

TSV05

TSV05 CTRL LT1
CVTSAA0

AH-T094A-MC
FICHE 1 OF 2

SEP 1982
COPYRIGHT © 1982
MADE IN USA



The main body of the document is a large grid of approximately 20 columns and 20 rows of small, illegible text blocks. Each block appears to be a page of a document, possibly containing technical specifications or data. The text is too small to be read accurately.

.REM_
IDENTIFICATION

PRODUCT ID: AC-T093A-MC
PRODUCT TITLE: CVTSAAO TSV05 CTRL LT1
AUTHOR: DICK GORDON
MAINTAINER: SCOTT SNOWDON
DATE: MARCH 08, 1982

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

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

COPYRIGHT (C) 1982,1982 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL
DEC

PDP
DECUS

UNIBUS
DECTAPE

MASSBUS

TABLE OF CONTENTS

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

1.0 GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

THIS IS A PDP-11/23 RESIDENT DIAGNOSTIC WHICH CHECKS THE FUNCTIONALITY OF A TSV05 MAGTAPE SUBSYSTEM WHILE CONNECTED TO A PDP-11/23 SYSTEM (Q-BUS). THE PROGRAM PROVIDES ERROR MESSAGES WHICH IDENTIFY FAILING FUNCTIONS THAT AID IN THE REPAIR OF THE DEVICE. THIS DIAGNOSTIC CONSIST OF 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

PDP-11/23 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 (HSAAA.SYS VERSION 34 OR LATER)
PDP-11 DIAGNOSTIC LOADER/MONITOR (XXDP+)

1.3 RELATED DOCUMENTS AND STANDARDS

DIGITAL EQUIPMENT CORPORATION DOCUMENTS:

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

1.4 DIAGNOSTIC HIERARCY PREREQUISITES

FUNCTIONAL PDP-11/23 CENTRAL PROCESSOR AND MEMORY
FUNCTIONAL CONSOLE TERMINAL
FUNCTIONAL STANDALONE DIAGNOSTIC SUPERVISOR
FUNCTIONAL DIAGNOSTIC LOADER/MONITOR (XXDP+)

1.5 ASSUMPTIONS

ALL HARDWARE EXCEPT THE HARDWARE UNDER TEST IS ASSUMED TO WORK PROPERLY OR FALSE ERRORS CAN BE REPORTED.
THE TAPE BEING USED ON THE 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 PDP-11/23 DIAGNOSTIC SUPERVISOR COMPATIBLE PROGRAM. ALL LOADING AND RUNTIME INSTRUCTIONS CAN BE REFERENCED IN THE CHQUS XXDP+ USERS GUIDE, DOCUMENT NUMBER AC-F348E-MC. THE USER ENTRY IS IN QUOTES.

BOOT THE DIAGNOSTIC MEDIA

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

2.2 SWITCHES

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

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

EXAMPLE OF SWITCH USAGE:

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

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

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

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

2.3 FLAGS

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

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

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

*ERROR MESSAGES ARE DESCRIBED IN SECTION 3.1

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

```
/FLAGS:LOE:IER:BOE
```

2.4 HARDWARE QUESTIONS

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

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

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

TSBA/TSDB = 172520, VECTOR = 224

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

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

UNIT 0

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

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

THE ADDRESS AND VECTOR QUESTIONS WILL BE ASKED FOR EACH OF THE NUMBER OF UNITS (CONTROLLERS) SPECIFIED IN THE "# UNITS?" QUESTION. LOGICAL UNIT NUMBERS ARE ASSIGNED IN ORDER, BEGINNING AT 0. UP TO FOUR UNITS CAN BE SELECTED FOR TESTING AS FOLLOWS:
UP TO 4 TSV05 CONTROLLERS PER 11/23 AND UP TO 2 DRIVES PER CONTROLLER

2.5 SOFTWARE QUESTIONS

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

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

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

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

2.6 EXTENDED P-TABLE DIALOGUE

WHEN YOU ANSWER THE HARDWARE QUESTIONS, YOU ARE BUILDING ENTRIES

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

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

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

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

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

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

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

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

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

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

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

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

UNIT 7
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 6,7<CR>
Q-FACTOR (O) 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 (O) ? 160000<CR>
SUB-DEVICE # (O) ? 0-7<CR>
Q-FACTOR (O) 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: 100020	XOR: 000000
WORD #1	EXPD: 000012	RECV: 000012	XOR: 000000
WORD #2	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #3	EXPD: 000010	RECV: 000010	XOR: 000000
WORD #4	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #5	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #6	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #7	EXPD: 000000	RECV: 000000	XOR: 000000
WORD #8	EXPD: 070217	RECV: 070217	XOR: 000000
WORD #9	EXPD: 000074	RECV: 000034	XOR: 000040

ERROR MESSAGE EXAMPLE 2

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

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 (PDP-11/23)

DR>STA/FLA:PNT:HOE

UNITS (D) ? 1

UNIT 0

DEVICE ADDRESS (O) 172520 ? <CR>

VECTOR (O) 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 PDP-11/23 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 ITERATION 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 11/23 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 DC005 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 11/23 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 XST0 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 XST0 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

```

2          .TITLE  TSV2 - PROGRAM HEADER
3          .SBTTL  PROGRAM HEADER
4
10         .MCALL  SVC
11 000000  SVC          ; INITIALIZE SUPERVISOR MACROS
12         .ENABLE LC
13         .NLIST  BEX,CND
19 000000  .ENABL  ABS,AMA
20         .=2000
21 002000  BGNMOD  TSV2
   002000
22
23         :++
24         : THE PROGRAM HEADER IS THE INTERFACE BETWEEN
25         : THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
26         :--
27
28
29 002000  POINTER BGNSW,BGNSFT,BGNAU,BGNDU,BGNRPT
30 002000  HEADER  CVTSA,A,0,655.,0
   002000  L$NAME::          ;DIAGNOSTIC NAME
   002000      103          .ASCII /C/
   002001      126          .ASCII /V/
   002002      124          .ASCII /T/
   002003      123          .ASCII /S/
   002004      101          .ASCII /A/
   002005      000          .BYTE  0
   002006      000          .BYTE  0
   002007      000          .BYTE  0
   002010  L$REV::          ;REVISION LEVEL
   002010      101          .ASCII /A/
   002011  L$DEPO::        ;0
   002011      060          .ASCII /0/
   002012  L$UNIT::        ;NUMBER OF UNITS
   002012  000000          .WORD   0
   002014  L$TIML::        ;LONGEST TEST TIME
   002014  001217          .WORD  655.
   002016  L$HPCP::        ;PTR. TO H.W. QUES.
   002016  045634          .WORD  L$HARD
   002020  L$SPCP::        ;PTR. TO S.W. QUES.
   002020  045766          .WORD  L$SOFT
   002022  L$HPTP::        ;PTR. TO DEF. H.W. PTABLE
   002022  002154          .WORD  L$HW
   002024  L$SPTP::        ;PTR. TO S.W. PTABLE
   002024  002164          .WORD  L$SW
   002026  L$LADP::        ;DIAG. END ADDRESS
   002026  046404          .WORD  L$LAST
   002030  L$STA::         ;RESERVED FOR APT STATS
   002030  000000          .WORD   0
   002032  L$CO::          ;
   002032  000000          .WORD   0
   002034  L$DTYP::        ;DIAGNOSTIC TYPE
   002034  000000          .WORD   0
   002036  L$APT::         ;APT EXPANSION
   002036  000000          .WORD   0
   002040  L$DTP::         ;PTR. TO DISPATCH TABLE
   002040  002124          .WORD  L$DISPATCH

```

002042		L\$PRIO::		;DIAGNOSTIC RUN PRIORITY
002042	000000	.WORD	0	
002044		L\$ENVI::		;FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000	.WORD	0	
002046		L\$EXP1::		;EXPANSION WORD
002046	000000	.WORD	0	
002050		L\$MREV::		;SVC REV AND EDIT #
002050	003	.BYTE	C\$REVISION	
002051	003	.BYTE	C\$EDIT	
002052		L\$EF::		;DIAG. EVENT FLAGS
002052	000000	.WORD	0	
002054	000000	.WORD	0	
002056		L\$SPC::		
002056	000000	.WORD	0	
002060		L\$DEVP::		; POINTER TO DEVICE TYPE LIST
002060	003376	.WORD	L\$DVTYP	
002062		L\$REPP::		;PTR. TO REPORT CODE
002062	022654	.WORD	L\$RPT	
002064		L\$EXP4::		
002064	000000	.WORD	0	
002066		L\$EXP5::		
002066	000000	.WORD	0	
002070		L\$AUT::		;PTR. TO ADD UNIT CODE
002070	022342	.WORD	L\$AU	
002072		L\$DUT::		;PTR. TO DROP UNIT CODE
002072	022440	.WORD	L\$DU	
002074		L\$LUN::		;LUN FOR EXERCISERS TO FILL
002074	000000	.WORD	0	
002076		L\$DESP::		;POINTER TO DIAG. DESCRIPTION
002076	003404	.WORD	L\$DESC	
002100		L\$LOAD::		;GENERATE SPECIAL AUTOLOAD EMT
002100	104035	EMT	ESLOAD	
002102		L\$ETP::		;POINTER TO ERRTABL
002102	000000	.WORD	0	
002104		L\$ICP::		;PTR. TO INIT CODE
002104	021546	.WORD	L\$INIT	
002106		L\$CCP::		;PTR. TO CLEAN-UP CODE
002106	022626	.WORD	L\$CLEAN	
002110		L\$ACP::		;PTR. TO AUTO CODE
002110	022546	.WORD	L\$AUTO	
002112		L\$PRT::		;PTR. TO PROTECT TABLE
002112	021536	.WORD	L\$PROT	
002114		L\$TEST::		;TEST NUMBER
002114	000000	.WORD	0	
002116		L\$DLY::		;DELAY COUNT
002116	000000	.WORD	0	
002120		L\$HIME::		;PTR. TO HIGH MEM
002120	000000	.WORD	0	

33
34
35
36
37
38
39
40
41

.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
002122		.WORD 11
002122	000013	
002124		L\$DISPATCH::
002124	023436	.WORD T1
002126	023656	.WORD T2
002130	024354	.WORD T3
002132	025046	.WORD T4
002134	026402	.WORD T5
002136	027506	.WORD T6
002140	030764	.WORD T7
002142	034352	.WORD T8
002144	035256	.WORD T9
002146	040372	.WORD T10
002150	043474	.WORD T11

```
43 .SBTTL DEFAULT HARDWARE P-TABLE
44
45 :++
46 : THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
47 : THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
48 : IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
49 :--
50 002152 BGNHW DFPTBL ;DEFAULT HARD-P-TABLE
    002152 000003 .WORD L10000-L$HW/2
    002154 L$HW::
    002154 DFPTBL::
51
52 002154 172520 .WORD 172520 ; 1ST (OF 2) REGISTERS.
53 002156 000224 .WORD 224 ; INTERRUPT VECTOR
54 002160 000200 .WORD PRI04 ; INTERRUPT PRIORITY.
55 002162 ENDPHW
    002162 L10000:
```



```
57                                     .SBTTL  SOFTWARE P-TABLE
58
59                                     :++
60                                     : THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
61                                     : PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
62                                     :--
63 002162                               BGNSW   SFPTBL
    002162 000004                       .WORD  L10001-L$$W/2
    002164
    002164
64
65 002164 000000                       L$$W::
66 002166 000000                       SFPTBL::
67
68                                     TRANSTST:: .WORD  0      : ENABLE TEST OF TRANSPORT(S) IF =1
69 002170 000017                       NOITS::   .WORD  0      : INHIBIT ITERATION OPTION.
70 002172 000310                       : ... 0 = ITERATE.
71 002174                               : ...NZ = INHIBIT ITERATE.
    002174                               LERRMAX:: .WORD  15.   : LOCAL (PER TEST) ERROR LIMIT
72                                     GERRMAX:: .WORD  200.  : GLOBAL (PER UNIT) ERROR LIMIT
73 002174                               L10001:
                                     ENDSW
                                     ENDMOD
```

7
8
13
19
20 002174
002174
21
22
23
24
25
26
27
28
29
33 002174

.TITLE TSV3 - GLOBAL AREAS
.SBTTL GLOBAL EQUATES SECTION

BGNMOD TSV3
TSV3::

.SBTTL GLOBAL EQUATES SECTION

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

EQUALS ; GET STANDARD EQUATES.

: BIT DIFINITIONS

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

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

: EVENT FLAG DEFINITIONS
: EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

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

:
:

```
000340      ; PRIORITY LEVEL DEFINITIONS
000300      .
000240      PRI07== 340
000200      PRI06== 300
000140      PRI05== 240
000100      PRI04== 200
000040      PRI03== 140
000000      PRI02== 100
           PRI01== 40
           PRI00== 0
```

```
           ; OPERATOR FLAG BITS
000004      .
000010      EVL==      4
000020      LOT==     10
000040      ADR==     20
000100      IDU==     40
000200      ISR==    100
000400      UAM==    200
001000      BOE==    400
002000      PNT==   1000
004000      PRI==   2000
010000      IXE==   4000
020000      IBE==  10000
040000      IER==  20000
100000      LOE==  40000
           HGE== 100000
```

34
35 002174

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

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

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

40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96

.SBTTL TSV05 REGISTER AND PACKET DEFINITIONS

:
: SOME GENERAL EQUATES.
:

000004	ERRVEC==	4	: POINTER TO ERROR VECTOR FOR BUS TIME OUT.
000060	TTIVEC==	60	: INTERRUPT VECTOR FOR CONSOLE INPUT
177560	TTICSR==	177560	: BUS ADDRESS OF CONSOLE INPUT
177562	TTIBFR==	177562	: CONSOLE INPUT DATA BUFFER
177520	BDVPCR==	177520	: BDV11 PAGE CONTROL REGISTER

:+
:BIT DEFINITIONS FOR TSSR REGISTER
:-

100000	SC=	BIT15	:SPECIAL CONDITION
040000	BIE=	BIT14	:BUS INTERFACE ERROR
020000	SCE=	BIT13	:SANITY CHECK ERROR
010000	RMR=	BIT12	:MODIFICATION REFUSED
004000	NXM=	BIT11	:NONEXISTANT MEMORY ERROR
002000	NBA=	BIT10	:NEED BUFFER ADDRESS
001400	HIADDR=	BIT9!BIT8	:EXTENDED ADDRESS BITS
000200	SSR=	BIT7	:SUB SYSTEM READY
000100	OFL=	BIT6	:OFF LINE BIT
000060	FATERR=	BIT4!BIT5	:FATAL TERMINATION ERROR CODES
000016	TERCLS=	BIT3!BIT2!BIT1	:TERMINATION CODES

:+
:BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 0
:(XST0)
:-

100000	XSOTMK=	BIT15	:TAPE MARK DETECTED
040000	XSORLS=	BIT14	:RECORD LENGTH SHORT
020000	XSOLET=	BIT13	:LOGICAL END OF TAPE
010000	XSORLL=	BIT12	:RECORD LENGTH LONG
004000	XSOWLE=	BIT11	:WRITE LOCK ERROR
002000	XSONEF=	BIT10	:NON EXECUTABLE FUNCTION
001000	XSOILC=	BIT9	:ILLEGAL COMMAND
000400	XSOILA=	BIT8	:ILLEGAL ADDRESS
000200	XSOMOT=	BIT7	:TAPE IN MOTION
000100	XSOONL=	BIT6	:TRANSPORT ON LINE
000040	XSOIE=	BIT5	:INTERRUPT ENABLE
000020	XSOVCK=	BIT4	:VOLUME CHECK BIT
000010	XSOPED=	BIT3	:PHASE ENCODED DRIVE
000004	XSOWLK=	BIT2	:WRITE LOCKED
000002	XSOBOT=	BIT1	:BEGINNING OF TAPE
000001	XSOEOT=	BIT0	:END OF TAPE

:+
:BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 1
:(XST1)

```

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

```

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



```

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

```

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

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

340
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

```
.SBTTL SPECIAL MACROS AND OPDEFS.  
  
:+  
:SAVE GENERAL REGS 1 TO 5  
:-  
  
    .MACRO SAVREG  
    JSR    R5,REGSAV  
    .ENDM  
  
:+  
: MACRO TO FORCE AN ERROR  
:-  
    .MACRO FORCERROR TAG,NOTSSR  
    .NLIST  
    .IF NDF LISTALL, .NLIST  
    .LIST  
    .IF B NOTSSR  
        MOV    TSSR(R5),R1    ;READ TSSR  
    .ENDC  
        MOV    FORCER,FORCER  ;IS FORCER SET? (LEAVE C BIT ALONE)  
        BNE    TAG            ;BR IF YES  
    .NLIST  
    .IF NDF LISTALL, .LIST  
    .LIST  
    .ENDM  
  
:+  
: MACRO TO FORCE AN EXIT TO AVOID SECTION ITERATIONS  
: WILL EXIT TO A LABEL IF FORCER IS NEGATIVE  
: SO TO FORCE ERRORS AND EXIT ON 1 ERROR SET  
: FORCER TO 177777  
: TO FORCE ERRORS AND ITERATIONS SET FORCER TO 1.  
:-  
    .MACRO FORCEEXIT TAG  
    .NLIST  
    .IF NDF LISTALL, .NLIST  
    .LIST  
        MOV    FORCER,FORCER  ;IS FORCER NEGATIVE?  
        BMI    TAG            ;BR IF YES  
    .NLIST  
    .IF NDF LISTALL, .LIST  
    .LIST  
    .ENDM  
  
:+  
: MACRO TO INCREMENT ERROR COUNTS  
:-  
    .MACRO NEXT.ERRNO  
    .NLIST  
    ::: .IF NDF LISTALL, .NLIST  
        ERRNO=ERRNO+1  
    ::: .IF NDF LISTALL, .LIST  
    .LIST  
    .ENDM  
  
:+
```

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

000000

002174 000000

;MACRO TO PERFORM XOR
;:-

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

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

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

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

.SBTTL GLOBAL DATA SECTION

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

```

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

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

```

463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479 002750
 480 002750 000000
 481 002752 177777
 482 002754 000001
 483 002756 000002
 484 002760 000004
 485 002762 000010
 486 002764 000020
 487 002766 000040
 488 002770 000100
 489 002772 000200
 490 002774 000400
 491 002776 001000
 492 003000 002000
 493 003002 004000
 494 003004 010000
 495 003006 020000
 496 003010 040000
 497 003012 100000
 498 003014 177776
 499 003016 177775
 500 003020 177773
 501 003022 177767
 502 003024 177737
 503 003026 177677
 504 003030 177577
 505 003032 177377
 506 003034 176777
 507 003036 175777
 508 003040 173777
 509 003042 167777
 510 003044 157777
 511 003046 137777
 512 003050 077777
 513 003052 125252
 514 003054 052525
 515 003056

.SBTTL TSTBLK - TEST DATA TABLE

```

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

```

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

TBLEND==.
  
```

```

517                                     .SBTTL GLOBAL ENVIRONMENT STORAGE
518
519                                     ; STORAGE FOR DEVICE REGISTERS
520
521 003056 000000 100000 000000 DUMMY: 0,100000,0,0 ; DUMMY DEVICE REGISTERS...
522 003066 000000 000000 000000 0,0,0,0,0,0,0,0,0 ; ...FOR MULTI-UNIT CHECKOUT.
523
524
525
526 003106 000000 DUFLG:: .WORD 0 ; 'DROPPED UNIT' FLAG.
527                                     ; INHIBITS CODE IN 'CLEAN-UP'.
528 003110 000000 NODEV:: .WORD 0 ; FLAG TO SAY NO DEVICE.
529
530 003112 000000 TEMP1:: .WORD 0 ; SOME TEMP LOCATIONS.
531 003114 000000 TEMP2:: .WORD 0
532 003116 000000 XXCOMM:: .WORD 0 ; XXDP+ COMM BLOCK POINTER.
533 003120 000000 FREE:: .WORD 0 ; 1ST FREE MEMORY ADDRESS...
534 003122 000000 FRESIZ:: .WORD 0 ; ...AND SIZE (IN WORDS).
535 003124 000000 FREEHI: .WORD 0 ; LAST WORD IN FREE SPACE
536 003126 000000 KTFLG:: .WORD 0 ; KT11, MEM AVAIL FLAG -
537                                     ; - .WORD 0 = <24K OR NO KT -
538                                     ; - NZ = >24K AND KT.
539 003130 000000 KTENABLE:: .WORD 0 ; SET BY TEST ROUTINES TO FLAG >28K UNDER TEST
540 003132 000000 NXMFLG:: .WORD 0 ; SET IF WE CAN TEST CLEARED OTHERWISE
541 003134 000000 NXMLO:: .WORD 0 ; NXM LO ADDRESS BITS
542 003136 000000 NXMHI:: .WORD 0 ; NXM HI ADDRESS BITS FOR DAL'S 16-21
543 003140 000000 T23A:: .WORD 0 ; 11/23A FLAG
544 003142 000000 T23B:: .WORD 0 ; 11/23B FLAG
545 003144 000000 T3BFLG:: .WORD 0 ; TEST 3B FLAG ^0
546 003146 002000 PST32W:: .WORD 2000 ; 32W BLOCK ADDRESS FOR 32K START
547 003150 000000 SIFLAG:: .WORD 0
548 003152 000000 BADDAT:: .WORD 0 ; ACTUAL DATA
549 003154 000000 GDDAT:: .WORD 0 ; EXPECTED DATA
550 003156 000000 LOOPFL:: .WORD 0
551 003160 CTAB:: .WORD 0 ; CONFIGURATION TABLES.
552 003160 000000 CTABM:: .WORD 0 ; CONFIG WORK.
553 003162 000000 .WORD 0
554 003164 000000 .WORD 0
555 003166 000000 .WORD 0
556 003170 177777 .WORD -1 ; END OF MEM TABLE.
557 003172
558 CTABE::
559 ; ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:
560
561 0 = UNIT NOT TESTED
562 100000 = UNIT ONLINE, NO ERRORS
563 10XXXX = UNIT ONLINE, ENCOUNTERED XXXX ERRORS
564 160000 = UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
565 160001 = UNIT DROPPED, NOT IDLE AT START
566 14XXXX = UNIT DROPPED, ENCOUNTERED XXXX ERRORS
567 003172
568 003372 000000 ERTABL: .BLKW 64.
569 ERTABE: .WORD 0
570 003374 000000 SKIPT: .WORD 0 ; 1=SKIP SUBTEST 0=NO SKIP OF SUBTEST
    
```



```

572                                     .SBTTL GLOBAL TEXT MESSAGES
573
574 :++
575 : THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
576 : MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
577 : MORE THAN ONE TEST.
578 :--
579
580
581 :+
582 : NAMES OF DEVICES SUPPORTED
583 :--
584
585 003376          DEVTYP <TSV05>
      003376          LSDVTYP::
      003376          .ASCIZ /TSV05/
      124          123          126          .EVEN
586
588 :+
589 : TEST DESCRIPTION
590 :--
591 003404          DESCRIPT <**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****>
      003404          L$DESC::
      003404          .ASCIZ /**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****/
      052          052          052          .EVEN
612
613
614
615 :+
616 : BIT TO ASCII CONVERSION FOR TSSR REGISTER
617 :--
618
619 003476 003536 003541 003545 TSSRBIT::          .WORD 1$,2$,3$,4$,5$,6$,7$,8$
620 003516 003577 003603 003607          .WORD 9$,10$,11$,12$,13$,14$,15$,16$
621 003536          123          103          000 1$:          .ASCIZ 'SC'
622 003541          102          111          105 2$:          .ASCIZ 'BIE'
623 003545          123          103          105 3$:          .ASCIZ 'SCE'
624 003551          122          115          122 4$:          .ASCIZ 'RMR'
625 003555          116          130          115 5$:          .ASCIZ 'NXM'
626 003561          116          102          101 6$:          .ASCIZ 'NBA'
627 003565          102          111          124 7$:          .ASCIZ 'BIT9'
628 003572          102          111          124 8$:          .ASCIZ 'BIT8'
629 003577          123          123          122 9$:          .ASCIZ 'SSR'
630 003603          117          106          114 10$:         .ASCIZ 'OFL'
631 003607          102          111          124 11$:         .ASCIZ 'BIT5'
632 003614          102          111          124 12$:         .ASCIZ 'BIT4'
633 003621          102          111          124 13$:         .ASCIZ 'BIT3'
634 003626          102          111          124 14$:         .ASCIZ 'BIT2'
635 003633          102          111          124 15$:         .ASCIZ 'BIT1'
636 003640          102          111          124 16$:         .ASCIZ 'BIT0'
637          .EVEN
638 003646          124          123          123 SFIERR: .ASCIZ 'TSSR ERROR AFTER SOFT INIT'
639 003701          124          123          123 SFHERR: .ASCIZ 'TSSR ERROR AFTER BUS RESET'
640 003734          040          040          116 NXR:      .ASCIZ / NON-EXISTANT DEVICE REGISTER/
641 003773          045          101          040 NXR:      .ASCIZ /% ADDRESS: %06/
642 004014          045          101          040 TSSX:     .ASCII  /% TSBA,TSSR EXP'D: %06%A,%06%N/
643 004054          045          101          040 TSSX:     .ASCII  /% TSBA,TSSR REC'D: %06%A,%06/

```

```
644 004113 045 116 045 FUSI: .ASCII /%N%/
645 004117 040 040 125 USI: .ASCIZ / UNEXPECTED INTERRUPT/
646 004146 040 040 111 NSI: .ASCIZ / INTERRUPT EXPECTED, NOT RECEIVED/
647 004211 045 116 045 FNOINTR: .ASCII /%N%/
648 004215 040 040 116 NOINTR: .ASCIZ / NO INTERRUPT WAS GENERATED/
649 004252 040 040 111 IFAULT: .ASCIZ / INTERRUPT FAULT/
650 004274 045 101 040 INTX: .ASCIZ /%A CPU PC: %06% TSBA: %06/
651 004331 040 040 042 NOINIT: .ASCIZ / 'BUS-INIT' DIDN'T INITIALIZE CONTROLLER/
652 004403 040 040 042 NSINIT: .ASCIZ / 'SOFT-INIT' DIDN'T INITIALIZE THE DPU/
653 004453 040 040 042 BRINIT: .ASCIZ / 'BUS-RESET' DIDN'T INITIALIZE THE DPU/
654
655 004523 000 NUL: .ASCIZ //
656 004524 045 116 000 NULCR: .ASCIZ /%N/
657 004527 045 101 040 EXPGOT: .ASCIZ /%A EXP'D: %06%, REC'D: %06/
658 004563 045 116 045 EXPGT2: .ASCIZ /%N% EXP'D: %06%, %06%N% REC'D: %0%, %06/
659 004637 045 101 040 DUAD12: .ASCIZ /%A REG(W) WRITTEN TO: %06% REG(R) READ; EXP'D: %06%, REC'D: %06/
660 004741 122 101 115 PKTRAM:: .ASCIZ 'RAM Contents Do Not Match Packet Sent'
661 005007 040 040 103 SCME: .ASCIZ / CONFIG DOESN'T MATCH MFG. MASTER/
662 005052 127 122 111 WRTMSG: .ASCIZ 'WRITE CHARACTERISTICS Failed'
663 005107 124 123 123 WRTERR: .ASCIZ 'TSSR Incorrect After WRITE Command, More Bits Set Than SSR'
664 005202 124 123 123 RDERR: .ASCIZ 'TSSR Incorrect After READ Command, More Bits Set Than SSR'
665 005274 106 101 124 SCHERR: .ASCIZ 'FATAL ERROR IN SUBTEST - CHECK TAPE, CABLES, TRANSPORT etc.'
666 005366 105 122 122 RETERR: .ASCIZ 'ERROR IN SUBTEST - WRITE DATA RETRY FIVE TIMES FAILED'
667 005454 045 116 045 NOMEM: .ASCIZ '%N% ***** NO NXM ADDRESS--CANNOT TEST NXM TIMEOUT. *****%N'
668 005550 045 116 045 M8186: .ASCIZ '%N% ***** 11/23A SYSTEM *****%N'
669 005641 045 116 045 M8189: .ASCIZ '%N% ***** 11/23B SYSTEM *****%N'
670
671 .EVEN
672
673
```

```

675
676
677
678
679
680
681
682
683 005732
    005732
684 005732
    005732 013746 003110
    005736 012746 003773
    005742 012746 000002
    005746 010600
    005750 104415
    005752 062706 000006
685 005756 004737 005764
686 005762
    005762
    005762 104423
687
688
689
690
691
692
693 005764 005727
694 005766 000000
695 005770 001402
696 005772 004777 177770
697 005776
    005776 012746 004524
    006002 012746 000701
    006006 010600
    006010 104415
    006012 062706 000004
698 006016 000207
    
```

.SBTTL GLOBAL ERROR REPORT SECTION

```

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

```

BGNMSG NXRERR ;NON-EXISTANT DEVICE REGISTER.
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,@EXTA ; APPEND EXTENSION TEXT.
1$: PRINTX #NULCR ; PRINT A BLANK LINE
MOV #NULCR,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #4,SP
RTS PC
    
```

```

701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719 006020
720 006020
721 006024 010104
722 006026
    006026 010446
    006030 012746 006411
    006034 012746 000002
    006040 010600
    006042 104414
    006044 062706 000006
723 006050 010400
724 006052 004737 016034
725 006056 103410
726 006060
    006060 012746 006631
    006064 012746 000001
    006070 010600
    006072 104415
    006074 062706 000004
727 006100 010403
728 006102 042703 001476
729 006106 001434
730 006110 012702 002630
731 006114 012701 003476
732 006120 005703
733 006122 001413
734 006124 000241
735 006126 006103
736 006130 103006
737 006132 011100
738 006134 112022
739 006136 001376
740 006140 112762 000054 177777
741 006146 005721
742 006150 000763
743 006152 105042
744 006154
    006154 012746 002630
    006160 012746 006602
    
```

```

.SBTTL PRITSSR - PRINT TSSR CONTENTS

:
:
:ROUTINE TO DISPLAY THE CONTENTS, AND BIT DEFINITIONS, OF
:THE TSSR REGISTER. THIS ROUTINE IS NORMALLY CALLED ONLY
:BY A MESSAGE PRINTING ROUTINE
:
:INPUTS:
:
:      R1      CONTENTS OF TSSR
:
:SUBORDINATE ROUTINES:
:
:      CHKAMB  CHECK FOR AMBIGUOUS CONTENTS
:
:
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$PNTB
    ADD #6,SP
    MOV R4,R0
    JSR PC,CHKAMB   ;GET TSSR BACK FOR CHKAMB
    BCS 5$         ;ARE CONTENTS AMBIGUOUS ?
    PRINTX #AMBTSSR ;BRANCH IF NOT
    MOV #AMBTSSR,-(SP) ;SHOW CONTENTS ARE AMBIGUOUS
    MOV #1,-(SP)
    MOV SP,R0
    TRAP C$PNTX
    ADD #4,SP
5$: MOV R4,R3 ;CONTENTS OF TSSR
    BIC #HIADDR!FATERR!TERCLS,R3 ;CLEAR ALL MULTIPLE BIT FIELDS
    BEQ 20$ ;NO BITS ARE SET
    MOV #TMPBFR,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 #' ,-(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)
    
```

```

006164 012746 000002      MOV      #2,-(SP)
006170 010600      MOV      SP,R0
006172 104415      TRAP    C$PNTX
006174 062706 000006      ADD     #6,SP
745
746 006200 010403      20$:    MOV      R4,R3          ;GET THE TSSR CONTENTS
747 006202 042703 177761      BIC     #^CTERCLS,R3    ;CLEAR ALL BUT TERMINATION
748 006206 016303 006672      MOV     TCOCOD(R3),R3   ;GET THE TERMINATION CODE MEANING
749 006212      PRINTX #TCOASC,R3       ;PRINT THE TERMINATION CODE
      006214 010346      MOV     R3,-(SP)
      006214 012746 006472      MOV     #TCOASC,-(SP)
      006220 012746 000002      MOV     #2,-(SP)
      006224 010600      MOV     SP,R0
      006226 104415      TRAP    C$PNTX
      006230 062706 000006      ADD     #6,SP
750 006234 010403      MOV     R4,R3          ;TSSR CONTENTS AGAIN
751 006236 042703 177717      BIC     #^CFATERR,R3   ;CLEAR ALL BUT FATAL TERMINATION
752 006242 001416      BEQ     25$           ;DON'T PRINT IF ZERO
753 006244 006203      ASR     R3
754 006246 006203      ASR     R3
755 006250 006203      ASR     R3          ;ALINE TERMINATION CODE FOR INDEX
756 006252 016303 007232      MOV     TSFCOD(R3),R3  ;GET THE FATAL TERMINATION CODE
757 006256      PRINTX #TFCASC,R3     ;PRINT THE FATAL TERMINATION CODE
      006256 010346      MOV     R3,-(SP)
      006260 012746 006533      MOV     #TFCASC,-(SP)
      006264 012746 000002      MOV     #2,-(SP)
      006270 010600      MOV     SP,R0
      006272 104415      TRAP    C$PNTX
      006274 062706 000006      ADD     #6,SP
758 006300 042704 176377      25$:    BIC     #^CHIADDR,R4   ;CLEAR ALL BUT EXTENDED ADDRESS
759 006304 001411      BEQ     30$           ;DON'T PRINT IF ZERO
760 006306      PRINTX #TEXASC,R4     ;PRINT THE EXTENDED ADDRESS BITS
      006306 010446      MOV     R4,-(SP)
      006310 012746 006431      MOV     #TEXASC,-(SP)
      006314 012746 000002      MOV     #2,-(SP)
      006320 010600      MOV     SP,R0
      006322 104415      TRAP    C$PNTX
      006324 062706 000006      ADD     #6,SP
761 006330 013703 002176      30$:    MOV     EPRTSW,R3      ;PRINT MEASGE BUFFER ADDRESS
762 006334      PRINTX R3             ;PRINT PROPER MESSAGE
      006334 010346      MOV     R3,-(SP)
      006336 012746 000001      MOV     #1,-(SP)
      006342 010600      MOV     SP,R0
      006344 104415      TRAP    C$PNTX
      006346 062706 000004      ADD     #4,SP
763 006352 000207      RTS     PC             ;RETURN TO CALLER
764
766 006354      EPRT2:
767 006354      045      116      045      EPRT1: .ASCIZ  '%NZA *****REPLACE M7196*****'
782 006411      045      116      045      TSSRFOR: .ASCIZ  '%NZA TSSR = %06'
783 006431      045      116      045      TEXASC: .ASCIZ  '%NZA Extended Address Bits = %06'
784 006472      045      116      045      TCOASC: .ASCIZ  '%NZA Termination Class Code = %T'
785 006533      045      116      045      TFCASC: .ASCIZ  '%NZA Fatal Termination Class Code = %T'
786 006602      045      116      045      TSSDEF: .ASCIZ  '%NZA TSSR Bits Set: %T'
787 006631      045      116      045      AMBTSSR: .ASCIZ  '%NZA TSSR Contents Are Ambiguous'
788
789 006672 006712 006735 006763 TCOCOD: .EVEN .WORD  1$,2$,3$,4$,5$,6$,7$,8$
    
```

790	006712	116	157	162	1\$:	.ASCIZ	'Normal Termination'
791	006735	124	145	162	2\$:	.ASCIZ	'Termination Condition'
792	006763	124	141	160	3\$:	.ASCIZ	'Tape Status Alert'
793	007005	106	165	156	4\$:	.ASCIZ	'Function Reject'
794	007025	122	145	143	5\$:	.ASCIZ	'Recoverable Error - Tape Position One Record Down'
795	007107	122	145	143	6\$:	.ASCIZ	'Recoverable Error - Tape Was Not Moved'
796	007156	125	156	162	7\$:	.ASCIZ	'Unrecoverable Error'
797	007202	106	141	164	8\$:	.ASCIZ	'Fatal Controller Error'
798						.EVEN	
799							
800	007232	007242	007276	007307	TSFCOD:	.WORD	1\$,2\$,3\$,4\$
801	007242	111	156	164	1\$:	.ASCIZ	'Internal Diagnostic Failure'
802	007276	122	145	163	2\$:	.ASCIZ	'Reserved'
803	007307	102	165	163	3\$:	.ASCIZ	'Bus Interface or Sanity Check Error'
804	007353	122	145	163	4\$:	.ASCIZ	'Reserved'
805						.EVEN	

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

808
 809
 810 :+ THIS ROUTINE PRINTS THE ADDRESS AND CONTENTS OF A COMMAND PACKET.
 811 : THIS ROUTINE IS NORMALLY ONLY CALLED FROM A PRINT ROUTINE.
 812

813 : INPUT:
 814
 815

816 R0 NUMBER OF WORDS IN PACKET
 817 R3 HIGH ORDER COMMAND PACKET ADDRESS
 818 R4 ADDRESS OF COMMAND PACKET

819 : NOTE: R3 IS IGNORED IF THE KTENABLE FLAG IS CLEAR.
 820
 821

822 PRIPKT::

823 SAVREG ;SAVE THE REGISTERS
 824 MOV R0,R5 ;SAVE NO. OF WORDS IN PACKET
 825 TST KTENABLE ;ABOVE 28K UNDER TEST?
 826 BNE 10\$;BR IF YES
 827 CLR R3 ;SET HIGH ORDER ADDRESS TO 0
 828 10\$: MOV R3,R1 ;COPY HIGH ORDER ADDRESS
 829 MOV R4,R0 ;GET LOWER ADDRESS
 830 ROL R0 ;SHIFT BIT 15 INTO C BIT
 831 ROL R1 ;AND INTO HIGH ORDER.
 832 PRINTB #PKTADD,R1,R4 ;PRINT PACKET ADDRESS

007412 010446
 007414 010146
 007416 012746 007550
 007422 012746 000003
 007426 010600
 007430 104414
 007432 062706 000010

833 15\$: MOV R3,R0 ;GET HIGH ORDER ADDRESS
 834 BEQ 20\$;BR IF NOT ABOVE 28K.
 835 MOV R4,R1 ;GET LOW ORDER ADDRESS
 836 JSR PC,SETMAP ;SETUP PAR6 MAPPING FOR 18 BIT ADDRESS
 837 MOV R0,R4 ;GET RETURNED PAR6 ADDRESS BIAS

