQUAL)
2831 042060          ERRHRD  ERRNO,T10MBF,EXPREC ;MESSAGE BUFFER WAS MODIFIED
      042060 104456          TRAP  C#ERHRD
      042062 002000          .WORD 1024
      042064 042754          .WORD T10MBF
      042066 015554          .WORD EXPREC
2832 042070          70$:
2833 042070 104406          CKLOOP  ;LOOP ON ERROR IF FLAG SET
      TRAP  C#CLP1
2834 042072 005037 002222          CLR  INTRECV    ;CLEAR INTERRUPT RECEIVED FLAG
2835 042076 004737 043506          JSR  PC,T10RST  ;RESET THE PACKETS AND COMMANDS
2836 042102 042714 100000          BIC  #100000,(R4) ;CLEAR THE ACK BIT
2837 042106 010465 000000          MOV  R4,TSDB(R5) ;SET THE PACKET ADDRESS
2838 042112 004737 016416          JSR  PC,CHKTSSR ;WAIT FOR SSR TO SET
2839 042116 103407          BCS  75$      ;BR IF CARRY SET (GOOD RETURN)
2840 042120 010001          MOV  R0,R1     ;SAVE CONTENTS OF TSSR
2845 042122          ERRDF  ERRNO,T10SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      042122 104455          TRAP  C#ERDF
      042124 002001          .WORD 1025
      042126 043210          .WORD T10SSR
      042130 012116          .WORD PKTSSR
2846 042132 005237 002220          INC  FATFLG  ;SET FATAL ERROR FLAG
2847 042136          75$: CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      042136 104406          TRAP  C#CLP1
2848 042140          ESCAPE  SEG  ;BY-PASS SUBTEST IF FATAL ERROR

```

TEST 10: BASIC PACKET PROTOCOL

	042140	104410				TRAP	C#ESCAPE
	042142	000062				.WORD	10001#-.
2849	042144	005737	002222	TST	INTRECV		
2850	042150	001006		BNE	82#		
2854	042152	016500	000002	MOV	TSSR(R5),R0		
2855	042156			ERRHRD	ERRNO,T10NINT,PKTSSR		
	042156	104456				TRAP	C#ERHRD
	042160	002002				.WORD	1026
	042162	043277				.WORD	T10NINT
	042164	012116				.WORD	PKTSSR
2856	042166	016501	000002	82#:	MOV	TSSR(R5),R1	
2857	042172	012702	000200		MOV	#SSR,R2	
2858	042176	032701	000100		BIT	#OFL,R1	
2859	042202	001402			BEQ	85#	
2860	042204	052702	000100		BIS	#OFL,R2	
2861	042210	020201		85#:	CMP	R2,R1	
2862	042212	001404			BEQ	90#	
2866	042214				ERRHRD	ERRNO,T10SSR,PKTSSR	
	042214	104456				TRAP	C#ERHRD
	042216	002003				.WORD	1027
	042220	043210				.WORD	T10SSR
	042222	012116				.WORD	PKTSSR
2867	042224			90#:			
2868	042224			ENDSEG			
	042224						
	042224	104405					
2869	042226	005737	002220	TST	FATFLG		
2870	042232	001403		BEQ	95#		
2871	042234	004737	017262	JSR	PC,CKDROP		
2872							
2873	042240			BGNSEG			
	042240	104404					
2874	042242	005037	002222	95#:	CLR	INTRECV	
2875	042246	004737	043506		JSR	PC,T10RST	
2876	042252	010465	000000		MOV	R4,TSDB(R5)	
2877	042256	004737	016416		JSR	PC,CHKTSSR	
2878	042262	103407			BCS	100#	
2879	042264	010001			MOV	R0,R1	
2883	042266				ERRDF	ERRNO,T10SSR,PKTSSR	
	042266	104455				TRAP	C#ERDF
	042270	002004				.WORD	1028
	042272	043210				.WORD	T10SSR
	042274	012116				.WORD	PKTSSR
2884	042276	005237	002220	100#:	INC	FATFLG	
2885	042302			CKLOOP			
	042302	104406					
2886	042304			ESCAPE	SEG		
	042304	104410					
	042306	000062					
2887	042310	005737	002222	TST	INTRECV		
2888	042314	001006		BNE	112#		
2892	042316	016500	000002	MOV	TSSR(R5),R0		
2893	042322			ERRHRD	ERRNO,T10NINT,PKTSSR		
	042322	104456				TRAP	C#ERHRD
	042324	002005				.WORD	1029
	042326	043277				.WORD	T10NINT
	042330	012116				.WORD	PKTSSR

```

;DID AN INTERRUPT OCCUR ?
;BRANCH IF YES
;GET TSSR FOR ERROR REPORT

;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
;DOES EXPECTED MATCH RECEIVED ?
;OKAY IF MATCH
;NBA NOT ZERO

;***** END SEGMENT *****
10001#:# TRAP C#ESEG
;ANY FATAL ERRORS
;BR, IF NO FATAL ERRORS
;TRY TO DROP THE UNIT

;***** BGN SEGMENT *****
TRAP C#BSEG
;CLEAR INTERRUPT RECEIVED FLAG
;RESET THE PACKETS AND COMMANDS
;SET THE PACKET ADDRESS
;WAIT FOR SSR TO SET
;BR IF CARRY SET (GOOD RETURN)
;SAVE CONTENTS OF TSSR
;DEVICE FATAL SSR FAILED TO SET

