

B1

iv wto v?
EQ 0001
USER DOCUMENTATION

MACRO M1200 23 MAR 84 09:44 PAGE 2

S

.REM

IDENTIFICATION

PRODUCT ID: AC-T814A-MC
PRODUCT TITLE: CNTSAAO TSV05 CTRL LT1
DECO/DEPO: 1.0
DEPARTMENT: ISS/DIAGNOSTIC SERVICES
DATE: APRIL 09, 1984

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) 1984 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

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 SBC-11/21, RESIDENT DIAGNOSTIC WHICH CHECKS THE FUNCTIONALITY OF A TSV05 MAGTAPE SUBSYSTEM WHILE CONNECTED TO A SBC-11/21, SYSTEM (Q-BUS). THE PROGRAM PROVIDES ERROR MESSAGES WHICH IDENTIFY FAILING FUNCTIONS THAT AID IN THE REPAIR OF THE DEVICE. THIS DIAGNOSTIC CONSISTS 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

SBC-11/21, 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 (MSAAA.SYS VERSION 34 OR LATER)
PDP-11 DIAGNOSTIC LOADER/MONITOR (XXDP+)

1.3 RELATED DOCUMENTS AND STANDARDS

DIGITAL EQUIPMENT CORPORATION DOCUMENTS:

1. XXDP+ USERS GUIDE
2. TSV05 TRANSPORT SUBSYSTEM USER'S GUIDE
3. TSV05 TRANSPORT SUBSYSTEM TECHNICAL MANUAL
4. TSV05 TRANSPORT SUBSYSTEM INSTALLATION MANUAL

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

FUNCTIONAL SBC-11/21, 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.

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 SBC-11/21+ DIAGNOSTIC SUPERVISOR COMPATIBLE
PROGRAM. ALL LOADING AND RUNTIME INSTRUCTIONS CAN BE REFERENCED IN THE
XXDP+ USERS GUIDE. THE USER ENTRY IS IN QUOTES.

BOOT THE DIAGNOSTIC MEDIA

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

2.2 SWITCHES

THERE ARE SEVERAL SWITCHES WHICH ARE USED TO MODIFY SUPERVISOR OPERATION.

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

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

EXAMPLE OF SWITCH USAGE:

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

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

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

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

2.3 FLAGS

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

CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

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

*ERROR MESSAGES ARE DESCRIBED IN SECTION 3.1

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

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

2.4 HARDWARE QUESTIONS

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

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

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

```
TSBA/TSDB * 176000, 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 47196 CONTROLLERS
PRESENT TO BE TESTED>

UNIT 0

DEVICE ADDRESS (O) 176000 ? <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.

0 UNITS (0) ? 8<CR>

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

AS YOU CAN SEE IN THE ABOVE DIALOGUE, THE RUNTIME SERVICES WILL BUILD AS MANY ENTRIES AS IT CAN WITH THE INFORMATION GIVEN IN ANY ONE PASS THROUGH THE QUESTIONS. IN THE FIRST PASS, TWO ENTRIES ARE BUILT SINCE TWO SUB-DEVICES AND Q-FACTORS WERE SPECIFIED. THE SERVICES ASSUME THAT THE CSR ADDRESS IS 160000 FOR BOTH SINCE IT WAS SPECIFIED ONLY ONCE. IN THE SECOND PASS, FOUR ENTRIES WERE BUILT. THIS IS BECAUSE FOUR SUB-DEVICES WERE SPECIFIED. THE "-" CONSTRUCT TELLS THE RUNTIME SERVICES TO INCREMENT THE DATA FROM THE FIRST NUMBER TO THE SECOND. IN THIS CASE, SUB-DEVICES 2, 3, 4 AND 5 WERE SPECIFIED. (IF THE SUB-DEVICE WERE SPECIFIED BY ADDRESSES, THE INCREMENT WOULD BE BY 2 SINCE ADDRESSES MUST BE ON AN EVEN BOUNDARY.) THE CSR ADDRESSES AND Q-FACTORS FOR THE FOUR ENTRIES ARE ASSUMED TO BE 160000 AND 0 RESPECTIVELY SINCE THEY WERE ONLY SPECIFIED ONCE. THE LAST TWO UNITS ARE SPECIFIED IN THE THIRD PASS.

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

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

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

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

2.7 QUICK START UP PROCEDURE (XXDP+)

TO START-UP THIS PROGRAM:

1. BOOT XXDP.
2. TYPE "R NAME", WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM
3. TYPE "START"
4. ANSWER THE "CHANGE HW" QUESTION WITH "Y"
5. ANSWER ALL THE HARDWARE QUESTIONS
6. ANSWER THE "CHANGE SW" QUESTION WITH "N"

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

3.0 ERROR INFORMATION

3.1 TYPES OF ERROR MESSAGES

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

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

WHERE: NAME = DIAGNOSTIC NAME
 TYPE = ERROR TYPE (SYS FATAL, DEV FATAL, HARD OR SOFT)
 NUMBER = ERROR NUMBER
 UNIT NUMBER = 0 - N (N IS LAST UNIT IN TABLE)
 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
 CNTSA 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.

CNTSA HRD ERR 00159 ON UNIT 00 TST 001 SUB 005 PC: 026202
 TSSR NOT CORRECT AFTER SPACE RECOFDS 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.

CNTS HRD ERR 00121 ON UNIT 00 TST 001 SUB 002 PC: 023306

MOT BIT (XSTO) 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 (D) 176000 ? <CR>
VECTOR (D) 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 THE FALCON PROCESSOR.

THE PROGRAM RUNS IN TWO MODES; NO ITERATIONS AND DEFAULT MODE. IN THE NO ITERATIONS MODE, EACH TEST IS RUN ONCE, WITH NO ITERATIONS. IN THE DEFAULT MODE EACH TEST IS REPEATED BY THE NUMBER OF TIMES INDICATED BY

THE ITERATION COUNT. NO ITERATIONS MODE IS SELECTED BY ANSWERING THE INHIBIT ITERATIONS QUESTION WITH A "Y" (YES).

TIMES REQUIRED TO RUN TESTS 1 THROUGH 12:

Q.V. 2 MIN 15 SECONDS
DEFAULT 16 MINS

MORE EXHAUSTIVE CHECKS ARE AVAILABLE BY ALLOWING THE DIAGNOSTIC PROGRAMS TO RUN FOR MORE THAN ONE PASS. THE SECOND PASS OF THE PROGRAM IS MORE COMPREHENSIVE THAN THE FIRST PASS. ALL ITERATIONS AFTER THE FIRST PASS ARE THE SAME, HOWEVER, THEY ARE SUBSTANTIALLY LONGER.

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) 176000 ? <ENTER THE ADDRESS OF THE
TSBA/TSD8 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 11/21 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 11/21+ 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 XSTO, IS SET BY INITIALIZE AND CLEARED BY EXECUTING A WRITE CHARACTERISTICS COMMAND WITH THE CVC BIT SET. IT IS ALSO VERIFIED THAT A WRITE CHARACTERISTICS COMMAND WITH THE CVC BIT CLEAR DOES NOT AFFECT THE STATE OF THE VOLUME CHECK BIT. THE ACTUAL FUNCTION OF VOLUME CHECK, THAT OF PREVENTING OR ALLOWING A TAPE MOTION COMMAND DEPENDING UPON WHETHER VOLUME CHECK IS SET OR CLEAR, IS NOT CHECKED BY THIS TEST; THIS FUNCTIONALITY IS CHECKED IN THE INDIVIDUAL TESTS OF TAPE MOTION COMMANDS.

TEST 9: COMPLETION INTERRUPT

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

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

TEST 10: BASIC PACKET PROTOCOL

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

TEST 11: NON-TAPE MOTION COMMANDS

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

7.0 MAINTENANCE HISTORY

REVISION A - MARCH 1982

CVTSAAO => CNTSAAO

JAKI BERG

9-APR-1984

CHANGES WERE MADE TO CVTSAAO TO PRODUCE CNTSAAO FOR THE FALCON-PLUS PROJECT (SBC-11/21*). CHANGES, MARKED BY ";JB REV A-0", ARE:

- SET THE ODT BREAK VECTOR (LOCATED IN 140) TO THE STARTING ADDRESS OF FALCON'S JDT RIM (170000-OCTAL).
- LOWER THE GENERAL INTERRUPT PRIORITY FROM 7 TO 6.
- CHANGE DEFAULT CSR ADDRESS FROM .72540 TO .76000.