838 20\$: CLR R1 ;SAVE WORD NUMBER
 839 25\$: MOV (R4)+,R2 ;GET PACKET CONTENTS
 840 PRINTB #PKTFRM,R1,R2 ;PRINT THE DATA

007456 010246
 007460 010146
 007462 012746 007512
 007466 012746 000003
 007472 010600
 007474 104414
 007476 062706 000010

841 INC R1 ;NEXT WORD NUMBER
 842 CMP R1,R5 ;DONE ALL PACKET WORDS?
 843 BLT 25\$;LOOP TILL ALL DONE
 844 RTS PC ;RETURN

845
 846 007512 045 116 045 PKTFRM: .ASCIZ '%N% Packet Word #%D1% = %06'
 847 007550 045 116 045 PKTADD: .ASCIZ '%N% Packet Address = %01%05'
 848 .EVEN
 849

```

851                                     .SBTTL PRIBXOR - PRINT EXPD, RECV AND XOR BYTE
852
853
854
855 :PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE DATA BYTE
856 :THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
857
858 :INPUTS:
859
860         R1      RECEIVED DATA
861         R2      EXPECTED DATA
862
863 :OUTPUT:
864
865         R0      XOR OF EXPECTED/RECEIVED DATA
866
867 :-
868
869 007606 PRIBXOR::
870 007606 SAVREG                                ;SAVE THE REGISTERS
871 007612 010203 MOV R2,R3                    ;EXPECTED DATA
872 007614 XOR R1,R3                        ;FORM THE EXCLUSIVE OR
873 007624 012700 177400 MOV #^C<377>,R0          ;BYTE MASK
874 007630 040001 BIC R0,R1                ;SAVE LOW BYTE RECV
875 007632 040002 BIC R0,R2                ;SAVE LOW BYTE EXPD
876 007634 040003 BIC R0,R3                ;SAVE LOW BYTE XOR
877 007636 PRINTB #XORBFOR,R2,R1,R3 ;PRINT THE MESSAGE
      007636 010346 MOV R3,-(SP)
      007640 010146 MOV R1,-(SP)
      007642 010246 MOV R2,-(SP)
      007644 012746 007670 MOV #XORBFOR,-(SP)
      007650 012746 000004 MOV #4,-(SP)
      007654 010600 MOV SP,R0
      007656 104414 TRAP C$PNTB
      007660 062706 000012 ADD #12,SP
878 007664 010300 MOV R3,R0                    ;R0 HAS XOR ON RETURN
879 007666 000207 RTS                          ;RETURN TO CALLER
880
881 007670 045 116 045 XORBFOR: .ASCIZ '%N% EXPD: %03% RECV: %03% XOR: %03'
882 .EVEN
883

```


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

.SBTTL PRI XOR - 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

```

:-

PRI XOR::

```

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

```

```

007736 010203
007742 010346
007754 010146
007756 010246
007760 012746 010006
007762 012746 000004
007766 010600
007772 104414
007774 062706 000012
010002 010300
010004 000207

```

```

XORFOR: .ASCIZ '%N% EXPD: %06% RECV: %06% XOR: %06%'
.EVEN

```

```

914 .SBTTL PRIEQU - PRINT BIT NUMBERS AS ASCII EQUIVALENT
915
916
917
918 :ROUTINE TO CONVERT BIT VALUES TO ASCII AND PRINT THE STRING
919 :THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
920
921 :INPUTS:
922
923 : R0 OCTAL VALUE TO CONVERT
924 : R1 TABLE OF POINTERS TO ASCII EQUIVALENT
925
926
927
928 010054 PRIEQU: SAVREG ;SAVE THE REGISTERS
929 010054 RTS PC ;RETURN TO CALLER
930 010060 000207
931
932
933
934
935 .SBTTL PRIRAM - PRINT RAM ADDRESS
936
937
938 :PRINT CONTROLLER RAM ADDRESS.
939 :THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
940
941 :INPUTS:
942
943 : R4 RAM ADDRESS
944
945
946 010062 PRIRAM: SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
947 010062 PRINTB #RAMFOR,R4 ;PRINT RAM ADDRESS IN ERROR
948 010066 010446 MOV R4,-(SP)
949 010112 045 116 045 RAMFOR: .ASCIZ '%N%A CONTROLLER RAM ADDRESS = %06'
950 010070 012746 010112 MOV #RAMFOR,-(SP)
951 010074 012746 000002 MOV #2,-(SP)
952 010100 010600 MOV SP,R0
953 010102 104414 TRAP C$PNTB
954 010104 062706 000006 ADD #6,SP
955 010110 000207 RTS PC ;RETURN
956
957
958 .SBTTL PRIADD - PRINT MEMORY ERROR ADDRESS
959
960 :PRINT MEMORY ADDRESS
961 :THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
962
963 :IMPLICIT INPUTS
964
965 : ERRHI - HIGH ORDER ADDRESS
966 : ERRLO - LOW ORDER ADDRESS
    
```

```

965
966
967 010154
968 010154
969 010160 013700 002234
970 010164 013701 002236
971 010170 010102
972 010172 006101
973 010174 006100
974 010176
    010176 010246
    010200 010046
    010202 012746 010224
    010206 012746 000003
    010212 010600
    010214 104414
    010216 062706 000010
975 010222 000207
976
977 010224 045 116 045 PRIA0: .ASCIZ '%NZA MEMORY ERROR ADDRESS = %01X05'
978 .EVEN
979
980
981 .SBTTL PRITADD - PRINT MEMORY TEST ADDRESS
982
983
984 :PRINT MEMORY ADDRESS
985 :THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
986
987 :IMPLICIT INPUTS
988
989 ERRHI - HIGH ORDER ADDRESS
990 ERRLO - LOW ORDER ADDRESS
991
992
993 PRITADD:
994 010270
995 010270
996 010274 013702 002234
997 010300 013701 002236
998
999
1000 010304
    010304 010146
    010306 012746 010352
    010312 012746 000C02
    010316 010600
    010320 104414
    010322 062706 000006
1001 010326
    010326 010246
    010330 012746 010415
    010334 012746 000002
    010340 010600
    010342 104414
    010344 062706 000006
1002 010350 000207

```

```

:
:-
PRIADD:
    SAVREG
    MOV ERRHI,R0
    MOV ERRLO,R1
    MOV R1,R2
    ROL R1
    ROL R0
    PRINTB #FRIA0,R0,R2
    MOV R2,-(SP)
    MOV R0,-(SP)
    MOV #PRIA0,-(SP)
    MOV #3,-(SP)
    MOV SP,R0
    TRAP C$PNTB
    ADD #10,SP
    RTS PC
    :SAVE R1-R5 UNTIL NEXT RETURN
    :GET HIGH ADDRESS
    :GET LOW ADDRESS
    :COPY LOW ADDRESS
    :SHIFT BIT 15 TO C BIT
    :SHIFT INTO HIGH ORDER
    :PRINT MEMORY ADDRESS IN ERROR
    :RETURN

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

.SBTTL PRITADD - PRINT MEMORY TEST ADDRESS

:PRINT MEMORY ADDRESS
:THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.

:IMPLICIT INPUTS
ERRHI - HIGH ORDER ADDRESS
ERRLO - LOW ORDER ADDRESS

PRITADD:
    SAVREG
    MOV ERRHI,R2
    MOV ERRLO,R1
    :MOV R1,R2
    :ROL R1
    :ROL R0
    PRINTB #PRIT0,R1
    MOV R1,-(SP)
    MOV #PRIT0,-(SP)
    MOV #2,-(SP)
    MOV SP,R0
    TRAP C$PNTB
    ADD #6,SP
    PRINTB #PRIT1,R2
    MOV R2,-(SP)
    MOV #PRIT1,-(SP)
    MOV #2,-(SP)
    MOV SP,R0
    TRAP C$PNTB
    ADD #6,SP
    RTS PC
    :SAVE R1-R5 UNTIL NEXT RETURN
    :GET HIGH ADDRESS
    :GET LOW ADDRESS
    :COPY LOW ADDRESS
    :SHIFT BIT 15 TO C BIT
    :SHIFT INTO HIGH ORDER
    :PRINT MEMORY ADDRESS LOW IN ERROR
    :PRINT MEMORY ADDRESS HIGH IN ERROR
    :RETURN

```


1011 .SBTTL SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND

1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045

```

: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:
      -
    
```

1046 010462
1047 010462
1048 010466 012737 000764 010650
1049 010474 012737 140010 010640
1050 010502 005703
1051 010504 100403
1052 010506 010337 010642
1053 010512 000407
1054 010514 042703 100000 5\$:
1055 010520 010337 010642
1056 010524 052737 000400 010640
1057 010532 012704 010640 10\$:
1058 010536 010465 000000
1059 010542 004737 016240 15\$:
1060 010546 103420
1061 010550
010550 012727 000250
010554 000000
010556 013727 002116
010562 000000
010564 005367 177772
010570 001375

```

SPACE::
      SAVREG          ;SAVE THE GENERAL REGISTERS
      MOV #500,SDELAY ;SET UP DELAY
      MOV #140010,80$ ;SET UP COMMAND, SPACE FORWARD
      TST R3          ;CHECK FOR DIRECTION
      BMI 5$         ;BR, IF REVERSE INDICATED
      MOV R3,90$     ;LOAD UP NUMBER OF RECORDS TO SPACE
      BR 10$        ;GO DO COMMAND
5$:   BIC #BIT15,R3  ;CLEAR DIRECTION BIT
      MOV R3,90$     ;LOAD UP NUMBER OF RECORDS TO SPACE
      BIS #BIT8,80$  ;SET REVERSE BIT IN COMMAND PACKET
10$:  MOV #80$,R4    ;SET UP R4 WITH PACKET ADDRESS
      MOV R4,TSDB(R5);SEND OUT COMMAND
15$:  JSR PC,WAITF   ;WAIT FOR SSR
      BCS 20$        ;BR, IF SSR IS SET AND OK
      DELAY 250     ;DELAY ABOUT .25 SECONDS
      MOV #250,(PC)+
      .WORD 0
      MOV L$DLY,(PC)+
      .WORD 0
      DEC -6(PC)
      BNE .-4
    
```

	010572	005367	177756		DEC	-22(PC)	
	010576	001367			BNE	.-20	
1062	010600	005337	010650		DEC	SDELAY	:BUMP DELAY COUNTER DOWN
1063	010604	001356			BNE	15\$:BR, IF MORE DELAY
1064	010606	000411			BR	60\$:BR IF TROUBLE CARRY = CLEAR
1065	010610	016501	000002	20\$:	MOV	TSSR(R5),R1	:READ TSSR
1066	010614	012702	000200		MOV	#SSR,R2	:SET UP EXPECTED
1067	010620	020201		25\$:	CMP	R2,R1	:ARE THEY OK
1068	010622	001401			BEQ	40\$:BR, IF EQUAL = OK
1069	010624	000402			BR	60\$:TROUBLE EXIT
1070	010626	000261		40\$:	SEC		:SET CARRY NO TROUBLE
1071	010630	000401			BR	70\$:EXIT
1072	010632	000241		60\$:	CLC		:CARRY CLEAR = ERROR
1073	010634			70\$:			
1074	010634	010400			MOV	R4,R0	:PASS PACKET ADDRESS
1075	010636	000207			RTS	PC	:RETURN

```
1077  
1078  
1079  
1080  
1081  
1085  
1086  
1087 010640 000000  
1088  
1089 010642 000000  
1090 010644 000000  
1091 010646 000000  
1092 010650 000000  
1093
```

 :
 :
 :
 : PACKET FOR SPACE COMMAND
 :
 :
 : COMMAND WORD
80\$: : .WORD
 : NUMBER OF RECORDS TO BE SPACED OVER WORD
90\$: : .WORD
 : .WORD
 : .WORD
SDELAY: : .WORD 0 : DELAY COUNTER
 : .EVEN

1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127 010652
1128 010652
1129 010656 005037 002226
1130 010662 005037 002224
1131 010666 010465 000000
1132 010672 004737 016326
1133 010676 103401
1134 010700 000435
1135 010702 016501 000002
1136 010706 012702 000200
1137 010712 032701 000100
1138 010716 001402
1139 010720 052702 000100
1140 010724 020201
1141 010726 001401
1142 010730 000421
1143 010732 062704 000010
1144 010736 011403
1145 010740 032763 000200 000012
1146 010746 001402
1147 010750 005237 002224
1148 010754
1149 010754 032763 000100 000012
1150 010762 001402
1151 010764 005237 002226

.SBTTL WRTCHR - WRITE CHARACTERISTICS COMMAND

```

:ROUTINE TO ISSUE A WRITE CHARACTERISTICS
:COMMAND SO THAT OTHER COMMANDS WILL BE ACCEPTED

:INPUT:
R4 ADDRESS OF PACKET FROM TEST
R5 FIRST DEVICE UNIBUS ADDRESS
REQUIRES A CALL TO SOFINIT BE DONE PREVIOUSLY

:OUTPUT:
R0 TSSR CONTENTS
CARRY SET - WRITE CHARACTERISTICS COMMAND OK
CLR - WRITE CHARACTERISTICS FAILED

:IMPLICIT OUTPUT:
MESSAGE BUFFER AND OTHER BUFFERS ALL SET UP
SOFTWARE SWITCHES SET AS FOLLOWS:
EXTFEA = EXTENDED FEATURES PRESENT
BENBSW = BUFFER ENABLE SWITCH ON OR OFF
    
```

:SIDE EFFECTS:

WRTCHR::

```

SAVREG ;SAVE THE GENERAL REGISTERS
CLR BENBSW ;CLEAR BUFFER ENABLE SWITCH
CLR EXTFEA ;CLEAR EXTENDED FEATURES SW SWITCH
10$: MOV R4,TSDB(R5) ;SEND OUT COMMAND
JSR PC,CHKTSSR ;WAIT FOR SSR
BCS 20$ ;BR, IF SSR IS SET AND OK
BR 60$ ;BR IF TROUBLE CARRY = CLEAR
20$: MOV TSSR(R5),R1 ;READ TSSR
MOV #SSR,R2 ;SET UP EXPECTED
BIT #OFL,R1 ;WAS OFF LINE SET IN TSSR
BEQ 25$ ;BR, IF NO OFL SET
BIS #OFL,R2 ;MAKE THEM LOOK ALIKE
25$: CMP R2,R1 ;ARC THEY OK
BEQ 40$ ;BR, IF EQUAL = OK
BR 60$ ;TROUBLE EXIT
40$: ADD #8,R4 ;POINT TO WRT CHARA DATA PACKET
MOV (R4),R3 ;GET ADDRESS OF MESSAGE BUFFER
BIT #X2.EXTF,XST2(R3) ;EXTENDED FEATURES BIT SET?
BEQ 45$ ;BR IF NO
INC EXTFEA ;SET EXTENDED FEATURES SW SWITCH
45$: BIT #X2.BUFE,XST2(R3) ;BUFFER ENABLE SWITCH SET
BEQ 50$ ;BR, IF SWITCH NOT SET
INC BENBSW ;SET SOFTWARE SWITCH FOR ENABLED
    
```


1152	010770			50\$:				
1153	010770	000261			SEC			;SET CARRY NO TROUBLE
1154	010772	000401			BR	70\$;EXIT
1155	010774	000241		60\$:	CLC			;CARRY CLEAR = ERROR
1156	010776	016500	000002	70\$:	MOV	TSSR(R5),R0		;RETURN TSSR CONTENTS
1157	011002	000207			RTS	PC		;RETURN
1158								
1159								

1161
 1162
 1163
 1164
 1165
 1166
 1167
 1168
 1169
 1170
 1171
 1172
 1173
 1174
 1175
 1176
 1177
 1178
 1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1188
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1200
 1201
 1202
 1203
 1205
 1207
 1208
 1209
 1210
 1211

.SBTTL REWIND - POSITION TAPE (REWIND) COMMAND

```

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

```

: CALLING SEQUENCE:

```

DO A SOFT INIT
DO A WRITE CHARACTERISTICS
JSR PC,REWIND

```

: INPUT:

R5 FIRST DEVICE UNIBUS ADDRESS

: OUTPUT

R0 THE CONTENTS OF R4 IS PASSED TO R0

REWIND::

```

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

```

RWPACK: .=<.+10>&177770

```

.WORD 102010 ;POSTION COMMAND (REWIND)
.WORD 0 ;NOT USED

```

```

011004
011004
012704 011100
010465 000000
012703 000550
004737 016240
103417
012727 000372
000000
013727 002116
000000
005367 177772
001375
005367 177756
001367
005303
001357
000241
010400
000207
011100
011100 102010
011102 000000

```

TSV3 - GLOBAL AREAS MACRO M1113 25-MAY-82 08:41 PAGE 30-1
REWIND - POSITION TAPE (REWIND) COMMAND

F 5

SEQ 0057

1212
1213
1214

1216
 1217
 1218
 1219
 1220
 1221
 1222
 1223
 1224
 1225
 1226
 1227
 1228
 1229
 1230
 1231
 1232
 1233
 1234
 1235
 1236
 1237
 1238
 1239
 1240
 1241
 1242
 1243
 1244
 1245
 1246
 1247
 1248
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257
 1258
 1259
 1260
 1261
 1262
 1263
 1264
 1265
 1266
 1267
 1268
 1269

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

```

011104
011104
011110 012701 002240
011114 012702 000201
011120 005003
011122 004737 016326
011126 112765 000000 000000
011134 004737 016326 10$:
011140 010265 000000
011144 004737 016326
011150 116511 000000
011154 122124
011156 001401
011160 005203
011162 005202 20$:
011164 020227 000210
011170 003761
011172 005703
011174 001402
011176 000241
011200 000401
011202 000261 30$:
011204 012737 000010 002300 50$:
011212 000207
  
```

```

1271 .SBTTL CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA
1272
1273
1274 :ROUTINE TO READ THE FIRST 8 OR 10 BYTES FROM RAM
1275 :MEMORY AND COMPARE THIS DATA TO A CHARACTERISTICS DATA BLOCK.
1276
1277 :INPUT:
1278
1279 :       R4      ADDRESS OF THE CHARACTERISTICS DATA
1280 :       R5      FIRST DEVICE UNIBUS ADDRESS
1281
1282 :OUTPUT:
1283
1284 :       CARRY   SET - RAM MATCHES PACKET
1285 :              CLR - RAM DOES NOT MATCH PACKET
1286
1287 :IMPLICIT OUTPUT:
1288
1289 :       THE TABLE RAMDATA IS FILLED WITH THE
1290 :       DATA HELD IN RAM.
1291 :       RAMSIZ IS SET TO 8. OR 10. FOR PRAMPKT ROUTINE
1292
1293 :SIDE EFFECTS:
1294
1295 :       THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
1296
1297 :-
1298

```

```

1299 011214 CKRAM2:: SAVREG ;SAVE THE GENERAL REGISTERS
1300 011214 MOV #RAMDATA,R1 ;ADDRESS TO SAVE THE RAM DATA
1301 011220 012701 002240 MOV #RMCHBEG,R2 ;BYTE ADDRESS OF FIRST RAM DATA
1302 011224 012702 000167 CLR R3 ;CLEAR THE ERROR FLAG
1303 011230 005003 JSR PC,CHKTSSR ;WAIT FOR SSR
1304 011232 004737 016326 MOVB #0,TSDB(R5) ;SET MAINTENANCE MODE
1305 011236 112765 000000 000000 10$: JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
1306 011244 004737 016326 MOV R2,TSDB(R5) ;SELECT NEXT RAM ADDRESS
1307 011250 010265 000000 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
1308 011254 004737 016326 MOVB TSBA(R5),(R1) ;READ THE RAM DATA
1309 011260 116511 000000 CMPB (R1)+,(R4)+ ;COMPARE TO EXPECTED
1310 011264 122124 BEQ 20$ ;BRANCH IF OK
1311 011266 001401 INC R3 ;SET ERROR FLAG
1312 011270 005203 INC R2 ;ADDRESS OF NEXT RAM LOCATION
1313 011272 005202 20$: MOV #8.,RAMSIZ ;ASSUME EXTFEA NOT SET
1314 011274 012737 000010 002300 TST EXTFEA ;IS THE SOFTWARE EXTENDED FEATURES SET
1315 011302 005737 002224 BEQ 25$ ;BR, IF NOT SET
1316 011306 001407 MOV #10.,RAMSIZ ;SET RAMSIZ FOR EXTEND FEATURES
1317 011310 012737 000012 002300 CMP R2,#RMCHEND ;AT END OF EXTENDED BUFFER
1318 011316 020227 000200 BLE 10$ ;BR, IF NOT AT END YET
1319 011322 003750 BR 27$ ;AT END BRANCH
1320 011324 000403 BR 27$ ;AT END BRANCH
1321 011326 020227 000176 25$: CMP R2,#RMCHEND-2 ;REACHED END YET ?
1322 011332 003744 BLE 10$ ;BRANCH TILL ALL READ
1323 011334 005703 27$: TST R3 ;WAS AN ERROR FOUND ?
1324 011336 001402 BEQ 30$ ;BRANCH IF NOT
1325 011340 000241 CLC ;CLEAR CARRY TO SHOW ERROR
1326 011342 000401 BR 50$ ;AND EXIT
1327 011344 000261 30$: SEC ;SHOW GOOD COMPARE

```

TSV3 - GLOBAL AREAS MACRO M1113 25-MAY-82 08:41 PAGE 32-1 I 5
CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA

SEQ 0060

1328 011346 000207
1329

SOS: RTS PC

:RETURN

1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356 011350
1357 011350
1358 011354 010037 002302
1359 011360 010137 002304
1360 011364 005737 003130
1361 011370 001403
1362 011372 004737 017306
1363 011376 010001
1364 011400 005004
1365 011402 005003
1366 011404 010205
1367 011406 011264 002320
1368 011412 011164 002464
1369 011416 022221
1370 011420 001401
1371 011422 005203
1372 011424 062704 000002
1373 011430 020427 000014
1374 011434 003764
1375 011436 032765 000200 000012
1376 011444 001403
1377 011446 020427 000016
1378 011452 003755
1379 011454 005703
1380 011456 001402
1381 011460 000241
1382 011462 000401
1383 011464 000261
1384 011466 000207
1385

```
.SBTTL CKMSG - COMPARE WRITE CHAR. MESSAGE BUFFERS
:
:ROUTINE TO COMPARE A WRITE CHARACTERISTICS EXPD AND RECV
:BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
:ERROR PRINT ROUTINES.
:
:INPUT:
:
:      R0      RECV MESSAGE BUFFER HIGH ORDER ADDRESS
:      R1      RECV MESSAGE BUFFER LOW ORDER ADDRESS
:      R2      EXPD MESSAGE BUFFER ADDRESS
:
:OUTPUT:
:
:      CARRY   SET - MESSAGE BUFFERS MATCH
:             CLR -MESSAGE BUFFERS DON'T MATCH
:
:IMPLICIT OUTPUT:
:
:      EXPMSG   BUFFER IS SET TO EXPD DATA
:      RECVMSG  BUFFER IS SET TO RECV DATA
:      RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
:      RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
:
:CKMSG::
:      SAVREG   ;SAVE R1-R5 UNTIL NEXT RETURN
:      MOV      R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
:      MOV      R1,RCVLOADD ;SAVE RECV LOW ADDRESS
:      TST      KTENABLE    ;TESTING ABOVE 28K?
:      BEQ      10$         ;BR IF NO
:      JSR      PC,SETMAP   ;RETURN ADDRESS BIASED TO PAR6 IN R0
:      MOV      R0,R1      ;GET RETURNED ADDRESS BIASED TO PAR6
:      CLR      R4         ;WORD IN BUFFER
:      CLR      R3         ;CLEAR ERROR SEEN FLAG
:      MOV      R2,R5      ;GET EXPD BUFFER ADDRESS
:      MOV      (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
:      MOV      (R1),RECVMSG(R4) ;SAVE RECV FOR ERROR REPORT
:      CMP      (R2)+,(R1)+ ;EXPD EQUAL RECV?
:      BEQ      25$         ;BR IF YES
:      INC      R3         ;SET ERROR SEEN FLAG
:      ADD      #2,R4      ;POINT TO NEXT WORD ADDRESS
:      CMP      R4,#14     ;DONE FIRST 7 WORDS?
:      BLE     15$         ;BR IF NO
:      BIT      #X2.EXTF,XST2(R5) ;IS EXTENDED FEATURES SET IN EXPD?
:      BEQ      50$         ;BR IF NO
:      CMP      R4,#16     ;DONE EXTENDED FEATURES WORD?
:      BLE     55$         ;BR IF NO
:      TST      R3         ;ANY ERRORS SEEN?
:      BEQ      55$         ;BR IF NO
:      CLC      ;SET FAILURE
:      BR      60$
:
:      10$:
:
:      15$:
:
:      25$:
:
:      50$:
:
:      55$:
:      SEC      ;SET SUCCESS
:      60$:
:      RTS      PC        ;RETURN
```

```

1387          .SBTTL  CKMSG2  - COMPARE EXPD RECV MESSAGE BUFFERS
1388
1389
1390          :ROUTINE TO COMPARE AN EXPECTED AND RECEIVED MESSAGE
1391          :BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
1392          :ERROR PRINT ROUTINES.
1393
1394          :INPUT:
1395
1396          R0      RECV MESSAGE BUFFER HIGH ORDER ADDRESS
1397          R1      RECV MESSAGE BUFFER LOW ORDER ADDRESS
1398          R2      EXPD MESSAGE BUFFER ADDRESS
1399          R3      NUMBER OF BYTES TO COMPARE
1400
1401          :OUTPUT:
1402
1403          CARRY   SET - MESSAGE BUFFERS MATCH
1404          CLR     - MESSAGE BUFFERS DON'T MATCH
1405
1406          :IMPLICIT OUTPUT:
1407
1408          EXPMSG   BUFFER IS SET TO EXPD DATA
1409          RECMSG   BUFFER IS SET TO RECV DATA
1410          RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
1411          RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
1412
1413          -
1414          CKMSG2::
1415          SAVREG          :SAVE R1-R5 UNTIL NEXT RETURN
1416          CMP            R3,#RECVMSG-EXPMSG;@@D IS COUNT ABOVE MAX ALLOWED?
1417          BLE            5$          :@@D BR IF NO
1418          MOV            #RECVMSG-EXPMSG,R3;@@D
1419          PRINTF         #DEBUGMSG          :@@D
1420          MOV            #DEBUGMSG,-(SP)
1421          MOV            #1,-(SP)
1422          MOV            SP,R0
1423          TRAP          C$PRINTF
1424          ADD            #4,SP
1425          5$: MOV        R0,RCVHIADD      :SAVE RECV HIGH ADDRESS
1426          MOV        R1,RCVLOAD        :SAVE RECV LOW ADDRESS
1427          TST        KTENABLE          :TESTING ABOVE 28K?
1428          BEQ        10$              :BR IF NO
1429          JSR        PC,SETMAP         :RETURN ADDRESS BIASED TO PAR6 IN R0
1430          MOV        R0,R1            :GET RETURNED ADDRESS BIASED TO PAR6
1431          10$: CLR     R4              :WORD IN BUFFER
1432          CLR     R5                  :CLEAR ERROR SEEN FLAG
1433          15$: MOVB   (R2),EXPMSG(R4)  :SAVE EXPD FOR ERROR REPORT
1434          MOVB   (R1),RECMSG(R4)      :SAVE RECV FOR ERROR REPORT
1435          CMPB   (R2)+,(R1)+          :EXPD EQUAL RECV?
1436          BEQ    25$                  :BR IF YES
1437          INC    R5                    :SET ERROR SEEN FLAG
1438          25$: ADD    #1,R4            :POINT TO NEXT BYTE
1439          CMP    R4,R3                 :DONE ALL BYTES?
1440          BGE    50$                  :BR IF YES
1441          BR     15$                  :DO NEXT BYTE
1442          50$: TST    R5                :ANY ERRORS SEEN?
1443          BEQ    55$                  :BR IF NO
  
```



```
1439 011612 000241                    CLC                    ;SET FAILURE
1440 011614 000401                    BR                    60$                    ;
1441 011616 000261                    55$:                  SEC                    ;SET SUCCESS
1442 011620 000207                    60$:                  RTS                    ;RETURN
1443
1444 011622            120            122            117 DEBUGMSG:            .ASCIZ 'PROGRAM INTERNAL ERROR -CKMSG2 MESSAGE BUFFER EXCEEDED-';@@
1445 011712            045            116            045 FERCM:            .ASCII /XNZA ***/
1446 011723            040            040            124 ERCM:            .ASCIZ / TSSR ERROR CODE REC'D = /
1447 011756            056            056            056 SIMSG:            .ASCIZ /... AFTER DOING SOFT INIT/
1448 012011            124            105            123 TINERR:            .ASCIZ /TEST: .../
1449                                    .EVEN
```

```

1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467 012024
      012024
1468 012024 004737 006020
1469 012030 004737 017172
1470 012034
      012034
      012034 104423
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483 012036
      012036
1484 012036 004737 006020
1485 012042 012700 000004
1486 012046 004737 007364
1487 012052
      012052
      012052 104423
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500 012054
      012054
  
```

```

: +
: PRINT ROUTINE TO FATAL SOFT INIT ERRORS
: INPUT:
:       R1      CONTENTS OF TSSR AT ERROR
: SIDE EFFECTS:
:       EXECUTES DROP UNIT TO CEASE TESTING
: -
:
: BGNMSG  SFMSG
SFMSG:: JSR    PC,PRITSSR    ;PRINT CONTENTS OF TSSR REGISTER
        JSR    PC,CKDROP   ;DROP UNIT, IF ALLOWED
        ENDMSG
L10003: TRAP   C$MSG
:
: +
: PRINT ROUTINE TO PRINT THE CONTENTS OF
: TSSR AND A COMMAND PACKET OTHER THAN GET STATUS COMMAND PACKET.
: INPUTS:
:       R1      TSSR CONTENTS
:       R4      ADDRESS OF COMMAND PACKET
: -
:
: BGNMSG  PKTSSR
PKTSSR:: JSR    PC,PRITSSR    ;PRINT THE CONTENTS OF TSSR REGISTER
        MOV    #4,R0        ;NO. OF WORDS IN PACKET
        JSR    PC,PRIPKT   ;PRINT THE CONTENTS OF COMMAND PACKET
        ENDMSG
L10004: TRAP   C$MSG
:
: +
: PRINT ROUTINE TO PRINT THE CONTENTS OF
: TSSR AND A GET STATUS COMMAND PACKET.
: INPUTS:
:       R1      TSSR CONTENTS
:       R4      ADDRESS OF COMMAND PACKET
: -
:
: BGNMSG  PKTGETS
PKTGETS::
  
```

```

1501 012054 004737 006020      JSR    PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
1502 012060 012700 000002      MOV    #2,R0          ;NO. OF WORDS IN GET STATUS PACKET
1503 012064 004737 007364      JSR    PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
1504 012070
      012070
      012070 104423      L10005:  TRAP    C$MSG

1505
1506
1507
1508      ;+
1509      ;PRINT TSSR ERRORS FOR INITIALIZATION TESTS
1510      ;INPUTS:
1511      ;
1512      ;       R1      TSSR CONTENTS
1513      ;       R4      ADDRESS OF COMMAND PACKET
1514      ;-
1515
1516 012072      BGNMSG  SFFMSG
      012072
1517 012072 004737 006020      SFFMSG:: JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
1518 012076
      012076
      012076 104423      L10006:  TRAP    C$MSG

1519
1520
1521      .SBTTL  PKTMES - PRINT TSSR AND MESSAGE BUFFER
1522      ;+
1523      ;PRINT ROUTINE TO PRINT THE CONTENTS OF TSSR AND MESSAGE
1524      ;BUFFER FOR ERROR REPORTS
1525      ;INPUTS:
1526      ;
1527      ;       R1      CONTENTS OF TSSR
1528      ;       R2      LOW ORDER MESSAGE BUFFER
1529      ;       R3      HIGH ORDER MESSAGE BUFFER ADDRESS
1530      ;       NOTE: R3 IS IGNORED IF KTENABLE FLAG IS CLEAR
1531      ;-
1532
1533
1534 012100      BGNMSG  PKTMES
      012100
1535 012100 004737 006020      PKTMES:: JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR
1536 012104 010200
1537 012106 010301
1538 012110 004737 014232      MOV    R2,R0          ;LOW ORDER ADDRESS
1539 012114
      012114
      012114 104423      MOV    R3,R1          ;HIGH ORDER ADDRESS
1540
      L10007:  JSR    PC,PRMESS      ;PRINT THE MESSAGE BUFFER
      TRAP    C$MSG
  
```

1542
 1543
 1544
 1545
 1546
 1547
 1548
 1549
 1550
 1551
 1552
 1553
 1554 012116
 012116
 1555 012116 004737 010270
 1556 012122 016501 000002
 1557 012126 004737 006C20
 1558 012132
 012132
 012132 104423
 1559
 1560
 1561
 1562
 1563
 1564
 1565
 1566
 1567
 1568
 1569
 1570
 1571
 1572
 1573 012134
 012134
 1574 012134 012700 000007
 1575 012140 005737 002224
 1576 012144 001402
 1577 012146 012700 000010
 1578 012152 004737 014542
 1579 012156
 012156
 012156 104423
 1580
 1581

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

```
.SBTTL MSGEXP - PRINT WRITE CHAR. EXPD-RECV MESSAGE BUFFERS
:
:PRINT ROUTINE TO PRINT WRITE CHARACTERISTIC MESSAGE BUFFER
:
:IMPLICIT INPUTS:
:
:      EXPMSG  - EXPECTED MESSAGE BUFFER
:      RECMSG  - RECEIVED MESSAGE BUFFER
:      RCVHIADU- 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    $$          ;BR IF NO
:          MOV    #8.,R0      ;EXT FEATURE BUFFER IS 8 WORDS
:          JSR    PC,PRMSGEXP ;PRINT EXPD/RECV MESSAGE BUFFERS
:          ENDMSG
L10011:  TRAP     C$MSG
```

1583
 1584
 1585
 1586
 1587
 1588
 1589
 1590
 1591
 1592
 1593
 1594
 1595 012160
 012160
 1596 012160
 012160 010146
 012162 012746 012232
 012166 012746 000002
 012172 010600
 012174 104415
 012176 062706 000006
 1597 012202
 012202 012746 012301
 012206 012746 000001
 012212 010600
 012214 104415
 012216 062706 000004
 1598 012222 010100
 1599 012224 004737 015112
 1600 012230
 012230
 012230 104423
 1601 012232 045 116
 1602 012301 045 116
 1603
 1604

```

.SBTTL FIFEXP - PRINT FIFO EXP/RECV DATA
:
:PRINT ROUTINE TO PRINT FIFO EXP/RECV DATA
:
: R1 - BYTE COUNT
:
:IMPLICIT INPUTS:
:
: EXPMSG - EXPECTED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
: RECMSG - RECEIVED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
:
:
: BGNMSG FIFEXP
FIFEXP:
PRINTX #FIF1MSG,R1 ;PRINT BYTES TRANSFERRED
MOV R1,-(SP)
MOV #FIF1MSG,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #6,SP
PRINTX #FIF2MSG ;PRINT HEADER MSG
MOV #FIF2MSG,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #4,SP
MOV R1,R0 ;GET BYTE COUNT
JSR PC,PRBYTEXP ;PRINT FIFO BYTES IN ERROR
ENDMSG
L10012:
TRAP C$MSG
FIF1MSG: .ASCIZ '%X%A NUMBER OF BYTES TRANSFERRED = %u2'
FIF2MSG: .ASCIZ '%X%A FIFO DATA BYTES IN ERROR:'
.EVEN
  
```

1606
 1607
 1608
 1609
 1610
 1611
 1612
 1613
 1614
 1615
 1616
 1617
 1618
 1619 012340
 012340
 1620 012340 012701 012402
 1621 012344 012100
 1622 012346 001410
 1623 012350
 012350 010046
 012352 012746 000001
 012356 010600
 012360 104415
 012362 062706 000004
 1624 012366 000766
 1625 012370 012700 000012
 1626 012374 004737 014542
 1627 012400
 012400
 012400 104423
 1628
 1629 012402 012420 012462 012553
 1630 012420 045 116 045
 1631 012462 045 116 045
 1632 012553 045 116 045
 1633 012644 045 116 045
 1634 012735 045 116 045
 1635 012777 045 116 045
 1636
 1637
 1638
 1639
 1640
 1641
 1642
 1643
 1644
 1645
 1646
 1647
 1648
 1649
 1650
 1651
 1652 013054
 013054
 1653 013054 012701 013116

```

.SBTTL MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS
:
:PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RECV
:
:IMPLICIT INPUTS:
:
:   EXPMSG - EXPECTED MESSAGE BUFFER
:   RECMSG - RECEIVED MESSAGE BUFFER
:   RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
:   RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
:
:
:   BGNMSG MSGSTAT
MSGSTAT:
:   MOV #STATCOD,R1 ;ASCII ADDRESS TABLE
10$:   MOV (R1)+,R0 ;DONE ALL MSG LINES?
:   BEQ 20$ ;BR IF YES
:   PRINTX R0 ;PRINT STATUS BIT NAMES
:   MOV R0,-(SP)
:   MOV #1,-(SP)
:   MOV SP,R0
:   TRAP C$PNTX
:   ADD #4,SP
:   BR 10$ ;DO ANOTHER MSG LINE
20$:   MOV #10,R0 ;NUMBER OF WORDS IN A READ STATUS BUFFER
:   JSR PC,PRMSGEXP ;PRINT EXPD/RECV MESSAGE BUFFERS
:   ENDMMSG
L10013:
:   TRAP C$MSG
:
:   STATCOD: .WORD 1$,2$,3$,4$,5$,6$,0
1$: .ASCIIZ '%NZA Tape Bus Signals in Word #8:'
2$: .ASCIIZ '%NZA PARERR<15> IEOT <12> IFMK <9> IRDY<6> IRWD<2>'
3$: .ASCIIZ '%NZA IRESV2<14> IIDENT<11> IHER <8> IONL<5> IFBY<1>'
4$: .ASCIIZ '%NZA IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>'
5$: .ASCIIZ '%NZA Tape Bus Signals in Word #9:'
6$: .ASCIIZ '%NZA DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>'
:   .EVEN
    