;SET FATAL ERROR FLAG
;LOOP ON ERROR, IF FLAG SET
;BY-PASS SUBTEST IF FATAL ERROR

;DID AN INTERRUPT OCCUR ?
;BRANCH IF YES
;GET TSSR FOR ERROR REPORT

```


L13

SEQ 0167

TEST 10: BASIC PACKET PROTOCOL

```

2987 042650 100204          .WORD 100204          ;WRITE CHAR COMMAND, WITH IE, ACK
2988 042652 042660          .WORD T10DATA        ;ADDRESS OF CHARACTERISTICS BLOCK
2989 042654 000000          .WORD 0              ;STARTING VALUE OF BLOCK SIZE
2990 042656 000010          .WORD 8              ;CHARACTERISTICS DATA BLOCK
2991                                     ;ADDRESS OF MESSAGE BUFFER
2992 042660          T10DATA:          .WORD T10BFR
2993 042660 042672          .WORD 0              ;LENGTH OF MESSAGE BUFFER
2994 042662 000000          .WORD 14
2995 042664 000016          .WORD 0,0           ;MESSAGE BUFFER
2996 042666 000000 000000          T10BFR: .BLKW 8
2997                                     ;*
2998 042672                                     ;TEST DATA FOR SUBTEST FOUR
2999                                     ;*
3000                                     ;COMMAND PACKET FOR TEST
3001                                     ;WRITE CHAR COMMAND, WITH IE, ACK
3002                                     ;ADDRESS OF CHARACTERISTICS BLOCK
3003                                     ;STARTING VALUE OF BLOCK SIZE
3004 042712 100204          T10PKT:          .WORD 100204
3005 042712 042722          .WORD T10DTA        ;CHARACTERISTICS DATA BLOCK
3006 042714 000000          .WORD 0              ;ADDRESS OF MESSAGE BUFFER
3007 042716 000010          .WORD 8              ;LENGTH OF MESSAGE BUFFER
3008 042720 000000          .WORD 0,0           ;MESSAGE BUFFER
3009                                     ;*
3010 042722          T10DTA:          .WORD T10BUFR
3011 042722 042734          .WORD 0              ;LOCAL TEXT MESSAGES FOR TEST
3012 042724 000000          .WORD 14
3013 042726 000016          .WORD 0,0
3014 042730 000000 000000          T10BUFR: .BLKW 8
3015 042734                                     ;*
3016                                     ;LOCAL TEXT MESSAGES FOR TEST
3017                                     ;*
3018                                     ;MESSAGE BUFFER Modified after MESSAGE BUFFER RELEASE Command'
3019                                     ;NBA Not Clear After WRITE CHARACTERISTICS Command'
3020                                     ;NBA Set After MESSAGE BUFFER RELEASE Command'
3021                                     ;Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
3022                                     ;Expected Interrupt Not Received On WRITE CHARACTERISTICS'
3023 042754 115 145 163 T10MBF: .ASCIZ 'Message Buffer Modified after MESSAGE BUFFER RELEASE Command'
3024 043051 116 102 101 T10NBA: .ASCIZ 'NBA Not Clear After WRITE CHARACTERISTICS Command'
3025 043133 116 102 101 T10NNBA: .ASCIZ 'NBA Set After MESSAGE BUFFER RELEASE Command'
3026 043210 103 157 156 T10SSR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
3027 043277 105 170 160 T10INT: .ASCIZ 'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
3028 043370 125 156 145 T10INT: .ASCIZ 'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
3029 043457 102 141 163 TST10ID: .ASCIZ 'Basic Packet Protocol'
3030                                     ;*
3031                                     ;EVEN
3032                                     ;*
3033                                     ;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
3034                                     ;*
3035                                     ;*
3036                                     ;*
3037                                     ;*
3038                                     ;*
3039                                     ;*
3040 043506          T10RST:          SAVREG          ;SAVE THE REGISTERS
3041 043506          MOV #T10PACKET,R1    ;START OF THE PACKET
3042 043512 012701 042650  MOV #100204,(R1)  ;WRITE CHARACTERISTICS WITH ACK, IE
3043 043516 012721 100204

```


M13

TEST 10: BASIC PACKET PROTOCOL

```

3044 043522 012721 042660      MOV    #T10DATA,(R1)• ;ADDRESS OF CHAR DATA BLOCK
3045 043526 005021             CLR    (R1)•          ;EXTENDED ADDRESS
3046 043530 012721 000010      MOV    #8,(R1)•       ;SIZE OF DATA BLOCK IN BYTES
3047 043534 012721 042672      MOV    #T10BFR,(R1)• ;ADDRESS OF MESSAGE BUFFER
3048 043540 005021             CLR    (R1)•          ;EXTENDED ADDRESS
3049 043542 012721 000016      MOV    #14,(R1)•     ;LENGTH OF MESSAGE BUFFER
3050 043546 005021             CLR    (R1)•          ;EXTENDED ADDRESS
3051 043550 005011             CLR    (R1)•          ;EXTENDED ADDRESS
3052 043552 005037 042672      CLR    T10BFR        ;CLEAR 1ST LOC IN MESSAGE BUFFER
3053 043556 000207             RTS     PC            ;RETURN
3054
3055
3056
3057
3058
3059
3060 043560
3061 043560
3062 043564 012701 042712      T10RT2: SAVREG        ;SAVE THE REGISTERS
3063 043570 012721 100204      MOV    #T10PKT,R1    ;START OF THE PACKET
3064 043574 012721 042722      MOV    #100204,(R1)• ;WRITE CHARACTERISTICS WITH ACK, IE
3065 043600 005021             CLR    (R1)•          ;EXTENDED ADDRESS
3066 043602 012721 000010      MOV    #8,(R1)•       ;SIZE OF DATA BLOCK IN BYTES
3067 043606 012721 042734      MOV    #T10BUFR,(R1)• ;ADDRESS OF MESSAGE BUFFER
3068 043612 005021             CLR    (R1)•          ;EXTENDED ADDRESS
3069 043614 012721 000016      MOV    #14,(R1)•     ;LENGTH OF MESSAGE BUFFER
3070 043620 005021             CLR    (R1)•          ;EXTENDED ADDRESS
3071 043622 005011             CLR    (R1)•          ;EXTENDED ADDRESS
3072 043624 005037 042734      CLR    T10BUFR        ;CLEAR 1ST LOC IN MESSAGE BUFFER
3073 043630 000207             RTS     PC            ;RETURN
3074 043632
3075 043632 104401             L10075: TRAP    C#ETST
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088 043634
3089 043634
3093 043634 012700 045642
3094 043640 004737 016570
3095 043644 012737 000024 002214
3096 043652
3097 043652
3098 043652 104402
3099 043654

```