```

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
22         TSV2::
23         ;**
24         ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
25         ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
26         ;--
27
28
29 002000          POINTER BGNSW,BGNSFT,BGNA',BGNDU,BGNRPT
30 002000          HEADER CNTSA,A,0,655.,0
002000          L$NAME::          ;DIAGNOSTIC NAME
002000          103          .ASCII /C/
002001          116          .ASCII /N/
002002          124          .ASCII /T/
002003          123          .ASCII /S/
002004          101          .ASCII /A/
002005          000          .BYTE 0
002006          000          .BYTE 0
002007          000          .BYTE 0
002010          L$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. PTABLE
002016          045654          .WORD L$HARD
002020          L$SPCP::          ;PTR. TO S.W. PTABLE
002020          046006          .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$CO::          .WORD 0
002032          000000          .WORD 0
002034          L$DTYP::          ;DIAGNOSTIC TYPE
002034          000000          .WORD 0
002036          L$APT::          ;APT EXPANSION
002036          000000          .WORD 0
002040          L$DTP::          ;PTR. TO DISPATCH TABLE
002040          002124          .WORD L$DISPATCH

```

PROGRAM HEADER

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	022702	.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	022370	.WORD	L\$AU	
002072		L\$DUT::		;PTR. TO DROP UNIT CODE
002072	022466	.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	E\$LOAD	
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	022654	.WORD	L\$CLEAN	
002110		L\$ACP::		;PTR. TO AUTO CODE
002110	022574	.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	

DISPATCH TABLE

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.

002122
002122 000013
002124
002124 023464
002126 023704
002130 024402
002132 025074
002134 026430
002136 027534
002140 031004
002142 034372
002144 035276
002146 040412
002150 043514

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

DEFAULT HARDWARE P-TABLE

```

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  DF,PTBL      ;DEFAULT HARD-P-TABLE
      002152  000003      .WORD  L10000-L$HW/2
      002154
      002154
51
52 002154  176000      .WORD  176000      ; 1ST (OF 2) REGISTERS.
53 002156  000224      .WORD  224        ; INTERRUPT VECTOR
54 002160  000200      .WORD  PRI04     ; INTERRUPT PRIORITY.
55 002162
      002162      ENDPHW
      L10000:

```

SOFTWARE P-TABLE

```

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          .WORD  L10001-L$SW/2
002164          L$SW::
002164          SFPTBL::
64
65 002164          TRANSTST::      .WORD  0          ; ENABLE TEST OF TRANSPORT(S) IF =1
66 002166          NOITS::        .WORD  0          ; INHIBIT ITERATION OPTION.
67                                     ; ... 0 = ITERATE.
68                                     ; ...NZ = INHIBIT ITERATE.
69 002170          LERRMAX::      .WORD  15.        ; LOCAL (PER TEST) ERROR LIMIT
70 002172          GERRMAX::      .WORD  200.       ; GLOBAL (PER UNIT) ERROR LIMIT
71 002174          ENDSW
002174          L10001;
72
73 002174          ENDMOD

```

SOFTWARE P-TABLE

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
;
; BIT POSITION IN SECOND STATUS WORD
000040          EF.START== 32. ; (100000) START COMMAND WAS ISSUED
000037          EF.RESTART== 31. ; (040000) RESTART COMMAND WAS ISSUED
000036          EF.CONTINUE== 30. ; (020000) CONTINUE COMMAND WAS ISSUED
000035          EF.NEW== 29. ; (010000) A NEW PASS HAS BEEN STARTED
000034          EF.PWR== 28. ; (004000) A POWER-FAIL/POWER-UP OCCURRED

```



```

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

```

```

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

```

34
35 002174

```

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

```

MEMORY MANAGEMENT DEFINITIONS

```
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
```

MEMORY MANAGEMENT DEFINITIONS

```

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
; FNOC
; *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