```

```

.SBTTL MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS
:
:PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RECV
:
:IMPLICIT INPUTS:
:
:   EXPMSG - EXPECTED MESSAGE BUFFER
:   RECMSG - RECEIVED MESSAGE BUFFER
:   RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
:   RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
:
:
:   BGNMSG MSGLOOP
MSGLOOP:
:   MOV #LOOPCOD,R1 ;ASCII ADDRESS TABLE
    
```

```

1654 013060 012100          10$:  MOV      (R1)+,R0      ;DONE ALL MSG LINES?
1655 013062 001410          BEQ      20$      ;BR IF YES
1656 013064          PRINTX  R0      ;PRINT STATUS BIT NAMES
      013064 010046          MOV      R0,-(SP)
      013066 012746 000001  MOV      #1,-(SP)
      013072 010600          MOV      SP,R0
      013074 104415          TRAP    C$PNTX
      013076 062706 000004  ADD      #4,SP
1657 013102 000766          BR      10$      ;DO ANOTHER MSG LINE
1658 013104 012700 000012  20$:  MOV      #10.,R0    ;NUMBER OF WORDS IN A READ STATUS BUFFER
1659 013110 004737 014542  JSR      PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
1660 013114          ENDMSG
      013114          L10014:
      013114 104423          TRAP    C$MSG
1661
1662 013116 013136 01321: 013310 LOOPCOD: .WORD 1$,2$,3$,4$,5$,6$,7$,0
1663 013136          045 116 045 1$: .ASCIZ '%NZA Tape Bus Loopback Signals in Word #8:'
1664 013211          045 116 045 2$: .ASCIZ '%NZA PARERR<15> IRESV2<14>
1665 013310          045 116 045 3$: .ASCIZ '%NZA IHISP=>IEOT<12> IWRT=>IIDENT<11> IREV =>ICER <10>'
1666 013407          045 116 045 4$: .ASCIZ '%NZA IWFM =>IFMK<09> IEDIT=>IHER <08> IFAD =>ISPEED<07>'
1667 013506          045 116 045 5$: .ASCIZ '%NZA ITADO=>IRDY<06> ITAD1=>IONL <05> IERASE=>ILDV <04>'
1668 013605          045 116 045 6$: .ASCIZ '%NZA IREW =>IDBY<03> IRWU =>IRWD <02> IFEN =>IFBY <01>'
1669 013704          045 116 045 7$: .ASCIZ '%NZA IGO =>IFPT<00>'
1670          .EVEN
1671

```

1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686 013732
013732
1687 013732 012700 000012
1688 013736 004737 014542
1689 013742
013742
013742 104423
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707 013744
013744
1708 013744 004737 010154
1709 013750 013701 002230
1710 013754 013702 002232
1711 013760 004737 007736
1712 013764
013764
013764 104423
1713

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

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

1763
1764
1765

.EVEN

```

1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784 014232
1785 014232
1786 014236 010005
1787 014240 005737 003130
1788 014244 001001
1789 014246 005001
1790 014250 010103
1791 014252 006100
1792 014254 006101
1793 014256
    014256 010546
    014260 010146
    014262 012746 014410
    014266 012746 000003
    014272 010600
    014274 104415
    014276 062706 000010
1794 014302
    014302 012746 014455
    014306 012746 000001
    014312 010600
    014314 104415
    014316 062706 000004
1795 014322 005004
1796 014324 010501
1797 014326 010300
1798 014330 001403
1799 014332 004737 017306
1800 014336 010005
1801 014340
    014340 012546
    014342 010446
    014344 012746 014513
    014350 012746 000003
    014354 010600
    014356 104415
    014360 062706 000010
1802 014364 005204
1803 014366 020427 000007
1804 014372 003005
    
```

```

.SBTTL PRMESS - PRINT CONTENTS OF MESSAGE BUFFER
:
: THIS ROUTINE PRINTS THE CONTENTS OF
: THE 7 OR 8 WORD MESSAGE BUFFER RETURNED BY THE
: TSV-05.
:
: INPUT:
:
: R0 LOW ORDER ADDRESS OF MESSAGE BUFFER
: R1 HIGH ORDER ADDRESS OF MESSAGE BUFFER
: NOTE: R1 IS IGNORED IF KENABLE FLAG IS CLEAR
:
: THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
:
: -
PRMESS:
    SAVREG                ;SAVE THE REGISTERS
    MOV R0,R5              ;SAVE LOW ORDER ADDRESS
    TST KENABLE            ;ADDRESS ABOVE 28K?
    BNE 10$                ;BR IF YES
    CLR R1                 ;SET HIGH ORDER ADDRESS TO 0
    MOV R1,R3              ;SAVE HIGH ORDER ADDRESS
    ROL R0                  ;SHIFT BIT15 TO C BIT
    ROL R1                  ;SHIFT TO HIGH ORDER FOR PRINTOUT
    PRINTX #PROASC,R1,R5   ;PRINT MESSAGE BUFFER ADDRESS
    MOV R5,-(SP)
    MOV R1,-(SP)
    MOV #PROASC,-(SP)
    MOV #3,-(SP)
    MOV SP,R0
    TRAP C$PNTX
    ADD #10,SP
    PRINTX #PRIASC         ;PRINT HEADER FOR CONTENTS
    MOV #PRIASC,-(SP)
    MOV #1,-(SP)
    MOV SP,R0
    TRAP C$PNTX
    ADD #4,SP
    CLR R4                 ;NUMBER OF THE NEXT WORD
    MOV R5,R1              ;COPY LOW ORDER ADDRESS
    MOV R3,R0              ;COPY HIGH ORDER ADDRESS
    BEQ 20$                ;BR IF NOT ABOVE 28K
    JSR PC,SETMAP         ;SETUP PAR ADDRESS IN R0
    MOV R0,R5              ;GET PAR FORMAT ADDRESS ABOVE 28K
    PRINTX #PRASC,R4,(R5)+ ;PRINT THE CONTENTS OF MEMORY BUFFER
    MOV (R5)+,-(SP)
    MOV R4,-(SP)
    MOV #PRASC,-(SP)
    MOV #3,-(SP)
    MOV SP,R0
    TRAP C$PNTX
    ADD #10,SP
    INC R4                 ;NUMBER OF THE NEXT
    CMP R4,#7              ;DONE ALL YET?
    BGT 50$                ;BRANCH IF ALL DONE
    
```

1805	014374	002761				BLT	20\$:PRINT FIRST 7 WORDS
1806	014376	032763	000200	000012		RIT	#X2.EXTF,XST2(R3)	:EXTENDED FEATUTES ON ?
1807	014404	001355				B:IE	20\$:PRINT EXTENDED STATUS WORD
1808	014406	000207			50\$:	RTS	PC	:RETURN
1809								
1810	014410	045	116	045	PROASC:	.ASCIZ	'%NZA	Message Buffer Address = %01%05'
1811	014455	045	116	045	PR1ASC:	.ASCIZ	'%NZA	Message Buffer Contents:'
1812	014513	045	116	045	PRASC:	.ASCIZ	'%NZA	Word%D1%A: %0'
1813						.EVEN		

```

1815 .SBTTL PRMSGEXP - PRINT EXPD/RCV MESSAGE BUFFERS
1816
1817
1818 :ROUTINE TO PRINT EXPECTED AND RECEIVED MESSAGE BUFFERS
1819
1820 RO - NUMBER OF WORDS IN BUFFER
1821
1822 :IMPLICIT INPUTS:
1823
1824 EXPMSG - EXPECTED MESSAGE BUFFER
1825 RECMSG - RECEIVED MESSAGE BUFFER
1826 RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1827 RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1828
1829 PRMSGEXP::
1830 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1831 MOV RO,R5 ;SAVE NUMBER OF WORDS
1832 MOV RCVLOADD,RO ;GET RCV LOW ADDRESS
1833 MOV RO,R4 ;COPY LOW ADDRESS
1834 MOV RCVHIADD,R1 ;GET RCV HIGH ADDRESS
1835 ROL RO ;SHIFT BIT15 TO C BIT
1836 ROL R1 ;SHIFT TO HIGH ORDER FOR PRINTOUT
1837 PRINTX #PRMSG0,R1,R4 ;PRINT MESSAGE BUFFER ADDRESS
1838 MOV R4,-(SP)
1839 MOV R1,-(SP)
1840 MOV #PRMSG0,-(SP)
1841 MOV #3,-(SP)
1842 MOV SP,RO
1843 TRAP C$PNTX
1844 ADD #10,SP
1845 PRINTX #PRMSG1 ;PRINT HEADER FOR CONTENTS
1846 MOV #PRMSG1,-(SP)
1847 MOV #1,-(SP)
1848 MOV SP,RO
1849 TRAP C$PNTX
1850 ADD #4,SP
1851 CLR R4 ;NUMBER OF THE CURRENT WORD
1852 MOV #EXPMSG,R1 ;GET EXPD BUFFER ADDRESS
1853 MOV #RECMSG,R2 ;GET RCV BUFFER ADDRESS
20$: MOV (R1),RO ;GET EXPD
1854 MOV (R2),R3 ;GET RCV
1855 XOR RO,R3 ;XOR EXPD/RCV
1856 PRINTX #PRMSG2,R4,(R1)+,(R2)+,R3
1857 MOV R3,-(SP)
1858 MOV (R2)+,-(SP)
1859 MOV (R1)+,-(SP)
1860 MOV R4,-(SP)
1861 MOV #PRMSG2,-(SP)
1862 MOV #5,-(SP)
1863 MOV SP,RO
1864 TRAP C$PNTX
1865 ADD #14,SP
1866 INC R4 ;NUMBER OF THE NEXT
1867 CMP R4,R5 ;DONE ALL YET?
1868 BGE 50$ ;BR IF YES
1869 BR 20$ ;DO ANOTHER
1870 50$: RTS PC ;RETURN
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
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
1995
1996
1997
1998
1999
2000
    
```

1851					
1852	014722	045	116	045	PRMSG0: .ASCIZ '%N% Message Buffer Address = %01%05'
1853	014767	045	116	045	PRMSG1: .ASCIZ '%N% Message Buffer Contents:'
1854	015025	045	116	045	PRMSG2: .ASCIZ '%N% WORD #%D2% EXPD: %06% RECV: %06% XOR: %06'
1855					.EVEN
1856					

```

1858 .SBTTL PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER
1859
1860
1861 :+
1862 :ROUTINE TO PRINT ERROR BYTES IN MESSAGE BUFFERS
1863 :ONLY THE FIRST 8 ERRORS ENCOUNTERED ARE PRINTED DUE TO SCREEN SPACE
1864 :
1865 :RO - NUMBER OF BYTES IN BUFFER
1866 :
1867 :IMPLICIT INPIJTS:
1868 :
1869 :EXPMSG - EXPECTED MESSAGE BUFFER
1870 :RECMSG - RECEIVED MESSAGE BUFFER
1871 PRBYTEXP::
1872 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1873 MOV R0,R5 ;SAVE NUMBER OF BYTES
1874 CLR PRMNO ;INIT ERROR COUNT
1875 CLR R4 ;NUMBER OF THE CURRENT BYTE
1876 MOV #EXPMSG,R1 ;GET EXPD BUFFER ADDRESS
1877 MOV #RECMSG,R2 ;GET RECV BUFFER ADDRESS
1878 20$: MOVB (R1),R0 ;GET EXPD BYTE
1879 BIC #^C<377>,R0 ;CLEAR UPPER BYTE
1880 MOVB R0,PRBEXP ;SAVE FOR ERROR REPORT
1881 MOVB (R2),R3 ;GET RECV BYTE
1882 BIC #^C<377>,R3 ;CLEAR UPPER BYTE
1883 MOVB R3,PRBREC ;FOR ERROR REPORT
1884 XOR R0,R3 ;XOR EXPD/RECV
1885 CMPB (R1)+,(R2)+ ;EXPD = RECV?
1886 BEQ 30$ ;BR IF YES
1887 INC PRMNO ;UPDATE ERROR COUNT
1888 000010 CMP PRMNO,#8. ;PRINTED 8?
1889 BHI 30$ ;BR IF YES
1890 27$: PRINTX #PRBMSG,R4,PRBEXP,PRBREC,R3
1891 MOV R3,-(SP)
1892 MOV PRBREC,-(SP)
1893 MOV PRBEXP,-(SP)
1894 MOV R4,-(SP)
1895 MOV #PRBMSG,-(SP)
1896 MOV #5,-(SP)
1897 MOV SP,R0
1898 TRAP C$PNTX
1899 000014 ADD #14,SP
1900 FORCEEXIT 50$ ;aod
1901 BR 35$ ;aod
1902 30$: FORCERROR 27$,NOTSSR ;aod
1903 35$: ;aod
1904 INC R4 ;NUMBER OF THE NEXT
1905 CMP R4,R5 ;DONE ALL YET?
1906 BGE 50$ ;BR IF YES
1907 BR 20$ ;DO ANOTHER
1908 50$: PRINTX #PRBTOT,PRMNO ;PRIN TOTAL ERROR COUNT
1909 MOV PRMNO,-(SP)
1910 MOV #PRBTOT,-(SP)
1911 MOV #2,-(SP)
1912 MOV SP,R0
1913 TRAP C$PNTX

```

```
1901 015320 062706 000006          ADD      #6,SP
1902 015324 000207          RTS      PC          ;RETURN
1903 015326      045      116      045 PRBMSG: .ASCIZ  '%N%A  BYTE #D2%A  EXPD: %03%A  RECV: %03%A  XOR: %03%'
1904 015413      045      116      045 PRBTOT: .ASCIZ  '%N%A  NUMBER OF BYTES IN ERROR = %D2%'
1905                                .EVEN
1906 015460 000000          PRBEXP: .WORD  0          ;EXPD
1907 015462 000000          PRBREC: .WORD  0          ;RECV
1908
```


1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922 015464
015464
1923 015464 004737 007736
1924 015470
015470
015470 104423
1925
1926

```
.SBTTL EXPREC - PRINT EXPD/RECV WORD DATA
:+
:PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
:INPUTS:
:      R1      RECEIVED DATA
:      R2      EXPECTED DATA
:-

      BGNMSG EXPREC
EXPREC:: JSR PC,PRIXOR      :PRINT THE DATA
          ENDMSG
L10017: TRAP C$MSG
```

1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941 015472
015472
1942 015472 004737 007606
1943 015476
015476
015476 104423
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968 015500
015500
1969 015500 004737 013766
1970 015504
015504
015504 104423
1971
1972
1973
1974
1975
1976
1977
1978

.SBTTL EXPBREC - PRINT EXPD/RECV BYTE DATA

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

:INPUTS:

: R1 RECEIVED DATA BYTE
: R2 EXPECTED DATA BYTE

:-

BGNMSG EXPBREC
EXPBREC:: JSR PC,PRIBXOR ;PRINT THE DATA
ENDMSG
L10020: TRAP C\$MSG

.SBTTL RAMERR - PRINT RAM AND PACKET DATA

:+
:PRINT ROUTINE TO DISPLAY RAM/PACKET DATA

:INPUTS:

: R4 POINTER TO COMMAND PACKET

:IMPLICIT INPUTS:

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

:IMPLICIT OUTPUTS:

: RAMSIZ SET TO 0

:-

BGNMSG RAMERR
RAMERR:: JSR PC,PRAMPKT ;PRINT RAM/PACKET DATA
ENDMSG
L10021: TRAP C\$MSG

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

:+
:PRINT ROUTINE TO DISPLAY RAM/PACKET DATA

:INPUTS:

1979
 1980
 1981
 1982
 1983
 1984
 1985
 1986
 1987
 1988
 1989
 1990
 1991
 1992
 1993
 1994
 1995 015506
 015506
 1996 015506 004737 010270
 1997 015512 004737 013766
 1998 015516
 015516
 015516 104423
 1999
 2000
 2001
 2002
 2003
 2004
 2005
 2006
 2007
 2008
 2009
 2010
 2011
 2012
 2013 015520
 015520
 2014 015520 042701 177400
 2015 015524 042702 177400
 2016 015530 004737 010062
 2017 015534 004737 007736
 2018 015540
 015540
 015540 104423
 2019
 2020
 2021
 2022
 2023
 2024
 2025
 2026
 2027
 2028
 2029

```

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

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

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

```

2030
2031
2032 015542          :-
      015542          BGNMSG  TIMEXP
2033 015542          TIMEXP::
      015542 012746 015570 PRINTX  #TIMSGO          ;PRINT HEADER
      015546 012746 000001 MOV     #TIMSGO,-(SP)
      015552 010600      MOV     #1,-(SP)
      015554 104415      MOV     SP,R0
      015556 062706 000004 TRAP   C$PNTX
2034 015562 004737 007736 ADD     #4,SP
2035 015566          JSR     PC,PRIXOR          ;PRINT THE DATA
      015566          ENDMSG
      015566 104423      L10024:
      015566          TRAP   C$MSG
2036
2037
2038 015570          045    116    045  TIMSGO: .ASCIZ  '%X%A TIMER A STATUS IS IN BIT 3%X%A TIMER B STATUS IS IN BIT 2'
2039          .EVEN
    
```


2064
2065
2066
2067
2068
2069
2070

.SBTTL GLOBAL SUBROUTINES SECTION

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

2072 .SBTTL SOFINIT - SOFT INITIALIZE OF CONTROLLER

2073
 2074
 2075
 2076 :ROUTINE TO DO A SCFT INITIALIZE OF THE CONTROLLER
 2077 :BY WRITING INTO THE TSSR REGISTER. AFTER THE INIT,
 2078 :THE TSSR REGISTER IS TESTED FOR ERRORS. ANY ERRORS
 2079 :DETECTED SHOULD BE TREATED AS DEVICE FATAL ERRORS.

2080
 2081 :INPUTS:
 2082
 2083 R5 ADDRESS OF FIRST REGISTER

2084
 2085 :OUTPUTS:
 2086
 2087 R0 CONTENTS OF TSSR, IF ERROR
 2088 CARRY SET IF INIT WAS OKAY
 2089 CLEAR IF FATAL ERROR

2090
 2091 :CALLING SEQUENCE:
 2092
 2093 MOV #ADDRESS,R5
 2094 JSR PC,SOFINIT
 2095 BCS CONTINUE
 2096 ERRDF ;REPORT FATAL ERROR
 2097
 2098
 2099

2100	015764				SOFINIT::		
2101	015764				SAVREG		; SAVE THE REGISTERS
2102	015770	012765	000000	000002	MOV #0,TSSR(R5)		; DO THE INIT.
2103	015776	004737	016240		JSR PC,WAITF		; WAIT FOR SSR
2104	016002	016500	000002		MOV TSSR(R5),R0		;GET THE TSSR REGISTER
2105	016006	010004			MOV R0,R4		;TSSR CONTENTS
2106	016010	042704	176277		BIC #^C<HIADDR!OFL>,R4		;R4 HAS EXPECTED CONTENTS
2107	016014	052704	002200		BIS #SSR!NBA,R4		;ONLY EXPECTED BITS SET ?
2108	016020	020400			CMP R4,R0		;BRANCH IF OKAY
2109	016022	001402			BEQ \$\$;CLEAR THE CARRY FOR ERROR
2110	016024	000241			CLC		;GO TO EXIT
2111	016026	000401			BR 10\$;SET THE CARRY BIT
2112	016030	000261			5\$: SEC		;RETURN TO CALLER
2113	016032	000207			10\$: RTS	PC	

```

2115 .SBTTL CHKAMB - CHECK TSSR FOR AMBIGUITY
2116
2117
2118
2119 :+
2120 :THIS ROUTINE TESTS THE CONTENTS OF THE TSSR REGISTER
2121 :FOR AMBIGUITY
2122 :INPUT:
2123
2124 RO CONTENTS UF TSSR
2125
2126 :OUTPUT:
2127
2128 RO CONTENTS OF TSSR
2129
2130 CARRY SET - NO AMBIGUITY
2131 CLR - AMBIGUOUS CONTENTS
2132
2133 :-
2134
2135 016034 CHKAMB: SAVREG ;SAVE THE GENERAL REGISTERS
2136 016034 MOV RO,R4 ;CONTENTS OF TSSR
2137 016040 010004 BIT #SC,RO ;IS BIT 15 SET ?
2138 016042 032700 100000 BNE 5$ ;BRANCH IF YES
2139 016046 001004 BIT #^C<NBA!OFL!SSR!HIADDR>,RO ;ANY OTHER BITS SET ?
2140 016050 032700 174077 BNE 40$ ;MUST BE AN ERROR
2141 016054 001023 BR 45$ ;RETURN WITH SUCCESS
2142 016056 000424 5$: BIT #SSR,RO ;IS READY BIT SET ?
2143 016060 032700 000200 BNE 10$ ;BRANCH IF READY BIT IS SET.
2144 016064 001011 BIT #BITS,RO ;IS FATAL ERROR BIT SET ?
2145 016066 032700 000040 BEQ 40$ ;ERROR IF NOT
2146 016072 001414 BIC #^CTERCLS,R4 ;CLEAR ALL BUT TERMINATION CODE
2147 016074 042704 177761 CMP R4,#16 ;ALL THREE BITS MUST BE SET
2148 016100 020427 000016 BNE 40$ ;ERROR IF NOT SET
2149 016104 001007 BR 45$ ;OK IF ALL ARE SET
2150 016106 000410 10$: BIT #BITS,RO ;IS FATAL ERROR BIT SET ?
2151 016110 032700 000040 BEQ 45$ ;ERROR IF BIT IS SET WITH SSR
2152 016114 001405 BIT #BIT2!BIT1,RO ;IS THIS A FUNCTION REJECT
2153 016116 032700 000006 BNE 45$ ;BR, IF TSSR IS OK
2154 016122 001002 40$: CLC ;AMBIGUOUS CONTENTS
2155 016124 000241 BR 50$
2156 016126 000401 45$: SEC ;SHOW SUCCESS - NO AMBIGUITY
2157 016130 000261 50$: RTS PC ;RETURN TO CALLER
2158 016132 000207
2159
    
```



```

2161                .SBTTL ENAINT,DSBINT - ENABLE/DISABLE INTERRUPTS
2162                :
2163                : DEFAULT DISPLAY INTERRUPT HANDLERS.
2164                : IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
2165                : OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
2166                :
2167                :
2168                : BIT DEFINITIONS FOR "INTMASK" AND "INTFLAG" BYTES:
2169                :
2170                :       IOKCKIN=BIT7      ; DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
2171                :       IOKSTP=BIT0       ; EXPECT "STOP" INTERRUPT.
2172                :
2173                : INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
2174 016134          INTMASK: .BYTE 0
2175                : INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
2176 016135          INTFLAG: .BYTE 0
2177                :
2178                : SAVED INTERRUPT VECTOR:
2179 016136          INTVEC: .WORD 0
2180                : SAVE CPU PC
2181 016140          INTCPC: .WORD 0
2182                :
2183                : SUBROUTINE TO ENABLE INTERRUPTS:
2184 016142          ENAINT: MOV    RO,-(SP)      ;SAVE RO
2185 016144          MOV    IVEC,RO          ;GET POINTER TO VECTORS
2186 016150          MOV    #INTR,(RO)+     ;SET UP INTERRUPT VECTOR
2187 016154          MOV    #PRI07,(RO)+
2188 016160          MOV    (SP)+,RO        ;RESTORE RO
2189 016162          MOV    (SP),-(SP)
2190 016164          MOV    #0,2(SP)       ;SET CPU TO LEVEL 0
2191 016172          RTI
2192                :
2193                : SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 7)
2194 016174          DSBINT: MOV    (SP),-(SP)
2195 016176          MOV    #PRI07,2(SP)
2196 016204          RTI
2197

```

```
2199 .SBTTL INTR - INTERRUPT HANDLERS
2200
2201 016206 BGNSRV INTR ;DEFINE INTERRUPT ENTRY
      016206 INTR::
2202 016206 012737 000001 002222 MOV #1,INTRECV ;SET FLAG TO SHOW INTERRUPT RECEIVED
2203 016214 105037 016135 CLRB INTFLAG ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2204 016220 132737 000001 016134 BITB #IOKSTP,INTMASK ;EXPECTING STOP INTERRUPT?
2205 016226 001003 BNE 1$ ;BR IF YES
2206 016230 152737 000001 016135 BISB #IOKSTP,INTFLAG ;NO. SET THE ERROR FLAG.
2207
2208 ;SAVE REGISTERS, MSG BUFFER, ETC.
2209 016236 1$:
2210 016236 ENDSRV
      016236 L10026:
      016236 RTI
2211
2212
```

```

2214 .SBTTL WAITF - WAIT FOR SUBSYSTEM READY
2215
2216 : SUBROUTINE TO WAIT FOR THE SUBSYSTEM READY FLAG
2217 :
2218 : INPUTS:
2219
2220 R5 ADDRESS OF FIRST DEVICE REGISTER
2221
2222 : OUTPUTS:
2223
2224 R0 CONTENTS OF LAST TSSR READ
2225 CARRY SET - READY BIT SET
2226 CLR - TIMEOUT WAITING FOR READY
2227
2228 016240 000401 WAITF:: BR 1$ ;NOP WHEN SUPER FIXED
2229 016242 104422 BREAK ; DO A SUPVSR BREAK FIRST.
016242 104422 TRAP CSBRK
2230 016244 012746 003000 1$: MOV #3000,-(SP) ;300 MSEC TIMER
2231 016250 016500 000002 2$: MOV TSSR(R5),R0 ;READ THE TSSR REGISTER
2232 016254 105700 TSTB R0 ;TEST FOR READY BIT SET
2233
2234 016256 100420 BMI 3$ ; EXIT ON STOP FLAG.
2235 016260 DELAY 1 ; WAIT 100 USEC
016260 012727 000001 MOV #1,(PC)+
016264 000000 .WORD 0
016266 013727 002116 MOV LSDLY,(PC)+
016272 000000 .WORD 0
016274 005367 177772 DEC -6(PC)
016300 001375 BNE -4
016302 005367 177756 DEC -22(PC)
016306 001367 BNE -20
2236 016310 005316 DEC (SP) ;REDUCE DELAY COUNT
2237 016312 001356 BNE 2$ ;RETRY UNTIL TIMER EXPIRES
2238 016314 000241 CLC ; C = 0, CONTROLLER STILL RUNNING...
2239 016316 000401 BR 4$ ;...OR HUNG-UP AFTER 300 MSEC.
2240 016320 000261 3$: SEC ; C = 1, CONTROLLER IS STOPPED.
2241 016322 005326 4$: DEC (SP)+ ;RESTORE STACK WITHOUT CHANGING CARRY BIT
2242 016324 000207 RTS PC
    
```

2244 .SBTTL CHKTSSR - CHECK TSSR FOR READY
 2245

2246 :+
 2247 :
 2248 :THIS ROUTINE WAITS FOR READY IN THE TSSR
 2249 :AND TESTS FOR AMBIGUOUS BIT SETTINGS IN TSSR.
 2250

2251 :INPUT:
 2252

2253 : R5 ADDRESS OF CSR REGISTERS
 2254

2255 :OUTPUT:
 2256

2257 : R0 CONTENTS OF TSSR
 2258 : CARRY SET - OKAY
 2259 : CLR - NOT READY AMBIGUOUS, OR SC SET
 2260
 2261 :-
 2262

2263 016326
 2264 016326 004737 016240
 2265 016332 103014
 2266 016334 004737 016034
 2267 016340 103006
 2268 016342 032700 100000
 2269 016346 001405
 2270 016350 032700 074000
 2271 016354 001402
 2272 016356 000241
 2273 016360 000401
 2274 016362 000261
 2275 016364 000207

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

```

2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288 016366 012737 016420 000004
2289 016374 012737 000200 000006
2290 016402 005003
2291 016404 005711
2292
2293 016406 020102
2294 016410 001407
2295 016412 062701 000002
2296 016416 000772
2297
2298 016420 005103
2299 016422 012716 016430
2300 016426 000002
2301 016430
      016430 012700 000004
      016434 104436
2302 016436 005703
2303 016440 001401
2304 016442 000261
2305 016444 000207
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318 016446
2319 016446 005737 002166
2320 016452 001006
2321 016454 005737 002202
2322 016460 100403
2323 016462 005337 002214
2324 016466 001002
2325 016470 000241
2326 016472 000401
2327 016474 000261
2328 016476 000207

      .SBTTL XNXM - CHECK FOR NONEXISTENT MEMORY
      :+
      : ROUTINE TO TEST FOR A NEXM IN THE RANGE (R1) THRU (R2).
      : ON RETURN, IF 'C' = 1, (R1) = NEXM ADDRESS.
      : 'C' = 0, ALL ADDRESSES OK.
      :
      : CALL: MOV ADR1,R1
      :        MOV ADR2,R2
      :        JSR PC,NXM
      :        RETURN          ;TEST 'C' AND PROCEED.
XNXM: MOV #2$,@#4          ; SET BUSERR VECTOR.
      MOV #PRI04,@#6
      CLR R3                ; FLAG.
1$:   TST (R1)              ; TEST THE ADDRESS(ES).
      ; IF ANY TRAP, CONTINUE AT 2$.
      ; OTHERWISE, CONTINUE HERE.
      ; BR IF FINISHED (NO NEXM'S).
      ; SET NEXT ADDRESS...
      ; ...AND CONTINUE.
      CMP R1,R2
      BEQ 3$
      ADD #2,R1
      BR 1$
2$:   COM R3                ; GOT ONE, SET FLAG...
      MOV #3$, (SP)
      RTI                    ; ...AND DISMISS INTERRUPT...
3$:   CLRVEC #4              ; ...AND GIVE BACK THE VECTOR.
      MOV #4,R0
      TRAP C$CVEC
      TST R3
      BEQ .+4                ; DID WE CATCH ONE ??
      SEC                    ; NO, 'C' = 0, SKIP NEXT.
      RTS PC                 ; YES, 'C' = 1, (R1) = NEXM ADDR.

      .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

```

2330
 2331
 2332
 2333
 2334
 2335
 2336
 2337
 2338
 2339
 2340
 2341
 2342
 2343
 2344
 2345
 2346
 2347
 2348
 2349
 2350
 2351
 2352
 2353
 2354
 2355
 2356
 2357

```

.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
:
: -
  
```

2358 016500
 2359 016500 010046
 2360 016502 005037 003150
 2361 016506 005037 016746
 2362 016512 005037 005766
 2363 016516 105037 016134
 2364 016522 013700 002200
 2365 016526 006300
 2366 016530 005737 003110
 2367 016534 001430
 2368 016536 1000:0
 2369 016540 052760 160000 003172
 2370 016546
 016546 104455
 016550 000001
 016552 003734
 016554 005732
 2371 016556 000407
 2372 016560 052760 160001 003172 3\$:
 2373 016566
 016566 104455
 016570 000002
 016572 004331
 016574 000000
 2374 016576 012737 177777 003106 2\$:
 2375 016604
 016604 013700 002200
 016610 104451
 2376 016612

```

TSTSETUP::
MOV R0,-(SP) ;SAVE THE TEST ID MESSAGE
CLR SIFLAG ; CLEAR "SOFT INIT" FLAG
CLR ERRK ; CLEAR LOCAL ERROR COUNTER.
CLR EXTA ; CLEAR ERROR EXTENSION FLAG.
CLRB INTMASK ; CLEAR INTERRUPT MASK (CHECK ERROR)
MOV UNITN,R0 ; GET THE UNIT NUMBER,
ASL R0 ; ... AND MAKE IT A WORD OFFSET.
TST NODEV ; DID STARTUP FIND THE DEVICE?
BEQ 4$ ; BR IF YES
BPL 3$ ; BR IF NOT IDLE
BIS #160000,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
ERRDF 1,NXR,NXRERR ; NO DEVICE HERE -- PRINT IT
TRAP C$ERDF
.WORD 1
.WORD NXR
.WORD NXRERR
BR 2$
BIS #160001,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
ERRDF 2,NOINIT ; DEVICE NOT IDLE
TRAP C$ERDF
.WORD 2
.WORD NOINIT
.WORD 0
MOV #-1,DUFLG ; DROP THE UNIT
DODU UNITN
MOV UNITN,R0
TRAP C$DODU
DOCLN ; ABORT THE PASS
  
```

2377	016612	104444			TRAP	CSDCLN		
2378	016614	000423			BR	5\$		
2379	016616		4\$:		RFLAGS	RO		; GET THE OPERATOR FLAGS.
2380	016620	032700		001000	TRAP	CSRFLA		
2381	016624	001412			BIT	#PNT,RO		; PRINT THE TEST NUMBERS?
2382	016626	011600			BEG	1\$; BR IF NO
2383	016630				MOV	(SP),RO		;GET THE ID MESSAGE
	016630	010046			PRINTF	#TNAM,RO		;DISPLAY THE TEST ID
	016632	012746		016674	MOV	RO,-(SP)		
	016636	012746		000002	MOV	#TNAM,-(SP)		
	016642	010600			MOV	#2,-(SP)		
	016644	104417			MOV	SP,RO		
	016646	062706		000006	TRAP	C\$PNTF		
2384	016652	005237		002212	ADD	#6,SP		; BUMP TEST COUNTER.
2385	016656				1\$:	INC	TSTCNT	;PRIORITY THAT OF DEVICE
	016656	013700		002210	SETPRI	IPRI		
	016662	104441			MOV	IPRI,RO		
2386	016664	005726			TRAP	C\$SPRI		
2387	016666	013705		002204	5\$:	TST	(SP)+	;FIX UP THE STACK
2388	016672	000207			MOV	CSRADDR,R5		; ADDRESS OF TSV REGISTERS ON UNIBUS
2389	016674	045	123	045	RTS	PC		
2390					TNAM:	.ASCIZ	'%S%T%A Test'	
						.EVEN		

```

2392
2393
2394
2395
2396
2397 016710
      016710 104421
2398 016712 030027 020000
2399 016716 001412
2400 016720
      016720 013746 016746
      016724 012746 016750
      016730 012746 000002
      016734 010600
      016736 104417
      016740 062706 000006
2401 016744 000207
2402
2403 016746 000000
2404 016750 045 101 040
2405 016767 105 122 122
2406
2407
2408
2409
2410
2411
2412 017034 005237 016746
2413 017040 010046
2414 017042 013700 002200
2415 017046 006300
2416 017050 062700 003172
2417 017054 005210
2418 017056 032710 007777
2419 017062 001001
2420 017064 005310
2421 017066 012600
2422 017070 000207
2423
2424 017072 010046
2425 017074 013700 002200
2426 017100 006300
2427 017102 016000 003172
2428 017106 042700 170000
2429 017112 020037 002172
2430 017116 103004
2431 017120 023737 016746 002170
2432 017126 103417
2433 017130
      017130 104421
2434 017132 032700 000040
2435 017136 001013
2436 017140 012737 177777 003106
2437 017146
      017146 104455
      017150 000004
      017152 016767
    
```

```

.SBTTL TSTEND - PRINT ERRORS RECEIVED
:
: AT END OF EACH TEST, PRINT THE NUMBER OF ERRORS RECEIVED
: IF NORMAL ERROR REPORTING IS DISABLED (FLA:IER).
:
TSTEND: RFLAGS R0
        TRAP CSRFLA
        BIT R0,#IER
        BEQ 1$ ; BR IF 'IER' NOT SET.
        PRINTF #ESUM,ERRK ; PRINT ERROR COUNT.
        MOV ERRK,-(SP)
        MOV #ESUM,-(SP)
        MOV #2,-(SP)
        MOV SP,R0
        TRAP C$PNTF
        ADD #6,SP
1$: RTS PC

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

.SBTTL INCERK - INCREMENT LOCAL ERROR COUNT
:
: +
: ROUTINES TO INCREMENT LOCAL ERROR COUNT AND CHECK FOR LIMIT:
: -
INCERK: INC ERRK ; INCREMENT LOCAL ERROR COUNT
        MOV R0,-(SP) ; SAVE R0
        MOV UNITN,R0 ; GET UNIT NUMBER,
        ASL R0 ; ... AND MAKE IT A WORD OFFSET.
        ADD #ERTABL,R0 ; R0 GETS ADDRESS OF ERROR TABLE ENTRY.
        INC (R0) ; INCREMENT THE DEVICE ERROR COUNT
        BIT #7777,(R0) ; DID WE OVERFLOW THE FIELD?
        BNE 1$ ; BR IF NO.
        DEC (R0) ; YES -- BACK IT UP TO 7777.
1$: MOV (SP)+,R0 ; RESTORE R0
        RTS PC ; RETURN TO CALLER.

CKEMAX: MOV R0,-(SP) ; SAVE R0
        MOV UNITN,R0 ; GET UNIT NUMBER
        ASL R0 ; ... AND MAKE IT A WORD OFFSET
        MOV ERTABL(R0),R0 ; GET ERROR TABLE ENTRY
        BIC #170000,R0 ; EXTRACT ERROR COUNT FIELD
        CMP R0,GERRMAX ; IS GLOBAL LIMIT EXCEEDED FOR THIS UNIT?
        BHIS 1$ ; BR IF YES
        CMP ERRK,LERRMAX ; IS LOCAL LIMIT EXCEEDED FOR THIS TEST?
        BLO 2$ ; BR IF NO
1$: RFLAGS R0 ; GET OPERATOR FLAGS
        TRAP CSRFLA
        BIT #IDU,R0 ; IS DROPPING INHIBITED?
        BNE 2$ ; BR IF YES.
        MOV #-1,DUFLG ; NO -- DROP THE UNIT
        ERRDF 4,EMAXDU
        TRAP C$ERDF
        .WORD 4
        .WORD EMAXDU
    
```