```

;
;
;ROUTINE TO RESTORE COMMAND PACKET #2 TO START-UP (DEFAULT) VALUES
;
;-

```

```

T10RT2: SAVREG        ;SAVE THE REGISTERS
MOV    #T10PKT,R1    ;START OF THE PACKET
MOV    #100204,(R1)• ;WRITE CHARACTERISTICS WITH ACK, IE
MOV    #T10DTA,(R1)• ;ADDRESS OF CHAR DATA BLOCK
CLR    (R1)•          ;EXTENDED ADDRESS
MOV    #8,(R1)•       ;SIZE OF DATA BLOCK IN BYTES
MOV    #T10BUFR,(R1)• ;ADDRESS OF MESSAGE BUFFER
CLR    (R1)•          ;EXTENDED ADDRESS
MOV    #14,(R1)•     ;LENGTH OF MESSAGE BUFFER
CLR    (R1)•          ;EXTENDED ADDRESS
CLR    T10BUFR        ;CLEAR 1ST LOC IN MESSAGE BUFFER
RTS     PC            ;RETURN
ENDTST

```

```

L10075: TRAP    C#ETST

```

.SBTTL TEST 11: NON-TAPE MOTION COMMANDS

```

;
;THIS TEST VERIFIES PROPER OPERATION OF THE INITIALIZE
;COMMAND. TWO SUBTESTS ARE USED. THE FIRST VERIFIES THAT
;THE COMMAND RUNS TO COMPLETION AND STORES A VALID
;MESSAGE PACKET. THE SECOND VERIFIES THAT NON-ZERO
;VALUES IN THE COMMAND MODE FIELD CAUSES COMMAND REJECT.
;
;-

```

```

BGNTST
MOV    #TST11ID,R0    ;ASCII MESSAGE TO IDENTIFY TEST
JSR    PC,TSTSETUP    ;DO INITIAL TEST SETUP
MOV    #20,LOOPCNT    ;PERFORM 20 ITERATIONS
T11LOOP: BGNSUB        ;//////////////// BEGIN SUBTEST //////////////////
T11.1: TRAP    C#BSUB
SETPRI #PRI00        ;LOWER PRIORITY TO ALLOW INTERRUPTS

```


TEST 11: NON-TAPE MOTION COMMANDS

```

3331 044600 010001      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
3332 044602      ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
                                TRAP      C#ERDF
                                .WORD    1119
                                .WORD    SFIERR
                                .WORD    SFIMSG
3333 044612      3#:
3334 044612 012704 045070      MOV      @T11PK2,R4 ;WRITE CHARACTERISTICS PACKET
3335 044616 004737 010662      JSR      PC,WRTCHR ;ISSUE WRITE CHARACTERISTICS
3336 044622 103404      BCS     4#          ;BR, IF COMMAND ISSUED OK
3340 044624      ERRHRD  ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTICSC FAILED
                                TRAP      C#ERHRD
                                .WORD    1120
                                .WORD    WRTMSG
                                .WORD    SFIMSG
3341 044634      4#:
3342 044634 004737 045674      JSR      PC,T11REST ;SET UP PACKET FOR COMMAND
3343 044640 012704 045020      MOV      @T11PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
3344 044644      5#:
3345 044644 005037 002222      CLR      INTRECV   ;CLEAR INTERRUPT RECEIVED FLAG
3346 044650 052714 007000      BIS      #007000,(R4) ;SET TO NON-ZERO MODE
3347 044654 010465 000000      MOV      R4,TSD8(R5) ;SET THE PACKET ADDRESS
3348 044660 004737 016416      JSR      PC,CHKTSSR ;WAIT FOR SSR TO SET
3349 044664 103405      BCS     15#        ;BR IF CARRY SET (GOOD RETURN)
3350 044666 010001      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
3354 044670      ERRDF   ERRNO,T11SR2,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C#ERDF
                                .WORD    1121
                                .WORD    T11SR2
                                .WORD    PKTSSR
3355 044700      15#:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
3356 044702      104406      ESCAPE   SUB        ;BY-PASS SUBTEST IF FATAL ERROR
                                TRAP      C#CLP1
                                .WORD    C#ESCAPE
                                .WORD    L10106-.
3357 044706 005737 002222      TST      INTRECV   ;DID AN INTERRUPT OCCUR ?
3358 044712 001004      BNE     22#        ;BRANCH IF YES
3362 044714      ERRHRD  ERRNO,T11NINT,PKTSSR
                                TRAP      C#ERHRD
                                .WORD    1122
                                .WORD    T11NINT
                                .WORD    PKTSSR
3363 044724 016501 000002      22#:      MOV      TSSR(R5),R1 ;GET THE CONTENTS OF TSSR
3364 044730 012702 100206      MOV      @SC!SSR!TSREJ,R2 ;EXPECTED CONTENTS OF TSSR
3365 044734 032701 000100      BIT      @OFL,R1   ;IS OFF-LINE BIT SET ?
3366 044740 001402      BEQ     25#        ;BRANCH IF NOT OFF-LINE
3367 044742 052702 000100      BIS      @OFL,R2   ;SET OFF-LINE IN EXPECTED DATA
3368 044746 020201      25#:      CMP      R2,R1     ;DOES EXPECTED MATCH RECEIVED ?
3369 044750 001404      BEQ     30#        ;OKAY IF MATCH
3373 044752      ERRHRD  ERRNO,T114REJ,PKTSSR ;COMMAND NOT REJECTED
                                TRAP      C#ERHRD
                                .WORD    1123
                                .WORD    T114REJ
                                .WORD    PKTSSR
3374 044762      30#:
3375
3376 044762 004737 011154      35#:      JSR      PC,CKRAM  ;CHECK RAM TO MEMORY

```


TEST 11: NON-TAPE MOTION COMMANDS

3431							
3432							
3433	045132	111	116	111	T11NBA:	.ASCIZ	'INITIALIZE Command Not Accepted'
3434	045172	111	116	111	T112REJ:		.ASCIZ 'INITIALIZE Not Rejected With Non-Zero Mode Field'
3435	045253	107	105	124	T113REJ:		.ASCIZ 'GET STATUS Not Accepted'
3436	045303	107	105	124	T114REJ:		.ASCIZ 'GET STATUS Not Rejected With Non-Zero Mode Field'
3437	045364	103	157	156	T11SSR:	.ASCIZ	'Contents of TSSR Incorrect After INITIALIZE'
3438	045440	103	157	156	T11SR2:	.ASCIZ	'Contents of TSSR Incorrect After GET STATUS'
3439	045514	105	170	160	T11NINT:		.ASCIZ 'Expected Interrupt Not Received On INITIALIZE'
3440	045572	111	156	143	T11TSBA:		.ASCIZ 'Incorrect TSBA Address After INITIALIZE'
3441	045642	116	157	156	TST11ID:		.ASCIZ 'Non-Tape Motion Commands'
3442							.EVEN
3443							
3444							
3445							
3446							
3447							
3448							
3449							
3450							
3451							
3452	045674				T11REST:		
3453	045674				SAVREG		;SAVE THE REGISTERS
3454	045700	012701	045020		MOV	#T11PACKET,R1	;START OF THE PACKET
3455	045704	012721	100213		MOV	#100213,(R1)+	;INITIALIZE WITH ACK, IE
3456	045710	005021			CLR	(R1)+	;ADDRESS OF CHAR DATA BLOCK
3457	045712	005021			CLR	(R1)+	;EXTENDED ADDRESS
3458	045714	005021			CLR	(R1)+	;SIZE OF DATA BLOCK IN BYTES
3459	045716	005021			CLR	(R1)+	;ADDRESS OF MESSAGE BUFFER
3460	045720	005021			CLR	(R1)+	
3461	045722	005021			CLR	(R1)+	;LENGTH OF MESSAGE BUFFER
3462	045724	005021			CLR	(R1)+	
3463	045726	005011			CLR	(R1)	
3464	045730	005037	045042		CLR	T11BFR	;CLEAR 1ST LOC IN MESSAGE BUFFER
3465	045734	000207			RTS	PC	;RETURN
3466							
3467							
3468							
3469							
3470							
3471							
3472							
3473	045736				T11RT2:		
3474	045736				SAVREG		;SAVE THE REGISTERS
3475	045742	012701	045020		MOV	#T11PACKET,R1	;START OF THE PACKET
3476	045746	012721	100217		MOV	#100217,(R1)+	;GET STATUS WITH ACK, IE
3477	045752	005021			CLR	(R1)+	;ADDRESS OF CHAR DATA BLOCK
3478	045754	005021			CLR	(R1)+	;EXTENDED ADDRESS
3479	045756	005021			CLR	(R1)+	;SIZE OF DATA BLOCK IN BYTES
3480	045760	005021			CLR	(R1)+	;ADDRESS OF MESSAGE BUFFER
3481	045762	005021			CLR	(R1)+	
3482	045764	005021			CLR	(R1)+	;LENGTH OF MESSAGE BUFFER
3483	045766	005021			CLR	(R1)+	
3484	045770	005011			CLR	(R1)	
3485	045772	005037	045042		CLR	T11BFR	;CLEAR 1ST LOC IN MESSAGE BUFFER
3486	045776	000207			RTS	PC	;RETURN
3487	046000				ENDTST		

I14

TEST 11: NON-TAPE MOTION COMMANDS

046000
046000 104401
3488 046002

ENDMOD

L10102: TRAP C\$ETST

TEST 11: NON-TAPE MOTION COMMANDS