```

C3

TSV05 REGISTER AND PACKET DEFINITIONS

```

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

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

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     ;*
107     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 2
108     ;(XST2)
109     ;*
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     ;*
120     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 3
121     ;(XST3)
122     ;*
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     ;*
133     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 4
134     ;(XST4)
135     ;*
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
144     ;*
145     ;TSSR TERMINATION CODES (BIT 0-2)
146     ;*
147
148     000006      TSREJ = 3+2      ;COMMAND REJECTED
149     000006      UNREC = 6      ;UNRECOVERABLE ERROR
150
151
152
153     ;*
153     ;DEVICE REGISTER OFFSETS

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

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      ;
166      ; TSDB ADDRESS BIT DEFINITIONS
167      ;
168      000003      A1716 = BIT1+BIT0      ;ADDRESS BITS 17:16 ARE IN 1:0
169
170      ;
171      ; COMMAND DEFINITIONS
172      ;
173      000017      P.GETSTAT = 17      ;GET STATUS
174      000013      P.INIT = 13      ;INITIALIZE
175      000012      P.CONTROL = 12      ;CONTROL COMMANDS
176      000011      P.FORMAT = 11      ;FORMAT
177      000010      P.POSITION = 10      ;POSITION
178      000006      P.WRTSUR = 6      ;SUBSYSTEM WRITE
179      000005      P.WRITE = 5      ;WRITE
180      000004      P.WRTCHAR = 4      ;WRITE CHARACTERISTICS
181      000001      P.READ = 1      ;READ
182
183      ;
184      ; COMMAND PACKET HEADER WORD BIT DEFINITIONS
185      ;
186      100000      P.ACK = BIT15      ;BUFFER AVAIL FOR CONTROLLER
187      040000      P.CVC = BIT14      ;CLEAR VOLUME CHECK
188      020000      P.OPP = BIT13      ;REVERSE SEQUENCE OF DATA BITS
189      010000      P.SWB = BIT12      ;SWAP BYTES IN MEMORY
190      007400      P.MODE = BIT11!BIT10!BIT9!BIT8 ;EXTENDED COMMAND MODE FIELD
191      000200      P.IE = BIT7      ;INTERRUPT ENABLE
192      000140      P.FMT = BIT6!BITS5 ;PACKET HEADER TYPE (ALWAYS=0)
193      000037      P.CMD = 37      ;MAJOR COMMAND FIELD
194
195      ;
196      ; CONTROL COMMAND MODE CODES
197      ;
198      000000      PC.RELEASE = 0*256. ;RELEASE BUFFER
199      000400      PC.REWIND = 1*256. ;REWIND
200      001000      PC.NOOP = 2*256. ;NO-OP
201      002000      PC.IEREW = 4*256. ;REWIND IMMEDIATE INTERRUPT
202      002400      PC.ERASE = 5*256. ;SECURITY ERASE
203
204      ;
205      ; CONTROLLER RAM DEFINITIONS
206      ;
207      000167      RMCHBEG = 167      ;CHARACTERISTICS IO DATA BEGIN RAM ADDRESS
208      000200      RMCHEND = 200      ;CHARACTERISTICS IO DATA END RAM ADDRESS
209      000201      RMPKTBEG = 201      ;COMMAND PACKET BEGIN RAM ADDRESS
210      000210      RMPKTEND = 210      ;COMMAND PACKET END RAM ADDRESS
211      000215      RMMMSGBEG = 215      ;MESSAGE BUFFER BEGIN RAM ADDRESS
212      000234      RMMMSGEND = 234      ;MESSAGE BUFFER END RAM ADDRESS

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

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.RDXT  = 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

```

TSV05 REGISTER AND PACKET DEFINITIONS

```

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

```


H3

TSV05 REGISTER AND PACKET DEFINITIONS

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	;	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					

J3

SPECIAL MACROS AND OPDEFS.

```

397                   ;MACRO TO PERFORM XOR
398                   ;
399                   ;
400                   ;MACRO XOR    A,B
401                   MOV    A,-(SP)
402                   BIC    B,(SP)
403                   BIC    A,B
404                   BIS    (SP)+,B
405                   .ENDM
406                   ;
407                   000000                   EN=0                   ; INITIALIZE ERROR NUMBER
408                   .SBTTL FORCER   - FORCE ERROR FLAG
409                   ;
410                   ;
411                   ; THE FOLLOWING LOCATIONS MAY BE PATCHED BY THE USER
412                   ; TO OBTAIN THE RESULTS DESCRIBED FOR EACH.
413                   ;
414                   ;
415                   002174 000000           FORCER::           0           ; FORCE TYPE ALL HARD ERRORS (THE ONES CALLED -
416                   ; - BY THE MACRO "IFERROR"). AN ERROR NEED NOT -
417                   ; - EXIST, JUST ASSUME AND TYPE THE MESSAGE.
418                   ;
419                   ;
420                   ;

```

GLOBAL DATA SECTION

```

422          .SBTTL  GLOBAL DATA SECTION
423
424          ;++
425          ;THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
426          ;IN MORE THAN ONE TEST.
427          ;--
428
429          ;
430          ;THE FOLLOWING DATA ARE SET FOR EACH UNIT AT INIT TIME.
431          ;SINGLE UNIT DEFAULTS (LISTED) ARE IN THE DEFAULT P-TABLE.
432          ;
433 002176 000000 EPRTSW::      .WORD 0      ;PRINT SWITCH
434 002200 000000 UNITN::      .WORD 0      ;UNIT # UNDER TEST.
435 002202 000000 QVP::        .WORD 0      ;QUICK VERIFY FLAG.
436 002204 000000 CSRADDR::   .WORD 0      ;ADDRESS OF CSR FOR CURRENT DEVICE
437 002206 000224 IVEC::        .WORD 224    ;INTERRUPT VECTOR
438 002210 000200 IPRI::        .WORD PRI04  ;INTERRUPT PRIORITY.
439 002212 000000 TSTCNT::   .WORD 0      ;NUMBER OF TESTS RUN IN THIS PASS
440 002214 000000 LOOPCNT::  .WORD 0      ;REMAINING ITERATION COUNT FOR TEST
441 002216 000000 DEVCNT::   .WORD 0      ;NUMBER OF DEVICE UNDER TEST
442 002220 000000 FATFLG::   .WORD 0      ;SET IF FATAL ERROR IS DETECTED IN TEST
443 002222 000000 INTRECV::  .WORD 0      ;SET IF TAPE INTERRUPT WAS RECEIVED
444 002224 000000 EXTFEA::   .WORD 0      ;EXTENDED FEATURES SOFTWARE SW 0=OFF;1=ON
445 002226 000000 BENBSW::   .WORD 0      ;BUFFER ENABLE SWITCH SW 0=OFF;1=ON
446 002230 000000 EXPD::      .WORD 0      ;EXPECTED RAM DATA FOR PRAMPKT ROUTINE
447 002232 000000 RECV::      .WORD 0      ;RECEIVED RAM DATA FOR PRAMPKT ROUTINE
448 002234 000000 ERRHI::    .WORD 0      ;HIGH ADDRESS MEMORY ERROR
449 002236 000000 ERRLO::    .WORD 0      ;LOW ADDRESS MEMORY ERROR
450 002240      RAMDATA::  .BLKW 16.    ;DATA READ FROM RAM PACKET OR MESSAGE BUF AREA
451 002300 000000 RAMSIZ::   .WORD 0      ;RAM DATA SIZE FOR PRAMPKT ROUTINE
452 002302 000000 RCVHIADD:: .WORD 0      ;RECEIVED BUFFER HIGH ADDRESS
453 002304 000000 RCVLOADD:: .WORD 0      ;RECEIVED BUFFER LOW ADDRESS
454 002306 000000 COUNT::   .WORD 0      ;TEST COUNT PATTERN
455 002310 000000 DATA::   .WORD 0      ;TEST DATA
456 002312 000000 TSTFLAG:: .WORD 0      ;TEST FLAG WORD
457 002314 000000 TSTPTR::  .WORD 0      ;TSTBLK POINTER
458 002316 000000 PRMNO::   .WORD 0      ;PRINT ROUTINE TEMP
459 002320      EXPMSG::   .BLKB 100.   ;EXPECTED MESSAGE BUFFER DATA
460 002464      RECMMSG::  .BLKB 100.   ;RECEIVED MESSAGE BUFFER DATA
461 002630      TMPBFR::   .BLKB 80.    ;TEMPORARY STORAGE FOR PRINT