2438	017154	0G0000		.WORD	0	
	017156			DODU	UNITN	
	017156	013700	002200	MOV	UNITN R0	
	017162	104451		TRAP	CSDODL	
2439	017164			DOCLN		
	017164	104444		TRAP	CSDCLN	
2440	017166	012600	2\$:	MOV	(SP)+,R0	; RESTORE R0
2441	017170	000207		RTS	PC	; RETURN TO CALLER
2442						

```
2444 .SBTTL CKDROP - CHECK IF UNIT SHOULD BE DROPPED
2445
2446 :+ CHECK IF UNIT SHOULD BE DROPPED
2447 :-
2448 CKDROP: MOV R0,-(SP)
2449 FORCERROR R0 1$,NOTSSR
2450 RFLAGS R0
2451 TRAP CSRFLA
2452 BIT #IDU,R0
2453 BNE 1$
2454 MOV (SP),R0
2455 MOV #-1,DUFLG
2456 DODU UNITN
2457 MOV UNITN,R0
2458 TRAP CSDODU
2459 DOCLN ;ABORT THE PASS
2460 TRAP CSDCLN
2461 1$: MOV (SP)+,R0
2462 RTS PC
2463
2464 .SBTTL CONFIG - DETERMINE CONFIGURATION OF SYSTEM
2465 : SUBROUTINE - DETERMINE CONFIGURATION OF TSV05 SYSTEM.
2466 :
2467 CONFIG:
2468 JSR PC,SOFINIT
2469 RTS PC
2470
2471
2472
```

```
2474 .SBTTL KTON,KTOFF - ENABLE/DISABLE MEMORY MANAGEMENT
2475 .
2476 : SUBROUTINE - ENABLE MEM MGT.
2477 :
2478 017246 005737 003.26 KTON: TST KFLG : GOT KT?
2479 017252 001403 BEQ 1$ : NO.
2480 017254 012737 000001 177572 MOV #1,SRO : YES. ENABLE KT11.
2481 017262 000207 1$: RTS PC
2482
2483
2484
2485 :
2486 : SUBROUTINE - DISABLE MEM MGT.
2487 :
2488 017264 005737 003126 KTOFF: TST KFLG : GOT KT11?
2489 017270 001405 BEQ 1$ : NO.
2490 017272 000240 NOP
2491 017274 000240 NOP
2492 017276 012737 000000 177572 MOV #0,SRO : DISABLE KT.
2493 017304 000207 1$: RTS PC
2494
2495
```

2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536

.SBTTL SETMAP - SETUP PAR6 MAPPING

```

:
:
: THIS ROUTINE SETS UP KERNEL PAR6 TP HANDLE
: AN 18 BIT ADDRESS. THE OFFSET INTO THE PAGE
: IS RETURNED BIASED TO PAR6.
    
```

: INPUTS:

```

: R0 HIGH ORDER ADDRESS BITS
: R1 LOW ORDER ADDRESS BITS
    
```

: OUTPUTS:

```

: R0 OFFSET INTO BLOCK WITH PAR6 BIAS (I.E. THE ADDRESS)
: CARRY SET IF SUCCESS
: CLR IF ERROR
    
```

SETMAP:

```

SAVREG ;SAVE R1-R4 UNTIL NEXT RETURN
TST KTF LG ;SYSTEM HAVE ABOVE 28K?
BEQ 10$ ;BR IF NO
MOV R1,R2 ;SAVE LOW ORDER BITS
.REPT 6
ASR R0 ;CONVERT WORD ADDRESS TO 32W BLOCKS
ROR R1 ;MAKE IT DOUBLE PRECISION
.ENDR
BIC #177,R1 ;ALINE FOR LOWER 4K BOUNDARY
CMP R1,KTF LG ;HIGHER THAN EXISTING MEMORY?
BHS 10$ ;BR IF YES
MOV R1,@#KIPAR6 ;SETUP MAPPING REGISTER PAR6
BIC #160000,R2 ;SETUP DISPLACEMENT IN PAGE
ADD #140000,R2 ;ADD IN PAR6 BIAS
MOV R2,R0 ;RETURN IN R0
SEC ;SET SUCCESS
BR 15$
10$: CLC ;SET FAILURE
15$: RTS PC ;RETURN
    
```

```

017306
017306
005737 003126
017312
001433
017316
010102
017320
000006
017352 042701 000177
017356 020137 003126
017362 103011
017364 010137 172354
017370 042702 160000
017374 062702 140000
017400 010200
017402 000261
017404 000401
017406 000241
017410 000207
    
```

```

10$:
15$:
    
```

```

2538 .SBTTL FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN
2539
2540 * FILL MEMORY WITH A BACKGROUND PATTERN
2541
2542 : INPUTS:
2543
2544 : RO = BACKGROUND PATTERN
2545 : FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
2546 : KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
2547
2548 : OUTPUTS:
2549
2550 : NONE
2551
2552 -
2553 FILLMEM:
2554 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
2555 JSR PC,KTOFF ;DISABLE KT.
2556 MOV RO,R3 ;COPY TEST PATTERN
2557 MOV FREE,R1 ;GET FIRST FREE LOCATION
2558 MOV FRESIZ,R2 ;SIZE OF FREE SPACE BELOW 28K.
2559 10$: MOV R3,(R1)+ ;STORE A BACKGROUND WORD
2560 DEC R2 ;DONE ALL MEMORY IN FREE SPACE?
2561 BGT 10$ ;BR IF NO
2562 TST KTFLG ; GOT KT?
2563 BEQ 55$ ; NO. GET OUT.
2564 JSR PC,KTON ; YES. ENABLE KT.
2565 CLR RO ;HIGH ORDER ADDRESS START
2566 MOV PST32W,R1 ;GET >28K START ADDRESS (IN 32W BLOCKS)
2567 .REPT 6
2568 CLC ;CLEAR C BIT
2569 ROL R1 ;CONVERT BLOCKS TO WORDS
2570 ROL RO ;MAKE IT DOUBLE PRECISION
2571 .ENDR
2572 JSR PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
2573 30$: MOV R3,(R0)+ ;STORE TEST PATTERN IN >28K ADDRESS
2574 CMP RO,#160000 ;END OF PAR6 MAPPING AREA?
2575 BLO 30$ ;BR IF NO
2576 SUB #20000,RO ;BACKUP INTO PAR6 MAPPING BEGIN
2577 ADD #200,@#KIPAR6 ;POINT TO NEXT 4K BLOCK >28K.
2578 CMP @#KIPAR6,KTFLG ;END OF MEMORY?
2579 BEQ 50$ ;BR IF YES
2580 TST T23A ;11/23A?
2581 BEQ 35$ ;NO KEEP GOING
2582 MOV SR0,R4 ;GET SR0 CONTENTS
2583 BIC #177761,R4 ;CLEAR ALL BUT PAGE NUMBER
2584 CMP #16,R4 ;SEE IF PAGE 7
2585 BEQ 50$ ;EXIT IF THERE
2586 35$: TST T23B ;11/23B?
2587 BEQ 45$ ;NO KEEP GOING
2588 CMP @#KIPAR6,#7600 ;REACHED 18 BITS?
2589 BHIS 40$ ;YES
2590 BR 45$ ;NO KEEP GOING
2591 40$: MOV #20,SR3 ;SET 22 BIT RELOCATION
2592 45$: JMP 30$ ;KEEP GOING ON ETC.
2593 50$: JSR PC,KTOFF ;DISABLE KT.
2594 55$: RTS PC
    
```

2595
2596

2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654

017650
017650
010003
004737 017264
013701 003120
013702 003122
020311
001411
010137 002236
005037 002234
010337 002230
011137 002232
000474
005721
005302
003362
005737 003126
001472
004737 017246
005000
013701 003146
000006
042701 000177
010046
010146
004737 017306
010004
012601
012600
020314
001411
010037 002234

```

.SBTTL CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN
+
COMPARE MEMORY WITH A BACKGROUND PATTERN
INPUTS:
    RO = BACKGROUND PATTERN
    FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
    KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
OUTPUTS:
    CARRY - SET IF NO ERROR
    CARRY - CLR IF ERROR
IMPLICIT OUTPUTS:
    ERRHI - ERROR HIGH ADDRESS
    ERRLO - ERROR LOW ADDRESS
    EXPD  - EXPECTED DATA
    RECV  - RECEIVED DATA
-
CMPMEM:
    SAVREG
    MOV    RO,R3
    JSR   PC,KTOFF
    MOV   FREE,R1
    MOV   FRESIZ,R2
10$:    CMP   R3,(R1)
    BEQ   15$
    MOV   R1,ERRLO
    CLR   ERRHI
    MOV   R3,EXPD
    MOV   (R1),RECV
    BR    50$
15$:    TST   (R1)+
    DEC   R2
    BGT   10$
    TST   KTFLG
    BEQ   55$
    JSR   PC,KTON
    CLR   RO
    MOV   PST32W,R1
    .REPT 6
    ROL   R1
    ROL   RO
    .ENDR
    BIC   #177,R1
    MOV   RO,-(SP)
    MOV   R1,-(SP)
    JSR   PC,SETMAP
    MOV   RO,R4
    MOV   (SP)+,R1
    MOV   (SP)+,RO
30$:    CMP   R3,(R4)
    BEQ   32$
    MOV   RO,ERRHI
  
```

```

:SAVE R1-R5 UNTIL NEXT RETURN
:COPY TEST PATTERN
:DISABLE KT.
:GET FIRST FREE LOCATION
:SIZE OF FREE SPACE BELOW 28K.
:FREE SPACE LOCATION EQUAL TO EXPD?
:BR IF YES
:SAVE ADDRESS IN ERROR
:NO HIGH ADDRESS
:SAVE EXPD FOR ERROR REPORT
:SAVE RECV FOR ERROR REPORT
:POINT TO NEXT ADDRESS
:DONE ALL MEMORY IN FREE SPACE?
:BR IF NO
:GOT KT?
:NO. GET OUT.
:YES. ENABLE KT.
:HIGH ORDER ADDRESS START
:GET >28K START ADDRESS (IN 32W BLOCKS)
:CONVERT BLOCKS TO WORDS
:MAKE IT DOUBLE PRECISION
:ALINE 4K BOUNDARY
:SAVE HIGH ORDER
:SAVE LOW ORDER
:SETUP PAR6 MAPPING REGISTER
:COPY ADDRESS BIASED TO PAR6
:RESTORE LOW ORDER IN NON PAR6 FORMAT
:RESTORE HIGH ORDER IN NON PAR6 FORMAT
:ABOVE 28K LOCATION EQUAL EXPD?
:BR IF YES
:SAVE HIGH ORDER IN ERROR
  
```

2655	020030	010137	002236		MOV	R1,ERRLO	:SAVE LOW ORDER IN ERROR
2656	020034	010337	002230		MOV	R3,EXPD	:SAVE EXPD FOR ERROR REPORT
2657	020040	011437	002232		MOV	(R4),RECV	:SAVE RECV FOR ERROR REPORT
2658	020044	000421			BR	50\$:
2659	020046	06270*	000002	32\$:	ADD	#2,R1	:UPDATE NON PAR6 ADDRESS
2660	020052	005500			ADC	R0	:MAKE IT DOUBLE PRECISION ADD
2661	020054	062704	000002		ADD	#2,R4	:UPDATE PAR FORMAT ADDRESS
2662	020060	020427	160000		CMP	R4,#160000	:END OF PAR6 MAPPING AREA?
2663	020064	103755			BLO	30\$:BR IF NO
2664	020066	162704	020000		SUB	#20000,R4	:BACKUP INTO PAR6 MAPPING BEGIN
2665	020072	062737	000200	172354	ADD	#200,@#KIPAR6	:POINT TO NEXT 4K BLOCK >28K.
2666	020100	023737	172354	003126	CMP	@#KIPAR6,KTFLG	:END OF MEMORY?
2667	020106	101744			BLOS	30\$:BR IF NO
2668	020110	004737	017264	50\$:	JSR	PC,KTOFF	:TURN OFF MEMORY MAPPING
2669	020114	000241			CLC		:SET FAILURE
2670	020116	000403			BR	60\$:
2671	020120	004737	017264	55\$:	JSR	PC,KTOFF	:TURN OFF MEMORY MAPPING
2672	020124	000261			SEC		:SET SUCCESS
2673	020126	000207		60\$:	RTS	PC	
2674							


```

2676 .SBTTL REGSAV - SAVE R1-R5 ON STACK
2677 :+
2678 :ROUTINE TO
2679 :SAVE R1 THROUGH R5 ON THE STACK
2680 :CALLING SEQUENCE:
2681 :
2682 :       JSR     R5,REGSAV
2683 :
2684 :THIS IS A COOROUTINE WHICH TRANSFER CONTROL BACK TO
2685 :THE CALLING ROUTINE. AT THE END OF THE CALLING ROUTINE,
2686 :THE RTS PC RETURNS CONTROL TO THIS ROUTINE TO RESTORE
2687 :REGISTERS.
2688 :
2689 :THIS ROUTINE SHOULD ONLY BE CALLED FROM ROUTINES WHICH ARE
2690 :CALLED VIA A JSR PC INSTRUCTION
2691 :-
2692 :
2693 :
2694 :
2695 :
2696 020130 REGSAV:
2697 020130 010446      MOV     R4,-(SP)
2698 020132 010346      MOV     R3,-(SP)
2699 020134 010246      MOV     R2,-(SP)
2700 020136 010146      MOV     R1,-(SP)
2701 020140 010546      MOV     R5,-(SP)
2702 020142 016605 000012  MOV     10.(SP),R5
2703 020146 004736      JSR     PC,@(SP)+
2704 020150 012601      MOV     (SP)+,R1
2705 020152 012602      MOV     (SP)+,R2
2706 020154 012603      MOV     (SP)+,R3
2707 020156 012604      MOV     (SP)+,R4
2708 020160 012605      MOV     (SP)+,R5
2709 020162 000207      RTS     PC
2710

```

```

2712 .SBTTL GETPAT - GET 8 BIT PATTERN FROM OPERATOR
2713
2714 :+
2715 :ROUTINE TO REQUEST AN 8 BIT DATA PATTERN FROM THE OPERATOR
2716
2717 :INPUTS:
2718
2719 NONE.
2720
2721 :OUTPUTS:
2722
2723 RO OCTAL NUMBER FROM THE OPERATOR
2724
2725 :CALLING SEQUENCE:
2726
2727 JSR PC,GETPAT
2728
2729 :-
2730
2731 GETPAT::
2732 SAVREG ;SAVE THE GENERAL REGISTERS
2733 1$: GMANID DATASC,PATDAT,0,377,0,377,NO
      TRAP CS$GMAN
      BR 10000$
      .WORD PATDAT
      .WORD T$CODE
      .WORD DATASC
      .WORD 377
      .WORD T$LLOLIM
      .WORD T$HILIM
2734 10000$: BNCOMPLETE 1$ ;RETRY IF ERROR
      BCC 1$
      MOV PATDAT,RO ;DATA PATTERN FROM OPERATOR
      RTS PC ;RETURN TO CALLER
2735
2736 :+
2737 :LOCAL DATA AREA
2738
2739 :-
2740
2741
2742 020164 104443
2743 020170 000406
2744 020172 020220
2745 020174 000022
2746 020176 000022
2747 020200 000377
2748 020202 000000
2749 020204 000377
2750 020206 000377
2751 020210
2752 020210 103367
2753 020212 013700 020220
2754 020216 000207
2755
2756 PATDAT: .WORD 0 ;TEMPORARY STORAGE FOR DATA
2757 DATASC: .ASCIZ 'ENTER DATA PATTERN'
2758 .EVEN
  
```

```

2746          .SBTTL  GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE
2747
2748          :
2749          :ROUTINE TO ISSUE A MENU AND GET
2750          :THE OPERATOR'S RESPONSE.
2751          :
2752          :INPUTS:
2753
2754          :      R0      ADDRESS OF ASCIZ STRING OF MENU
2755          :      R1      MAXIMUM ALLOWABLE OPERATOR RESPONSE
2756
2757          :OUTPUTS:
2758
2759          :      R0      NUMBER OF THE OPERATOR'S SELECTION
2760          :
2761          :-
2762
2763          GETSEL::
2764          SAVREG          ;SAVE GENERAL REGISTERS
2765          MOV            R0,R2          ;SAVE THE MENU ADDRESS
2766          MOV            R2,R3          ;START OF MENU STRING
2767          TST            (R3)          ;END OF ASCII ?
2768          BEQ            3$           ;BRANCH IF ALL LINES DISPLAYED
2769          PRINTF        #SELASC,(R3)+ ;DISPLAY THE MENU
2770          MOV            (R3)+,-(SP)
2771          MOV            #SELASC,-(SP)
2772          MOV            #2,-(SP)
2773          MOV            SP,R0
2774          TRAP          C$PNTF
2775          ADD            #6,SP
2776          BR            2$
2777          3$:           GMANID        MENASC,MENRES,D,-1,0,-1,NO
2778          TRAP          C$GMAN
2779          BR            10001$
2780          .WORD        MENRES
2781          .WORD        T$CODE
2782          .WORD        MENASC
2783          .WORD        -1
2784          .WORD        T$LOLIM
2785          .WORD        T$HILIM
2786          10001$:
2787          BNCOMPLETE    1$           ;RETRY IF ERROR
2788          BCC            1$
2789          MOV            MENRES,R0     ;GET THE OPERATOR'S REPLY
2790          CMP            R0,R1         ;COMPARE TO MAXIMUM ALLOWED
2791          BLOS          5$           ;BRANCH IF OK
2792          PRINTF        #MENERR        ;DISPLAY ERROR MESSAGE
2793          MOV            #MENERR,-(SP)
2794          MOV            #1,-(SP)
2795          MOV            SP,R0
2796          TRAP          C$PNTF
2797          ADD            #4,SP
2798          BR            1$           ;RETRY
2799          5$:           RTS            PC          ;RETURN TO CALLER
2800          MENERR:      .ASCIZ        '%N%A *** Menu Selection Too Large ***'
2801          SELASC:      .ASCIZ        '%N%T'
2802          MENASC:      .ASCIZ        'Enter Menu Selection: '
  
```

2782
2783 020466 000000

MENRES: .EVEN .WORD 0

```

2785 .SBTTL CHKMAN - CHECK MANUAL INTERVENTION LEGALITY
2786
2787
2788 :ROUTINE TO TEST FOR MANUAL INTERVENTION LEGALITY.
2789
2790 :INPUT:
2791
2792 NONE.
2793
2794 :OUTPUT:
2795
2796 CARRY 0 MANUAL INTERVENTION NOT ALLOWED
2797 1 MANUAL INTERVENTION IS OK
2798
2799 :SIDE EFFECTS:
2800
2801 A MESSAGE IS DISPLAYED WARNING THAT TEST IS
2802 NOT EXECUTED IF MANUAL INTERVENTION IS NOT
2803 ALLOWED.
2804
2805 :-
2806
2807 020470 CHKMAN:: SAVREG ;SAVE THE REGISTERS
2808 020470 MANUAL ;SEE IF MANUAL INTERVENTION OK
2809 020474 104450 TRAP CSMANI
2810 020476 BCOMPLETE 1$ ;BRANCH IF ALLOWED
2811 020476 103411 BCS 1$
2812 020500 PRINTF #NOMAN ;PRINT THE WARNING MESSAGE
2813 020500 012746 020524 MOV #NOMAN,-(SP)
2814 020504 012746 000001 MOV #1,-(SP)
2815 020510 010600 MOV SP,R0
2816 020512 104417 TRAP C$PNTF
2817 020514 062706 000004 ADD #4,SP
2818 020520 000241 CLC ;CLEAR CARRY FOR ERROR
2819 020522 000207 RTS PC ;RETURN
2820
2821 1$:
2822
2823 045 116 045 NOMAN: .ASCIZ '%NZA *** Manual Intervention not Allowed - Test Aborted ***'
2824 .even
  
```

```

2818                                     .SBTTL ENVIRN - SETUP FREE DIAGNOSTIC SPACE
2819                                     :
2820                                     : SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
2821                                     :
2822 020620 ENVIRN: MEMORY R0
      020620 104431 TRAP C$MEM
2823 020622 010037 003120 MOV R0,FREE ; GET 1ST FREE ADDRESS...
2824 020526 062737 000002 003120 ADD #2,FREE
2825 020634 011037 003122 MOV (R0),FRESIZ ;...AND WORD COUNT.
2826 020640 162737 000004 003122 SUB #4,FRESIZ
2827 020646 013702 002012 MOV L$UNIT,R2 ; GET NUMBER OF UNITS
2828 020652 162737 000007 003122 10$: SUB #7,FRESIZ ; TAKE AWAY 7 WORDS PER UNIT
2829 020660 005302 DEC R2
2830 020662 001373 BNE 10$
2831 020664 013700 003120 MOV FREE,R0 ;GET FIRST FREE ADDRESS
2832 020670 063700 003122 ADD FRESIZ,R0 ;POINT TO LAST FREE ADDRESS
2833 020674 162700 000002 SUB #2,R0 ;BACKUP 1 WORD
2834 020700 010037 003124 MOV R0,FREEHI ;STORE LAST FREE ADDRESS
2835 020704 000240 NOP ;*****
2836 020706 012701 177520 MOV #BDVPCR,R1 ;GET BDV11 PCR ADDRESS
2837 020712 010102 MOV R1,R2 ;COPY TO R2
2838 020714 062702 000002 ADD #2,R2 ;SET THE RANGE
2839 020720 004737 016366 JSR PC,XNXM ;SEE IF WE HAVE ONE
2840 020724 103001 BCC 15$ ;OK TO SET FLAGS
2841 020726 000445 BR 40$ ;RETURN WITH FLAGS CLEAR
2842 020730 013701 177520 15$: MOV BDVPCR,R1 ;SAVE PCR CONTENTS
2843 020734 062701 000001 ADD #1,R1 ;ADD ONE TO IT
2844 020740 012702 177520 MOV #BDVPCR,R2 ;GET BDV11 PCR ADDRESS
2845 020744 005212 INC (R2) ;TRY TO WRITE TO IT
2846 020746 013703 177520 MOV BDVPCR,R3 ;GET RESULTS
2847 020752 020103 CMP R1,R3 ;DID IT CHANGE?
2848 020754 001017 BNE 20$ ;NO, MUST BE 11/23B
2849 020756 005237 003140 INC T23A ;SET THE FLAG
2850 020762 042737 170000 002120 BIC #170000,L$HIME ;SUPERVISOR COULD BE WRONG
2851 020770 000240 NOP ;BR 40$ FOR RELEASE
2852 020772 PRINTF #M8186 ;TELL THE SYSTEM TYPE
      020772 012746 005550 MOV #M8186,-(SP)
      020776 012746 000001 MOV #1,-(SP)
      021002 010600 MOV SP,R0
      021004 104417 TRAP C$PNTF
2853 021006 062706 000004 ADD #4,SP
2854 021012 000413 BR 40$ ;RETURN
2855 021014 005237 003142 20$: INC T23B ;SET THE FLAG
2856 021022 PRINTF #M8189 ;TELL THE SYSTEM TYPE
      021022 012746 005641 MOV #M8189,-(SP)
      021026 012746 000001 MOV #1,-(SP)
      021032 010600 MOV SP,R0
      021034 104417 TRAP C$PNTF
2857 021036 062706 000004 ADD #4,SP
2858 021042 000207 40$: RTS PC ;RETURN
  
```

```

2860                                     .SBTTL  KTINIT  -  SETUP  K111  MEMORY  MANAGEMENT  REGISTERS
2861                                     :+
2862                                     :ROUTINE  TO  INIT  KT-11
2863                                     :-
2864
2865
2866
2867 021044      KTINIT:
2868 021044      005037  003126      CLR      KTF LG      ;  INIT  >28K  MEMORY  FLAG
2869 021050      005037  003130      CLR      KTENABLE  ;  INIT  TEST  >28K  FLAG
2870 021054      023727  002120  001577  CMP      LSHIME,#1577 ;  GOT  ENOUGH  MEMORY  (>28K)?
2871 021062      101444      BLOS     9$          ;  NO.
2872 021064      013700  000004      MOV      @#ERRVEC,R0 ;  SAVE  OLD  ERR  VEC  PTR.
2873 021070      012737  021162  000004  MOV      #2$,@#ERRVEC ;  SET  ERR  VEC  PTR.
2874 021076      005737  177572      TST     @#SRO       ;  GOT  KT11?
2875 021102      000240      NOP          ;  (TRAP  IF  NO).
2876 021104      013737  002120  003126  MOV      LSHIME,KTF LG ;  YES.  SET  KT  FLAG.
2877 021112      042737  000177  003126  BIC     #177,KTF LG
2878 021120      010037  000004      MOV      R0,@#ERRVEC ;  RESTORE  OLD  ERR  VEC  PTR.
2879 021124      005000      CLR      R0          ;  R0  =  AR  DATA.
2880 021126      012701  172340      MOV      #KIPAR0,R1  ;  R1  =  KI  REGS  PTR.
2881 021132      012761  077406  177740  1$:  MOV      #77406,-40(R1) ;  SET  DESCRIPTOR  REG.
2882 021140      010021      MOV      R0,(R1)+   ;  SET  KIPAR  REG.
2883 021142      062700  000200      ADD     #200,R0     ;  BUMP  AR  DATA  BY  '4k'.
2884 021146      020027  002000      CMP     R0,#2000   ;  AT  'I/O'?
2885 021152      001367      BNE     1$          ;  NO.
2886 021154      012741  177600      MOV      #177600,-(R1) ;  YES.  SET  KTPAR7  FOR  I/O.
2887 021160      000405      BR      9$
2888
2889 021162      012716  021170      2$:  MOV      #6$,(SP)   ;  SET  L/P  RETURN
2890 021166      000002      RTI          ;  RTI  TO  NEXT  LOCATION
2891
2892 021170      010037  000004      6$:  MOV      R0,@#ERRVEC ;  RESTORE  OLD  ERR  VEC  PTR.
2893
2894 021174      000207      9$:  RTS     PC
2895
    
```

```

2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910 021176
2911
2912 021176 005737 002224
2913 021202 001020
2914 021204 012737 100206 021250
2915 021212 012737 021260 021252
2916 021220 012737 000006 021256
2917 021226 012737 100010 021260
2918 021234 012704 021250
2919 021240 004737 010652
2920 021244 000207
2921
2922
2923
2924
2925
2926
2927 021250 000000
2928 021252 000000
2929 021254 000000
2930 021256 000000
2931
2932
2933
2934
2935 021260 000000
2936 021262 000000
2937 021264 000000
2938
2939
    
```

```

: +
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
    
```



```

2941          :+
2942          :          SUBROUTINE TO CHECK WETHER OR NOT WE'LL TEST NXM
2943          :
2944          :
2945          : INPUTS:
2946          : OUTPUTS:
2947          :          The NXMFLG is set if we can test.
2948          :          The NXMLO and NXMHI addresses are setup.
2949          :
2950          :
2951 021266     MEMCK::
2952
2953 021266     SAVREG          :SAVE THE REGISTERS
2954 021272 005037 003132     CLR          NXMFLG      :CLEAR THE FLAG
2955 021276 005037 003134     CLR          NXMLO       :CLEAR THE TEST ADDRESS LO
2956 021302 005037 003136     CLR          NXMHI       :CLEAR THE TEST ADDRESS HI
2957 021306 005737 003142     TST          T23B        :IS IT A 11/23B?
2958 021312 001407           BEQ          1$          :NO
2959 021314 023727 002120 007777     CMP          L$HIME,#7777 : GREATER THAN 128K
2960 021322 103406           BLO          2$          : NO
2961 021324 004737 021442     JSR          PC,NXMTST   :SETUP THE ADDRESS
2962 021330 000427           BR           13$         :SET THE FLAG AND EXIT
2963 021332 005737 003140     TST          T23A        :IS IT A 11/23A?
2964 021336 001413           BEQ          4$          :NO
2965 021340 023727 002120 005777     CMP          L$HIME,#5777 :GREATER THAN 96K
2966 021346 101023           BHI          14$         :YES,23A/23B WITH 128K MEMORY
2967 021350 023727 002120 003777     CMP          L$HIME,#3777 :GREATER THAN 64K BUT LESS THAN 92K?
2968 021356 103403           BLO          4$          :NO, CHECK 24K
2969 021360 004737 021442     JSR          PC,NXMTST   :SETUP THE ADDRESS
2970 021364 000411           BR           13$         :SET THE FLAG AND EXIT
2971 021366 023727 002120 001577     CMP          L$HIME,#1577 :GREATER THAN 24K BUT LESS THAN 64K?
2972 021374 103410           BLO          14$         :NO, TELL THEM AND EXIT WITH FLAG CLEAR
2973 021376 004737 021442     JSR          PC,NXMTST   :SETUP THE ADDRESS
2974 021402 062737 000077 003136     ADD          #77,NXMHI   :FOOL THE 11/02 & 11/03
2975 021410 005237 003136     INC          NXMFLG      :SET THE FLAG
2976 021414 000411           BR           15$         :EXIT
2977 021416 000410           BR           14$         :NOP FOR PRINTOUT
2978 021420     PRINTF          #NOMEM :TELL THEM & EXIT ***NO PRINT*****
      021420 012746 005454     MOV          #NOMEM,-(SP)
      021424 012746 000001     MOV          #1,-(SP)
      021430 010600           MOV          SP,R0
      021432 104417           TRAP         C$PNTF
      021434 062706 000004     ADD          #4,SP
2979 021440 000207           RTS          PC          :RETURN
2980
2981
2982          :+
2983          :          SUBROUTINE TO SETUP THE NXM ADDRESS FOR TESTING
2984          :
2985          : OUTPUTS:NXMLO,NXMHI          :SETUP WITH NXM ADDRESS
2986          :
2987          :-
2988
2989 021442 013701 002120     NXMTST: MOV          L$HIME,R1   :GET TOP OF MEMORY
2990 021446 062701 000200     ADD          #200,R1     :MAKE IT I/O BLOCK OR OTHER NXM
2991 021452 042701 000177     BIC          #177,R1
2992 021456 010102           MOV          R1,R2      :RESAVE RESULTS

```

```
2993          000006          .REPT      6
2994          .ASL          R1          ;PUT IN PLACE FOR XFER
2995          .ENDR
2996 021474 010137 003134    MOV      R1,NXMLO          ;SAVE TEST ADDRESS LOW
2997          000012          .REPT     10
2998          .ASR          R2          ;PUT IN PLACE FOR XFER
2999          .ENDR
3000 021524 042702 1777C0    BIC      #177700,R2        ;DON'T WANT ILA!
3001 021530 010237 003136    MOV      R2,NXMHI          ;SAVE TEST ADDRESS HIGH
3002 021534 000207          RTS      PC                ;RETURN
3003
3004
3005
3006
3007 021536          ENDMOD
```

7
8
9 021536
021536
10
16

.TITLE TSV4 - MISCELLANEOUS SECTIONS
BGNMOD TSV4
TSV4::

```
18  
19 021536                            .SBTTL PROTECTION TABLE  
   021536                            BGNPROT  
20 021536 177777 177777 177777    L$PROT::  
21 021546                            .WORD    -1, -1, -1, -1  
22                                    ENDPROT
```

;NO DEVICE PROTECTION REQUIRED.

```

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                                     :IF "START" OR "RESTART", SET QUICK-PASS FLAG AND BUS-INIT.
31                                     :IF "CONTINUE", NOTHING IS REQUIRED.
32
33                                     :--
34                                     :+
35                                     :INSERT TEMPORARY JUMP TO ODT
36                                     :--
37 021546                                BGNINIT
    021546                                L$INIT::
38 021546 005037 002224                                40$: CLR     EXTFEA
39 021552 005037 003132                                CLR     NXMFLG
40 021556 012737 006354 002176                    MOV     #EPT1,EPTSW ;SET UP PRIMARY MESSAGE FOR REPLACEMENT
41 021564 005037 003150                                CLR     SIFLAG      ;CLEAR "SOFT INIT" FLAG
42 021570 005037 003130                                CLR     KTENABLE    ;CLEAR TEST ABOVE 28K FLAG
43 021574 005037 002300                                CLR     RAMSIZ      ;CLEAR RAM SIZE FOR RAMERR ROUTINE
44 021600                                READEF  #EF.CONTINUE
    021600 012700 000036                                MOV     #EF.CONTINUE,R0
    021604 104447                                TRAP    C$REFG
45 021606                                BNCOMPLETE 1$
    021606 103023                                BCC     1$
46 021610 023737 002200 002012                    CMP     UNITN,L$UNIT ;UNIT IN RANGE?
47 021616 103070                                BHIS    4$           ;BR IF NO.
48 021620 005737 003106                                TST    DUFLG        ;DROPPED UNIT?
49 021624 100472                                BMI     NXTU         ;BR IF YES
50 021626 013701 002200                                MOV     UNITN,R1
51 021632 006301                                ASL    R1
52 021634 005761 003172                                TST    ERTABL(R1)
53 021640 001516                                BEQ    SETU
54 021642 032761 040000 003172                    BIT     #BIT14,ERTABL(R1) ;DROPPED?
55 021650 001060                                BNE    NXTU
56 021652                                EXIT    INIT        ;DO NOTHING IF "CONTINUE".
    021652 104432                                TRAP    C$EXIT
    021654 000416                                .WORD  L10030-.
57 021656                                1$: READEF  #EF.NEW
    021656 012700 000035                                MOV     #EF.NEW,R0
    021662 104447                                TRAP    C$REFG
58 021664                                BNCOMPLETE NXTU    ;TAKE NEXT UNIT IF NOT NEW PASS.
    021664 103052                                BCC    NXTU
59 021666                                READEF  #EF.START
    021666 012700 000040                                MOV     #EF.START,R0
    021672 104447                                TRAP    C$REFG
60 021674                                BCOMPLETE 2$
    021674 103404                                BCS    2$
61 021676                                READEF  #EF.RESTART
    021676 012700 000037                                MOV     #EF.RESTART,R0
    021702 104447                                TRAP    C$REFG
62 021704                                BNCOMPLETE 31$
    021704 103031                                BCC    31$
63 021706                                2$: BRESET
64 021706                                TRAP    C$RESET    ;1ST PASS, BUS-INIT...
    021706 104433                                ;BUS RESET.
    
```

65	021710	005037	002212		CLR	TSTCNT		:NUMBER OF TESTS RUN IN PASS
66	021714	005037	002220		CLR	FATFLG		:CLEAR FATAL ERROR COUNT
67	021720	005037	003140		CLR	T23A		:CLEAR 11/23A FLAG
68	021724	005037	003142		CLR	T23B		:CLEAR 11/23B FLAG
69					MOV	#340,-(SP)		
70					MOV	#20,-(SP)		:RETURN TO DEBUGGER
71					JMP	O.ODT	::@ENTER THE DEBUGGER	
72	021730	005037	003374		CLR	SKIPT		:CLEAR THE SUBTEST "SKIPPER"
73	021734			20\$:				
74	021734	012737	177777	002202	MOV	#-1,QVP		:...QUICK VERIFY...
75	021742	004737	020620		JSR	PC,ENVIRN		:SET ENVIRONMENT.
76	021746	004737	021044		JSR	PC,KTINIT		:INITIALIZE KT MEMORY MANAGEMENT
77	021752	012700	003172		MOV	#ERTABL,RO		
78	021756	005020		30\$:	CLR	(RO)+		:CLEAR THE ERROR TABLE
79	021760	020027	003372		CMP	RO,#ERTABE		
80	021764	103774			BLO	30\$		
81	021766	000404			BR	4\$		
82	021770	005037	002202	31\$:	CLR	QVP		
83	021774	000137	022044		JMP	PASRPT		:GO REPORT THE STATUS
84								
85	022000			4\$:				
86	022000	012737	177777	002200	NEWPAS:	MOV #-1,UNITN		:INIT UNIT NUMBER...
87	022006	005037	002216		CLR	DEV CNT		:CLEAR COUNT OF DEVICES RUNNING
88	022012			NXTU:	BREAK			
89	022014	104422			TRAP	C\$BRK		
90	022020	005237	002200		INC	UNITN		:...AND SET NEXT UNIT NUMBER.
91	022026	023737	002200	002012	CMP	UNITN,LSUNIT		
92	022030	103423			BLO	SETU		
93	022036	012737	177777	003106	MOV	#-1,DUFLG		
94	022040	000401			BR	11\$		
95	022042	104444			DOCLN			:ABORT, NO MORE UNITS.
96	022044	000240		11\$:	TRAP	C\$DCLN		
97	022044	023727	002012	000001	PASRPT:	NOP		
98	022052	101752			CMP	LSUNIT,#1		:HOW MANY UNITS SELECTED?
99	022054	005737	002216		BLOS	NEWPAS		:BR IF ONLY 1
100	022060	001747			TST	DEV CNT		:ARE ANY STILL RUNNING?
101	022062				BEQ	NEWPAS		:BR IF NO
102	022064	104421			RFLAGS	RO		
103	022070	032700	000100		TRAP	C\$RFLA		
104	022070	001343			BIT	#ISR,RO		:SHOULD WE PRINT STATISTICS
105	022072				BNE	NEWPAS		:BR IF NO
106	022074	104424			DORPT			
107	022076	000741			TRAP	C\$DRPT		
108	022076			10\$:	BR	NEWPAS		
109	022076			SETU:				
110	022104	013700	002200		GPHARD	UNITN,RO		:GET UNIT N P-TABLE POINTER.
111	022106	104442			MOV	UNITN,RO		
112	022112	005037	003106		TRAP	C\$GPHRD		
113	022116	005237	002216		BNCOMPLETE	NXTU		:BR IF UNIT NOT AVAILABLE.
114	022120	012001	002204		BCC	NXTU		
		010137			CLR	DUFLG		:CLEAR 'DROPPED' FLAG.
					INC	DEV CNT		
					MOV	(RO)+,R1		:GET 1ST REGISTER ADDRESS.
					MOV	R1,CSRADDR		:ADDRESS OF REGISTERS OF UNIT UNDER TEST

```

115
116 022124 012001          MOV      (R0)+,R1          ;GET VECTOR ADDRESS.
117                      ;MOV      (R0),R2          ;GET INTERRUPT PRIORITY
118                      ;MOV      R2,IPRI        ;SET INTERRUPT PRIORITY.
119 022126 010137 002206  MOV      R1,IVEC        ;SET INTERRUPT VECTOR POINTER...
120 022132 012721 016206  MOV      #INTR,(R1)+    ;...VECTOR...
121 022136 013721 002210  MOV      IPRI,(R1)+    ;...AND PRIORITY.
122
123 022142                1$:
124                      ;          TST      QVP          ;1ST PASS ??
125                      ;          BEQ      5$          ;NO, SKIP THE PASS 1 STUFF.
126
127
128                      ;1ST PASS, CHECK THAT DEVICE ADDRESSES ARE VALID, AND
129                      ;THAT THE DISPLAY STATUS IS PROPERLY INITIALIZED.
130
131 022142 013701 002200          MOV      UNITN,R1
132 022146 006301          ASL      R1
133 022150 052761 100000 003172  BIS      #BIT15,ERTABL(R1) ;SAY DEVICE RUNNING
134 022156 005037 005766          CLR      EXTA          ;CLEAR ERROR EXTENSION FLAG.
135 022162 023727 002012 000001  CMP      LSUNIT,#1     ;ARE WE TESTING MULTIPLE UNITS?
136 022170 101416          BLOS    10$          ;BR IF NO.
137 022172          RFLAGS  RO          ;YES -- GET OPERATOR FLAGS.
      022172 104421          TRAP    CSRFLA
138 022174 032700 001000          BIT      #PNT,RO      ;SHOULD WE PRINT UNIT #?
139 022200 001412          BEQ     10$          ;BR IF NOT.
140 022202          PRINTF #PUNIT,UNITN ;PRINT THE UNIT #
      022202 013746 002200          MOV      UNITN,-(SP)
      022206 012746 022274          MOV      #PUNIT,-(SP)
      022212 012746 000002          MOV      #2,-(SP)
      022216 010600          MOV      SP,RO
      022220 104417          TRAP    C$PNTF
      022222 062706 000006          ADD     #6,SP
141 022226                10$:
142 022226 005037 003110          CLR      NODEV
143 022232 013701 002204          MOV      CSRADDR,R1 ;ADDRESS OF FIRST REGISTER
144 022236 010102          MOV      R1,R2      ;START OF REGISTERS
145 022240 062702 000002          ADD     #TSSR,R2   ;ADDRESS OF TSSR REGISTER
146 022244 004737 016366          JSR     PC,XNXM    ;TEST BOTH CONTROLLER REGISTERS...
147 022250 103005          BCC     2$          ;...AND BR IF ALL OK.
148 022252 010137 003110          MOV      R1,NODEV  ;FLAG DEVICE AS NON-EXISTENT
149 022256 012737 177777 003106  MOV      #-1,DUFLG ;DROP THIS UNIT.
150 022264                2$:
151
152                      ;FINALLY, SET CPU PRIORITY AND WE'RE DONE.
153
154 022264                5$:
      022264 012700 000000          SETPRI #PRI00        ;ENABLE INTERRUPTS.
      022270 104441          MOV      #PRI00,RO
155 022272          TRAP    C$SPRI
      022272          ENDINIT
156 022272 104411                L10030:
      022272          TRAP    C$INIT
157 022274 045 116 045 PUNIT: .ASCIZ /%N%N%A***** TESTING UNIT %D2%A *****/
158                      .EVEN
    
```

160
161
162
163
164
165
166
167 022342
022342
168 022342 010001
169 022344 006301
170 022346 052761 100000 003172
171 022354 042761 040000 003172
172 022362
022362 010046
022364 012746 022410
022370 012746 000002
022374 010600
022376 104417
022400 062706 000006
173 022404
022404 000167
022406 000026
174 022410 045 116 045 1\$:
175
176
177 022436
022436
022436 104452
178
179
180
181
182
183
184
185
186
187
188
189 022440
022440
190 022440 012737 177777 003106
191 022446 010001
192 022450 006301
193 022452 052761 140000 003172
194 022460 000240 000240 000240
195 022466
022466 010046
022470 012746 022514
022474 012746 000002
022500 010600
022502 104417
022504 062706 000006
196 022510
022510 000167
022512 000030

.SBTTL ADD AND DROP UNITS SECTIONS

;++
: THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
: TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
: OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
:--

```
LSAU:: BGNU
MOV RO,R1 ; GET UNIT TO BE ADDED (RO)
ASL R1 ; MAKE IT A WORD INDEX
BIS #100000,ERTABL(R1) ; SET THE "ACTIVE" BIT
BIC #40000,ERTABL(R1) ; CLEAR THE "DROPPED" BIT
PRINTF #1$,RO
MOV RO,-(SP)
MOV #1$,-(SP)
MOV #2,-(SP)
MOV SP,RO
TRAP C$PNTF
ADD #6,SP
EXIT AU
.WORD JSJMP
.WORD L10031-2-.
.ASCIZ /%N% UNIT %D% ADDED/
.EVEN
```

ENDAU ; UNUSED.

```
L10031: TRAP CSAU
```

;++
: THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
: TO BE REMOVED FROM THE TEST LIST.