```

1          .TITLE  TSV6 - PARAMETER CODING
7
12
18
19 046002      BGNMOD  TSV6
   046002      TSV6::
20
21          .SBTTL  HARDWARE PARAMETER CODING SECTION
22
23          ;**
24          ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
25          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
26          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
27          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
28          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
29          ; WITH THE OPERATOR.
30          ;--
31 046002      BGNHRD
   046002      .WORD  L10107-L$HARD/2
   046004      L$HARD::
32
33 046004      GPRMA  HPM1,0,0,160010,177776,YES      ;GET TSBA/TSDB REGISTER ADDRESS.
   046004      .WORD  T$CODE
   046006      .WORD  HPM1
   046010      .WORD  T$LLOLIM
   046012      .WORD  T$HILIM
34 046014      GPRMA  HPM2,2,0,0,776,YES              ;GET VECTOR ADDRESS.
   046014      .WORD  T$CODE
   046016      .WORD  HPM2
   046020      .WORD  T$LLOLIM
   046022      .WORD  T$HILIM
35          ;GPRMD  HPM3,4,0,340,0,7,YES              ;GET INTERRUPT PRIORITY.
36 046024      ENDRD
   .EVEN
   046024      L10107:
37 046024      104    105    126  HPM1:  .ASCIZ  'DEVICE ADDRESS (TSBA/TSDB) '
38 046060      111    116    124  HPM2:  .ASCIZ  'INTERRUPT VECTOR '
39 046104      111    116    124  HPM3:  .ASCIZ  'INTERRUPT PRIORITY '
40          .EVEN

```

SOFTWARE PARAMETER CODING SECTION