```

TSTBLK - TEST DATA TABLE

463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515

002750
002750 000000
002752 177777
002754 000001
002756 000002
002760 000004
002762 000010
002764 000020
002766 000040
002770 000100
002772 000200
002774 000400
002776 001000
003000 002000
003002 004000
003004 010000
003006 020000
003010 040000
003012 100000
003014 177776
003016 177775
003020 177773
003022 177767
003024 177737
003026 177677
003030 177577
003032 177377
003034 176777
003036 175777
003040 173777
003042 167777
003044 157777
003046 137777
003050 077777
003052 125252
003054 052525
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 ;DATA FOR WALKING ZEROS
.WORD +CBIT0
.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**

```

GLOBAL ENVIRONMENT STORAGE

```

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 ; ...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 000000 CTAB::      .WORD 0 ; CONFIGURATION TABLES.
552 003160 000000 CTAB::      .WORD 0 ; CONFIG WORK.
553 003162 000000          .WORD 0
554 003164 000000          .WORD 0
555 003166 000000          .WORD 0
556 003170 177777          .WORD 0 ; END OF MEM TABLE.
557 003172
558          CTAB::
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 OFFLINE, NO EXISTENT DEVICE REGISTER
565          ; 16XXXX = UNIT OFFLINE, EXISTENT REGISTER AT START
566          ; 14XXXX = UNIT DROPPED, COUNTS XXXX ERRORS
567 003172          ERTABL:      .BLKW 64,
568 003372 000000          ERTABE:      .WORD 0
569
570 003374 000000 SKIPT: .WORD 0 ; 1=SKIP SUBTEST 0=NO SKIP OF SUBTEST

```

GLOBAL TEXT MESSAGES

```

572 .SBTTL GLOBAL TEXT MESSAGES
573 ;++
574 ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
575 ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
576 ; MORE THAN ONE TEST.
577 ;--
578
579
580
581 ;+
582 ; NAMES OF DEVICES SUPPORTED
583 ;-
584
585 003376          DEVTYP <TSV05>
003376          L$DVTYP::
003376          124      123      126      .ASCIZ *TSV05*
                    .EVEN
586
587
588 ;+
589 ; TEST DESCRIPTION
590 ;-
591 003404          DESCRIPT <**** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR ****>
003404          L$DESC::
003404          052      052      052      .ASCIZ /***** TSV05 LOGIC DIAGNOSTIC - REPLACE M7196 IF ERROR *****/
                    .EVEN
612
613
614
615
616 ;+
617 ; BIT TO ASCII CONVERSION FOR TSSR REGISTER
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$: .ASCIZ 'BIT3'
634 003628      102      111      124 14$: .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 /*A ADDRESS: #06/
642 004014      045      101      040 TSSX: .ASCIZ /*A TSBA,TSSR EXP'D: #06#A,#06#N/
643 004054      045      101      040 TSSX: .ASCIZ /*A TSBA,TSSR REC'D: #06#A,#06/

```

GLOBAL TEXT MESSAGES

```

644 004113 045 116 045 FUSI: .ASCII /#N#A/
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#A/
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#A 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#A, REC'D: #06/
658 004563 045 116 045 EXPGT2: .ASCIZ /#N#A EXP'D: #06#A, #06#N#A REC'D: #0#A, #06/
659 004637 045 101 040 DUAD12: .ASCIZ /#A REG(W) WRITTEN TO: #06#A REG(R) READ; EXP'D: #06#A, 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 WRMSG: .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#A ***** NO NXM ADDRESS - CANNOT TEST NXM TIMEOUT. *****#N'
668 005550 045 116 045 M8186: .ASCIZ '#N#A ***** 11/23A SYSTEM *****#N'
669 005641 045 116 045 M8189: .ASCIZ '#N#A ***** 11/23B SYSTEM *****#N'
670
671
672
673

```


GLOBAL ERROR REPORT SECTION

```

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 000001
    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.
        ENDMMSG
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

```

PRITSSR - PRINT TSSR CONTENTS