: SUPVSR DOES THE "DROPPING". THIS IS JUST TO TELL THE MAN.
: "DROPPED" UNITS ARE RE-SELECTED ON OPERATOR "STA" OR "ADD"
: COMMAND, OTHERWISE REMAIN INACTIVE. THE "DISPLAY" COMMAND
: WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
: WHICH ARE STILL ACTIVE.
: UPON ENTRY, RO CONTAINS THE UNIT TO BE DROPPED.

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



```

197 022514    045    116    045 1$:    .ASCIZ /%N% UNIT %D% DROPPED/
198                                    .EVEN
199 022544                            ENDDU
     022544                            L10032:
     022544    104453                    TRAP    C$DU
200                                    :++
201                                    : AUTO-DROP CODE SECTION.
202                                    :--
203 022546                            BGAUTO
     022546                            L$AUTO::
204 022546    013705    002204            MOV    CSRADDR,R5                    :POINT TO DEVICE REGISTER
205 022552    012703    000550            MOV    #360.,R3                    :ENOUGH TIME FOR 2400' REEL TO REWIND
206 022556    004737    016240            10$: JSR    PC,WAITF                    :WAIT FOR SSR TO SET
207 022562    103420                    BCS    20$                        :LEAVE WHEN SSR IS SET
208 022564                            DELAY 250.                        :WAIT FOR .25 SECONDS
     022564    012727    000372            MOV    #250.,(PC)+
     022570    000000                    .WORD 0
     022572    013727    002116            MOV    L$DLY,(PC)+
     022576    000000                    .WORD 0
     022600    005367    177772            DEC    -6(PC)
     022604    001375                    BNE    .-4
     022606    005367    177756            DEC    -22(PC)
     022612    001367                    BNE    .-20
209 022614    005303                    DEC    R3                        :BUMP COUNTER DOWN
210 022616    001357                    BNE    10$                        :KEEP GOING
211 022620    004737    017172            JSR    PC,CKDROP                    :TRY AND DROP UNIT
212 022624                            20$:
213 022624                            ENDAUTO                            ; UNUSED.
     022624                            L10033:
     022624    104461                    TRAP    C$AUTO
  
```

```

215                                     .SBTTL  CLEAN-UP AND REPORT CODING SECTIONS
216
217
218                                     :++
219                                     : THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS
220                                     : EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).
221                                     : USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.
222                                     :--
222 022626                                     BGNCLN
223 022626 L$CLEAN::
224 022626 013705 002204                       MOV     CSRADDR,R5           ;POINT TO DEVICE REGISTER
225 022632 005737 003106                       TST     DUFLG                ;'DROPPED' FLAG IS SET ON...
226 022636 100405                               BMI     1$                   ;...AND GROSS CONTROLLER FAULT...
227                                     ;...DON'T TRY TO XCT CLEANUP CODE.
228 022640 012765 000000 000002               MOV     #0,TSSR(R5)         ;DO SOFT INIT
229 022646 004737 016240                       JSR     PC,WAITF
230 022652                                     1$:
231 022652                                     2$:
232 022652 L10034:                               ENDCLN
233 022652 104412                               TRAP    C$CLEAN
234
235                                     :++
236                                     : THE REPORT CODING SECTION CONTAINS THE
237                                     : 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
238                                     :--
239                                     BGNRPT
240 L$RPT::
241 PRINTS  #DEVSUM
242 MOV     #DEVSUM,-(SP)
243 MOV     #1,-(SP)
244 MOV     SP,R0
245 TRAP    C$PNTS
246 ADD     #4,SP
247 MOV     R2,-(SP)
248 MOV     R3,-(SP)
249 MOV     R4,-(SP)
250 MOV     #ERTABL,R4           ; GET START OF ERROR TABLE.
251 CLR     R3                   ; CLEAR UNIT NUMBER
252 1$: MOV     (R4),R2           ; GET ERROR TABLE ENTRY & TEST IT.
253 BEQ     4$                   ; ZERO IF UNIT NOT RUN
254 BPL     4$
255 BIT     #BIT14,R2           ; WAS UNIT DROPPED?
256 BNE     2$                   ; BR IF YES
257 BIC     #^C7777,R2         ; GET ERROR COUNT FIELD
258 PRINTS  #DEVONL,R3,R2       ; PRINT
259 MOV     R2,-(SP)
260 MOV     R3,-(SP)
261 MOV     #DEVONL,-(SP)
262 MOV     #3,-(SP)
263 MOV     SP,R0
264 TRAP    C$PNTS
265 ADD     #10,SP
266 BR     4$
267 2$: CMP     R2,#160000       ; WAS UNIT NON-EXISTENT?
268 BNE     3$                   ; BR IF NO
269 PRINTS  #DEVNXR,R3
270 MOV     R3,-(SP)
271 MOV     #DEVNXR,-(SP)
    
```

```

022772 012746 000002      MOV      #2,-(SP)
022776 010600      MOV      SP,R0
023000 104416      TRAP     C$PNTS
254 023002 062706 000006      ADD      #6,SP
255 023006 000431      BR       4$
255 023010 020227 160001      3$:     CMP      R2,#160001      ; WAS UNIT NOT READY AT STARTUP?
256 023014 001012      BNE     30$      ; BR IF NO.
257 023016      PRINTS  #DEVNRD,R3
023016 010346      MOV      R3,-(SP)
023020 012746 023305      MOV      #DEVNR,-(SP)
023024 012746 000002      MOV      #2,-(SP)
023030 010600      MOV      SP,R0
023032 104416      TRAP     C$PNTS
023034 062706 000006      ADD      #6,SP
258 023040 000414      BR       4$
259 023042 042702 170000      30$:    BIC      #^C7777,R2
260 023046      PRINTS  #DEVDRD,R3,R2
023046 010246      MOV      R2,-(SP)
023050 010346      MOV      R3,-(SP)
023052 012746 023366      MOV      #DEVDRD,-(SP)
023056 012746 000003      MOV      #3,-(SP)
023062 010600      MOV      SP,R0
023064 104416      TRAP     C$PNTS
261 023066 062706 000010      ADD      #10,SP
262 023072 062704 000002      4$:     ADD      #2,R4
263 023076 005203      INC      R3
264 023100 020427 003372      CMP      R4,#ERTABE
265 023104 103701      BLO     1$
266 023106 012604      MOV      (SP)+,R4
267 023110 012603      MOV      (SP)+,R3
268 023112 012602      MOV      (SP)+,R2
268 023114      ENDRPT      ; UNUSED.
268 023114      L10035:
023114 104425      TRAP     C$RPT
269
270
271 023116      045      116      045  DEVSUM: .ASCIZ /%N%ADEVICE STATUS SUMMARY:%N/
272 023153      045      101      040  DEVONL: .ASCIZ /%A UNIT %D3%A ONLINE, ERRORS = %D%N/
273 023223      045      101      040  DEVNXR: .ASCIZ /%A UNIT %D3%A DROPPED, NON-EXISTENT REGISTER%N/
274 023305      045      101      040  DEVNRD: .ASCIZ /%A UNIT %D3%A DROPPED, NOT READY AT STARTUP%N/
275 023366      045      101      040  DEVDRD: .ASCIZ /%A UNIT %D3%A DROPPED, ERRORS = %D%N/
276      .EVEN
277
278 023436      ENDMOD
279
280

```

1
2
9
10
16
24

.TITLE TSV5A - HARDWARE TESTS

TSV5:: BGNMOD TSV5

023436
023436

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
64
65
70
71
72
73
74
75
76
77
78
79
80
81
82

.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.

```

65 023436          BGNTST
    023436
70 023436 012700 023634      MOV   #TST1ID,R0      ;ASCII MESSAGE TO IDENTIFY TEST
71 023442 004737 016500      JSR   PC,TSTSETUP    ;DO INITIAL TEST SETUP
72 023446 012737 000024 002214  MOV   #20.,LOOPCNT  ;PERFORM 20 ITERATIONS
73 023454          T1LOOP:
74 023454 005003          CLR   R3              ;USE R3 AS FATAL ERROR FLAG
75
76 023456          BGNSUB      ;//////////////// BEGIN SUBTEST //////////////////
    023456
    023456 104402          T1.1:
77                                     TRAP   C$SUB
78 023460          BRESET      ;ISSUE A BUS RESET
    023460 104433          TRAP   C$RESET
79 023462 004737 016240      JSR   PC,WAITF      ;WAIT FOR READY
80 023466 016501 000002      MOV   TSSR(R5),R1  ;GET THE CONTENTS OF TSSR
81 023472 010102          MOV   R1,R2         ;CONTENTS OF TSSR
82 023474 042702 176277      BIC   #^C<HIADDR!OFL>,R2 ;THESE BITS MAY BE SET
    
```

```
83 023500 052702 002200      BIS      #SSR!NBA,R2      ;READY AND NEW DATA SHOULD BE SET
84 023504 020102              CMP      R1,R2          ;COMPARE EXPECTED TO RECEIVED
85 023506 001405              BEQ      10$            ;BRANCH IF COMPARE
89 023510              ERRDF  ERRNO,SFHERR,SFFMSG ;REPORT A FATAL ERROR
    023510 104455              TRAP      C$ERDF
    023512 000145              .WORD    101
    023514 003701              .WORD    SFHERR
    023516 012072              .WORD    SFFMSG
90 023520 005203              INC      R3              ;SET THE FATAL ERROR FLAG
91 023522              10$:  ENDSUB
92 023522              ;////////// END SUBTEST ////////////
    023522              L10037:
93 023522 104403              TRAP      C$ESUB
```

```

95 023524 005703          TST      R3          ;DID WE HAVE FATAL ERROR ?
96 023526 001402          BEQ      20$         ;BRANCH IF NOT
97 023530 004737 017172   JSR      PC,CKDROP  ;GO DROP THIS UNIT, IF ALLOWED
98 023534 005003          CLR      R3          ;RESET FATAL ERROR FLAG
99
100
101 023536          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
    023536          T1.2:          TRAP      C$BSUB
    023536 104402
102
103 023540 005065 000002   CLR      TSSR(R5)   ;WRITE TO ISSUE A SOFT RESET
104 023544 004737 016240   JSR      PC,WAITF   ;WAIT FOR READY TO SET
105 023550 016501 000002   MOV      TSSR(R5),R1 ;GET REGISTER TSSR DATA
106 023554 010102          MOV      R1,R2      ;CONTENTS OF TSSR
107 023556 042702 176277   BIC      #^C<HIADDR!OFL>,R2 ;THESE BITS MAY BE SET
108 023562 052702 002200   BIS      #SSR!NBA,R2 ;READY AND NEW DATA SHOULD BE SET
109 023566 020102          CMP      R1,R2      ;COMPARE EXPECTED TO RECEIVED
110 023570 001405          BEQ      10$        ;BRANCH IF COMPARE
114 023572          ERRDF  ERRNO,SFIERR,SFFMSG ;REPORT A FATAL ERROR
    023572 104455          TRAP      C$ERDF
    023574 000146          .WORD    102
    023576 003646          .WORD    SFIERR
    023600 012072          .WORD    SFFMSG
115 023602 005203          INC      R3          ;SET THE ERROR FLAG
116 023604          10$:          ENDSUB          ;//////////////// END SUBTEST //////////////////
117 023604          L10040:        TRAP      C$ESUB
    023604 104403
118
119
120 023606 005703          TST      R3          ;FATAL ERROR DETECTED ?
121 023610 001402          BEQ      20$         ;BRANCH IF NOT
122 023612 004737 017172   JSR      PC,CKDROP  ;SEE IF TIME TO DROP UNIT
123 023616 004737 016446   JSR      PC,TSTLOOP ;SHOULD WE DO ITERATIONS ?
124 023622 103002          BCC      40$        ;BRANCH IF NOT
125 023624 000137 023454   JMP      T1LOOP     ;LOOP UNTIL COUNT EXPIRED
126 023630          40$:          TST
    023630 104432          TRAP      C$EXIT
    023632 000022          .WORD    L10036-.
127
128          ;+
129          ;LOCAL TEXT MESSAGES FOR TEST
130          ;-
131
132 023634          111      156      151  TST1ID: .ASCIZ 'Initialization'
133          .EVEN
134 023654          ENDTST
    023654          L10036:        TRAP      C$ETST
    023654 104401
135
136

```

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
169
170
171
172
173
178
179
180
181
182
183
184
185
186
187
191
192
193
194
195
196

023656
023656
023656 012700 024324
023662 004737 016500
023666 012737 000024 002214
023674 005004
023676 012703 177777
023702 005703
023704 001412
023706 005003
023710 004737 015764
023714 103406
023716 010001
023720
023720 104455
023722 000311
023724 003646
023726 012024
023730 005203
023732 005037 002220
023736

.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).

BGNTST

```
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  
:BRANCH IF NOT  
:DON'T NEED INIT NEXT TIME THRU  
:DO SOFT INIT OF CONTROLLER  
:BR IF SOFT INIT = OK  
:SAVE CONTENTS OF TSSR  
:DEVICE FATAL ERROR DURING INIT  
TRAP C$ERDF  
.WORD 201  
.WORD SFIERR  
.WORD SFIMSG  
10$: INC R3 :FORCE SOFT INIT ON NEXT PASS  
CLR FATFLG :CLEAR FATAL ERROR FLAG  
BGNSEG :>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>>>>>
```



```
246 024122 004737 016446      JSR      PC,TSTLOOP      ;SHOULD WE DO ITERATIONS ?
247 024126 103002              BCC      40$             ;BRANCH IF NOT
248 024130 000137 023674      JMP      T2LOOP         ;LOOP UNTIL COUNT EXPIRED
249 024134              40$:  EXIT      TST      ;ALL DONE THIS TEST
    024134 104432
    024136 000214              TRAP    C$EXIT
                                .WORD    L10041-.

250
251
252          ;+
253          ;LOCAL TEXT MESSAGES FOR TEST
254          ;-
255 024140          124      123      102  T2TSBA: .ASCIZ  'TSBA Incorrect After TSDB High Write'
256 024205          124      123      123  T2TSSR: .ASCIZ  'TSSR Incorrect After TSDB High Write'
257 024252          116      157      040  T2SSR:  .ASCIZ  'No Sub-System Ready After TSDB High Write'
258 024324          127      162      141  TST2ID: .ASCIZ  'Wrap Data - High Byte'
259              .EVEN
260 024352              ENDTST
    024352
    024352 104401              L10041: TRAP    C$TST
261
262
```



```
370 ;LOCAL TEXT MESSAGES FOR TEST
371 :-
372
373 024636 124 123 102 T3TSBA: .ASCIZ 'TSBA Incorrect After TSDB Low Write'
374 024702 124 123 123 T3TSSR: .ASCIZ 'TSSR Incorrect After TSDB Low Write'
375 024746 116 157 040 T3SSR: .ASCIZ 'No Sub-System Ready After TSDB Low Write'
376 025017 127 162 141 TST3ID: .ASCIZ 'Wrap Data - Low Byte'
377 .EVEN
378 025044 .ENDTST
025044
025044 104401 L10042: TRAP CSETST
379
380
```

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
 437
 438
 439

025046
 025046
 025046
 025046 104402
 025050 012700 026356
 025054 004737 016500
 025060 012737 000005 002214

.SBTTL 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). THESE TESTS ARE PERFORMED BY THREE SUBTESTS, DESCRIBED BELOW. A 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 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.

BGNTST

T4::

BGNSUB

:/: BEGIN SUBTEST /:

T4.1:

TRAP CSBSUB

MOV #TST4ID,R0
 JSR PC,TSTSETUP
 MOV #5,LOOPCNT

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


```

559 025460          ERRHRD  ERRNO,TSBAM3,EXPREC      ;CHARACTERISTICS DATA NOT CORRECT
      025460 104456          TRAP  C$ERHRD
      025462 000627          .WORD  407
      025464 026276          .WORD  TSBAM3
      025466 015464          .WORD  EXPREC
560 025470 012702 000377      43$:  MOV  #000377,R2      ;SET ALL ONES WORD
561 025474 010465 000000      MOV  R4,TSDB(R5)      ;LOAD UP RAM ADDRESS POINTER
562 025500 004737 016326      JSR  PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
563 025504 110265 000000      MOVB R2,TSDB(R5)      ;WRITE DATA INTO RAM
564 025510 004737 016326      JSR  PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
565 025514 016501 000000      MOV  TSBA(R5),R1     ;READ RAM CONTENTS BACK
566 025520 120102          CMPB R1,R2           ;CHECK WITH DATA WRITTEN
567 025522 001404          BEQ  45$             ;BR IF OK, DATA IN = DATA OUT
571 025524          ERRHRD  ERRNO,TSBAM2,EXPREC      ;WRITTEN DATA NOT = TO READ
      025524 104456          TRAP  C$ERHRD
      025526 000630          .WORD  408
      025530 026214          .WORD  TSBAM2
      025532 015464          .WORD  EXPREC
572 025534          45$:  CKLOOP      ;SCOPE LOOP
      025534 104406          TRAP  C$CLP1
573 025536 004737 016326      JSR  PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
574 025542 010465 000000      MOV  R4,TSDB(R5)      ;WORD WRITE TO SET UP ADDRESS
575 025546 004737 016326      JSR  PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
576 025552 116501 000001      MOVB TSBAH(R5),R1     ;HIGH BYTE READ OF TSBA
577 025556 010403          MOV  R4,R3           ;DATA PATTERN WRITTEN
578 025560 000303          SWAB R3             ;HIGH TO LOW
579 025562 060403          ADD  R4,R3           ;TOTAL OF BYTES IN LOW BYTE
580 025564 120103          CMPB R1,R3           ;SUM OF BYTES WRITTEN TO TSDB = TSBAH
581 025566 001404          BEQ  50$             ;BR IF OK, THEY SHOULD BE
585 025570          ERRHRD  ERRNO,M2901,EXPREC      ;2901 PROBLEM ADDER
      025570 104456          TRAP  C$ERHRD
      025572 000631          .WORD  409
      025574 026124          .WORD  M2901
      025576 015464          .WORD  EXPREC
586 025600          50$:  CKLOOP      ;SCOPE LOOP
      025600 104406          TRAP  C$CLP1
587 025602 005304          DEC  R4              ;DROP RAM ADDRESS POINTER
588 025604 002312          BGE  40$             ;NOT AT LOC. ZERO YET
589
590 025606          ENDSUB      ;////////////////// END SUBTEST ////////////////////
      025606          L10045:
      025606 104403          TRAP  C$ESUB
591

```



```

641 025760 016501 000000      MOV      TSBA(R5),R1      ;PICK UP RAM CONTENTS
642 025764 120102             CMPB     R1,R2           ;IS MEMORY STILL ALL ONES
643 025766 001404             BEQ      43$            ;BR, IF OK (ALL ONES)
647 025770             ERRHRD   ERRNO,TSBAM3,EXPREC ;MEMORY CHANGED AFTER ALL ONES
                                TRAP      C$ERHRD
                                .WORD     412
                                .WORD     TSBAM3
                                .WORD     EXPREC
                                WRITE
648 026000 005002             43$:    CLR      R2            ;SET UP NEW EXPECTED
649 026002 010465 000000      MOV      R4,TSDB(R5)    ;LOAD UP RAM ADDRESS POINTER
650 026006 004737 016326      JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
651 026012 110265 000000      MOVB     R2,TSDB(R5)    ;WRITE DATA INTO RAM
652 026016 004737 016326      JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
653 026022 016501 000000      MOV      TSBA(R5),R1    ;READ RAM CONTENTS BACK
654 026026 120102             CMPB     R1,R2           ;CHECK WITH DATA WRITTEN
655 026030 001404             BEQ      45$            ;BR IF OK, DATA IN = DATA OUT
659 026032             ERRHRD   ERRNO,TSBAM2,EXPREC ;WRITTEN DATA NOT = TO READ
                                TRAP      C$ERHRD
                                .WORD     413
                                .WORD     TSBAM2
                                .WORD     EXPREC
660 026042             45$:    CKLOOP          ;SCOPE LOOP
                                TRAP      C$CLP1
661 026044 004737 016326      JSR      PC,CHKTSSR    ;WAIT FOR READY, NON-AMBIGUOUS
662 026050 116501 000001      MOVB     TSBAH(R5),R1  ;HIGH BYTE READ OF TSBA
663 026054 010203             MOV      R2,R3          ;DATA PATTERN WRITTEN
664 026056 000303             SWAB     R3             ;HIGH TO LOW
665 026060 060203             ADD      R2,R3          ;TOTAL OF BYTES IN LOW BYTE
666 026062 120103             CMPB     R1,R3          ;SUM OF BYTES WRITTEN TO TSDB = TSBAH
667 026064 001404             BEQ      50$            ;BR IF OK, THEY SHOULD BE
671 026066             ERRHRD   ERRNO,M2901,EXPREC ;2901 PROBLEM ADDER
                                TRAP      C$ERHRD
                                .WORD     414
                                .WORD     M2901
                                .WORD     EXPREC
672 026076             50$:    CKLOOP          ;SCOPE LOOP
                                TRAP      C$CLP1
673 026100 005304             DEC      R4             ;DROP RAM ADDRESS POINTER
674 026102 001315             BNE      40$            ;NOT AT LOC. ZERO YET
675
676 026104             ENDSUB          ;////////////////// END SUBTEST ////////////////////
                                L10046:
                                TRAP      C$ESUB
677
678 026106 004737 016446      JSR      PC,TSTLOOP    ;DO WE NEED TO ITERATE TEST ?
679 026112 103002             BCC      63$            ;BRANCH IF NOT
680 026114 000137 025066      JMP      T4LOOP        ;EXECUTE AGAIN
681 026120             63$:    EXIT      TST      ;ALL DONE THIS TEST
                                TRAP      C$EXIT
                                .WORD     L10043-.
682
683      ;+
684      ;LOCAL TEXT MESSAGES FOR TEST
685      ;-
686 026124      040      124      123      M2901: .ASCIZ ' TSBA High Byte Not Sum of Last TSDB Write (2901 Error)'
687 026214      040      127      162      TSBAM2: .ASCIZ ' Write to TSDB Not Equal to Read of TSBA Low Byte'
688 026276      127      162      151      TSBAM3: .ASCIZ 'Write To RAM Location Modified Another Location'

```

689 026356 122 101 115 TST4ID: .ASCIZ 'RAM Verification'
690 .EVEN
691 026400 .ENDTST
026400
692 026400 104401

L10043: TRAP CSETST

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

.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.

```

731 026402          BGNTST
    026402
736 026402 012700 027354          MOV    #TST5ID,RO          T5::
737 026406 004737 016500          JSR    PC,TSTSETUP      ;ASCII MESSAGE TO IDENTIFY TEST
738 026412 012737 000024 002214  MOV    #20,LOOPCNT      ;DO INITIAL TEST SETUP
739 026420          TSLOOP:    ;PERFORM 20 ITERATIONS
740 026420 005037 002220          CLR    FATFLG           ;CLEAR THE FATAL ERROR FLAG
741
742 026424          BGNSUB      ;//////////////// BEGIN SUBTEST //////////////////
    026424
    026424 104402          T5.1:
743                                     TRAP    C$BSUB
744 026426 004737 015764          JSR    PC,SOFINIT      ;DO A SOFT TO START
745 026432 103404          BCS    10$            ;BRANCH IF O.K.
749 026434          ERRDF     ERRNO,SFIERR,SFIMSG ;REPORT ERROR AND DROP DRIVE
    026434 104455          TRAP    C$ERDF
    026436 000765          .WORD  501
    026440 003646          .WORD  SFIERR
    026442 012024          .WORD  SFIMSG
750 026444 012702 177777          10$:  MOV    #-1,R2      ;ALL ONE DATA PATTERN
  
```

751	026450	005004		CLR	R4		:STARTING RAM ADDRESS	
752	026452	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
753	026456	105065	000000	15\$: CLR	TSDB(R5)		:SET MAINTENANCE MODE	
754	026462	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
755	026466	010465	000000	MOV	R4,TSDB(R5)		:SET THE NEXT RAM ADDRESS	
756	026472	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
757	026476	110265	000000	MOVB	R2,TSDB(R5)		:LOAD TEST DATA	
758	026502	005204		INC	R4		:NEXT ADDRESS TO TEST	
759	026504	020427	007777	CMP	R4,#7777		:COMPARE TO LAST ADDRESS	
760	026510	003762		BLE	15\$:BRANCH TILL ALL DATA WRITTEN	
761	026512			BRESET			:ISSUE A BUS RESET	
	026512	104433						TRAP C\$RESET
762	026514	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
763	026520	016501	000002	MOV	TSSR(R5),R1		:GET THE CONTENTS OF TSSR	
764	026524	010102		MOV	R1,R2		:CONTENTS OF TSSR	
765	026526	042702	176277	BIC	#^C<HIADDR!OFL>,R2		:THESE BITS MAY BE SET	
766	026532	052702	002200	BIS	#SSR!NBA,R2		:READY AND NEW DATA SHOULD BE SET	
767	026536	020102		CMP	R1,R2		:COMPARE EXPECTED TO RECEIVED	
768	026540	001406		BEQ	20\$:BRANCH IF COMPARE	
772	026542			ERRDF	ERRNO,SFHERR,SFFMSG		:REPORT A FATAL ERROR	
	026542	104455						TRAP C\$ERDF
	026544	000766						.WORD 502
	026546	003701						.WORD SFHERR
	026550	012072						.WORD SFFMSG
773	026552	005237	002220	INC	FATFLG		:SET FATAL ERROR FLAG	
774	026556			20\$: CKLOOP			:LOOP ON ERROR IF FLAG SET	
	026556	104406						TRAP C\$CLP1
775	026560			ESCAPE	SUB		:EXIT IF FATAL ERROR DETECTED	
	026560	104410						TRAP C\$ESCAPE
	026562	000170						.WORD L10050-
776	026564	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR SSR TO SET	
777	026570	105065	000000	CLRB	TSDB(R5)		:PUT BACK INTO MAINTENANCE MODE	
778	026574	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
779	026600	005065	000000	CLR	TSDB(R5)		:SET ADDRESS BACK TO 0000	
780	026604	012702	000377	MOV	#377,R2			
781	026610	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
782	026614	110265	000000	MOVB	R2,TSDB(R5)		:SHOULD POINT TO RAM 0	
783	026620	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
784	026624	005065	000000	CLR	TSDB(R5)		:SELECT LOCATION 0	
785	026630	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	
786	026634	116501	000000	MOVB	TSBA(R5),R1		:READ RAM LOCATION SPECIFIED	
787	026640	120102		CMPB	R1,R2		:LOCATION SHOULD BE 377 OCTAL	
788	026642	001406		BEQ	25\$:BR IF OK	
789	026644			ERRDF	ERRNO,TSADDR,EXPREC		:WASN'T POINTING TO CORRECT LOC.	
	026644	104455						TRAP C\$ERDF
	026646	000766						.WORD 502
	026650	027442						.WORD TSADDR
	026652	015464						.WORD EXPREC
790	026654	005237	002220	INC	FATFLG		:SET THE FATAL ERROR FLAG	
791	026660			25\$: CKLOOP			:SCOPE LOOP	
	026660	104406						TRAP C\$CLP1
792	026662			ESCAPE	SUB		:NO MORE CHECKS IF FATAL ERROR	
	026662	104410						TRAP C\$ESCAPE
	026664	000066						.WORD L10050-
793	026666	012704	000310	MOV	#310,R4		:START WITH LOC 310	
794	026672	005002		CLR	R2		:MEMORY EXPECTED SHOULD BE 000000	
795	026674	004737	016326	JSR	PC,CHKTSSR		:WAIT FOR READY, NON-AMBIGUOUS	

817	026772			BGNSUB		:/:////// BEGIN SUBTEST //://////		
	026772					T5.2:		
	026772	104402					TRAP	C\$BSUB
818								
819	026774	004737	015764	JSR	PC,SOFINIT	:DO A SOFT TO START		
820	027000	103404		BCS	10\$:BRANCH IF O.K.		
824	027002			ERRDF	ERRNO,SFIERR,SFIMSG	:REPORT ERROR AND DROP DRIVE		
	027002	104455					TRAP	C\$ERDF
	027004	000767					.WORD	503
	027006	003646					.WORD	SFIERR
	027010	012024					.WORD	SFIMSG
825	027012	012702	177777	10\$:	MOV	#-1,R2		
826	027016	005004			CLR	R4	:ALL ONE DATA PATTERN	
827	027020	004737	016326		JSR	PC,CHKTSSR	:STARTING RAM ADDRESS	
828	027024	105065	000000	15\$:	CLRB	TSDB(R5)	:WAIT FOR READY, NON-AMBIGUOUS	
829	027030	004737	016326		JSR	PC,CHKTSSR	:SET MAINTENANCE MODE	
830	027034	010465	000000		MOV	R4,TSDB(R5)	:WAIT FOR READY, NON-AMBIGUOUS	
831	027040	004737	016326		JSR	PC,CHKTSSR	:SET THE NEXT RAM ADDRESS	
832	027044	110265	000000		MOVB	R2,TSDB(R5)	:WAIT FOR READY, NON-AMBIGUOUS	
833	027050	005204			INC	R4	:LOAD TEST DATA	
834	027052	020427	007777		CMP	R4,#7777	:NEXT ADDRESS TO TEST	
835	027056	003762			BLE	15\$:COMPARE TO LAST ADDRESS	
836	027060	005065	000002		CLR	TSSR(R5)	:BRANCH TILL ALL DATA WRITTEN	
837	027064	004737	016326		JSR	PC,CHKTSSR	:ISSUE A SOFT RESET	
838	027070	016501	000002		MOV	TSSR(R5),R1	:WAIT FOR READY, NON-AMBIGUOUS	
839	027074	010102			MOV	R1,R2	:GET THE CONTENTS OF TSSR	
840	027076	042702	176277		BIC	#^C<HIADDR!OFL>,R2	:CONTENTS OF TSSR	
841	027102	052702	002200		BIS	#SSR!NBA,R2	:THESE BITS MAY BE SET	
842	027106	020102			CMP	R1,R2	:READY AND NEW DATA SHOULD BE SET	
843	027110	001406			BEQ	20\$:COMPARE EXPECTED TO RECEIVED	
847	027112				ERRDF	ERRNO,SFHERR,SFFMSG	:BRANCH IF COMPARE	
	027112	104455					:REPORT A FATAL ERROR	
	027114	000770					TRAP	C\$ERDF
	027116	003701					.WORD	504
	027120	012072					.WORD	SFHERR
	027122	005237	002220				.WORD	SFFMSG
848	027122				INC	FATFLG	:SET FATAL ERROR FLAG	
849	027126			20\$:	CKLOOP		:LOOP ON ERROR IF FLAG SET	
	027126	104406					TRAP	C\$CLP1
850	027130				ESCAPE	SUB	:EXIT IF FATAL ERROR DETECTED	
	027130	104410					TRAP	C\$ESCAPE
	027132	000170					.WORD	L10051-
851	027134	004737	016326		JSR	PC,CHKTSSR	:WAIT FOR SSR TO SET	
852	027140	105065	000000		CLRB	TSDB(R5)	:PUT BACK INTO MAINTENANCE MODE	
853	027144	004737	016326		JSR	PC,CHKTSSR	:WAIT FOR READY, NON-AMBIGUOUS	
854	027150	005065	000000		CLR	TSDB(R5)	:SET ADDRESS BACK TO 0000	
855	027154	012702	000377		MOV	#377,R2		
856	027160	004737	016326		JSR	PC,CHKTSSR	:WAIT FOR READY, NON-AMBIGUOUS	
857	027164	110265	000000		MOVB	R2,TSDB(R5)	:SHOULD POINT TO RAM 0	
858	027170	004737	016326		JSR	PC,CHKTSSR	:WAIT FOR READY, NON-AMBIGUOUS	
859	027174	005065	000000		CLR	TSDB(R5)	:SELECT LOCATION 0	
860	027200	004737	016326		JSR	PC,CHKTSSR	:WAIT FOR READY, NON-AMBIGUOUS	
861	027204	116501	000000		MOVB	TSBA(R5),R1	:READ RAM LOCATION SPECIFIED	
862	027210	120102			CMPB	R1,R2	:LOCATION SHOULD BE 377 OCTAL	
863	027212	001406			BEQ	25\$:BR IF OK	
864	027214				ERRDF	ERRNO,TSADDR,EXPREC	:WASN'T POINTING TO CORRECT LOC.	
	027214	104455					TRAP	C\$ERDF
	027216	000770					.WORD	504

```

TEST 5: SECOND INITIALIZATION TEST

      027220 027442
      027222 015464
865 027224 005237 002220      INC      FATFLG      ;SET THE FATAL ERROR FLAG
866 027230      25$:      CKLOOP      ;SCOPE LOOP
      027230 104406
867 027232      ESCAPE      SUB      ;NO MORE CHECKS IF FATAL ERROR
      027232 104410
      027234 000066
      027236 012704 000310      MOV      #310,R4      ;START WITH LOC 310
869 027242 005002      CLP      R2      ;MEMORY EXPECTED SHOULD BE 000000
870 027244 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
871 027250 010465 000000      30$:      MOV      R4,TSDB(R5)      ;SELECT LOCATION SPECIFIED
872 027254 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
873 027260 116501 000000      MOVB     TSBA(R5),R1      ;READ LOC CONTENTS
874 027264 120102      CMPB     R1,R2      ;CHECK MEMORY FOR 000000
875 027266 001406      BEQ      40$      ;BRANCH IF DATA OKAY
876 027270      ERRDF     ERRNO,TSMEM,SFFMSG      ;MEMORY NOT ZERO AFTER INIT.
      027270 104455
      027272 000770
      027274 027404
      027276 012072
877 027300 005237 002220      40$:      INC      FATFLG      ;SET THE FATAL ERROR FLAG
878 027304      CKLOOP
      027304 104406
879 027306      ESCAPE      SUB      ;EXIT ON FATAL ERROR
      027306 104410
      027310 000012
      027312 005204
880 027312 005204      INC      R4      ;LOOK AT NEXT RAM LOC.
881 027314 020427 000400      CMP      R4,#400      ;AT TOP OF RAM ADDRESS SPACE
882 027320 001353      BNE      30$      ;BRANCH TILL ALL MEMORY TESTED
883
884 027322      ENDSUB      ;////////////////// END SUBTEST ////////////////////
      027322
      027322 104403
      027322 104403
885
886 027324 005737 002220      TST      FATFLG      ;IS FATAL ERROR FLAG SET ?
887 027330 001402      BEQ      50$      ;BRANCH IF NOT
888 027332 004737 017172      JSR      PC,CKDROP      ;NO LOOP, TRY TO DROP DEVICE
889 027336 004737 016446      50$:      JSR      PC,TSTLOOP      ;SHOULD WE DO ITERATIONS ?
890 027342 103002      BCC      60$      ;BRANCH IF NOT
891 027344 000137 026420      JMP      TSLOOP      ;LOOP UNTIL COUNT EXPIRED
892 027350      60$:      EXIT      TST      ;ALL DONE THIS TEST
      027350 104432
      027352 000132
893
894
895      ;+
896      ;LOCAL TEXT MESSAGES FOR TEST
897      ;-
898 027354      105      170      164 TST5ID: .ASCIZ 'Extended Initialization'
899 027404      111      156      143 TSMEM: .ASCIZ 'Incorrect RAM Data After Init'
900 027442      111      156      143 TSADDR: .ASCIZ 'Incorrect RAM Address After Init'
901
902 027504
      027504
      027504 104401
903

```

L10047: TRAP CSETST

904

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

.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 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. 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.
2. TSDB IS WRITTEN WITH ADDRESS OF THE COMMAND BUFFER TO START PROCESSING.
3. THE PROGRAM WAITS FOR SSR TO SET; IF SSR DOES NOT SET, AN ERROR REPORT IS ISSUED AND THE TEST IS ABORTED.
4. THE CONTENTS OF TSSR ARE CHECKED. TSSR IS CORRECT IF IT CONTAINS EITHER OCTAL 102206 OR 102306 (BIT 6 DEPENDS UPON THE STATE OF THE TAPE TRANSPORT).
5. THE CONTENTS OF TSBA ARE CHECKED. TSBA SHOULD CONTAIN THE INITIAL COMMAND BUFFER ADDRESS (LOADED IN STEP 2) PLUS 10 (OCTAL); I.E., TSBA SHOULD POINT TO THE WORD JUST AFTER THE COMMAND PACKET (NOTE THAT 4 COMMAND PACKET WORDS ARE ALWAYS FETCHED).
6. USING THE MAINTENANCE MODE WRAPAROUND FUNCTIONS, THE COMMAND IMAGE BLOCK IN THE M7196'S RAM (LOCATIONS 201-210 (OCTAL)) ARE CHECKED; THE IMAGE SHOULD CONTAIN A COPY OF THE FOUR COMMAND PACKET WORDS AS SET UP IN CPU MEMORY.
7. THE COMMAND WORD IN THE COMMAND BUFFER IS INCREMENTED TO THE NEXT PATTERN NOT CONTAINING WRITE CHARACTERISTICS OR IE. THE REMAINING THREE WORD OF THE COMMAND BUFFER ARE SEQUENCED WITH PSEUDO-RANDOM DATA. IF THE COMMAND WORD HAS NOT REACHED ITS MAXIMUM VALUE (177777+1), THE TEST SEQUENCE IS REPEATED.