```

42          .SBTTL  SOFTWARE PARAMETER CODING SECTION
43
44
45          ;**
46          ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
47          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
48          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
49          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
50          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
51          ; WITH THE OPERATOR.
52          ;--
53          BGNSFT
54          .WORD L10110-L$SOFT/2
55          L$SOFT::
56          ; GPRML  SPM1,0,-1,YES          ; GET TRANSPORT TEST FLAG.
57          ; GPRML  SPM4,2,-1,YES         ; GET ITERATION CONTROL.
58          .WORD  T$CODE
59          .WORD  SPM4
60          .WORD  -1
61          ; GPRMD  SPM6,4,D,7777,0,7777,YES  ; GET LOCAL ERROR LIMIT
62          ; GPRMD  SPM7,6,D,7777,0,7777,YES  ; GET GLOBAL ERROR LIMIT
63          ENDSFT
64          .EVEN
65          L10110:
66          SPM1:  .ASCIZ  'ENABLE TRANSPORT TESTS '
67          SPM4:  .ASCIZ  'INHIBIT ITERATIONS '
68          SPM6:  .ASCIZ  'PER TEST ERROR LIMIT '
69          SPM7:  .ASCIZ  'PER UNIT ERROR LIMIT '
70          .SBTTL  PATCH AREA
71
72          ;
73          ; FINALLY A GENEROUS PATCH AREA.
74          ;
75          ; AND AN ADJUSTMENT TO ACCOUNT FOR THE "LASTAD BIT7" HACK
76          ; DESCRIBED IN "SUPPRG.MEM" (FOR REV C).
77          ;
78          PATCH::
79          .BLKW  32.
80          . = !377*1
81          LASTAD          ;SET LAST USED ADDRESS.
82          .EVEN
83          .WORD  0
84          .WORD  0
85          L$LAST::
86          ENDMOD
87          .END

```

Symbol table

ADDSSR 012176 G	C#AU = 000052	DEVDR0 023412	FRESIZ 003126 G	INTFLA 016225
ADR = 000020 G	C#AUTO= 000061	DEVNRD 023331	FUSI 004117	INTMAS 016224
AMBTSS 006635	C#BRK = 000022	DEVNXR 023247	F#AU = 000015	INTR 016276 G
ASSEMB= 000010	C#BSEG= 000004	DEVONL 023177	F#AUTO= 000020	INTREC 002222 G
A1716 = 000003	C#BSUB= 000002	DEVSUM 023142	F#BGN = 000040	INTVEC 016226
BADDAT 003156 G	C#CEFG= 000045	DFPTBL 002154 G	F#CLEA= 000007	INTX 004300
BADSSR 015760 G	C#CLCK= 000062	DIAGMC= 000000	F#DU = 000016	INVERT 021222 G
BDVPCR= 177520 G	C#CLEA= 000012	DICEA = 000001	F#END = 000041	IOKCKI= 000200
BENBSW 002230 G	C#CLOS= 000035	DSBINT 016264	F#HARD= 000004	IOKSTP= 000001
BIE = 040000	C#CLP1= 000006	DUAD12 004643	F#HW = 000013	IPRI 002210 G
BIT0 = 000001 G	C#CVEC= 000036	DUFLG 003112 G	F#INIT= 000006	ISR = 000100 G
BIT00 = 000001 G	C#DCLN= 000044	DUMMY 003062	F#JMP = 000050	IVEC 002206 G
BIT01 = 000002 G	C#DODU= 000051	EF.CON= 000036 G	F#MOD = 000000	IXE = 004000 G
BIT02 = 000004 G	C#DRPT= 000024	EF.NEW= 000035 G	F#MSG = 000011	I#AU = 000041
BIT03 = 000010 G	C#DU = 000053	EF.PWR= 000034 G	F#PROT= 000021	I#AUTO= 000041
BIT04 = 000020 G	C#EDIT= 000003	EF.RES= 000037 G	F#PWR = 000017	I#CLN = 000041
BIT05 = 000040 G	C#ERDF= 000055	EF.STA= 000040 G	F#RPT = 000012	I#DU = 000041
BIT06 = 000100 G	C#ERHR= 000056	EMAXDU 017057	F#SEG = 000003	I#HRD = 000041
BIT07 = 000200 G	C#ERRO= 000060	EN = 000000	F#SOFT= 000005	I#INIT= 000041
BIT08 = 000400 G	C#ERSF= 000054	ENAINI 016232	F#SRV = 000010	I#MOD = 000041
BIT09 = 001000 G	C#ERSO= 000057	ENVIRN 020710	F#SUB = 000002	I#MSG = 000041
BIT1 = 000002 G	C#ESCA= 000010	EPRTSW 002176 G	F#SW = 000014	I#PROT= 000040
BIT10 = 002000 G	C#ESEG= 000005	EPRT1 006360	F#TEST= 000001	I#PTAB= 000041
BIT11 = 004000 G	C#ESUB= 000003	EPRT2 006360	GDDAT 003160 G	I#PWR = 000041
BIT12 = 010000 G	C#ETST= 000001	ERRM 012003	GERRMA 002172 G	I#RPT = 000241
BIT13 = 020000 G	C#EXIT= 000032	ERRHI 002236 G	GETPAT 020254 G	I#SEG = 000041
BIT14 = 040000 G	C#GETB= 000026	ERRK 017036	GETSEL 020336 G	I#SETU= 000041
BIT15 = 100000 G	C#GETW= 000027	ERRLO 002240 G	G#CNT0= 000200	I#SFT = 000041
BIT2 = 000004 G	C#GHAN= 000043	ERRNO = 002144	G#DELM= 000372	I#SRV = 000041
BIT3 = 000010 G	C#GPHR= 000042	ERRVEC= 000004 G	G#DISP= 000003	I#SUB = 000041
BIT4 = 000020 G	C#GPLO= 000030	ERTABE 003376	G#EXCP= 000400	I#TST = 000041