```

701                                     .SBTTL PRITSSR - PRINT TSSR CONTENTS
702
703                                     ;*
704                                     ;
705                                     ;ROUTINE TO DISPLAY THE CONTENTS, AND BIT DEFINITIONS, OF
706                                     ;THE TSSR REGISTER. THIS ROUTINE IS NORMALLY CALLED ONLY
707                                     ;BY A MESSAGE PRINTING ROUTINE
708                                     ;
709                                     ;INPUTS:
710                                     ;
711                                     ;       R1       CONTENTS OF TSSR
712                                     ;
713                                     ;SUBORDINATE ROUTINES:
714                                     ;
715                                     ;       CHKAMB   CHECK FOR AMBIGUOUS CONTENTS
716                                     ;
717                                     ;-
718
719 PRITSSR:
720     SAVREG                               ;SAVE GENERAL REGISTERS
721     MOV     R4,R4                         ;SAVE THE TSSR CONTENTS
722     PRINTB @TSSRFOR,R4                   ;PRINT THE CONTENTS OF TSSR
723     MOV     R4,-(SP)
724     MOV     @TSSRFOR,-(SP)
725     MOV     @2,-(SP)
726     MOV     SP,R0
727     TRAP   C:PNTB
728     ADD     @6,SP
729     MOV     R4,R0                         ;GET TSSR BACK FOR CHKAMB
730     JSR    PC,CHKAMB                       ;ARE CONTENTS AMBIGUOUS ?
731     BCS    5:                             ;BRANCH IF NOT
732     PRINTX @AMBTSSR                       ;SHOW CONTENTS ARE AMBIGUOUS
733     MOV     @AMBTSSR,-(SP)
734     MOV     @1,-(SP)
735     MOV     SP,R0
736     TRAP   C:PNTX
737     ADD     @4,SP
738     MOV     R4,R3                         ;CONTENTS OF TSSR
739     BIC    @HIADDR!FATERR!TERCLS,R3      ;CLEAR ALL MULTIPLE BIT FIELDS
740     BEQ    20:                             ;NO BITS ARE SET
741     MOV     @TMPBFR,R2                     ;TEMPORARY ASCII BUFFER
742     MOV     @TSSRBIT,R1                   ;ASCII EQUIVALENT OF BITS
743     TST    R3                             ;REMAINING BITS TO CONVERT
744     BEQ    15:                             ;BRANCH WHEN ALL ARE DONE
745     CLC                                     ;CLEAR CARRY FOR SHIFT
746     ROL    R3                             ;SHIFT NEXT BIT TO CARRY
747     BCC    13:                             ;BRANCH IF BIT NOT SET
748     MOV     (R1),R0                         ;POINTER TO BIT DEFINITION
749     MOV     (R0)+,(R2)+                     ;MOVE ASCII TO BUFFER
750     MOV     11:                             ;MOVE ALL BITS
751     MOV     @'+,1(R2)                       ;INSERT A COMMA TO TERMINATE
752     TST    (R1)+                             ;POINT TO NEXT DESCRIPTION
753     BR     10:                             ;GET THE REMAINING BITS
754     CLRB  -(R2)                             ;TERMINATE THE LINE
755     PRINTX @TSSDEF,@TMPBFR               ;PRINT THE BIT DEFINITIONS
756     MOV     @TMPBFR,-(SP)
757     MOV     @TSSDEF,-(SP)

```

PRI1SSR - PR1NT TSSR CONTENTS

```

006164 012746 000002      MOV      02,-(SP)
006170 010600      MOV      SP,R0
006172 104415      TRAP    C:PNTX
006174 062706 000006      ADD      06,SP
745
746 006200 010403      20$:    MOV      R4,R3          ;GET THE TSSR CONTENTS
747 006202 042703 177761      BIC      0+CTERCLS,R3    ;CLEAR ALL BUT TERMINATION
748 006206 016303 006672      MOV      TCOCOD(R3),R3   ;GET THE TERMINATION CODE MEANING
749 006212      PRINTX  0+TCOASC,R3     ;PRINT THE TERMINATION CODE
      006212 010346      MOV      R3,-(SP)
      006214 012746 006472      MOV      0+TCOASC,-(SP)
      006220 012746 000002      MOV      02,-(SP)
      006224 010600      MOV      SP,R0
      006226 104415      TRAP    C:PNTX
      006230 062706 000006      ADD      06,SP
750 006234 010403      MOV      R4,R3          ;TSSR CONTENTS AGAIN
751 006236 042703 177717      BIC      0+CFATERR,R3   ;CLEAR ALL BUT FATAL TERMINATION
752 006242 001416      BEQ     25$             ;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  0+TFCASC,R3   ;PRINT THE FATAL TERMINATION CODE
      006256 010346      MOV      R3,-(SP)
      006260 012746 006533      MOV      0+TFCASC,-(SP)
      006264 012746 000002      MOV      02,-(SF)
      006270 010600      MOV      SP,R0
      006272 104415      TRAP    C:PNTX
      006274 062706 000006      ADD      06,SP
758 006300 042704 176377      25$:    BIC      0+CHIADDR,R4   ;CLEAR ALL BUT EXTENDED ADDRESS
759 006304 001411      BEQ     30$             ;DON'T PRINT IF ZERO
760 006306      PRINTX  0+TEXASC,R4   ;PRINT THE EXTENDED ADDRESS BITS
      006306 010446      MOV      R4,-(SP)
      006310 012746 006431      MOV      0+TEXASC,-(SP)
      006314 012746 000002      MOV      02,-(SP)
      006320 010600      MOV      SP,R0
      006322 104415      TRAP    C:PNTX
      006324 062706 000006      ADD      06,SP
761 006330 013703 002176      30$:    MOV      EPRTSW,R3      ;PRINT MEASGE BUFFER ADDRESS
762 006334      PRINTX  R3             ;PRINT PROPER MESSAGE
      006334 010346      MOV      R3,-(SP)
      006336 012746 000001      MOV      01,-(SP)
      006342 010600      MOV      SP,R0
      006344 104415      TRAP    C:PNTX
      006346 062706 000004      ADD      04,SP
763 006352 000207      RTS     PC              ;RETURN TO CALLER
764
766 006354      EPRT2:
767 006354      045    116    045    EPRT1: .ASCIZ '###A *****REPLACE M7196*****'
782 006411      045    116    045    1SSRFOR: .ASCIZ '###A TSSR = #06'
783 006431      045    116    045    TEXASC:  .ASCIZ '###A Extended Address Bits = #06'
784 006472      045    116    045    TCOASC:  .ASCIZ '###A Termination Class Code = #T'
785 006533      045    116    045    TFCASC:  .ASCIZ '###A Fatal Termination Class Code = #T'
786 006602      045    116    045    TSSDEF:  .ASCIZ '###A TSSR Bits Set; #T'
787 006631      045    116    045    AMBTSSR: .ASCIZ '###A TSSR Contents Are Ambiguous'
788
789 006672 006712 006735 006763 ICOCOD: .EVEN .WORD 1$,2$,3$,4$,5$,6$,7$,8$

```

PRITSSR - PRINT TSSR CONTENTS

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	

PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET

```

807 .SBTTL PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET
808
809
810 ;*
811 ;THIS ROUTINE PRINTS THE ADDRESS AND CONTENTS OF A COMMAND PACKET.
812 ;THIS ROUTINE IS NORMALLY ONLY CALLED FROM A PRINT ROUTINE.
813
814 ;INPUT:
815
816 R0 NUMBER OF WORDS IN PACKET
817 R3 HIGH ORDER COMMAND PACKET ADDRESS
818 R4 ADDRESS OF COMMAND PACKET
819
820 ; NOTE: R3 IS IGNORED IF THE KTENABLE FLAG IS CLEAR.
821 ;-
822
823 PRIPKT::
824 SAVREG ;SAVE THE REGISTERS
825 MOV R0,R5 ;SAVE NO. OF WORDS IN PACKET
826 TST KTENABLE ;ABOVE 28K UNDER TEST?
827 BNE 10$ ;BR IF YES
828 CLR R3 ;SET HIGH ORDER ADDRESS TO 0
829 MOV R3,R1 ;COPY HIGH ORDER ADDRESS
830 MOV R4,R0 ;GET LOWER ADDRESS
831 ROL R0 ;SHIFT BIT 15 INTO C BIT
832 ROL R1 ;AND INTO HIGH ORDER.
833 PRINTB @PKTADD,R1,R4 ;PRINT PACKET ADDRESS
834 MOV R4,-(SP)
835 MOV R1,-(SP)
836 MOV @PKTADD,-(SP)
837 MOV @3,-(SP)
838 MOV SP,R0
839 TRAP C#PNTB
840 ADD @10,SP
841 MOV R3,R0 ;GET HIGH ORDER ADDRESS
842 BEQ 20$ ;BR IF NOT ABOVE 28K.
843 MOV R4,R1 ;GET LOW ORDER ADDRESS
844 JSR PC,SETMAP ;SETUP PAR6 MAPPING FOR 18 BIT ADDRESS
845 MOV R0,R4 ;GET RETURNED PAR6 ADDRESS BIAS
846 CLR R1 ;SAVE WORD NUMBER
847 MOV (R4)+,R2 ;GET PACKET CONTENTS
848 PRINTB @PKTFRM,R1,R2 ;PRINT THE DATA
849 MOV R2,-(SP)
850 MOV R1,-(SP)
851 MOV @PKTFRM,-(SP)
852 MOV @3,-(SP)
853 MOV SP,R0
854 TRAP C#PNTB
855 ADD @10,SP
856 INC R1 ;NEXT WORD NUMBER
857 CMP R1,R5 ;DONE ALL PACKET WORDS?
858 BLT 25$ ;LOOP TILL ALL DONE
859 RTS PC ;RETURN
860
861 007364 010005 003130
862 007370 005737 003130
863 007372 005737 003130
864 007376 001001
865 007400 005003
866 007402 010301 10$:
867 007404 010400
868 007406 006100
869 007410 006101
870 007412
871 007412 010446
872 007414 010146
873 007416 012746 007550
874 007422 012746 000003
875 007426 010600
876 007430 104414
877 007432 062706 000010
878 007436 010300 15$:
879 007440 001404
880 007442 010401
881 007444 004737 017306
882 007450 010004
883 007452 005001 20$:
884 007454 012402 25$:
885 007456
886 007456 010246
887 007460 010146
888 007462 012746 007512
889 007466 012746 000003
890 007472 010600
891 007476 104414
892 007476 062706 000010
893 007502 005201
894 007504 020105
895 007506 002762
896 007510 000207
897
898
899
900
901
902
903
904
905
906 007512 045 116 045 PKTFRM: .ASCIZ '#N#A Packet Word #D1#A = #06'
907 007550 045 116 045 PKTADD: .ASCIZ '#N#A Packet Address = #01#05'
908 .EVEN
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930

```

PRIBXOR - PRINT EXPD, RECV AND XOR BYTE

```

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 PC ;RETURN TO CALLER
880
881 007670 045 116 045 XORBFOR: .ASCIZ '*N*A EXPD: *03*A RECV: *03*A XOR: *03*'
882 .EVEN
883

```

PRIXOR - PRINT EXPD, RECV AND XOR

```

885                                     .SBTTL PRIXOR - PRINT EXPD, RECV AND XOR
886
887
888
889                                     ;*
890                                     ;PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE TWO
891                                     ;THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
892
893                                     ;INPUTS:
894                                     ;
895                                     ;       R1      RECEIVED DATA
896                                     ;       R2      EXPECTED DATA
897
898                                     ;OUTPUT:
899                                     ;
900                                     ;       R0      XOR OF EXPECTED/RECEIVED DATA
901                                     ;
902                                     ;-
903 PRIXOR::
904     SAVREG                               ;SAVE THE REGISTERS
905     MOV      R2,R3                       ;EXPECTED DATA
906     XOR      R1,R3                       ;FORM THE EXCLUSIVE OR
907     PRINTB  @XORFOR,R2,R1,R3 ;PRINT THE MESSAGE
908     MOV      R5,-(SP)
909     MOV      R1,-(SP)
910     MOV      R2,-(SP)
911     MOV      @XORFOR,-(SP)
912     MOV      @4,-(SP)
913     MOV      SP,R0
914     TRAP    C#PNTB
915     ADD     @12,SP
916     MOV     R3,R0                       ;R0 HAS XOR ON; RETURN
917     RTS    PC                           ;RETURN TO CALLER
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

903 007736
904 007736
905 007742 010203
906 007744
907 007754
   007754 010346
   007756 010145
   007760 010246
   007762 012746 010006
   007766 012746 000004
   007772 01060C
   007774 104414
   007776 062706 000012
908 010002 010300
909 010004 000207
911 010006   045   116   045 XORFOR: .ASCIZ '##A EXPD: #06#A RECV: #06#A XOR: #06'
912

```

PRIEQU - PRINT BIT NUMBERS AS ASCII EQUIVALENT