SUBTEST 2 IS IDENTICAL TO SUBTEST 1, EXCEPT THAT THE PROGRAM

1057	030050			60\$:	BGNSUB	;////////// BEGIN SUBTEST ////////////	
	030050					T6.2:	
	030050	104402					TRAP CSBSUB
1058							
1059	030052				SETPRI #PRI00	;LOWER PRIORITY TO ALLOW INTERRUPTS	
	030052	012700	000000				MOV #PRI00,R0
	030056	104441					TRAP CSSPRI
1060	030060	012704	030420		MOV #T6PACKET,R4	;GET THE ADDRESS OF COMMAND PACKET	
1061	030064	012703	002750		MOV #TSTBLK,R3	;START OF TEST DATA	
1062	030070	012314		5\$:	MOV (R3)+,(R4)	;PLACE NEXT DATA WORD IN PACKET	
1063	030072				BGNSEG	;>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>	
	030072	104404					TRAP CSBSEG
1064	030074	004737	015764		JSR PC,SOFINIT	;DO SOFT INIT OF CONTROLLER	
1065	030100	103405			BCS 10\$;BR IF SOFT INIT = OK	
1069	030102	010001			MOV R0,R1	;SAVE CONTENTS OF TSSR	
1070	030104				ERRDF ERRNO,SFIERR,SFIMSG	;DEVICE FATAL ERROR DURING INIT	
	030104	104455					TRAP CSERDF
	030106	001137					.WORD 607
	030110	003646					.WORD SFIERR
	030112	012024					.WORD SFIMSG
1071	030114	005037	002220	10\$:	CLR FATFLG	;CLEAR FATAL ERROR FLAG	
1072	030120	005037	002222		CLR INTRECV	;CLEAR INTERRUPT RECEIVED FLAG	
1073	030124	004737	016326		JSR PC,CHKTSSR	;WAIT FOR READY, NON-AMBIGUOUS	
1074	030130	052714	060200		BIS #BIT7,(R4)	;ENABLE INTERRUPTS	
1075	030134	010465	000000		MOV R4,TSDB(R5)	;SET THE PACKET ADDRESS	
1076	030140	004737	016240		JSR PC,WAITF	;WAIT FOR SSR TO SET	
1077	030144	103407			BCS 15\$;BR IF CARRY SET (GOOD RETURN)	
1078	030146	010001			MOV R0,R1	;SAVE CONTENTS OF TSSR	
1082	030150				ERRDF ERRNO,T6SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	030150	104455					TRAP CSERDF
	030152	001140					.WORD 608
	030154	030455					.WORD T6SSR
	030156	012036					.WORD PKTSSR
1083	030160	005237	002220		INC FATFLG	;SET FATAL ERROR FLAG	
1084	030164			15\$:	CKLOOP	;LOOP ON ERROR, IF FLAG SET	
	030164	104406					TRAP C\$CLP1
1085	030166				ESCAPE SUB	;BY-PASS SUBTEST IF FATAL ERROR	
	030166	104410					TRAP C\$ESCAPE
	030170	000170					.WORD L10054-
1086	030172	005737	002222		TST INTRECV	;DID AN INTERRUPT OCCUR ?	
1087	030176	001004			BNE 22\$;BRANCH IF YES	
1091	030200				ERRHRD ERRNO,T6NINT,PKTSSR	;REPORT ERROR IF NO INTERRUPT	
	030200	104456					TRAP CSERHRD
	030202	001141					.WORD 609
	030204	030611					.WORD T6NINT
	030206	012036					.WORD PKTSSR
1092	030210	012702	102206	22\$:	MOV #SC!NBA!SSR!TSREJ,R2	;EXPECTED CONTENTS OF TSSR	
1093	030214	004737	016326		JSR PC,CHKTSSR	;WAIT FOR READY, NON-AMBIGUOUS	
1094	030220	016501	000002		MOV TSSR(R5),R1	;GET THE CONTENTS OF TSSR	
1095	030224	032701	000100		BIT #OFL,R1	;IS OFF-LINE BIT SET ?	
1096	030230	001402			BEQ 25\$;BRANCH IF NOT OFF-LINE	
1097	030232	052702	000100		BIS #OFL,R2	;SET OFF-LINE IN EXPECTED DATA	
1098	030236	020201		25\$:	CMP R2,R1	;DOES EXPECTED MATCH RECEIVED ?	
1099	030240	001404			BEQ 30\$;OKAY IF MATCH	
1103	030242				ERRHRD ERRNO,T6NBA,PKTSSR	;NBA NOT SET TO REJECT	
	030242	104456					TRAP CSERHRD
	030244	001142					.WORD 610

1152 030424 125252 .WORD 125252
1153 030426 052525 .WORD 052525

1154
1155
1156
1157
1158
1159
1159
1160 030430 103 157 155 T6NBA: .ASCIZ 'Command Not Rejected'

1161 030455 103 157 156 T6SSR: .ASCIZ 'Contents of TSSR Incorrect After Write Packet'
1162 030533 125 156 145 T6INT: .ASCIZ 'Unexpected Interrupt Received On Write Packet'
1163 030611 105 170 160 T6NINT: .ASCIZ 'Expected Interrupt Not Received On Write Packet'
1164 030671 111 156 143 T6TSBA: .ASCIZ 'Incorrect TSBA Address After Packet Write'
1165 030743 103 157 155 TST6ID: .ASCIZ 'Command Reject'
1166 .EVEN
1167 030762 .EVEN
030762 .EVEN
030762 104401 .EVEN
ENDTST

L10052: TRAP CSETST

1223	031056	005037	002220	10\$:	CLR	FATFLG	:CLEAR FATAL ERROR FLAG	
1224	031061	005037	002222		CLR	INTRECV	:CLEAR INTERRUPT RECEIVED FLAG	
1225	031066	010465	000000		MOV	R4,TSDB(R5)	:SET THE PACKET ADDRESS	
1226	031072	004737	016326		JSR	PC,CHKTSSR	:WAIT FOR SSR TO SET	
1227	031076	103407			BCS	15\$:BR IF CARRY SET (GOOD RETURN)	
1228	031100	010001			MOV	R0,R1	:SAVE CONTENTS OF TSSR	
1232	031102				ERRDF	ERRNO,T7SSR,PKTSSR	:DEVICE FATAL SSR FAILED TO SET	
	J31102	104455					TRAP	C\$ERDF
	031104	001276					.WORD	702
	031106	033721					.WORD	T7SSR
	031110	012036					.WORD	PKTSSR
1233	031112	005237	002220	15\$:	INC	FATFLG	:SET FATAL ERROR FLAG	
1234	031116				CKLOOP		:LOOP ON ERROR, IF FLAG SET	
	031116	104406					TRAP	C\$CLP1
1235	031120				ESCAPE	SEG	:BY-PASS SUBTEST IF FATAL ERROR	
	031120	104410					TRAP	C\$ESCAPE
	031122	000152					.WORD	10000\$-
1236	031124	005737	002222		TST	INTRECV	:DID AN INTERRUPT OCCUR ?	
1237	031130	001404			BEQ	22\$:BRANCH IF NOT	
1241	031132				ERRHRD	ERRNO,T7INT,PKTSSR		
	031132	104456					TRAP	C\$ERHRD
	031134	001277					.WORD	703
	031136	034101					.WORD	T7INT
	031140	012036					.WORD	PKTSSR
1242	031142	016501	000002	22\$:	MOV	TSSR(R5),R1	:GET THE CONTENTS OF TSSR	
1243	031146	012702	000200		MOV	#SSR,R2	:EXPECTED CONTENTS OF TSSR	
1244	031152	032701	000100		BIT	#OFL,R1	:IS OFF-LINE BIT SET ?	
1245	031156	001402			BEQ	25\$:BRANCH IF NOT OFF-LINE	
1246	031160	052702	000100		BIS	#OFL,R2	:SET OFF-LINE IN EXPECTED DATA	
1247	031164	020201		25\$:	CMP	R2,R1	:DOES EXPECTED MATCH RECEIVED ?	
1248	031166	001404			BEQ	30\$:OKAY IF MATCH	
1252	031170				ERRHRD	ERRNO,T7NBA,PKTSSR	:NBA NOT ZERO	
	031170	104456					TRAP	C\$ERHRD
	031172	001300					.WORD	704
	031174	033260					.WORD	T7NBA
	031176	012036					.WORD	PKTSSR
1253	031200			30\$:	CKLOOP		:LOOP ON ERROR ?	
	031200	104406					TRAP	C\$CLP1
1254	031202	004737	016326		JSR	PC,CHKTSSR	:WAIT FOR READY, NON-AMBIGUOUS	
1255	031206	016501	000000		MOV	TSBA(R5),R1	:GET TSBA REGISTER CONTENTS	
1256	031212	012702	033146		MOV	#T7BFR,R2	:START OF THE DATA BUFFER	
1257	031216	032762	000200	000012	BIT	#BIT7,XST2(R2)	:IS EXTENDED FEATURES BIT SET ?	
1258	031224	001404			BEQ	32\$:BRANCH IF EXTENDED FEATURES OFF	
1259	031226	005237	002224		INC	EXTFEA	:SET EXTENDED FEATURES FOR SUBTEST 6	
1260	031232	062702	000002		ADD	#2,R2	:EXTRA WORD IF SPECIAL FEATURES	
1261	031236	062702	000016	32\$:	ADD	#14,R2	:EXPECTED CONTENTS OF TSDA	
1262	031242	020102			CMP	R1,R2	:COMPARE EXPECTED TO RECEIVED	
1263	031244	001404			BEQ	35\$:ERROR IF NOT EQUAL	
1267	031246				ERRHRD	ERRNO,T7TSBA,EXPREC	:PRINT THE ERROR & EXPD/RCV	
	031246	104456					TRAP	C\$ERHRD
	031250	001301					.WORD	705
	031252	034170					.WORD	T7TSBA
	031254	015464					.WORD	EXPREC
1268								
1269								
1270	031256	004737	011104	35\$:	JSR	PC,CKRAM	:SEE IF DATA IN RAM IS CORRECT	
1271	031262	103404			BCS	40\$:BRANCH IF PACKET IN RAM IS CORRECT	


```

1414 031744 001402          BEQ      25$
1415 031746 052702 000100    BIS      #OFL,R2
1416 031752 020201          CMP      R2,R1
1417 031754 001414          BEQ      30$
1418 031756 010100          MOV      R1,R0
1419 031760          XOR      R2,R0
1420 031770 020027 002000    CMP      R0,#NBA
1421 031774 001404          BEQ      30$
1425 031776          ERRHRD  ERRNO,T73REJ,PKTSSR
      031776 104456
      032000 001313
      032002 033432
      032004 012036
1426 032006          30$:  CKLOOP
      032006 104406
1427 032010 032701 002000    BIT      #NBA,R1
1428 032014 001004          BNE      35$
1432 032016          ERRHRD  ERRNO,T72NBA,PKTSSR
      032016 104456
      032020 001314
      032022 033202
      032024 012036
1433 032026          35$:
1434 032026          ENDSEG
      032026
      032026 104405
1435
1436 032030 005203          INC      R3
1437 032032 020327 000006    CMP      R3,#6
1438 032036 002002          BGE      57$
1439 032040 000137 031616    JMP      5$
1440
1441 032044          57$:  ENDSUB
      032044
      032044 104403

```

```

;BRANCH IF NOT OFF-LINE
;SET OFF-LINE IN EXPECTED DATA
;DOES EXPECTED MATCH RECEIVED ?
;OKAY IF MATCH
;DATA FROM TSSR
;FIND BITS IN ERROR
;IS NBA ONLY BIT IN ERROR ?
;DON'T PRINT ERROR IF NBA ONLY BAD BIT
;COMMAND NOT REJECTED
      TRAP  C$ERHRD
      .WORD 715
      .WORD T73REJ
      .WORD PKTSSR
;LOOP ON ERROR ?
      TRAP  C$CLP1
;IS NBA BIT SET ?
;OKAY IF NBA SET
;NBA NOT SET
      TRAP  C$ERHRD
      .WORD 716
      .WORD T72NBA
      .WORD PKTSSR
;<<<<<<<<<<<<<<<< END SEGMENT<<<<<<<<<<<<<<<<
10000$:
      TRAP  C$ESEG
;NEXT BYTE COUNT
;TESTED ALL INVALID ?
;BRANCH IF TEST DONE
;BRANCH TILL BACK TO ZERO
;//////////////// END SUBTEST //////////////////
L10060:
      TRAP  C$ESUB

```



```

1443
1444
1445
1446
1447
1448
1449
1450
1451
1452 032046          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      032046          ;              T7.4:
      032046 104402          TRAP          C$BSUB
1453
1454 032050          SETPRI          #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
      032050 012700 000000          MOV          #PRI00,R0
      032054 104441          TRAP          C$SPRI
1455 032056 012703 033166          MOV          #T72DATA,R3          ;START OF TEST DATA FOR SUBTEST
1456 032062 012704 033120          MOV          #T7PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
1457 032066 004737 034302          JSR          PC,T7REST          ;RESTORE PACKET TO STARTING VALUES
1458
1459
1460 032072 004737 015764          JSR          PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
1461 032076 103405          BCS          10$          ;BR IF SOFT INIT = OK
1465 032100 010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
1466 032102          ERRDF          ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL ERROR DURING INIT
      032102 104455          TRAP          C$ERDF
      032104 001315          .WORD          717
      032106 003646          .WORD          SFIERR
      032110 012024          .WORD          SFIMSG
1467 032112 005037 002222          CLR          INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
1468 032116 052737 000001 033130          BIS          #1,T7DATA          ;MAKE ADDRESS ODD
1469 032124 010465 000000          MOV          R4,TSDB(R5)          ;SET THE PACKET ADDRESS
1470 032130 004737 016240          JSR          PC,WAITF          ;WAIT FOR SSR TO SET
1471 032134 103405          BCS          15$          ;BR IF CARRY SET (GOOD RETURN)
1472 032136 010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
1476 032140          ERRDF          ERRNO,T7SSR,PKTSSR          ;DEVICE FATAL SSR FAILED TO SET
      032140 104455          TRAP          C$ERDF
      032142 001316          .WORD          718
      032144 033721          .WORD          T7SSR
      032146 012036          .WORD          PKTSSR
1477 032150          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      032150 104406          TRAP          C$CLP1
1478 032152          ESCAPE          SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      032152 104410          TRAP          C$ESCAPE
      032154 000116          .WORD          L10061-.
1479 032156 005737 002222          TST          INTRECV          ;DID AN INTERRUPT OCCUR ?
1480 032162 001404          BEQ          22$          ;BRANCH IF NOT
1484 032164          ERRHRD          ERRNO,T7INT,PKTSSR
      032164 104456          TRAP          C$ERHRD
      032166 001317          .WORD          719
      032170 034101          .WORD          T7INT
      032172 012036          .WORD          PKTSSR
1485 032174 016501 000002          22$: MOV          TSSR(R5),R1          ;GET THE CONTENTS OF TSSR
1486 032200 012702 102206          MOV          #C$!SSR!T$REJ!NBA,R2          ;EXPECTED CONTENTS OF TSSR
1487 032204 032701 000100          BIT          #OFL,R1          ;IS OFF-LINE BIT SET ?
1488 032210 001402          BEQ          25$          ;BRANCH IF NOT OFF-LINE
1489 032212 052702 000100          BIS          #OFL,R2          ;SET OFF-LINE IN EXPECTED DATA
    
```

```

1490 032216 020201          25$:  CMP      R2,R1          ;DOES EXPECTED MATCH RECEIVED ?
1491 032220 001414          BEQ      30$          ;OKAY IF MATCH
1492 032222 010100          MOV      R1,R0          ;DATA FROM TSSR
1493 032224          XOR      R2,R0          ;FIND BITS IN ERROR
1494 032234 020027 002000    CMP      R0,#N3A        ;IS NBA ONLY BIT IN ERROR ?
1495 032240 001404          BEQ      30$          ;DON'T PRINT ERROR IF NBA ONLY BAD BIT
1499 032242          ERRHRD  ERRNO,T74REJ,PKTSSR ;COMMAND NOT REJECTED
      032242 104456          TRAP    C$ERHRD
      032244 001320          .WORD  720
      032246 033525          .WORD  T74REJ
      032250 012036          .WORD  PKTSSR
1500 032252          30$:  CKLOOP          ;LOOP ON ERROR ?
      032252 104406          TRAP    C$CLP1
1501 032254 032701 002000    BIT      #NBA,R1        ;IS NBA BIT SET ?
1502 032260 001004          BNE     35$          ;OKAY IF NBA SET
1506 032262          ERRHRD  ERRNO,T72NBA,PKTSSR ;NBA NOT SET
      032262 104456          TRAP    C$ERHRD
      032264 001321          .WORD  721
      032266 033202          .WORD  T72NBA
      032270 012036          .WORD  PKTSSR
1507
1508 032272          35$:  ENDSUB          ;////////// END SUBTEST ////////////
      032272          L10061:
1509 032272 104403          TRAP    C$ESUB
  
```

```

1511          :+
1512          :
1513          :TEST 7, SUBTEST 5
1514          :
1515          :CHECK THAT WRITE CHARACTERISTICS COMMAND IS REJECTED IF THE
1516          :MESSAGE BUFFER LENGTH SPECIFIES AN INVALID COUNT (LESS THAN 14)
1517          :
1518          :-
1519          :
1520 032274   BGNSUB               ;//////////////// BEGIN SUBTEST //////////////////
032274                       ; T7.5:
032274 104402                       TRAP   C$BSUB
1521
1522 032276   SETPRI #PRI00        ;LOWER PRIORITY TO ALLOW INTERRUPTS
032276 012700 000000                       MOV   #PRI00,R0
032302 104441                       TRAP   C$SPRI
1523 032304 012703 000001        ;STARTING BUFFER LENGTH
1524 032310 012704 033120        ;GET THE ADDRESS OF COMMAND PACKET
1525 032314 004737 034302        ;RESTORE PACKET TO STARTING VALUES
1526
1527 032320   BGNSEG               ;>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>
032320 104404                       TRAP   C$BSEG
1528
1529 032322 004737 015764        JSR   PC,SOFINIT            ;DO SOFT INIT OF CONTROLLER
1530 032326 103405              BCS   10$                  ;BR IF SOFT INIT = OK
1534 032330 010001              MOV   R0,R1                ;SAVE CONTENTS OF TSSR
1535 032332 104455              ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
032332 001322                       TRAP   C$ERDF
032336 003646                       .WORD 722
032340 012024                       .WORD SFIERR
1536 032342 005037 002222        10$: CLR   INTRECV            ;CLEAR INTERRUPT RECEIVED FLAG
1537 032346 010337 033134        MOV   R3,T7DATA+4          ;INSERT THE BAD MESSAGE LENGTH
1538 032352 010465 000000        MOV   R4,TSDB(R5)          ;SET THE PACKET ADDRESS
1539 032356 004737 016240        JSR   PC,WAITF             ;WAIT FOR SSR TO SET
1540 032362 103405              BCS   15$                  ;BR IF CARRY SET (GOOD RETURN)
1541 032364 010001              MOV   R0,R1                ;SAVE CONTENTS OF TSSR
1545 032366 104455              ERRDF  ERRNO,T7SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
032366 001323                       TRAP   C$ERDF
032370 001323                       .WORD 723
032372 033721                       .WORD T7SSR
032374 012036                       .WORD PKTSSR
1546 032376 104406              15$: CKLOOP             ;LOOP ON ERROR, IF FLAG SET
032376 104406                       TRAP   C$CLP1
1547 032400   ESCAPE SEG          ;BY-PASS SUBTEST IF FATAL ERROR
032400 104410                       TRAP   C$ESCAPE
032402 000116                       .WORD 10000$-
1548 032404 005737 002222        TST   INTRECV              ;DID AN INTERRUPT OCCUR ?
1549 032410 001404              BEQ   22$                  ;BRANCH IF NOT
1553 032412 104456              ERRHRD ERRNO,T7INT,PKTSSR
032412 001324                       TRAP   C$ERHRD
032414 034101                       .WORD 724
032416 012036                       .WORD T7INT
032420 012036                       .WORD PKTSSR
1554 032422 016501 000002        22$: MOV   TSSR(R5),R1        ;GET THE CONTENTS OF TSSR
1555 032426 012702 102206        MOV   #SC!SSR!TSREJ!NBA,R2 ;EXPECTED CONTENTS OF TSSR
1556 032432 032701 000100        BIT   #OFL,R1              ;IS OFF-LINE BIT SET ?

```



```

1681 033076
      033076
      033076 104403
1682
1683 033100 005737 002220
1684 033104 001402
1685 033106 004737 017172
1686 033112
1687 033112
      033112 104432
      033114 001234
1688
1689
1690
1691
1692
1694      033120
1696 033120
1697 033120 100004
1698 033122 033130
1699 033124 000000
1700 033126 000010
1701
1702 033130
1703 033130 033146
1704 033132 000000
1705 033134 000016
1706 033136 000000
1707 033140 000000
1708
1709 033142 000000 000000
1710 033146
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723 033166
1724 033166 000000 037140
1725 033172 000002 000001
1726 033176 000004 100100
1727      033202
1728
1729
1730
1731
1732
1733
1734 033202      116      102      101
1735 033260      127      122      111

```

55\$: ENDSUB

```

;////////// END SUBTEST ////////////
L10063:

```

TRAP C\$ESUB

```

TST FATFLG
BEQ 60$
JSR PC,CKDROP

```

```

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

```

```

60$: EXIT TST

```

;ALL DONE THIS TEST

```

TRAP C$EXIT
.WORD L10055-.

```

```

: +
: LOCAL STORAGE FOR THIS TEST
: -

```

.=<.+10>&177770

T7PACKET:

```

.WORD 100004
.WORD T7DATA
.WORD 0
.WORD 8.

```

```

;COMMAND PACKET FOR TEST
;WRITE CHARACTERISTICS COMMAND, WITH ACK
;ADDRESS OF CHARACTERISTICS BLOCK

```

;STARTING VALUE OF BLOCK SIZE

T7DATA:

```

.WORD T7BFR
.WORD 0
.WORD 14.
.WORD 0

```

```

;CHARACTERISTICS DATA BLOCK
;ADDRESS OF MESSAGE BUFFER

```

;LENGTH OF MESSAGE BUFFER

T7SP:

.WORD 0

;EXTFEA EXTRA WORD

```

T7BFR: .WORD 0,0
        .BLKW 8.

```

```

;SPACE
;MESSAGE BUFFER

```

```

: +
: TEST DATA FOR SUBTEST TWO
: -

```

: DATA HAS FORMAT:

```

          1ST WORD       OFFSET TO TEST WORD IN PACKET
          2ND WORD       BITS TO SET FOR TEST

```

T72DATA:

```

.WORD 0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
.WORD 2,BIT0
.WORD 4,BIT6!BIT15

```

T72DONE=.

```

: +
: LOCAL TEXT MESSAGES FOR TEST
: -

```

```

T72NBA: .ASCIZ 'NBA Not Set On Rejected WRITE CHARACTERISTICS'
T7NBA:  .ASCIZ 'WRITE CHARACTERISTICS Command Not Accepted'

```

1736	033333	127	122	111	T72REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
1737	033432	127	122	111	T73REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
1738	033525	127	122	111	T74REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
1739	033623	127	122	111	T75REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
1740	033721	103	157	156	T7SSR: .ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
1741	034010	105	170	160	T7NINT: .ASCIZ	'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
1742	034101	125	156	145	T7INT: .ASCIZ	'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
1743	034170	111	156	143	T7TSBA: .ASCIZ	'Incorrect TSBA Address After WRITE CHARACTERISTICS'
1744	034253	127	162	151	TST7ID: .ASCIZ	'Write Characteristics'
1745					.EVEN	
1746						

1748
 1749
 1750
 1751
 1752
 1753
 1754
 1755 034302
 1756 034302
 1757 034306 012701 033120
 1758 034312 012721 100004
 1759 034316 012721 033130
 1760 034322 005021
 1761 034324 012721 000010
 1762 034330 012721 033146
 1763 034334 005021
 1764 034336 012721 000020
 1765 034342 005021
 1766 034344 005011
 1767 034346 000207
 1768 034350
 034350
 034350 104401
 1769

```

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

```

```

T7REST:
  SAVREG          ;SAVE THE REGISTERS
  MOV #T7PACKET,R1 ;START OF THE PACKET
  MOV #100004,(R1)+ ;WRITE CHARACTERISTICS WITH ACK
  MOV #T7DATA,(R1)+ ;ADDRESS OF CHAR DATA BLOCK
  CLR (R1)+        ;EXTENDED ADDRESS
  MOV #8,(R1)+     ;SIZE OF DATA BLOCK IN BYTES
  MOV #T7BFR,(R1)+ ;ADDRESS OF MESSAGE BUFFER
  CLR (R1)+
  MOV #16,(R1)-    ;LENGTH OF MESSAGE BUFFER
  CLR (R1)+
  CLR (R1)
  RTS PC           ;RETURN
  ENDTST

```

```

L10055: TRAP CSETST

```

1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1810
1811
1812
1813
1814
1815
1816
1817
1821
1822
1823
1824
1825
1826
1827
1831

034352
034352
034352 012700 035237
034356 004737 016500
034362 012737 000024 002214
034370
034370 012704 034760
034374 004737 015764
034400 103405
034402 010001
034404 104455
034404 001441
034410 003646
034412 012024
034414 042714 040000
034420 010465 000000
034424 004737 016326
034430 103405
034432 010001
034434 104455

.....

.SBTTL 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.

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 XST0 IN THE RETURNED MESSAGE BUFFER IS EXAMINED; THE VCK BIT SHOULD BE CLEAR (0).
3. THE PREVIOUS STEP IS REPEATED TO VERIFY THAT VCK DOES NOT CHANGE (REMAINS AT 0).
4. A WRITE CHARACTERISTICS COMMAND IS ISSUED WITH CVC=1 AND THE VCK BIT IN XST0 IN THE MESSAGE BUFFER IS EXAMINED; THE VCK BIT SHOULD BE CLEAR (0).
5. A WRITE CHARACTERISTICS COMMAND IS ISSUED WITH CVC=0 AND THE VCK BIT IN XST0 IN THE MESSAGE BUFFER IS EXAMINED; THE VCK BIT SHOULD REMAIN CLEAR (0).

```

BGNFST
T8::
1810 034352 012700 035237      MOV   #TST8ID,R0      ;ASCII MESSAGE TO IDENTIFY TEST
1811 034356 004737 016500      JSR   PC,TSTSETUP    ;DO INITIAL TEST SETUP
1812 034362 012737 000024 002214  MOV   #20.,LOOPCNT  ;PERFORM 20 ITERATIONS
1813 034370
T8LOOP:
1815 034370 012704 034760      MOV   #T8PACKET,R4  ;PACKET FOR WRITE CHARACTERISTICS
1816 034374 004737 015764      JSR   PC,SOFINIT    ;DO SOFT INIT OF CONTROLLER
1817 034400 103405              BCS   10$           ;BR IF SOFT INIT = OK
1821 034402 010001              MOV   R0,R1         ;SAVE CONTENTS OF TSSR
1822 034404 104455              ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
                                TRAP   C$ERDF
                                .WORD  801
                                .WORD  SFIERR
                                .WORD  SFIMSG
1823 034414 042714 040000      10$:  BIC   #BIT14,(R4)   ;CLEAR THE CVC BIT
1824 034420 010465 000000      MOV   R4,TSD8(R5)  ;SET THE PACKET ADDRESS FOR WRITE CHAR
1825 034424 004737 016326      JSR   PC,CHKTSSR   ;WAIT FOR SSR TO SET
1826 034430 103405              BCS   15$           ;BR IF CARRY SET (GOOD RETURN)
1827 034432 010001              MOV   R0,R1         ;SAVE CONTENTS OF TSSR
1831 034434 104455              ERRDF ERRNO,T8SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP   C$ERDF
    
```

034436	001442							.WORD	802
034440	035150							.WORD	T8SSR
034442	012036							.WORD	PKTSSR
1832 034444		15\$:	CKLOOP				;LOOP ON ERROR, IF FLAG	SET	
034444	104406							TRAP	C\$CLP1
1833 034446			ESCAPE	TST			;EXIT IF FATAL ERROR		
034446	104410							TRAP	C\$ESCAPE
034450	000604							.WORD	L10064-
1834 034452	012702	035002	MOV	#T8BFR,R2			;ADDRESS OF THE MESSAGE BUFFER		
1835 034456	032762	000020	BIT	#XSOVCK,XSTO(R2)			;IS VOLUME CHECK SET IN XSTO ?		
1836 034464	001406		BEQ	20\$;OKAY IF VOLUME CHECK IS CLEAR		
1840 034466	016501	000002	MOV	TSSR(R5),R1			;CONTENTS OF TSSR FOR ERROR REPORT		
1841 034472			ERRHRD	ERRNO,T8NVCK,PKTMES			;VOLUME CHECK NOT CLEAR		
034472	104456							TRAP	C\$ERHRD
034474	001443							.WORD	803
034476	035057							.WORD	T8NVCK
034500	012100							.WORD	PKTMES
1842 034502		20\$:	CKLOOP				;LOOP ON ERROR ?		
034502	104406							TRAP	C\$CLP1
1843 034504	010465	000000	MOV	R4,TSDB(R5)			;SET THE PACKET ADDRESS FOR WRITE CHAR		
1844 034510	004737	016326	JSR	PC,CHKTSSR			;WAIT FOR SSR TO SET		
1845 034514	103405		BCS	25\$;BR IF CARRY SET (GOOD RETURN)		
1846 034516	010001		MOV	R0,R1			;SAVE CONTENTS OF TSSR		
1850 034520			ERRDF	ERRNO,T8SSR,PKTSSR			;DEVICE FATAL SSR FAILED TO SET		
034520	104455							TRAP	C\$ERDF
034522	001444							.WORD	804
034524	035150							.WORD	T8SSR
034526	012036							.WORD	PKTSSR
1851 034530		25\$:	CKLOOP				;LOOP ON ERROR, IF FLAG	SET	
034530	104406							TRAP	C\$CLP1
1852 034532			ESCAPE	TST			;EXIT IF FATAL ERROR		
034532	104410							TRAP	C\$ESCAPE
034534	000520							.WORD	L10064-
1853 034536	032762	000020	BIT	#XSOVCK,XSTO(R2)			;IS VOLUME CHECK SET IN XSTO ?		
1854 034544	001406		BEQ	30\$;OKAY IF VOLUME CHECK IS SET		
1858 034546	016501	000002	MOV	TSSR(R5),R1			;CONTENTS OF TSSR FOR ERROR REPORT		
1859 034552			ERRHRD	ERRNO,T8NVCK,PKTMES			;VOLUME CHECK NOT SET		
034552	104456							TRAP	C\$ERHRD
034554	001445							.WORD	805
034556	035057							.WORD	T8NVCK
034560	012100							.WORD	PKTMES
1860 034562		30\$:	CKLOOP				;LOOP ON ERROR ?		
034562	104406							TRAP	C\$CLP1
1861 034564	052714	040000	BIS	#BIT14,(R4)			;SET THE CVC BIT		
1862 034570	010465	000000	MOV	R4,TSDB(R5)			;SET THE PACKET ADDRESS FOR WRITE CHAR		
1863 034574	004737	016326	JSR	PC,CHKTSSR			;WAIT FOR SSR TO SET		
1864 034600	103405		BCS	35\$;BR IF CARRY SET (GOOD RETURN)		
1865 034602	010001		MOV	R0,R1			;SAVE CONTENTS OF TSSR		
1869 034604			ERRDF	ERRNO,T8SSR,PKTSSR			;DEVICE FATAL SSR FAILED TO SET		
034604	104455							TRAP	C\$ERDF
034606	001446							.WORD	806
034610	035150							.WORD	T8SSR
034612	012036							.WORD	PKTSSR
1870 034614		35\$:	CKLOOP				;LOOP ON ERROR, IF FLAG	SET	
034614	104406							TRAP	C\$CLP1
1871 034616			ESCAPE	TST			;EXIT IF FATAL ERROR		
034616	104410							TRAP	C\$ESCAPE

```

1872 034620 000434 000020 000006 BIT #XSOVCK,XSTO(R2) ;IS VOLUME CHECK CLEAR IN XSTO ?
1873 034622 032762 BEQ 40$ ;OKAY IF VOLUME CHECK IS CLEARED
1877 034630 001406 MOV TSSR(R5),R1 ;CONTENTS OF TSSR FOR ERROR REPORT
1878 034632 016501 000002 ERRHRD ERRNO,T8VCK,PKTMES ;VOLUME CHECK NOT CLEARED
      034636 104456 TRAP C$ERHRD
      034640 001447 .WORD 807
      034642 035022 .WORD T8VCK
      034644 012100 .WORD PKTMES
1879 034646 40$: CKLOOP ;LOOP ON ERROR ?
      034646 104406 TRAP C$CLP1
1880 034650 042714 040000 BIC #BIT14,(R4) ;CLEAR THE CVC BIT
1881 034654 010465 000000 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS FOR WRITE CHAR
1882 034660 004737 016326 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
1883 034664 103405 BCS 45$ ;BR IF CARRY SET (GOOD RETURN)
1884 034666 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
1888 034670 ERRDF ERRNO,T8SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      034670 104455 TRAP C$ERDF
      034672 001450 .WORD 808
      034674 035150 .WORD T8SSR
      034676 012036 .WORD PKTSSR
1889 034700 45$: CKLOOP ;LOOP ON ERROR, IF FLAG SET
      034700 104406 TRAP C$CLP1
1890 034702 ESCAPE TST ;EXIT IF FATAL ERROR
      034702 104410 TRAP C$ESCAPE
      034704 000350 .WORD L10064-.
1891 034706 032762 000020 000006 BIT #XSOVCK,XSTO(R2) ;IS VOLUME CHECK CLEAR IN XSTO ?
1892 034714 001406 BEQ 50$ ;OKAY IF VOLUME CHECK IS CLEARED
1896 034716 016501 000002 MOV TSSR(R5),R1 ;CONTENTS OF TSSR FOR ERROR REPORT
1897 034722 ERRHRD ERRNO,T8VCK,PKTMES ;VOLUME CHECK NOT CLEARED
      034722 104456 TRAP C$ERHRD
      034724 001451 .WORD 809
      034726 035022 .WORD T8VCK
      034730 012100 .WORD PKTMES
1898 034732 50$: CKLOOP ;LOOP ON ERROR ?
      034732 104406 TRAP C$CLP1
1899 034734 004737 016446 60$: JSR PC,TSTLOOP ;SHOULD WE DO ITERATIONS ?
1900 034740 103002 BCC 62$ ;BRANCH IF NOT
1901 034742 000137 034370 JMP T8LOOP ;LOOP UNTIL COUNT EXPIRED
1902 034746 62$: EXIT TST ;ALL DONE THIS TEST
      034746 104432 TRAP C$EXIT
      034750 000304 .WORD L10064-.
1903
1904
1905 ;+
1906 ;LOCAL STORAGE FOR THIS TEST
1907 ;-
1909 034760
1911 034760 T8PACKET: .=<. +10>&177770
1912 034760 100004 .WORD 100004 ;COMMAND PACKET FOR TEST
1913 034762 034770 .WORD T8DATA ;WRITE CHARACTERISTICS COMMAND
1914 034764 000000 .WORD 0 ;ADDRESS OF CHARACTERISTICS BLOCK
1915 034766 000010 .WORD 10 ;STARTING VALUE OF COUNTER
1916
1917 034770 T8DATA:
1918 034770 035002 .WORD T8BFR ;CHARACTERISTICS DATA BLOCK
1919 034772 000000 .WORD 0 ;ADDRESS OF MESSAGE BUFFER