BIT5 = 000040 G	C#GPRI= 000040	ERTABL 003176	G#HILI= 000002	J#JMP = 000167
BIT6 = 000100 G	C#INIT= 000011	ESUM 017040	G#LOLI= 000001	KIPAR0= 172340
BIT7 = 000200 G	C#INLP= 000020	EVL = 000004 G	G#NO = 000000	KIPAR1= 172342
BIT8 = 000400 G	C#MANI= 000050	EXBCNT= 000010	G#OFFS= 000400	KIPAR2= 172344
BIT9 = 001000 G	C#MEM = 000031	EXPBRE 015562 G	G#OF SI= 000376	KIPAR3= 172346
BOE = 000400 G	C#MSG = 000023	EXPD 002232 G	G#PRMA= 000001	KIPAR4= 172350
BRINIT 004457	C#OPEN= 000034	EXPGET 004533	G#PRMD= 000002	KIPAR5= 172352
BSEL0 = 000000	C#PNTB= 000014	EXPGET2 004567	G#PRML= 000000	KIPAR6= 172354
BSEL1 = 000001	C#PNTF= 000017	EXPMSG 002322 G	G#RADA= 000140	KIPAR7= 172356
CHKAMB 016124	C#PNTS= 000016	EXPREC 015554 G	G#RADB= 000000	KIPDR0= 172300
CHKMAN 020560 G	C#PNTX= 000015	EXTA 005772	G#RADD= 000040	KIPDR1= 172302
CHKTSS 016416	C#QIO = 000377	EXTEND 005770	G#RADL= 000120	KIPDR2= 172304
CKDROP 017262	C#RDBU= 000007	EXTFEA 002224 G	G#RADO= 000020	KIPDR3= 172306
CKEMAX 017162	C#REFG= 000047	E#END = 002100	G#XFER= 000004	KIPDR4= 172310
CKMSG 011420 G	C#RESE= 000033	E#LOAD= 000035	G#YES = 000010	KIPDR5= 172312
CKMSG2 011550 G	C#REVI= 000003	FATERR= 000060	HIADDR= 001400	KIPDR6= 172314
CKRAM 011154 G	C#RFLA= 000021	FATFLG 002220 G	HOE = 100000 G	KIPDR7= 172316
CKRAM2 011264 G	C#RPT = 000025	FERCM 011772	HPI1 046024	KTENAB 003134 G
CHDPKT 021274 G	C#SEFG= 000046	FIFEXP 012240 G	HPI2 046060	KTFLG 003132 G
CHPMEM 017740	C#SPRI= 000041	FIF1MS 012312	HPI3 046104	KTINIT 021070
CONFIG 017330	C#SVEC= 000037	FIF2MS 012361	IBE = 010000 G	KTOFF 017354
COUNT 002310 G	C#TPRI= 000013	FILLME 017502	IDU = 000040 G	KTON 017336
CSRADD 002204 G	DATA 002312 G	FNOINT 004215	IER = 020000 G	LERRMA 002170 G
CTAB 003164 G	DATASC 020312	FORCER 002174 G	IFALT 004256	LISTAL= 000001
CTABE 003176 G	DEBUGM 011702	FREE 003124 G	INCERK 017124	LOE = 040000 G
CTABM 003164 G	DEVcnt 002216 G	FREEMI 003130	INTCPC 016230	LOOPCN 002214 G

Symbol table

LOOPCO	013176	L10001	002174	L10073	037126	NXR	003740	PRI05	=	000240	G		
LOOPFL	003162	L10002	005766	L10074	037332	NXRERR	005736	PRI06	=	000300	G		
LOT	=	L10003	012114	L10075	043632	NXRX	003777	PRI07	=	000340	G		
L#ACP	002110	L10004	012132	L10076	041122	NXTU	022036	PRMESS	=	014312			
L#APT	002036	L10005	012150	L10077	041514	OFL	=	000100		PRMNO	002320	G	
L#AU	022366	L10006	012156	L10100	042404	ONEFIL	=	000000		PRMSG2	014632	G	
L#AUT	002070	L10007	012174	L10101	042632	O#APTS	=	000000		PRMSG0	015012		
L#AUTO	022572	L10010	012212	L10102	046000	O#AU	=	000001		PRMSG1	015057		
L#CCP	002106	L10011	012236	L10103	044100	O#BGNR	=	000001		PRMSG2	015115		
L#CLEA	022652	L10012	012310	L10104	044336	O#BGNS	=	000001		PROASC	014500		
L#CO	002032	L10013	012460	L10105	044556	O#DU	=	000001		PR1ASC	014545		
L#DEPO	002011	L10014	013174	L10106	045002	O#ERRT	=	000000		PST32W	003152	G	
L#DESC	003410	L10015	014022	L10107	046024	O#GNSW	=	000001		PUNIT	022320		
L#DESP	002076	L10016	014044	L10110	046144	O#POIN	=	000001		PW.D11	=	000021	
L#DEVP	002060	L10017	015560	MEMADD	014024	O#SETU	=	000000		PW.D13	=	000022	
L#DISP	002124	L10020	015566	MEMCK	021312	PASRPT	022070			PW.D22	=	000020	
L#DLY	002116	L10021	015574	MENASC	020527	PATCH	046304			PW.NOP	=	000000	
L#DTP	002040	L10022	015606	MENERR	020454	PATDAT	020310			PW.NO1	=	000023	
L#DTYP	002034	L10023	015630	MENRES	020556	PC.ERA	=	002400		PW.RDE	=	000024	
L#DU	022464	L10024	015656	MIVEC	=	000250				PW.RDR	=	000001	
L#DUT	002072	L10025	016016	MSA.FR	=	000006				PW.RDS	=	000005	
L#DVTY	003402	L10026	016326	MSA.NO	=	000000				PW.RFI	=	000003	
L#EF	002052	L10030	022316	MSA.NR	=	000004				PW.WCT	=	000006	
L#ENVI	002044	L10031	022462	MSA.VO	=	000002				PW.WFI	=	000004	
L#ETP	002102	L10032	022570	MSGEXP	012214					PW.WFM	=	000007	
L#EXP1	002046	L10033	022650	MSGLO	013134	PKHI	=	000004		PW.WMI	=	000010	
L#EXP4	002064	L10034	022676	MSGSTA	012420	PKLOW	=	000002		PW.WNP	=	000011	
L#EXPS	002066	L10035	023140	MSGSUB	014012	PKTADD	007554			PW.WTR	=	000002	