```

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 .SBTTL PRIRAM - PRINT RAM ADDRESS
935
936 ;*
937 ;
938 ;PRINT CONTRCLER 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 ENROR
948 010066 010446 MOV R4,-(SP)
949 010070 012746 010112 MOV RAMFOR,-(SP)
950 010074 012746 000002 MOV #2,-(SP)
951 010100 010600 MOV SP,R0
952 010102 104414 TRAP C#PNTB
953 010104 062706 000006 ADD #6,SP
954 010110 000207 RTS PC ;RETURN
955
956 010112 045 116 045 RAMFOR: .ASCIZ 'N/A CONTROLLER RAM ADDRESS = #06'
957 .EVEN
958
959 .SBTTL PRIADD - PRINT MEMORY ERROR ADDRESS
960
961 ;*
962 ;
963 ;PRINT MEMORY ADDRESS
964 ;THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
965 ;
966 ; IMPLICIT INPUTS
967 ;
968 ; ERRHI - HIGH ORDER ADDRESS
969 ; ERRLO - LOW ORDER ADDRESS

```


PRIADD - PRINT MEMORY ERROR 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 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 'NWA MEMORY ERROR ADDRESS = 0105'
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 010270
994 010270
995 010274 013702 002234
996 010300 013701 002236
997
998
999
1000 010304
    010304 010146
    010306 012746 010352
    010312 012746 000002
    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 ;SAVE R1-R5 UNTIL NEXT RETURN
; MOV ERRLO,R1 ;GET HIGH ADDRESS
; MOV R1,R2 ;GET LOW ADDRESS
; ROL R1 ;COPY LOW ADDRESS
; ROL R0 ;SHIFT BIT 15 TO C BIT
; PRINTB @PRIA0,R0,R2 ;SHIFT INTO HIGH ORDER
; MOV R2,-(SP) ;PRINT MEMORY ADDRESS IN ERROR
; MOV R0,-(SP)
; MOV @PRIA0,-(SP)
; MOV @3,-(SP)
; MOV SP,R0
; TRAP C:PNTB
; ADD @10,SP
; RTS PC ;RETURN
;
;
; PRIA0: .ASCIZ 'NWA MEMORY ERROR ADDRESS = 0105'
; .EVEN
;
; .SBTTL PRITADD - PRINT MEMORY TEST ADDRESS
;
; ;PRINT MEMORY ADDRESS
; ;THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
;
; ; IMPLICIT INPUTS
;
; ; ERRHI - HIGH ORDER ADDRESS
; ; ERRLO - LOW ORDER ADDRESS
;
;
; PRITADD:
; SAVREG
; MOV ERRHI,R2 ;SAVE R1-R5 UNTIL NEXT RETURN
; MOV ERRLO,R1 ;GET HIGH ADDRESS
; MOV R1,R2 ;GET LOW ADDRESS
; ROL R1 ;COPY LOW ADDRESS
; ROL R0 ;SHIFT BIT 15 TO C BIT
; PRINTB @PRIT0,R1 ;SHIFT INTO HIGH ORDER
; MOV R1,-(SP) ;PRINT MEMORY ADDRESS LOW IN ERROR
; MOV @PRIT0,-(SP)
; MOV @2,-(SP)
; MOV SP,R0
; TRAP C:PNTB
; ADD @6,SP
; PRINTB @PRIT1,R2 ;PRINT MEMORY ADDRESS HIGH IN ERROR
; MOV R2,-(SP)
; MOV @PRIT1,-(SP)
; MOV @2,-(SP)
; MOV SP,R0
; TRAP C:PNTB
; ADD @6,SP
; RTS PC ;RETURN

```


SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND

```

1011          .SBTTL SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND
1012
1013          ;+
1014          ;
1015          ;ROUTINE TO ISSUE A SPACE RECORDS
1016          ;COMMAND (FORWARD OR REVERSE)
1017          ;
1018          ;INPUT:
1019          ;
1020          ;       R3      NUMBER OF RECORDS TO BE SPACED OVER
1021          ;              BIT15 CONTROLS DIRECTION
1022          ;              BIT15 = 0 IS FORWARD
1023          ;              BIT15 = 1 IS REVERSE
1024          ;       R5      FIRST DEVICE UNIBUS ADDRESS
1025          ;
1026          ;       REQUIRES A WRITE CHARACTERISTICS DONE PREVIOUSLY
1027          ;
1028          ;OUTPUT:
1029          ;
1030          ;       CARRY   SET - SPACE RECORDS COMMAND OK
1031          ;              CLR - SPACE RECORDS FAILED
1032          ;
1033          ;
1034          ;       R0      THE CONTENTS OF R4 IS MOVED TO R0
1035          ;
1036          ;
1037          ;IMPLICIT OUTPUT:
1038          ;
1039          ;       TAPE HAS BEEN MOVED
1040          ;
1041          ;SIDE EFFECTS:
1042          ;
1043          ;
1044          ;-
1045
1046          SPACE::
1047          SAVREG          ;SAVE THE GENERAL REGISTERS
1048          MOV             #500.,SDFLAY ;SET UP DELAY
1049          MOV             #140010,80$ ;SET UP COMMAND, SPACE FORWARD
1050          TST            R3          ;CHECK FOR DIRECTION
1051          BMI            5$          ;BR, IF REVERSE INDICATED
1052          MOV            R3,90$      ;LOAD UP NUMBER OF RECORDS TO SPACE
1053          BR            10$         ;GO DO COMMAND
1054          BIC            #BIT15,R3  ;CLEAR DIRECTION BIT
1055          MOV            R3,90$      ;LOAD UP NUMBER OF RECORDS TO SPACE
1056          BIS            #BIT8,80$  ;SET REVERSE BIT IN COMMAND PACKET
1057          MOV            #80$,R4    ;SET UP R4 WITH PACKET ADDRESS
1058          MOV            R4,TSDB(R5);SEND OUT COMMAND
1059          JSR            PC,WAITF   ;WAIT FOR SSR
1060          BCS            20$         ;BR, IF SSR IS SET AND OK
1061          DELAY          250        ;DELAY ABOUT .25 SECONDS
1062          MOV            #250,(PC)+
1063          .WORD          0
1064          MOV            L$DLY,(PC)+
1065          .WORD          0
1066          DEC            -6(PC)
1067          BNE            ,-4

```

SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND

	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

B5

SPACE SPACE RECORDS (FORWARD AND REVERSE) COMMAND

1077					
1078					
1079					
1080				PACKET FOR SPACE COMMAND	
1081					
1085					
1086				COMMAND WORD	
1087	010640	000000	801:	.WORD	
1088				NUMBER OF RECORDS TO BE SPACED OVER WORD	
1089	010642	000000	901:	.WORD	
1090	010644	000000		.WORD	
1091	010646	000000		.WORD	
1092	010650	000000	SDELAY:	.WORD 0	DELAY COUNTER
1093				.EVEN	