```

1920 034774 000020 .WORD 16. ;LENGTH OF MESSAGE BUFFER
1921 034776 000000 000000 .WORD 0,0
1922
1923 035002 T8BFR: .BLKW 8. ;MESSAGE BUFFER
1924
1925

1926
1927 :+
1928 :LOCAL TEXT MESSAGES FOR TEST
1929 :-

1930 035022 126 157 154 T8VCK: .ASCIZ 'Volume Check Bit Not Cleared'
1931 035057 126 157 154 T8NVCK: .ASCIZ 'Volume Check Bit (VCK) Not Clear After Initialize (XST0)'
1932 035150 103 157 156 T8SSR: .ASCIZ 'Contents of TSSR Incorrect After Write Characteristics'
1933 035237 126 157 154 T8TID: .ASCIZ 'Volume Check'

1934
1935 035254 .EVEN
035254 LNDTST

L10064: TRAP CSETST

104401


```

TEST 9: COMPLETION INTERRUPT
          035402 001606
          035404 037737
1993 035406 012036
1994 035410 005237 002220           15$:  INC    FATFLG           ;SET FATAL ERROR FLAG
          035414 104406           CKLOOP           ;LOOP ON ERROR, IF FLAG SET
1995 035416 104406           ESCAPE  SEG      TRAP    C$CLP1
          035416 104410           TRAP    C$ESCAPE
          035420 000056           TRAP    10000$.
1996 035422 005737 002222           TST    INTRECV   ;DID AN INTERRUPT OCCUR ?
1997 035426 001004           BNE    22$      ;BRANCH IF YES
2001 035430           ERRHRD  ERRNO, T9NINT, PKTSSR
          035430 104456           TRAP    C$ERHRD
          035432 001607           .WORD  903
          035434 040026           .WORD  T9NINT
          035436 012036           .WORD  PKTSSR
2002 035440 016501 000002           22$:  MOV    TSSR(R5), R1  ;GET THE CONTENTS OF TSSR
2003 035444 012702 000200           MOV    #SSR, R2    ;EXPECTED CONTENTS OF TSSR
2004 035450 032701 000100           BIT    #OFL, R1    ;IS OFF-LINE BIT SET ?
2005 035454 001402           BEQ    25$         ;BRANCH IF NOT OFF-LINE
2006 035456 052702 000100           BIS    #OFL, R2    ;SET OFF-LINE IN EXPECTED DATA
2007 035462 020201           25$:  CMP    R2, R1     ;DOES EXPECTED MATCH RECEIVED ?
2008 035464 001404           BEQ    30$         ;OKAY IF MATCH
2012 035466           ERRHRD  ERRNO, T9NBA, PKTSSR ;NBA NOT ZERO
          035466 104456           TRAP    C$ERHRD
          035470 001610           .WORD  904
          035472 037276           .WORD  T9NBA
2013 035474 012036           .WORD  PKTSSR
2014 035476           30$:  ENDSEG          ;<<<<<<<<<<<<< END SEGMENT <<<<<<<<<<<<<
          035476 104405           .00000$. TRAP    C$ESEG
2015 035500 012364 000006           MOV    (R3)+, PKBCNT(R4) ;SET THE TEST WORD
2016 035504 020327 003056           CMP    R3, #TBLEND ;HAS ALL DATA BEEN TESTED ?
2017 035510 103002           BHIS   55$         ;BRANCH IF ALL DATA DONE
2018 035512 000137 035332           JMr    55$         ;BRANCH TILL BACK TO ZERO
2020 035516           55$:  ENDSUB        ;////////// END SUBTEST //////////
2021 035516           L10066: TRAP    C$ESUB
          035516 104403
2022 035520 005737 002220           TST    FATFLG     ;ANY FATAL ERRORS ?
2023 035524 001402           BEQ    60$         ;BRANCH IF NOT
2024 035526 004737 017172           JSR    PC, CKDROP  ;TRY TO DROP THE UNIT
2025 035532 033727 037254 000200           60$:  BIT    T9BFR+12, #BIT7 ;EXTENDED FEATURES SET?
2026 035540 001402           BEQ    70$         ;BR IF NO
2027 035542 005237 002224           INC    EXTFEA     ;SET EXT FEATURE FLAG
2029 035546           70$:

```

```

2031          :+
2032          :
2033          :TEST 9, SUBTEST 2
2034          :
2035          :CHECK THAT UNUSED BITS BEING SET CAUSES
2036          :WRITE CHARACTERISTICS COMMAND TO BE REJECTED
2037          :
2038          :-
2039
2040 035546          BGNSUB          ://////////////// BEGIN SUBTEST //////////////////
          035546          :                      T9.2:
          035546 104402          TRAP          C$BSUB
2041
2042 035550          SETPRI #PRI00          :LOWER PRIORITY TO ALLOW INTERRUPTS
          035550 012700 000000          MOV          #PRI00,R0
          035554 104441          TRAP          C$SPRI
2043 035556 012703 037262          :START OF TEST DATA FOR SUBTEST
2044 035562 012704 037220 5$: MOV #T92DATA,R3          :GET THE ADDRESS OF COMMAND PACKET
2045 035566 004737 040316          JSR PC,T9REST          :RESTORE PACKET TO STARTING VALUES
2046
2047 035572          BGNSEG          :>>>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>
          035572 104404          TRAP          C$BSEG
2048
2049 035574 004737 015764          JSR PC,SOFINIT          :DO SOFT INIT OF CONTROLLER
2050 035600 103405          BCS 10$          :BR IF SOFT INIT = OK
2054 035602 010001          MOV R0,R1          :SAVE CONTENTS OF TSSR
2055 035604          ERRDF ERRNO,SFIERR,SFIMSG          :DEVICE FATAL ERROR DURING INIT
          035604 104455          TRAP          C$ERDF
          035606 001611          .WORD          905
          035610 003645          .WORD          SFIERR
          035612 012024          .WORD          SFIMSG
2056 035614 005037 002222 10$: CLR INTRECV          :CLEAR INTERRUPT RECEIVED FLAG
2057 035620 010400          MOV R4,R0          :START OF THE COMMAND PACKET
2058 035622 061300          ADD (R3),R0          :OFFSET TO THE DATA WORD TO TEST
2059 035624 056310 000002          BIS 2(R3),(R0)          :SET THE DATA BITS TO BE TESTED
2060 035630 010465 000000          MOV R4,TSDB(R5)          :SET THE PACKET ADDRESS
2061 035634 004737 016240          JSR PC,WAITF          :WAIT FOR SSR TO SET
2062 035640 103405          BCS 15$          :BR IF CARRY SET (GOOD RETURN)
2063 035642 010001          MOV R0,R1          :SAVE CONTENTS OF TSSR
2067 035644          ERRDF ERRNO,T9SSR,PKTSSR          :DEVICE FATAL SSR FAILED TO SET
          035644 104455          TRAP          C$ERDF
          035646 001612          .WORD          906
          035650 037737          .WORD          T9SSR
          035652 012036          .WORD          PKTSSR
2068 035654          CKLOOP          :LOOP ON ERROR, IF FLAG SET
          035654 104406          TRAP          C$CLP1
2069 035656          ESCAPE SEG          :BY-PASS CHECKS IF FATAL ERROR
          035656 104410          TRAP          C$ESCAPE
          035660 000056          .WORD          10000$-
2070 035662 005737 002222          TST INTRECV          :DID AN INTERRUPT OCCUR ?
2071 035666 001004          BNE 22$          :BRANCH IF YES
2075 035670          ERRHRD ERRNO,T9NINT,PKTSSR
          035670 104456          TRAP          C$ERHRD
          035672 001613          .WORD          907
          035674 040026          .WORD          T9NINT
          035676 012036          .WORD          PKTSSR
2076 035700 016501 000002 22$: MOV TSSR(R5),R1          :GET THE CONTENTS OF TSSR
    
```



```

2097
2098
2099
2100
2101
2102
2103
2104
2105
2106 035760          BGNSUB              ;//////////////// BEGIN SUBTEST //////////////////
      035760          ;                              T9.3:
      035760 104402          TRAP          C$BSUB
2107
2108 035762          SETPRI   #PRI00        ;LOWER PRIORITY TO ALLOW INTERRUPTS
      035762 012700 000000          MOV          #PRI00,R0
      035766 104441          TRAP          C$SPRI
2109 035770 012703 000001          MOV          #1,R3            ;STARTING BYTE COUNT
2110 035774 012704 037220 5$:   MOV          #T9PACKET,R4        ;GET THE ADDRESS OF COMMAND PACKET
2111 036000 004737 040316          JSR          PC,T9REST         ;RESTORE PACKET TO STARTING VALUES
2112
2113 036004          BGNSEG              ;>>>>>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>>>
      036004 104404          TRAP          C$BSEG
2114
2115 036006 004737 015764          JSR          PC,SOFINIT        ;DO SOFT INIT OF CONTROLLER
2116 036012 103405          BCS          10$              ;BR IF SOFT INIT = OK
2120 036014 010001          MOV          R0,R1            ;SAVE CONTENTS OF TSSR
2121 036016          ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      036016 104455          TRAP          C$ERDF
      036020 001615          .WORD       909
      036022 003646          .WORD       SFIERR
      036024 012024          .WORD       SFIMSG
2122 036026 005037 002222 10$:  CLR          INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
2123 036032 010364 000006          MOV          R3,PKBCNT(R4)    ;INSERT THE BYTE COUNT FOR TEST
2124 036036 010465 000000          MOV          R4,TSDB(R5)     ;SET THE PACKET ADDRESS
2125 036042 004737 016240          JSR          PC,WAITF         ;WAIT FOR SSR TO SET
2126 036046 103405          BCS          15$              ;BR IF CARRY SET (GOOD RETURN)
2127 036050 010001          MOV          R0,R1            ;SAVE CONTENTS OF TSSR
2131 036052          ERRDFr  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      036052 104455          TRAP          C$ERDF
      036054 001616          .WORD       910
      036056 037737          .WORD       T9SSR
      036060 012036          .WORD       PKTSSR
2132 036062          15$:  CKLOOP
      036062 104406          ESCAPE   SFG                  ;LOOP ON ERROR, IF FLAG SET
      TRAP          C$CLP1
2133 036064          ESCAPE   SFG                  ;BY-PASS SUBTEST IF FATAL ERROR
      TRAP          C$ESCAPE
      036064 104410          .WORD       10000$-.
      036066 000056          TST          INTRECV          ;DID AN INTERRUPT OCCUR ?
2134 036070 005737 002222          BNE          22$              ;BRANCH IF YES
2135 036074 001004          ERRHRD   ERRNO,T9NINT,PKTSSR
2139 036076          ERRHRD   ERRNO,T9NINT,PKTSSR
      036076 104456          TRAP          C$ERHRD
      036100 001617          .WORD       911
      036102 040026          .WORD       T9NINT
      036104 012036          .WORD       PKTSSR
2140 036106 016501 000002 22$:  MOV          TSSR(R5),R1        ;GET THE CONTENTS OF TSSR
2141 036112 012702 102206          MOV          #SC!SSR!TSREJ!NBA,R2 ;EXPECTED CONTENTS OF TSSR
2142 036116 032701 000100          BIT          #OFL,R1          ;IS OFF-LINE BIT SET ?
  
```



```

2161
2162
2163
2164
2165
2166
2167
2168
2169
2170 036164          BGNSUB          :////////// BEGIN SUBTEST //////////
      036164          :                      T9.4:
      036164 104402          TRAP          C$BSUB
2171
2172 036166          SETPRI  #PRI00          :LOWER PRIORITY TO ALLOW INTERRUPTS
      036166 012700 000000          MOV          #PRI00,R0
      036172 104441          TRAP          C$SPRI
2173 036174 012703 037262          MOV          #T92DATA,R3          :START OF TEST DATA FOR SUBTEST
2174 036200 012704 037220          MOV          #T9PACKET,R4        :GET THE ADDRESS OF COMMAND PACKET
2175 036204 004737 040316          JSR          PC,T9REST          :RESTORE PACKET TO STARTING VALUES
2176
2177
2178 036210 004737 015764          JSR          PC,SOFINIT          :DO SOFT INIT OF CONTROLLER
2179 036214 103405          BCS          10$
2183 036216 010001          MOV          R0,R1          :BR IF SOFT INIT = OK
2184 036220          ERRDF  ERRNO,SFIERR,SFIMSG :SAVE CONTENTS OF TSSR
      036220 104455          :DEVICE FATAL ERROR DURING INIT
      036222 001621          TRAP          C$ERDF
      036224 003646          .WORD          913
      036226 012024          .WORD          SFIERR
2185 036230 005037 002222          .WORD          SFIMSG
2186 036234 052737 000001 037230 10$: CLR          INTRECV          :CLEAR INTERRUPT RECEIVED FLAG
2187 036242 010465 000000          BIS          #1,T9DATA          :MAKE ADDRESS ODD
2188 036246 004737 016240          MOV          R4,TSDB(R5)        :SET THE PACKET ADDRESS
2189 036252 103405          JSR          PC,WAITF          :WAIT FOR SSR TO SET
2190 036254 010001          BCS          15$
2194 036256          MOV          R0,R1          :BR IF CARRY SET (GOOD RETURN)
      036256 104455          ERRDF  ERRNO,T9SSR,PKTSSR      :SAVE CONTENTS OF TSSR
      036260 001622          :DEVICE FATAL SSR FAILED TO SET
      036262 037737          TRAP          C$ERDF
      036264 012036          .WORD          914
2195 036266          .WORD          T9SSR
      036266 104406          .WORD          PKTSSR
2196 036270          CKLOOP          :LOOP ON ERROR, IF FLAG SET
      036270 104410          ESCAPE  SUB          :BY-PASS SUBTEST IF FATAL ERROR
      036272 000056          TRAP          C$CLP1
2197 036274 005737 002222          TRAP          C$ESCAPE
2198 036300 001004          .WORD          L10071-
2202 036302          TST          INTRECV          :DID AN INTERRUPT OCCUR ?
      036302 104456          BNE          22$
      036304 001623          ERRHRD  ERRNO,T9NINT,PKTSSR  :BRANCH IF YES
      036306 040026          TRAP          C$ERHRD
      036310 012036          .WORD          915
2203 036312 016501 000002          .WORD          T9NINT
2204 036316 012702 102206          .WORD          PKTSSR
2205 036322 032701 000100          MOV          TSSR(R5),R1        :GET THE CONTENTS OF TSSR
2206 036326 001402          MOV          #SC!SSR!TSREJ!NBA,R2 :EXPECTED CONTENTS OF TSSR
2207 036330 052702 000100          BIT          #OFL,R1          :IS OFF-LINE BIT SET ?
      BEQ          25$
      BIS          #OFL,R2          :BRANCH IF NOT OFF-LINE
      :SET OFF-LINE IN EXPECTED DATA

```

```
2208 036334 020201          25$:  CMP      R2,R1          ;DOES EXPECTED MATCH RECEIVED ?
2209 036336 001404          BEQ      30$          ;OKAY IF MATCH
2213 036340          ERRHRD  ERRNO,T94REJ,PKTSSR ;COMMAND NOT REJECTED
      036340 104456          TRAP    C$ERHRD
      036342 001624          .WORD  916
      036344 037543          .WORD  T94REJ
      036346 012036          .WORD  PKTSSR
2214 036350          30$:
2215
2216 036350          ENDSUB          ;////////////////// END SUBTEST ////////////////////
      036350          L10071:
      036350 104403          TRAP    C$ESUB
2217
```

```
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228 036352          BGNSUB                  ;////////// BEGIN SUBTEST ////////////
     036352                               T9.5:
     036352   104402                TRAP      C$BSUB
2229
2230 036354          SETPRI   #PRICO          ;LOWER PRIORITY TO ALLOW INTERRUPTS
     036354   012700   000000                MOV      #PRIO0,R0
     036360   104441                                TRAP      C$SPRI
2231 036362   012703   000001                ;STARTING BUFFER LENGTH
2232 036366   012704   037220   5$:      MOV      #T9PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
2233 036372   004737   040316                JSR      PC,T9REST          ;RESTORE PACKET TO STARTING VALUES
2234
2235 036376          BGNSEG                  ;>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>
     036376   104404                                TRAP      C$BSEG
2236
2237 036400   004737   015764                JSR      PC,SOFINIT         ;DO SOFT INIT OF CONTROLLER
2238 036404   103405                                BCS      10$               ;BR IF SOFT INIT = OK
2242 036406   010001                                MOV      R0,R1             ;SAVE CONTENTS OF TSSR
2243 036410          ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
     036410   104455                                TRAP      C$ERDF
     036412   001625                                .WORD    917
     036414   003646                                .WORD    SFIERR
     036416   012024                                .WORD    SFIMSG
2244 036420   005037   002222   10$:      CLR      INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
2245 036424   010337   037234                MOV      R3,T9DATA+4        ;INSERT THE BAD MESSAGE LENGTH
2246 036430   010465   000000                MOV      R4,TSDB(R5)        ;SET THE PACKET ADDRESS
2247 036434   004737   016240                JSR      PC,WAITF          ;WAIT FOR SSR TO SET
2248 036440   103405                                BCS      15$               ;BR IF CARRY SET (GOOD RETURN)
2249 036442   010001                                MOV      R0,R1             ;SAVE CONTENTS OF TSSR
2253 036444          ERRDF   ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
     036444   104455                                TRAP      C$ERDF
     036446   001626                                .WORD    918
     036450   037737                                .WORD    T9SSR
     036452   012036                                .WORD    PKTSSR
2254 036454          CKLOOP                ;LOOP ON ERROR, IF FLAG SET
     036454   104406                                TRAP      C$CLP1
2255 036456          ESCAPE   SEG                ;BY-PASS SUBTEST IF FATAL ERROR
     036456   104410                                TRAP      C$ESCAPE
     036460   000056                                .WORD    10000$.-
2256 036462   005737   002222                TST      INTRECV           ;DID AN INTERRUPT OCCUR ?
2257 036466   001004                BNE      22$               ;BRANCH IF YES
2261 036470          ERRHRD   ERRNO,T9NINT,PKTSSR
     036470   104456                                TRAP      C$ERHRD
     036472   001627                                .WORD    919
     036474   040026                                .WORD    T9NINT
     036476   012036                                .WORD    PKTSSR
2262 036500   016501   000002   22$:      MOV      TSSR(R5),R1        ;GET THE CONTENTS OF TSSR
2263 036504   012702   102206                MOV      #SC!SSR!TSREJ!NBA,R2 ;EXPECTED CONTENTS OF TSSR
2264 036510   032701   000100                BIT      #OFL,R1           ;IS OFF-LINE BIT SET ?
```



```
2283 :+
2284 :
2285 :TEST 9, SUBTEST 6
2286 :
2287 :THIS SUBTEST IS EXECUTED ONLY IF THE EXTENDED
2288 :FEATURES MODE IS ENABLED (AS DETERMINED BY EXAMINING
2289 :XST2 AFTER A PREVIOUS EXECUTION OF WRITE CHARACTERISTICS).
2290 :IT VERIFIES THAT A FIFTH CHARACTERISTICS DATA WORD IS FETCHED
2291 :IF THE BYTE COUNT PARAMETER IN THE COMMAND PACKET IS 10 DECIMAL
2292 :OR GREATER.
2293 :
2294 :-
2295 036556        BGNSUB                                :///// BEGIN SUBTEST /////
036556                                                :      T9.6:
036556 104402
2296 036560 005737 002224       TST     EXTFEA                    :IS EXTENDED FEATURES SOFT. SW SET?
2297 036564 001002                BNE     4$                      :BR, IF SOFTWARE SWITCH IS SET (ON)
2298 036566 000137 037006       JMP     5$                      :EXIT SUBTEST
2299 036572 004737 040316       JSR     PC,T9REST               :SET PACKET TO START-UP VALUES
2300
2301 036576        SETPRI #PRI00                            :LOWER PRIORITY TO ALLOW INTERRUPTS
036576 012700 000000                MOV     #PRI00,R0
036602 104441                TRAP    C$SPRI
2302 036604 012703 002762       MOV     #TSTBLK+10.,R3         :START OF TEST DATA
2303 036610 012704 037220       MOV     #T9PACKET,R4         :GET THE ADDRESS OF COMMAND PACKET
2304 036614 012764 000012 000006 5$: MOV     #10.,PKBCNT(R4)       :START WITH EXTENDED FEATURES VALUE
2305 036622
2306 036622 104404       BGNSEG                                :>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>
036622                                                :      TRAP    C$BSEG
2307
2308 036624 004737 015764       JSR     PC,SOFINIT            :DO SOFT INIT OF CONTROLLER
2309 036630 103405                BCS    10$                     :BR IF SOFT INIT = OK
2313 036632 010001                MOV     R0,R1                 :SAVE CONTENTS OF TSSR
2314 036634                ERRDF  ERRNO,SFIERR,SFIMSG :DEVICE FATAL ERROR DURING INIT
036634 104455                TRAP    C$ERDF
036636 001631                .WORD  921
036640 003646                .WORD  SFIERR
036642 012024                .WORD  SFIMSG
2315 036644 005037 002220       10$: CLR     FATFLG                    :CLEAR FATAL ERROR FLAG
2316 036650 005037 002222       CLR     INTRECV               :CLEAR INTERRUPT RECEIVED FLAG
2317 036654 010465 000000       MOV     R4,TSDB(R5)           :SET THE PACKET ADDRESS
2318 036660 004737 016326       JSR     PC,CHKTSSR           :WAIT FOR SSR TO SET
2319 036664 103407                BCS    15$                     :BR IF CARRY SET (GOOD RETURN)
2320 036666 010001                MOV     R0,R1                 :SAVE CONTENTS OF TSSR
2324 036670                ERRDF  ERRNO,T9SSR,PKTSSR     :DEVICE FATAL SSR FAILED TO SET
036670 104455                TRAP    C$ERDF
036672 001632                .WORD  922
036674 037737                .WORD  T9SSR
036676 012036                .WORD  PKTSSR
2325 036700 005237 002220       15$: INC     FATFLG                    :SET FATAL ERROR FLAG
2326 036704                CKLOOP                          :LOOP ON ERROR, IF FLAG SET
036704 104406
2327 036706                ESCAPE SEG                       :BY-PASS SUBTEST IF FATAL ERROR
036706 104410                TRAP    C$CLP1
036710 000056                TRAP    C$ESCAPE
2328 036712 005737 002222       TST     INTRECV               :DID AN INTERRUPT OCCUR ?
2329 036716 001004                BNE    22$                     :BRANCH IF YES
```



```

2355
2356
2357
2358
2359
2360
2361
2362
2363
2364 037010          BGNSUB          ;////////// BEGIN SUBTEST ////////////
      037010          ;              T9.7:          TRAP      C$BSUB
      037010 104402
2365
2366 037012          SETPRI  #PRI00      ;LOWER PRIORITY TO ALLOW INTERRUPTS
      037012 012700 000000          MOV      #PRI00,R0
      037016 104441          TRAP      C$SPRI
2367 037020 012704 037220          MOV      #T9PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
2368 037024 004737 040316          JSR      PC,T9REST      ;SET UP A VALID PACKET
2369 037030 004737 015764          JSP      PC,SOFINIT      ;DO SOFT INIT OF CONTROLLER
2370 037034 103405          BCS      10$          ;BR IF SOFT INIT = OK
2374 037036 010001          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
2375 037040          ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      037040 104455          TRAP      C$ERDF
      037042 001635          .WORD   925
      037044 003646          .WORD   SFIERR
      037046 012024          .WORD   SFIMSG
2376 037050 005037 002222          10$:  CLR      INTRECV      ;CLEAR INTERRUPT RECEIVED FLAG
2377 037054 052714 000200          BIC      #BIT7,(R4)      ;ENABLE INTERRUPTS
2378 037060 010465 000000          MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS
2379 037064 004737 016326          JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2380 037070 103405          BCS      15$          ;BR IF CARRY SET (GOOD RETURN)
2381 037072 010001          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
2385 037074          ERRDF  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      037074 104455          TRAP      C$ERDF
      037076 001636          .WORD   926
      037100 037737          .WORD   T9SSR
      037102 012036          .WORD   PKTSSR
2386 037104          15$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      037104 104406          TRAP      C$CLP1
2387 037106          ESCAPE  SUB      ;BY-PASS SUBTEST IF FATAL ERROR
      037106 104410          TRAP      C$ESCAPE
      037110 000102          .WORD   L10074-
2388 037112 005737 002222          TST      INTRECV      ;DID AN INTERRUPT OCCUR ?
2389 037116 001004          BNE      22$          ;BRANCH IF YES
2393 037120          ERRHRD  ERRNO,T9NINT,PKTSSR
      037120 104456          TRAP      C$ERHRD
      037122 001637          .WORD   927
      037124 040026          .WORD   T9NINT
      037126 012036          .WORD   PKTSSR
2394 037130          22$:  CKLOOP      ;LOOP ON ERROR ?
      037130 104406          TRAP      C$CLP1
2395
2396 037132 005037 002222          CLR      INTRECV      ;CLEAR INTERRUPT RECEIVED FLAG
2397 037136 042714 000200          BIC      #BIT7,(R4)      ;DISABLE INTERRUPTS
2398 037142 010465 000000          MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS
2399 037146 004737 016326          JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2400 037152 103405          BCS      25$          ;BR IF CARRY SET (GOOD RETURN)
    
```

```

2401 037154 010001          MOV    R0,R1          ;SAVE CONTENTS OF TSSR
2405 037156          ERRDF  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      037156 104455          TRAP  C$ERRDF
      037160 001640          .WORD 928
      037162 037737          .WORD T9SSR
      037164 012036          .WORD PKTSSR
2406 037166          25$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      037166 104406          TRAP  C$CLP1
2407 037170          ESCAPE SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      037170 104410          TRAP  C$ESCAPE
      037172 000020          .WORD L10074-.
2408 037174 005737 002222  TST    INTRECV        ;DID AN INTERRUPT OCCUR ?
2409 037200 001404          BEQ   30$            ;BRANCH IF NOT
2413 037202          ERRHRD ERRNO,T9INT,PKTSSR
      037202 104456          TRAP  C$ERRHRD
      037204 001641          .WORD 929
      037206 040117          .WORD T9INT
      037210 012036          .WORD PKTSSR
2414 037212          30$:
2415 037212          ENDSUB          ;////////////////// END SUBTEST ////////////////////
      037212 104403          L10074:
      037212          TRAP  C$ESUB
2416 037214          EXIT  TST          ;ALL DONE THIS TEST
2417 037214 104432          TRAP  C$EXIT
      037216 001152          .WORD L10065-.
2418
2419          :+
2420          :LOCAL STORAGE FOR THIS TEST
2421          :-
2422
2426 037220          T9PACKET:          ;COMMAND PACKET FOR TEST
2427 037220 100204          .WORD 100204          ;WRITE CHAR COMMAND, WITH IE, ACK
2428 037222 037230          .WORD T9DATA          ;ADDRESS OF CHARACTERISTICS BLOCK
2429 037224 000000          .WORD 0
2430 037226 000010          .WORD 8.            ;STARTING VALUE OF BLOCK SIZE
2431
2432 037230          T9DATA:          ;CHARACTERISTICS DATA BLOCK
2433 037230 037242          .WORD T9BFR          ;ADDRESS OF MESSAGE BUFFER
2434 037232 000000          .WORD 0
2435 037234 000016          .WORD 14.           ;LENGTH OF MESSAGE BUFFER
2436 037236 000000 000000          .WORD 0.0
2437
2438 037242          T9BFR: .BLKW 8.          ;MESSAGE BUFFER
2439
2440          :+
2441          :TEST DATA FOR SUBTEST TWO
2442          :
2443          :DATA HAS FORMAT:
2444          :
2445          :
2446          :      1ST WORD      OFFSET TO TEST WORD IN PACKET
2447          :      2ND WORD      BITS TO SET FOR TEST
2448          :
2449          :
2450          :-
2451 037262          T92DATA:
  
```

2452	037262	000000	037140	.WORD	0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
2453	037266	000002	000001	.WORD	2,BIT0
2454	037272	000004	100100	.WORD	4,BIT6!BIT15
2455		037276			

T92DONE=.

2456
 2457
 2458
 2459
 2460
 2461
 2462
 2463
 2464
 2465
 2466
 2467
 2468
 2469
 2470
 2471
 2472
 2473

:+
 :LOCAL TEXT MESSAGES FOR TEST
 :-

2462	037276	127	122	111	T9NBA:	.ASCIZ	'WRITE CHARACTERISTICS Command Not Accepted'
2463	037351	127	122	111	T92REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
2464	037450	127	122	111	T93REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
2465	037543	127	122	111	T94REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
2466	037641	127	122	111	T95REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
2467	037737	103	157	156	T9SSR:	.ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
2468	040026	105	170	160	T9NINT:	.ASCIZ	'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
2469	040117	125	156	145	T9INT:	.ASCIZ	'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
2470	040206	111	156	143	T9TSBA:	.ASCIZ	'Incorrect TSBA Address After WRITE CHARACTERISTICS'
2471	040271	103	157	155	T9T9ID:	.ASCIZ	'Completion Interrupt'
						.EVEN	

2475
2476
2477
2478
2479
2480
2481
2482 040316
2483 040316
2484 040322 012701 037220
2485 040326 012721 100204
2486 040332 012721 037230
2487 040336 005021
2488 040340 012721 000010
2489 040344 012721 037242
2490 040350 005021
2491 040352 012721 000016
2492 040356 005021
2493 040360 005011
2494 040362 005037 037242
2495 040366 000207
2496 040370
040370
040370 104401

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

T9REST:

SAVREG ;SAVE THE REGISTERS
MOV #T9PACKET,R1 ;START OF THE PACKET
MOV #100204,(R1)+ ;WRITE CHARACTERISTICS WITH ACK, IE
MOV #T9DATA,(R1)+ ;ADDRESS OF CHAR DATA BLOCK
CLR (R1)+ ;EXTENDED ADDRESS
MOV #8,(R1)+ ;SIZE OF DATA BLOCK IN BYTES
MOV #T9BFR,(R1)+ ;ADDRESS OF MESSAGE BUFFER
CLR (R1)+
MOV #14,(R1)+ ;LENGTH OF MESSAGE BUFFER
CLR (R1)+
CLR (R1)
CLR T9BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
RTS PC ;RETURN
ENDTST

L10065: TRAP CSETST


```
2719 041340          ERRHRD  ERRNO,T10MBF,EXPREC      ;MESSAGE BUFFER WAS MODIFIED
      041340 104456
      041342 001770
      041344 042614
      041346 015464
      TRAP      C$ERHRD
      .WORD    1016
      .WORD    T10MBF
      .WORD    EXPREC

2720
2721 041350          70$:
2722 041350 005737 002220
2723 041354 001402
2724 041356 004737 017172
2725 041362          80$:
2726 041362          ENDSEG
      TRAP      C$ESEG
      .WORD    10001$

2727 041364 104405          ENDSUB
      TRAP      C$ESUB
      .WORD    L10077
      .WORD    104403
```



```

2907
2908      ;+
2909      ;TEST 10 SUBTEST 4
2910      ;
2911      ;CHECKS THAT THE REGISTER MODIFICATION REFUSED (RMR) BIT IN
2912      ;THE TSSR WILL BE SET IF A WRITE CHARACTERISTICS COMMAND
2913      ;BEING EXECUTED AND ANOTHER "WC" COMMAND IS ATTEMPTED
2914      ;
2915      ;-
2916      042256      BGNSUB      ;//////////////// BEGIN SUBTEST //////////////////
      042256      104402      T10.4:      TRAP      C$BSUB
2917
2918      042260      004737      043420      JSR      PC,T10RT2      ;SET SECOND PACKET UP
2919      042264      004737      043346      JSR      PC,T10RST      ;SET PACKET TO INITIAL VALUES
2920      042270      012700      000000      SETPRI      #PRI00      ;LOWER PRIORITY TO ALLOW INTERRUPTS
      042274      104441      MOV      #PRI00,R0
      042276      012704      042510      TRAP      C$SPRI
2921      042276      012704      042510      MOV      #T10PACKET,R4      ;GET THE ADDRESS OF COMMAND PACKET
2922      042302      012703      042552      MOV      #T10PKT,R3      ;GET THE ADDRESS OF 2ND CMD PACKET
2923      042306      012764      000010      000006      MOV      #8.,PKBCNT(R4)      ;START WITH MINIMUM ALLOWABLE VALUE
2924      042314      012763      000010      000006      MOV      #8.,PKBCNT(R3)      ;START WITH MINIMUM ALLOWABLE VALUE
2925      042322
2926      042322      104404      5$:      BGNSEG      ;>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>>>
      042322      004737      015764      TRAP      C$BSEG
2927      042324      103405      JSR      PC,SOFINIT      ;DO SOFT INIT OF CONTROLLER
2928      042330      010001      BCS      10$      ;BR IF SOFT INIT = OK
2932      042332      010001      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
2933      042334      104455      ERRDF      ERRNO,SFIERR,SFIMSG      ;DEVICE FATAL ERROR DURING INIT
      042334      002007      TRAP      C$ERDF
      042336      003646      .WORD      1031
      042340      012024      .WORD      SFIERR
      042342      005037      .WORD      SFIMSG
2934      042344      005037      002220      10$:      CLR      FATFLG      ;CLEAR FATAL ERROR FLAG
2935      042350      010465      000000      CLR      INTRECV      ;CLEAR INTERRUPT RECEIVED FLAG
2936      042354      004737      016240      MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS
2937      042360      016501      000002      MOV      R3,TSDB(R5)      ;SECOND COMMAND PACKET
2938      042364      032701      000200      JSR      PC,WAITF      ;WAIT FOR SSR TO SET
2939      042370      001006      104455      MOV      TSSR(R5),R1      ;GET CONTENTS OF TSSR REGISTER
2940      042374      002010      042404      BIT      #SSR,R1      ;CHECK FOR SSR (TSSR) SET
2941      042400      002010      042406      BNE      15$      ;BR, IF SSR SET (GOOD)
2945      042402      104455      ERRDF      ERRNO,T10SSR,PKTSSR      ;DEVICE FATAL SSR FAILED TO SET
      042402      002010      TRAP      C$ERDF
      042404      043050      .WORD      1032
      042410      012036      .WORD      T10SSR
      042412      005237      002220      .WORD      PKTSSR
2946      042412      005237      002220      15$:      INC      FATFLG      ;SET FATAL ERROR FLAG
2947      042416      104406      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      042416      104410      TRAP      C$CLP1
2948      042420      000056      ESCAPE      SEG      ;BY-PASS SUBTEST IF FATAL ERROR
      042422      005737      002222      TRAP      C$ESCAPE
2949      042424      001004      TST      INTRECV      ;DID AN INTERRUPT OCCUR ?
2950      042430      001004      BNE      22$      ;BRANCH IF YES
2951
2952
2956      042432      ERRHRD      ERRNO,T10NINT,PKTSSR
    
```