L#HARD	046004	L10036	023700	MS.ATT	=	000006				P.ACK	=	100000	
L#HIME	002120	L10037	023546	MS.EXT	=	000200				P.CMD	=	000037	
L#HPCP	002016	L10040	023630	MS.RSD	=	000001				P.CONT	=	000012	
L#HPTP	002022	L10041	024376	MS.RSF	=	000020				P.CVC	=	040000	
L#HM	002154	L10042	025070	MS.RST	=	000010				P.FMT	=	000140	
L#ICP	002104	L10043	026424	M2901	026150	PNT	=	001000		P.FORM	=	000011	
L#INIT	021572	L10044	025332	M8186	005554	PRAMPK	014046			P.GETS	=	000017	
L#LADP	002026	L10045	025632	M8189	005645	PRASC	014603			P.IE	=	000200	
L#LAST	047004	L10046	026130	NBA	=	002000				P.INIT	=	000013	
L#LOAD	002100	L10047	027530	NEWPAS	022024	PRBEXP	015550			P.MODE	=	007400	
L#LUN	002074	L10050	026776	NODEV	003114	PRBMSG	015416			P.OPP	=	020000	
L#PREV	002050	L10051	027346	NOINIT	004335	PRBREC	015552			P.POSI	=	000010	
L#NAME	002000	L10052	031002	NOINTR	004221	PRBTOT	015503			P.READ	=	000001	
L#PRIO	002042	L10053	030060	NOITS	002166	PRI	=	002000		P.SMB	=	010000	
L#PROT	021562	L10054	030404	NOMAN	020614	PRIADD	010160			P.WRIT	=	000005	
L#PRT	002112	L10055	034460	NOMEM	005460	PRIAO	010230			P.WRTC	=	000004	
L#REPP	002062	L10056	031410	NP.IR	=	000200				P.WRTS	=	000006	
L#REV	002010	L10057	031674	NP.LOO	=	000040				QVP	=	002202	G
L#RPT	022700	L10060	032140	NP.OUT	=	000100				RAMASC	014226		
L#SOFT	046136	L10061	032366	NP.WRP	=	000020				RAMDAT	002242	G	
L#SPC	002056	L10062	032632	NSI	004152	PRITAD	010274			RAMERR	015570	G	
L#SPCP	002020	L10063	033202	NSINIT	004407	PRITSS	006024			RAMEXP	015610	G	
L#SPTP	002024	L10064	035364	NUL	004527	PRIT0	010356			RAMFOR	010116		
L#STA	002030	L10065	040520	NULCR	004530	PRIT1	010421			RAMSIZ	002302	G	
L#SW	002164	L10066	035626	NXM	=	004000				RAMTAD	015576	G	
L#TEST	002114	L10067	036076	NXMFLG	003136	PRI00	=	000000		RCVHIA	002304	G	
L#TIML	002014	L10070	036302	NXMI	003142	PRI01	=	000040		RCVLOA	002306	G	
L#UNIT	002012	L10071	036470	NXML0	003140	PRI02	=	000100		RDERR	005206		
L10000	002162	L10072	036674	NXMTST	021466	PRI03	=	000140		RECMSG	002466	G	
						PRI04	=	000200					

Symbol table

RECV	002234	G	S1.ICE	002000	TST5ID	027400	T1LOOP	023500	T3TSSR	024726	
REGSAV	020220	S1.IEO	010000	TST6ID	030763	T1.1	023502	T4	025072	G	
RETERR	005372	S1.IFM	001000	TST7ID	034363	T1.2	023562	T4LOOP	025112		
REV	002226	G	S1.IHE	000400	TST8ID	035347	T10	040522	G	T4.1	025072
REWIND	011054	G	S1.IID	004000	TST9ID	040421	T10BFR	042672	T4.2	025334	
RMCHBE	000167	S1.IIR	020000	TSV2	002000	G	T10BUF	042734	T4.3	025634	
RMCHEN	000200	S1.I2R	040000	TSV3	002174	G	T10DAT	042660	T5	026426	G
RMMSGB	000215	S1.PAR	100000	TSV4	021562	G	T10DTA	042722	T5ADDR	027466	
RMMSGE	000234	S2.ATI	000010	TSV5	023462	G	T10INT	043370	T5LOOP	026444	
RMPKTB	000201	S2.BTI	000004	TSV6	046002	G	T10L00	040540	T5MEM	027430	
RMPKTE	000210	S2.DIM	000200	TTIBFR	177562	G	T10MBF	042754	T5.1	026450	
RMR	010000	S2.ILW	000100	TTICSR	177560	G	T10NBA	043051	T5.2	027016	
RMPACK	011150	S2.INR	000020	TTIVEC	000060	G	T10NIN	043277	T6	027532	G
SC	100000	S2.OUT	000040	T#ARGC	000003	T#OPAC	043133	T6INT	030553		
SCE	020000	S2.UND	000003	T#CODE	001130	T#OPKT	042650	T6LOOP	027550		
SCHERR	005300	TBLEND	003062	G	T#ERRN	002144	T10PKT	042712	T6NBA	030450	
SCHE	005013	TCOASC	006476	T#EXCP	000000	T10RST	043506	T6NINT	030631		
SDELAY	010660	TCOCOD	006676	T#FLAG	000040	T10RT2	043560	T6PACK	030440		
SELASC	020522	TEMP1	003116	G	T#GMAN	000000	T10SSR	043210	T6SSR	030475	
SELDAT	000004	TEMP2	003120	G	T#HILI	000776	T10.1	040540	T6TSBA	030711	
SEL2	000002	TERCLS	000016	T#LAST	000001	T10.2	041124	T6.1	027550		
SETMAP	017376	TESTNO	000013	T#LOLI	000000	T10.3	041516	T6.2	030074		
SETU	022122	TEXASC	006435	T#LSYM	010000	T10.4	042406	T7	031004	G	
SFFMSG	012152	G	TFCASC	006537	T#LTNO	000013	T11	043634	G	T7BFR	033256
SFHERR	003705	TIMEXP	015632	G	T#NEST	177777	T11BFR	045042	T7DATA	033240	
SFIERR	003652	TIMSG0	015660	T#NS0	000000	T11BF2	045112	T7INT	034211		
SFIMSG	012104	G	TINERR	012071	T#NS1	000005	T11DAT	045030	T7LOOP	031022	
SFPTBL	002164	G	TMPBFR	002632	G	T#NS2	000002	T11DTA	045100	T7NBA	033370
SIFLAG	003154	G	TNAM	016764	T#NS3	000003	T11L00	043652	T7NINT	034120	
SIMSG	012036	TRANST	002164	G	T#PTNU	000000	T11NBA	045132	T7PACK	033230	
SKIPT	003400	TSBA	000000	G	T#SAVL	177777	T11NIN	045514	T7REST	034412	
SOFINI	016054	G	TSBAH	000001	G	T#SEGL	177777	T11PAC	045020	T7SP	033250
SPACE	010466	G	TSBAH2	026240	T#SEKO	010000	T11PK2	045070	T7SSR	034031	
SPM1	046144	TSBAMS	026322	T#SUBN	000004	T11RES	045674	T7TSBA	034300		
SPM4	046174	TSDB	000000	G	T#TAGL	177777	T11RT2	045736	T7.1	031022	
SPM6	046224	TSDBH	000001	G	T#TAGN	010111	T11SR2	045440	T7.2	031424	
SPM7	046254	TSFCOD	007236	T#TEMP	000000	T11SSR	045364	T7.3	031676		
SRO	177572	TSREJ	000006	T#TEST	000013	T11TSB	045572	T7.4	032142		
SR1	177574	TSSDEF	006606	T#TSTM	177777	T11.1	043652	T7.5	032370		
SR2	177576	TSSR	000002	G	T#TSTS	000001	T11.2	044114	T7.6	032634	
SR3	172516	TSSRBI	003502	G	T##AU	010031	T11.3	044340	T72DAT	033276	
SSR	000200	TSSRFO	006415	G	T##AUT	010033	T11.4	044560	T72DON	033312	
STATCO	012462	TSSRM	000003	G	T##CLE	010034	T112RE	045172	T72NBA	033312	
SVCGBL	000000	TSSX	004020	TSTBLK	002752	G	T113RE	045253	T72REJ	033443	
SVCINS	000000	TSTCNT	002212	G	T#HAR	010107	T114RE	045303	T73REJ	033542	
SVCSUB	000001	TSTEND	017000	G	T##HMJ	010000	T2	023702	G	T74REJ	033635
SVCTAG	000000	TSTFLA	002314	G	T##INI	010030	T2LOOP	023720	T75REJ	033733	
SVCTST	000001	TSTL00	016536	G	T##MSG	010025	T2SSR	024276	T8	034462	G
S#LSYM	010000	TSTPTR	002316	G	T##PRO	010027	T2TSBA	024164	T8BFR	035112	
SO.IDB	000010	TSTSET	016570	G	T##RPT	010035	T2TSSR	024231	T8DATA	035100	
SO.IFB	000002	TST1ID	023660	G	T##SEG	010000	T23A	003144	G	T8LOOP	034500
SO.IFP	000001	TST10I	043457	G	T##SOF	010110	T23B	003146	G	T8NVCK	035167
SO.ILD	000020	TST10I	043457	G	T##SRV	010026	T3	024400	G	T8PACK	035070
SO.ION	000040	TST11I	045642	G	T##SUB	010106	T3BFLG	003150	G	T8SSR	035260
SO.IRD	000100	TST2ID	024350	G	T##SW	010001	T3LOOP	024416	T8VCK	035132	
SO.IRW	000004	TST3ID	025043	G	T#TES	010102	T3SSR	024772	T9	035366	G
SO.ISP	000200	TST4ID	026402	G	T1	023462	G	T3TSBA	024662	T9BFR	037372

B15

Symbol table

T9DATA	037360	T95REJ	037771	WF.I4R=	000001	XSONEF=	002000	X2.EXT=	000200
T9INT	040247	UAM	= 000200	WRTCHR	010662	XSOONL=	000100	X2.OPM=	100000
T9LOOP	035410	UNITN	002200	WRTERR	005113	XSOPED=	000010	X2.RCE=	040000
T9NBA	037426	UNREC	= 000006	WRTMSG	005056	XSORLL=	010000	X2.REV=	000077
T9NINT	040156	USI	004123	WSMBK	021304	XSORLS=	040000	X2.SPA=	035400
T9PACK	037350	WAITF	016330	XFERAS	016020	XSOTMK=	100000	X2.UNI=	000007
T9REST	040446	WC.IFA=	000200	XNXM	016456	XSOVCK=	000020	X2.WCF=	002000
T9SSR	040067	WC.IFE=	000002	XORBF0	007674	XSOWLE=	004000	X3.DCK=	000010
T9TSBA	040336	WC.IGO=	000001	XORFOR	010012	XSOWLK=	000004	X3.MBZ=	000006
T9.1	035410	WC.IRE=	000010	XSTO	= 000006	XXCOMM	003122	X3.MDE=	177400
T9.2	035666	WC.IRW=	000004	XST1	= 000010	X#ALWA=	000000	X3.OPI=	000100
T9.3	036100	WC.IOT=	000100	XST2	= 000012	X#FALS=	000040	X3.REV=	000040
T9.4	036304	WC.IIT=	000040	XST3	= 000014	X#OFFS=	000400	X3.RIB=	000001
T9.5	036472	WC.ISR=	000020	XST4	= 000016	X#TRUE=	000020	X3.SPA=	000200
T9.6	036676	WF.IED=	000010	XSOBOT=	000002	X1.COR=	020000	X3.TRF=	000020
T9.7	037130	WF.IER=	000004	XSOEOT=	000001	X1.DLT=	100000	X4.HSP=	100000
T92DAT	037412	WF.IHI=	000200	XSOIE	= 000040	X1.MBZ=	017375	X4.MBZ=	017400
T92DON=	037426	WF.IRE=	000040	XSOILA=	000400	X1.RBP=	000400	X4.RCE=	040000
T92REJ	037501	WF.IWF=	000020	XSOILC=	001000	X1.SPA=	040000	X4.TSM=	020000
T93REJ	037600	WF.IWR=	000100	XSOLET=	020000	X1.UNC=	000002	X4.WRC=	000377
T94REJ	037673	WF.I3R=	000002	XSOMOT=	000200	X2.BUF=	000100		

. ABS. 047004 000 (RW,I,GBL,ABS,OVR)
 000000 001 (RW,I,LCL,REL,CON)
 ABS 000000 002 (RW,I,LCL,REL,CON)

Errors detected: 0

*** Assembler statistics

Work file reads: 273
 Work file writes: 268
 Size of work file: 28912 Words (113 Pages)
 Size of core pool: 19684 Words (75 Pages)
 Operating system: RSX-11M/PLUS (Under VAX/VMS)

Elapsed time: 00:04:47.69
 CVTSAC,CVTSAC/-SP=SVC/PL,TSV1A,TSV22A,TSV3B,TSV4,TSV5A,TSV6