042432	104456							TRAP	C\$ERHRD
042434	002011							.WORD	1033
042436	043137							.WORD	T10NINT
042440	012036							.WORD	PKTSSR
2957	042442	016501	000002	22\$:	MOV	TSSR(R5),R1			
2958	042446	012702	110200		MOV	#SSR!RMR!SC,R2			
2959	042452	032701	000100		BIT	#OFL,R1			
2960	042456	001402			BEQ	25\$			
2961	042460	052702	000100		BIS	#OFL,R2			
2962	042464	020201		25\$:	CMP	R2,R1			
2963	042466	001404			BEQ	30\$			
2967	042470				ERRHRD	ERRNO,T10SSR,PKTSSR			
	042470	104456							
	042472	002012						TRAP	C\$ERHRD
	042474	043050						.WORD	1034
	042476	012036						.WORD	T10SSR
2968	042500			30\$:				.WORD	PKTSSR
2969	042500				ENDSEG				
	042500								
	042500	104405							
2970	042502				ENDSUB				
	042502								
	042502	104403							
2971	042504				EXIT	TST			
	042504	104432							
	042506	000764							

```

: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 ?
:OKA' IF MATCH
:NB' NOT ZERO

:<<<<<<<<<<<<<<<< END SEGMENT <<<<<<<<<<<<<<<<<<
10000$:
TRAP C$ESEG
:////////// END SUBTEST //////////
L10101:
TRAP C$ESUB
:ALL DONE WITH THIS TEST
TRAP C$EXIT
.WORD L10075-
    
```

```

2973
2974      ;+
2975      ;LOCAL STORAGE FOR THIS TEST
2976      ;-
2980 042510      T10PACKET:      ;COMMAND PACKET FOR TEST
2981 042510 100204      .WORD 100204      ;WRITE CHAR COMMAND, WITH IE, ACK
2982 042512 042520      .WORD T10DATA      ;ADDRESS OF CHARACTERISTICS BLOCK
2983 042514 000000      .WORD 0
2984 042516 000010      .WORD 8.      ;STARTING VALUE OF BLOCK SIZE
2985
2986 042520      T10DATA:      ;CHARACTERISTICS DATA BLOCK
2987 042520 042532      .WORD T10BFR      ;ADDRESS OF MESSAGE BUFFER
2988 042522 000000      .WORD 0
2989 042524 000016      .WORD 14.      ;LENGTH OF MESSAGE BUFFER
2990 042526 000000 000000      .WORD 0.0
2991
2992 042532      T10BFR: .BLKW 8.      ;MESSAGE BUFFER
2993
2994      ;+
2995      ;
2996      ;TEST DATA FOR SUBTEST FOUR
2997      ;
2998 042552      T10PKT:      ;COMMAND PACKET FOR TEST
2999 042552 100204      .WORD 100204      ;WRITE CHAR COMMAND, WITH IE, ACK
3000 042554 042562      .WORD T10DTA      ;ADDRESS OF CHARACTERISTICS BLOCK
3001 042556 000000      .WORD 0
3002 042560 000010      .WORD 8.      ;STARTING VALUE OF BLOCK SIZE
3003
3004 042562      T10DTA:      ;CHARACTERISTICS DATA BLOCK
3005 042562 042574      .WORD T10BUFR      ;ADDRESS OF MESSAGE BUFFER
3006 042564 000000      .WORD 0
3007 042566 000016      .WORD 14.      ;LENGTH OF MESSAGE BUFFER
3008 042570 000000 000000      .WORD 0.0
3009
3010 042574      T10BUFR: .BLKW 8.      ;MESSAGE BUFFER
3011
3012      ;+
3013      ;LOCAL TEXT MESSAGES FOR TEST
3014      ;-
3015
3016
3017 042614      115      145      163      T10MBF: .ASCIZ 'Message Buffer Modified after MESSAGE BUFFER RELEASE Command'
3018 042711      116      102      101      T10NBA: .ASCIZ 'NBA Not Clear After WRITE CHARACTERISTICS Command'
3019 042773      116      102      101      T10NNBA: .ASCIZ 'NBA Set After MESSAGE BUFFER RELEASE Command'
3020 043050      103      157      156      T10SSR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
3021 043137      105      170      160      T10NINT: .ASCIZ 'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
3022 043230      125      156      145      T10INT: .ASCIZ 'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
3023 043317      102      141      163      TST10ID: .ASCIZ 'Basic Packet Protocol'
3024
3025      .EVEN
  
```

3027
 3028
 3029
 3030
 3031
 3032
 3033
 3034 043346
 3035 043346
 3036 043352 012701 042510
 3037 043356 012721 100204
 3038 043362 012721 042520
 3039 043366 005021
 3040 043370 012721 000010
 3041 043374 012721 042532
 3042 043400 005021
 3043 043402 012721 000016
 3044 043406 005021
 3045 043410 005011
 3046 043412 005037 042532
 3047 043416 000207
 3048
 3049
 3050
 3051
 3052
 3053
 3054 043420
 3055 043420
 3056 043424 012701 042552
 3057 043430 012721 100204
 3058 043434 012721 042562
 3059 043440 005021
 3060 043442 012721 000010
 3061 043446 012721 042574
 3062 043452 005021
 3063 043454 012721 000016
 3064 043460 005021
 3065 043462 005011
 3066 043464 005037 042574
 3067 043470 000207
 3068 043472
 043472
 043472 104401

```

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

```

T10RST:
    SAVREG                ;SAVE THE REGISTERS
    MOV #T10PACKET,R1     ;START OF THE PACKET
    MOV #100204,(R1)+     ;WRITE CHARACTERISTICS WITH ACK, IE
    MOV #T10DATA,(R1)+    ;ADDRESS OF CHAR DAT. BLOCK
    CLR (R1)+             ;EXTENDED ADDRESS
    MOV #8,(R1)+         ;SIZE OF DATA BLOCK IN BYTES
    MOV #T10BFR,(R1)+    ;ADDRESS OF MESSAGE BUFFER
    CLR (R1)+
    MOV #14,(R1)+        ;LENGTH OF MESSAGE BUFFER
    CLR (R1)+
    CLR (R1)
    CLR T10BFR           ;CLEAR 1ST LOC IN MESSAGE BUFFER
    RTS PC               ;RETURN
    
```

```

: +
: 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)+
    MOV #14,(R1)+       ;LENGTH OF MESSAGE BUFFER
    CLR (R1)+
    CLR (R1)
    CLR T10BUFR         ;CLEAR 1ST LOC IN MESSAGE BUFFER
    RTS PC              ;RETURN
    ENDTST
    
```

```

L10075:
    TRAP CSETST
    
```



```

3382 044644          EXIT   TST           ;ALL DONE THIS TEST
      044644 104432
      044646 000762          TRAP   C$EXIT
                                   .WORD L10102-.
3383
3384
3385      ;+
3386      ;LOCAL STORAGE FOR THIS TEST
3387      ;-
3391 044650          T11PACKET:        ;COMMAND PACKET FOR TEST
3392 044650 100204          .WORD   100204      ;WRITE CHAR COMMAND, WITH IE, ACK
3393 044652 044660          .WORD   T11DATA     ;ADDRESS OF CHARACTERISTICS BLOCK
3394 044654 000000          .WORD   0
3395 044656 000010          .WORD   8.         ;STARTING VALUE OF BLOCK SIZE
3396
3397 044660          T11DATA:           ;CHARACTERISTICS DATA BLOCK
3398 044660 044672          .WORD   T11BFR     ;ADDRESS OF MESSAGE BUFFER
3399 044662 000000          .WORD   0
3400 044664 000016          .WORD   14.        ;LENGTH OF MESSAGE BUFFER
3401 044666 000000 000000  .WORD   0.0
3402
3403 044672          T11BFR: .BLKW  8.         ;MESSAGE BUFFER
3404
3405
3407          044720          .=<.+10>&177770
3409 044720          T11PK2:           ;COMMAND PACKET FOR TEST
3410 044720 100204          .WORD   100204     ;WRITE CHAR COMMAND, WITH IE, ACK
3411 044722 044730          .WORD   T11DTA     ;ADDRESS OF CHARACTERISTICS BLOCK
3412 044724 000000          .WORD   0
3413 044726 000010          .WORD   8.         ;STARTING VALUE OF BLOCK SIZE
3414
3415 044730          T11DTA:           ;CHARACTERISTICS DATA BLOCK
3416 044730 044742          .WORD   T11BF2     ;ADDRESS OF MESSAGE BUFFER
3417 044732 000000          .WORD   0
3418 044734 000016          .WORD   14.        ;LENGTH OF MESSAGE BUFFER
3419 044736 000000 000000  .WORD   0.0
3420
3421 044742          T11BF2: .BLKW  8.         ;MESSAGE BUFFER
3422
3423
3424
3425      ;+
3426      ;LOCAL TEXT MESSAGES FOR TEST
3427      ;-
3428
3429 044762 111 116 111 T11NBA: .ASCIZ 'INITIALIZE Command Not Accepted'
3430 045022 111 116 111 T112REJ: .ASCIZ 'INITIALIZE Not Rejected With Non-Zero Mode Field'
3431 045103 107 105 124 T113REJ: .ASCIZ 'GET STATUS Not Accepted'
3432 045133 107 05 124 T114REJ: .ASCIZ 'GET STATUS Not Rejected With Non-Zero Mode Field'
3433 045214 103 157 156 T11SSR: .ASCIZ 'Contents of TSSR Incorrect After INITIALIZE'
3434 045270 103 157 156 T11SR2: .ASCIZ 'Contents of TSSR Incorrect After GET STATUS'
3435 045344 105 170 160 T11NINT: .ASCIZ 'Expected Interrupt Not Received On INITIALIZE'
3436 045422 111 156 143 T11TSBA: .ASCIZ 'Incorrect TSBA Address After INITIALIZE'
3437 045472 116 157 156 TST11ID: .ASCIZ 'Non-Tape Motion Commands'
3438
3439          .EVEN
    
```

3441
 3442
 3443
 3444
 3445
 3446
 3447
 3448
 3449 045524
 3450 045524
 3451 045530 012701 044650
 3452 045534 012721 100213
 3453 045540 005021
 3454 045542 005021
 3455 045544 005021
 3456 045546 005021
 3457 045550 005021
 3458 045552 005021
 3459 045554 005021
 3460 045556 005011
 3461 045560 005037 044672
 3462 045564 000207
 3463
 3464
 3465
 3466
 3467
 3468
 3469
 3470 045566
 3471 045566
 3472 045572 012701 044650
 3473 045576 012721 100217
 3474 045602 005021
 3475 045604 005021
 3476 045606 005021
 3477 045610 005021
 3478 045612 005021
 3479 045614 005021
 3480 045616 005021
 3481 045620 005011
 3482 045622 005037 044672
 3483 045626 000207
 3484 045630
 045630
 045630 104401
 3485 045632

```

: +
: ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
: INITIALIZE COMMAND
: -
T11REST:
  SAVREG          ;SAVE THE REGISTERS
  MOV #T11PACKET,R1 ;START OF THE PACKET
  MOV #100213,(R1)+ ;INITIALIZE WITH ACK, IE
  CLR (R1)+        ;ADDRESS OF CHAR DATA BLOCK
  CLR (R1)+        ;EXTENDED ADDRESS
  CLR (R1)+        ;SIZE OF DATA BLOCK IN BYTES
  CLR (R1)+        ;ADDRESS OF MESSAGE BUFFER
  CLR (R1)+        ;LENGTH OF MESSAGE BUFFER
  CLR (R1)+
  CLR (R1)
  CLR T11BFR       ;CLEAR 1ST LOC IN MESSAGE BUFFER
  RTS PC           ;RETURN

```

```

: +
: ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
: GET STATUS COMMAND
: -
T11RT2:
  SAVREG          ;SAVE THE REGISTERS
  MOV #T11PACKET,R1 ;START OF THE PACKET
  MOV #100217,(R1)+ ;GET STATUS WITH ACK, IE
  CLR (R1)+        ;ADDRESS OF CHAR DATA BLOCK
  CLR (R1)+        ;EXTENDED ADDRESS
  CLR (R1)+        ;SIZE OF DATA BLOCK IN BYTES
  CLR (R1)+        ;ADDRESS OF MESSAGE BUFFER
  CLR (R1)+        ;LENGTH OF MESSAGE BUFFER
  CLR (R1)+
  CLR (R1)
  CLR T11BFR       ;CLEAR 1ST LOC IN MESSAGE BUFFER
  RTS PC           ;RETURN

```

ENDMOD

L10102: TRAP CSETST

```

1          .TITLE  TSV6 - PARAMETER CODING
7
12
18
19 045632      BGNMOD  TSV6
   045632      TSV6::
20
21
22          .SBTTL  HARDWARE PARAMETER CODING SECTION
23
24          :++
25          : THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
26          : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
27          : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
28          : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
29          : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
30          : WITH THE OPERATOR.
31          :--
32 045632      BGNHRD
   045632      .WORD  L10107-L$HARD/2
   045634      L$HARD::
33
34 045634      GPRMA  HPM1,0,0,160010,177776,YES      ;GET TSBA/TSDB REGISTER ADDRESS.
   045634      .WORD  T$CODE
   045636      .WORD  HPM1
   045640      .WORD  T$LLOLIM
   045642      .WORD  T$HILIM
35 045644      GPRMA  HPM2,2,0,0,776,YES              ;GET VECTOR ADDRESS.
   045644      .WORD  T$CODE
   045646      .WORD  HPM2
   045650      .WORD  T$LLOLIM
   045652      .WORD  T$HILIM
36          :GPRMD  HPM3,4,0,340,0,7,YES            ;GET INTERRUPT PRIORITY.
37 045654      ENDHRD
   045654      .EVEN
38 045654      104    105    126  HPM1:  .ASCIZ  'DEVICE ADDRESS (TSBA/TSDB) '
39 045710      111    116    124  HPM2:  .ASCIZ  'INTERRUPT VECTOR '
40 045734      111    116    124  HPM3:  .ASCIZ  'INTERRUPT PRIORITY '
41          .EVEN
42

```

```

44                                     .SBTTL SOFTWARE PARAMETER CODING SECTION
45
46
47                                     :++
48                                     : THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
49                                     : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
50                                     : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
51                                     : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
52                                     : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
53                                     : WITH THE OPERATOR.
54                                     :--
54 045764                                BGNSFT
54 045764 000003                        .WORD L10110-L$$SOFT/2
54 045766
55 L$$SOFT::
56 045766                                : GPRML SPM1,0,-1,YES ; GET TRANSPORT TEST FLAG.
56 045766 001130                        : GPRML SPM4,2,-1,YES ; GET ITERATION CONTROL.
56 045770 046024                        .WORD T$CODE
56 045772 177777                        .WORD SPM4
57                                     : .WORD -1
58                                     : GPRMD SPM6,4,D,7777,0,7777,YES ; GET LOCAL ERROR LIMIT
58                                     : GPRMD SPM7,6,D,7777,0,7777,YES ; GET GLOBAL ERROR LIMIT
59 045774                                ENDSFT
60                                     .EVEN
61 L10110:
62 045774 105 116 101 SPM1: .ASCIZ 'ENABLE TRANSPORT TESTS '
63 046024 111 116 110 SPM4: .ASCIZ 'INHIBIT ITERATIONS '
64 046054 120 105 122 SPM6: .ASCIZ 'PER TEST ERROR LIMIT '
65 046104 120 105 122 SPM7: .ASCIZ 'PER UNIT ERROR LIMIT '
66                                     .SBTTL PATCH AREA
67
68
69                                     :
70                                     : FINALLY A GENEROUS PATCH AREA.
71                                     :
72                                     : AND AN ADJUSTMENT TO ACCOUNT FOR THE 'LASTAD BIT7' HACK
73                                     : DESCRIBED IN 'SUPPRG.MEM' (FOR REV C).
74                                     :
75 046134                                PATCH::
76                                     .BLKW 32.
77 046134
78                                     .=.!377+1
80 046400 046400                        LASTAD ;SET LAST USED ADDRESS.
82 046400                                .EVEN
82 046400 000000                        .WORD 0
82 046402 000000                        .WORD 0
83 046404                                L$LAST::
83 046404                                ENDMOD
84 046404 000001                        .END
    
```

ADSSR 012116 G	C\$AU = 000052	DEVDR0 023366	FRESIZ 003122 G	INTFLA 016135
ADR = 000020 G	C\$AUTO= 000061	DEVNRD 023305	FUSI 004113	INTMAS 016134
AMBTSS 006631	C\$BRK = 000022	DEVNXR 023223	F\$AU = 000015	INTR 016206 G
ASSEMB= 000010	C\$BSEG= 000004	DEVONL 023153	F\$AUTO= 000020	INTREC 002222 G
A1716 = 000003	C\$BSUB= 000002	DEVSUM 023116	F\$BGN = 000040	INTVEC 016136
BADDAT 003152 G	C\$CEFG= 000045	DFPTBL 002154 G	F\$CLEA= 000007	INTX 004274
BADSSR 015670 G	C\$CLCK= 000062	DIAGMC= 000000	F\$DU = 000016	INVERT 021176 G
BDVPCR= 177520 G	C\$CLEA= 000012	DICEA = 000001	F\$END = 000041	IOKCKI= 000200
BENBSW 002226 G	C\$CLOS= 000035	DSBINT 016174	F\$HARD= 000004	IOKSTP= 000001
BIE = 040000	C\$CLP1= 000006	DUAD12 004637	F\$HW = 000013	IPRI 002210 G
BIT0 = 000001 G	C\$CVEC= 000036	DUFLG 003106 G	F\$INIT= 000006	ISR = 000100 G
BIT00 = 000001 G	C\$DCLN= 000044	DUMMY 003056	F\$JMP = 000050	IVEC 002206 G
BIT01 = 000002 G	C\$DODU= 000051	EF.CON= 000076 G	F\$MOD = 000000	IXE = 004000 G
BIT02 = 000004 G	C\$DRPT= 000024	EF.NEW= 000033 G	F\$MSG = 000011	ISAU = 000041
BIT03 = 000010 G	C\$DU = 000053	EF.PWR= 000034 G	F\$PROT= 000021	ISAUTO= 000041
BIT04 = 000020 G	C\$EDIT= 000003	EF.RES= 000037 G	F\$PWR = 000017	ISCLN = 000041
BIT05 = 000040 G	C\$ERDF= 000055	EF.STA= 000040 G	F\$RPT = 000012	ISDU = 000041
BIT06 = 000100 G	C\$ERHR= 000056	EMAXDU 016767	F\$SEG = 000003	ISHRD = 000041
BIT07 = 000200 G	C\$ERRO= 000060	EN = 000000	F\$SOFT= 000005	ISINIT= 000041
BIT08 = 000400 G	C\$ERSF= 000054	ENAINI 016142	F\$SRV = 000010	ISMOD = 000041
BIT09 = 001000 G	C\$ERSO= 000057	ENVIRN 020620	F\$SUB = 000002	ISMSG = 000041
BIT1 = 000002 G	C\$ESCA= 000010	EPRTSW 002176 G	F\$SW = 000014	ISPROT= 000040
BIT10 = 002000 G	C\$ESEG= 000005	EPRT1 006354	F\$TEST= 000001	ISPTAB= 000041
BIT11 = 004000 G	C\$ESUB= 000003	EPRT2 006354	GDDAT 003154 G	ISPWR = 000041
BIT12 = 010000 G	C\$ETST= 000001	ERCM 011723	GERRMA 002172 G	ISRPT = 000041
BIT13 = 020000 G	C\$EXIT= 000032	ERRHI 002234 G	GETPAT 020164 G	ISSEG = 000041
BIT14 = 040000 G	C\$GETB= 000026	ERRK 016746	GETSEL 020246 G	ISSETU= 000041
BIT15 = 100000 G	C\$GETW= 000027	ERRLO 002236 G	G\$CNT0= 000200	ISSFT = 000041
BIT2 = 000004 G	C\$GMAN= 000043	ERRNO = 002144	G\$DELM= 000372	ISSRV = 000041
BIT3 = 000010 G	C\$GPHR= 000042	ERRVEC= 000004 G	G\$DISP= 000003	ISSUB = 000041
BIT4 = 000020 G	C\$GPLO= 000030	ERTABE 003372	G\$EXCP= 000400	ISTST = 000041
BIT5 = 000040 G	C\$GPRI= 000040	ERTABL 003172	G\$HILI= 000002	JSJMP = 000167
BIT6 = 000100 G	C\$INIT= 000011	ESUM 016750	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 015472 G	G\$OFSI= 000376	KIPAR3= 172346
BOE = 000400 G	C\$MSG = 000023	EXPD 002230 G	G\$PRMA= 000001	KIPAR4= 172350
BRINIT 004453	C\$OPEN= 000034	EXPBOT 004527	G\$PRM0= 000002	KIPAR5= 172352
BSELO = 000000	C\$PNTB= 000014	EXPBT2 004563	G\$PRML= 000000	KIPAR6= 172354
BSEL1 = 000001	C\$PNTF= 000017	EXPMSG 002320 G	G\$RADA= 000140	KIPAR7= 172356
CHKAMB 016034	C\$PNTS= 000016	EXPREC 015464 G	G\$RADB= 000000	KIPDR0= 172300
CHKMAN 020470 G	C\$PNTX= 000015	EXTA 005766	G\$RADD= 000040	KIPDR1= 172302
CHKTSS 016326	C\$QIO = 000377	EXTEND 005764	G\$RADL= 000120	KIPDR2= 172304
CKDROP 017172	C\$RDBU= 000007	EXTFEA 002224 G	G\$RADO= 000020	KIPDR3= 172306
CKEMAX 017072	C\$REFG= 000047	E\$END = 002100	G\$XFER= 000004	KIPDR4= 172310
CKMSG 011350 G	C\$RESE= 000033	E\$LOAD= 000035	G\$YES = 000010	KIPDR5= 172312
CKMSG2 011470 G	C\$REVI= 000003	FATERR= 000060	HIADDR= 001400	KIPDR6= 172314
CKRAM 011104 G	C\$RFLA= 000021	FATFLG 002220 G	HOE = 100000 G	KIPDR7= 172316
CKRAM2 011214 G	C\$RPT = 000025	FERCM 011712	HPM1 045654	KTENAB 003130 G
CMDPKT 021250 G	C\$SEFG= 000046	FIFEXP 012160 G	HPM2 045710	KTF LG 003126 G
CMPMEM 017650	C\$SPRI= 000041	FIF1MS 012232	HPM3 045734	KTINIT 021044
CONFIG 017240	C\$SVEC= 000037	FIF2MS 012301	IBE = 010000 G	KTOFF 017264
COUNT 002306 G	C\$TPRI= 000013	FILLME 017412	IDU = 000040 G	KTON 017246
CSRADD 002204 G	DATA 002310 G	FNOINT 004211	IER = 020000 G	LERRMA 002170 G
CTAB 003160 G	DATASC 020222	FORCER 002174 G	IFAU LT 004252	LISTAL= 000001
CTABE 003172 G	DEBUGM 011622	FREE 003120 G	INCERK 017034	LOE = 040000 G
CTABM 003160 G	DEV CNT 002216 G	FREEHI 003124	INTCPC 016140	LOOPCN 002214 G

SYMBOL TABLE									
LOOPCO	013116	L10001	002174	L10073	037006	NXR	003734	PRI05	= 000240 G
LOOPFL	003156 G	L10002	005762	L10074	037212	NXRERR	005732 G	PRI06	= 000300 G
LOT	= 000010 G	L10003	012034	L10075	043472	NXRX	003773	PRI07	= 000340 G
LSACP	002110 G	L10004	012052	L10076	040772	NXTU	022012	PRMESS	014232
LSAPT	002036 G	L10005	012070	L10077	041364	OFL	= 000100	PRMNO	002316 G
LSAU	022342 G	L10006	012076	L10100	042254	ONEFIL	= 000000	PRMSGE	014542 G
LSAUT	002070 G	L10007	012114	L10101	042502	OSAPTS	= 000000	PRMSG0	014722
LSAUTO	022546 G	L10010	012132	L10102	045630	OSAU	= 000001	PRMSG1	014767
LSCCP	002106 G	L10011	012156	L10103	043740	OSBGNR	= 000001	PRMSG2	015025
LSCLEA	022626 G	L10012	012230	L10104	044176	OSBGNS	= 000001	PROASC	014410
LSCO	002032 G	L10013	012400	L10105	044416	OSDU	= 000001	PR1ASC	014455
LSDEPO	002011 G	L10014	013114	L10106	044642	OSERRT	= 000000	PST32W	003146 G
LSDESC	003404 G	L10015	013742	L10107	045654	OSGNSW	= 000001	PUNIT	022274
LSDESP	002076 G	L10016	013764	L10110	045774	OSPOIN	= 000001	PW.D11	= 000021
LSDEVP	002060 G	L10017	015470	MEMADD	013744 G	OSSETU	= 000000	PW.D13	= 000022
LSDISP	002124 G	L10020	015476	MEMCK	021266 G	PASRPT	022044	PW.D22	= 000020
LSDLY	002116 G	L10021	015504	MENASC	020437	PATCH	046134 G	PW.NOP	= 000000
LSDTP	002040 G	L10022	015516	MENERR	020364	PATDAT	020220	PW.NO1	= 000023
LSDTYP	002034 G	L10023	015540	MENRES	020466	PC.ERA	= 002400	PW.RDE	= 000024
LSDU	022440 G	L10024	015566	MMVEC	= 000250	PC.IER	= 002000	PW.RDR	= 000001
LSDUT	002072 G	L10025	015726	MSA.FR	= 000006	PC.NOO	= 001000	PW.RDS	= 000005
LSDVTY	003376 G	L10026	016236	MSA.NO	= 000000	PC.REL	= 000000	PW.RFI	= 000003
LSEF	002052 G	L10030	022272	MSA.NR	= 000004	PC.REW	= 000400	PW.WCT	= 000006
LSENVI	002044 G	L10031	022436	MSA.VO	= 000002	PKBCNT	= 000006	PW.WFI	= 000004
LSETP	002102 G	L10032	022544	MSGEXP	012134 G	PKHI	= 000004	PW.WFM	= 000007
LSEXP1	002046 G	L10033	022624	MSGLOO	013054 G	PKLOW	= 000002	PW.WMI	= 000010
LSEXP4	002064 G	L10034	022652	MSGSTA	012340 G	PKTADD	007550	PW.WNP	= 000011
LSEXP5	002066 G	L10035	023114	MSGSUB	013732 G	PKTFRM	007512	PW.WTR	= 000002
LSHARD	045634 G	L10036	023654	MS.ATT	= 000006	PKTGET	012054 G	P.ACK	= 100000
LSHIME	002120 G	L10037	023522	MS.EXT	= 000200	PKTMES	012100 G	P.CMD	= 000037
LSHPCP	002016 G	L10040	023604	MS.RSD	= 000001	PKTRAM	004741 G	P.CONT	= 000012
LSHPTP	002022 G	L10041	024352	MS.RSF	= 000020	PKTSSR	012036 G	P.CVC	= 040000
LSHW	002154 G	L10042	025044	MS.RST	= 000010	PNT	= 001000 G	P.FMT	= 000140
LSICP	002104 G	L10043	026400	M2901	026124	PRAMPK	013766	P.FORM	= 000011
LSINIT	021546 G	L10044	025306	M8186	005550	PRASC	014513	P.GETS	= 000017
LSLADP	002026 G	L10045	025606	M8189	005641	PRBEXP	015460	P.IE	= 000200
LSLAST	046404 G	L10046	026104	NBA	= 002000	PRBMSG	015326	P.INIT	= 000013
LSLOAD	002100 G	L10047	027504	NEWPAS	022000	PRBREC	015462	P.MODE	= 007400
LSLUN	002074 G	L10050	026752	NODEV	003110 G	PRBTOT	015413	P.OPP	= 020000
LSMREV	002050 G	L10051	027322	NOINIT	004331	PRBYTE	015112 G	P.POSI	= 000010
LSNAME	002000 G	L10052	030762	NOINTR	004215	PRI	= 002000 G	P.READ	= 000001
LSPRIO	002042 G	L10053	030034	NOITS	002166 G	PRIADD	010154	P.SWB	= 010000
LSPROT	021536 G	L10054	030360	NOMAN	020524	PRIAO	010224	P.WRIT	= 000005
LSPRT	002112 G	L10055	034350	NOMEM	005454	PRI BXO	007606 G	P.WRTC	= 000004
LSREPP	002062 G	L10056	031314	NP.IR	= 000200	PRIEQU	010054	P.WRTS	= 000006
LSREV	002010 G	L10057	031600	NP.LOO	= 000040	PRIPKT	007364 G	QVP	002202 G
LSRPT	02265 G	L10060	032044	NP.OUT	= 000100	PRIRAM	010062	RAMASC	014146
LSOFT	045766 G	L10061	032272	NP.WRP	= 000020	PRITAD	010270	RAMDAT	002240 G
LSGPC	002056 G	L10062	032536	NSI	004146	PRITSS	006020	RAMERR	015500 G
LSGPCP	002020 G	L10063	033076	NSINIT	004403	PRITO	010352	RAMEXP	015520 G
LSPTP	002024 G	L10064	035254	NUL	004523	PRITI	010415	RAMFOR	010112
LSSTA	002030 G	L10065	040370	NULCR	004524	PRIXOR	007736 G	RAMSIZ	002300 G
LSW	002164 G	L10066	035516	NXM	= 004000	PRI00	= 000000 G	RAMTAD	015506 G
LSTEST	002114 G	L10067	035756	NXMFLG	003132 G	PRI01	= 000040 G	RCVHIA	002302 G
LSTIML	002014 G	L10070	036162	NXMHI	003136 G	PRI02	= 000100 G	RCVLOA	002304 G
LSUNIT	002012 G	L10071	036350	NXMLO	003134 G	PRI03	= 000140 G	RDERR	005202
L10000	002162	L10072	036554	NXMTST	021442	PRI04	= 000200 G	RECMG	002464 G

RECV	002232	G	S1.IEO=	010000	TST6ID	030743	T1.1	023456	T4	025046	G			
REGSAV	020130		S1.IFM=	001000	TST7ID	034253	T1.2	023536	T4LOOP	025060				
RETERR	005366		S1.IHE=	000400	TST8ID	035237	T10	040372	G	T4.1	025046			
REWIND	011004	G	S1.IID=	004000	TST9ID	040271	T10BFR	042532		T4.2	025310			
RMCHBE=	000167		S1.IIR=	020000	TSV2	002000	G	T10BUF	042574		T4.3	025610		
RMCHEN=	000200		S1.I2R=	040000	TSV3	002174	G	T10DAT	042520		T5	026402	G	
RMM3GB=	000215		S1.PAR=	100000	TSV4	021536	G	T10DTA	042562		T5ADDR	027442		
RMM3GE=	000234		S2.ATI=	000010	TSV5	023436	G	T10INT	043230		T5LOOP	026420		
RMPKTB=	000201		S2.BTI=	000004	TSV6	045632	G	T10LOO	040410		T5MEM	027404		
RMPKTE=	000210		S2.DIM=	000200	TTIBFR=	177562	G	T10MBF	042614		T5.1	026424		
RMR	=	010000	S2.ILW=	000100	TTICSR=	177560	G	T10NBA	042711		T5.2	026772		
RWPACK	011100		S2.INR=	000020	TTIVEC=	000060	G	T10NIN	043137		T6	027506	C	
SC	=	100000	S2.OUT=	000040	TSARGC=	000003		T10NNB	042773		T6INT	030533		
SCE	=	020000	S2.UND=	000003	TSCODE=	001130		T10PAC	042510		T6LOOP	027524		
SCHERR	005274		TBLEND=	003056	TSEARN=	002144		T10PKT	042552		T6NBA	030430		
SCME	005007		TCOASC	006472	TSEXCP=	000000		T10RST	043346		T6NINT	030611		
SDELAY	010650		TCOCOD	006672	T\$FLAG=	000040		T10RT2	043420		T6PACK	030420		
SELASC	020432		TEMP1	003112	G	T\$GMAN=	000000		T10SSR	043050		T6SSR	030455	
SELDAT=	000004		TEMP2	003114	G	T\$HILI=	000776		T10.1	040410		T6TSBA	030671	
SEL2	=	000002	TERCLS=	000016		T\$LAST=	000001		T10.2	040774		T6.1	027524	
SETMAP	017306		TESTNO=	000013		T\$LOLI=	000000		T10.3	041366		T6.2	030050	
SETU	022076		TEXASC	006431		T\$LSYM=	010000		T10.4	042256		T7	030764	G
SFFMSG	012072	G	TFCASC	006533		T\$LTNO=	000013		T11	043474	G	T7BFR	033146	
SFHERR	003701		TIMEXP	015542	G	T\$NEST=	177777		T11BFR	044672		T7DATA	033130	
SFIERR	003646		TIMSGO	015570		T\$NSO =	000000		T11BF2	044742		T7INT	034101	
SFIMSG	012024	G	TINERR	012011		T\$NS1 =	000005		T11DAT	044660		T7LOOP	031002	
SFPTBL	002164	G	TMPBFR	002630	G	T\$NS2 =	000002		T11DTA	044730		T7NBA	033260	
SIFLAG	003150	G	TNAM	016674		T\$NS3 =	000003		T11LOO	043512		T7NINT	034010	
SIMSG	011756		TRANST	002164	G	T\$PTNU=	000000		T11NBA	044762		T7PACK	033120	
SKIPT	003374		TSBA =	000000	G	T\$SAVL=	177777		T11NIN	045344		T7REST	034302	
SOFINI	015704	G	TSBAH =	000001	G	T\$SEGL=	177777		T11PAC	044650		T7SP	033140	
SPACE	010462	G	TSBAM2	026214		T\$SEKO=	010000		T11PK2	044720		T7SSR	033721	
SPM1	045774		TSBAM3	026276		T\$SUBN=	000004		T11RES	045524		T7TSBA	034170	
SPM4	046024		TSDB =	000000	G	T\$TAGL=	177777		T11RT2	045566		T7.1	031002	
SPM6	046054		TSDBH =	000001	G	T\$TAGN=	010111		T11SR2	045270		T7.2	031330	
SPM7	046104		TSFCOD	007232		T\$TEMP=	000000		T11SSR	045214		T7.3	031602	
SRO	=	177572	TSREJ =	000006		T\$TEST=	000013		T11TSB	045422		T7.4	032046	
SR1	=	177574	TSSDEF	006602		T\$STM=	177777		T11.1	043512		T7.5	032274	
SR2	=	177576	TSSR =	000002	G	T\$STSS=	000001		T11.2	043754		T7.6	032540	
SR3	=	172516	TSSRBI	003476	G	T\$SAU =	010031		T11.3	044200		T72DAT	033166	
SSR	=	000200	TSSRFO	006411		T\$SAUT=	010033		T11.4	044420		T72DON=	033202	
STATCO	012402		TSSRH =	000003	G	T\$SCLE=	010034		T112RE	045022		T72NBA	033202	
SVCGBL=	000000		TSSX	004014		T\$SDU =	010032		T113RE	045103		T72REJ	033333	
SVCINS=	000000		TSTBLK	002750	G	T\$SHAR=	010107		T114RE	045133		T73REJ	033432	
SVCSUB=	000001		TSTCNT	002212	G	T\$SHW =	010000		T2	023656	G	T74REJ	033525	
SVCTAG=	000000		TSTEND	016710		T\$SINI=	010030		T2LOOP	023674	G	T75REJ	033623	
SVCTST=	000001		TSTFLA	002312	G	T\$MSG=	010025		T2SSR	024252		T8	034352	G
S\$LSYM=	010000		TSTL00	016446	G	T\$PRO=	010027		T2TSBA	024140		T8BFR	035002	
SO.IDB=	000010		TSTPTR	002314	G	T\$RPT=	010035		T2TSSR	024205		T8DATA	034770	
SO.IFB=	000002		TSTSET	016500	G	T\$SEGE=	010000		T23A	003140	G	T8LOOP	034370	
SO.IFP=	000001		TST1ID	023634		T\$SOF=	010110		T23B	003142	G	T8NVCK	035057	
SO.ILD=	000020		TST101	043317		T\$SRV=	010026		T3	024354	G	T8PACK	034760	
SO.ION=	000040		TST111	045472		T\$SUB=	010106		T3BFLG	003144	G	T8SSR	035150	
SO.IRD=	000100		TST2ID	024324		T\$SW =	010001		T3LOOP	024372		T8VCK	035022	
SO.IRW=	000004		TST3ID	025017		T\$TES=	010102		T3SSR	024746		T9	035256	G
SO.ISP=	000200		TST4ID	026356		T1	023436	G	T3TSBA	024636		T9BFR	037242	
SI.ICE=	002000		TST5ID	027354		T1LOOP	023454		T3TSSR	024702		T9DATA	037230	

T9INT	040117	UAM	= 000200 G	WRTCHR	010652 G	XSOONL	= 000100	X2.EXT	= 000200
T9LOOP	035300	UNITN	002200 G	WRTERR	005107	XSOPEL	= 000010	X2.OPM	= 100000
T9NBA	037276	UNREC	= 000006	WRTMSG	005052	XSORLL	= 010000	X2.RCE	= 040000
T9NINT	040026	USI	004117	WSMBK	021260 G	XSORLS	= 040000	X2.REV	= 000077
T9PACK	037220	WAITF	016240 G	XFERAS	015730	XSOVMK	= 100000	X2.SPA	= 035400
T9REST	040316	WC.IFA	= 000200	XNXM	016366	XSOVCK	= 000020	X2.UNI	= 000007
T9SSR	037737	WC.IFE	= 000002	XORBFO	007670	XSOWLE	= 004000	X2.WCF	= 002000
T9TSBA	040206	WC.IGO	= 000001	XORFOR	010006	XSOWLK	= 000004	X3.DCK	= 000010
T9.1	035300	WC.IRE	= 000010	XST0	= 000006 G	XXCOMM	003116 G	X3.MBZ	= 000006
T9.2	035546	WC.IRW	= 000004	XST1	= 000010 G	XSALWA	= 000000	X3.MDE	= 177400
T9.3	035760	WC.IOT	= 000100	XST2	= 000012 G	XSALS	= 000040	X3.OPI	= 000100
T9.4	036164	WC.IIT	= 000040	XST3	= 000014 G	XSOFFS	= 000400	X3.REV	= 000040
T9.5	036352	WC.ISR	= 000020	XST4	= 000016 G	XSTRUE	= 000020	X3.RIB	= 000001
T9.6	036556	WF.IED	= 000010	XSOBOT	= 000002	X1.COR	= 020000	X3.SPA	= 000200
T9.7	037010	WF.IER	= 000004	XSOEOT	= 000001	X1.DLT	= 100000	X3.TRF	= 000020
T92DAT	037262	WF.IHI	= 000200	XSOIE	= 000040	X1.MBZ	= 017375	X4.HSP	= 100000
T92DON	= 037276	WF.IRE	= 000040	XSOILA	= 000400	X1.RBP	= 000400	X4.MBZ	= 017400
T92REJ	037351	WF.IWF	= 000020	XSOILC	= 001000	X1.SPA	= 040000	X4.RCE	= 040000
T93REJ	037450	WF.IWR	= 000100	XSOLET	= 020000	X1.UNC	= 000002	X4.TSM	= 020000
T94REJ	037543	WF.I3R	= 000002	XSOMOT	= 000200	X2.BUF	= 000100	X4.WRC	= 000377
T95REJ	037641	WF.I4R	= 000001	XSONEF	= 002000				

. ABS. 046404 000
 000000 001
 ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 28224 WORDS (111 PAGES)
 DYNAMIC MEMORY: 20346 WORDS (78 PAGES)
 ELAPSED TIME: 00:34:19
 CVTSA0, CVTSA0/-SP=SVC/ML, TSV1A, TSV22A, TSV3A, TSV4, TSV5A, TSV6