D5

WRCHR - WRITE CHARACTERISTICS COMMAND

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								

REWIND - POSITION TAPE (REWIND) COMMAND

```

1161 .SBTTL REWIND - POSITION TAPE (REWIND) COMMAND
1162
1163 ;*
1164 ;
1165 ; THIS ROUTINE WILL REWIND THE SELECTED TAPE.
1166 ;
1167 ; CAUTION: THE ROUTINE DOES NOT WAIT FOR BOT
1168 ; TO ARRIVE, ALSO THE CALLER MUST CHECK FOR
1169 ; SSR TO SET IN THE TSSR
1170 ;
1171 ;
1172 ; CALLING SEQUENCE:
1173 ;
1174 ; DO A SOFT INIT
1175 ; DO A WRITE CHARACTERISTICS
1176 ; JSR PC,REWIND
1177 ;
1178 ; INPUT:
1179 ;
1180 ; R5 FIRST DEVICE UNITBUS ADDRESS
1181 ;
1182 ;
1183 ; OUTPUT
1184 ;
1185 ; R0 THE CONTENTS OF R4 IS PASSED TO R0
1186 ;
1187 ;
1188 ;
1189 011004 REWIND: SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1190 0110C4 MOV #RWPACK,R4 ;GET PACKET ADDRESS
1191 011010 012704 011100 MOV R4,TSD8(R5) ;SEND PACKET ADDRESS TO EXECUTE
1192 011014 010465 000000 MOV #360,R3 ;ENOUGH TIME FOR 2400' REEL TO REWIND
1193 011020 012703 000550 JSR PC,WAITF ;WAIT FOR SSR TO SET
1194 011024 004737 016240 BCS 20$ ;LEAVE WHEN SSR IS SET
1195 011030 103417 DELAY 250. ;WAIT FOR .25 SECONDS
1196 011032 MOV #250.,(PC)+
011032 012727 000372 .WORD 0
011036 000000 MOV L#DLY,(PC)+
011040 013727 002116 .WORD 0
011044 000000 DEC -6(PC)
011046 005367 177772 BNE -4
011052 001375 DEC -22(PC)
011054 005367 177756 BNE -20
1197 011062 005303 DEC R3 ;BUMP COUNTER DOWN
1198 011064 001357 BNE 10$ ;KEEP GOING
1199 011066 000241 CLC ;CLEAR CARRY TO SET ERROR
1200 011070 010400 20$: MOV R4,R0 ;PASS THE PACKET ADDRESS
1201 011072 000207 RTS PC ;RETURN
1202
1203
1204
1205 011100 RWPACK: .=<.>10>E177770
1207 011100 .WORD 102010 ;POSITION COMMAND (REWIND)
1208 011100 102010 .WORD 0 ;NOT USED
1209 011102 000000
1210

```


CKRAM - COMPARE RAM TO I/O PACKET

1212
1213
1214
1215
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

```

.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
  MOV  #0,TSDB(R5)                       ;SET MAINTENANCE MODE
  JSR  PC,CHKTSSR                         ;WAIT FOR SSR TO SET
  MOV  R2,TSDB(R5)                       ;SELECT NEXT RAM ADDRESS
  JSR  PC,CHKTSSR                         ;WAIT FOR SSR TO SET
  MOV  TSBA(R5),(R1)                     ;READ THE RAM DATA
  CMPB (R1)+,(R4)+                       ;COMPARE TO EXPECTED
  BEQ  20$                                ;BRANCH IF OK
  INC  R3                                 ;SET ERROR FLAG
  INC  R2                                 ;ADDRESS OF NEXT RAM LOCATION
  CMP  R2,#RMPKTEND                      ;REACHED END YET ?
  BLE  10$                                ;BRANCH TILL ALL READ
  TST  R3                                 ;WAS AN ERROR FOUND ?
  BEQ  30$                                ;BRANCH IF NOT
  CLC                                     ;CLEAR CARRY TO SHOW ERROR
  BR   50$                                ;AND EXIT
  SEC                                     ;SHOW GOOD COMPARE
  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
011204 012737 000010 002300 30$:
011212 000207 50$:

```

CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA

```

1267          .SBTTL CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA
1268          ;*
1269          ;
1270          ;ROUTINE TO READ THE FIRST 8 OR 10 BYTES FROM RAM
1271          ;MEMORY AND COMPARE THIS DATA TO A CHARACTERISTICS DATA BLOCK.
1272          ;
1273          ;INPUT:
1274          ;
1275          ;      R4      ADDRESS OF THE CHARACTERISTICS DATA
1276          ;      R5      FIRST DEVICE UNIBUS ADDRESS
1277          ;
1278          ;OUTPUT:
1279          ;
1280          ;      CARRY   SET - RAM MATCHES PACKET
1281          ;            CLR - RAM DOES NOT MATCH PACKET
1282          ;
1283          ;IMPLICIT OUTPUT:
1284          ;
1285          ;      THE TABLE RAMDATA IS FILLED WITH THE
1286          ;      DATA HELD IN RAM.
1287          ;      RAMSIZ IS SET TO 8. OR 10. FOR PRAMPKT ROUTINE
1288          ;
1289          ;SIDE EFFECTS:
1290          ;
1291          ;      THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
1292          ;
1293          ;-
1294
1295          CKRAM2::
1296          SAVREG          ;SAVE THE GENERAL REGISTERS
1297          MOV             @RAMDATA,R1      ;ADDRESS TO SAVE THE RAM DATA
1298          MOV             @RMCHBEG,R2     ;BYTE ADDRESS OF FIRST RAM DATA
1299          CLR             R3              ;CLEAR THE ERROR FLAG
1300          JSR             PC,CHKTSSR      ;WAIT FOR SSR
1301          MOVB            @0,TSDB(R5)     ;SET MAINTENANCE MODE
1302          JSR             PC,CHKTSSR      ;WAIT FOR SSR TO SET
1303          MOV             R2,TSDB(R5)     ;SELECT NEXT RAM ADDRESS
1304          JSR             PC,CHKTSSR      ;WAIT FOR SSR TO SET
1305          MOVB            TSBA(R5),(R1)   ;READ THE RAM DATA
1306          CMPB           (R1)+,(R4)+     ;COMPARE TO EXPECTED
1307          BEQ             20$            ;BRANCH IF OK
1308          INC             R3              ;SET ERROR FLAG
1309          INC             R2              ;ADDRESS OF NEXT RAM LOCATION
1310          MOV             @8.,RAMSIZ      ;ASSUME EXTFEA NOT SET
1311          TST             EXTFEA          ;IS THE SOFTWARE EXTENDED FEATURES SET
1312          BEQ             25$            ;BR, IF NOT SET
1313          MOV             @10.,RAMSIZ     ;SET RAMSIZ FOR EXTEND FEATURES
1314          CMP             R2,@RMCHEND    ;AT END OF EXTENDED BUFFER
1315          BLE             10$            ;BR, IF NOT AT END YET
1316          BR             27$            ;AT END BRANCH
1317          CMP             R2,@RMCHEND-2  ;REACHED END YET ?
1318          BLE             10$            ;BRANCH TILL ALL READ
1319          TST             R3              ;WAS AN ERROR FOUND ?
1320          BEQ             30$            ;BRANCH IF NOT
1321          CLC              ;CLEAR CARRY TO SHOW ERROR
1322          BR             50$            ;AND EXIT
1323          SEC              ;SHOW GOOD COMPARE

```

TSV3 - GLOBAL AREAS MACRO M1200 23-MAR-84 09:44 PAGE 32-1

H5

SEQ 0059

CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA

1324 011346 000207
1325

50\$: RTS PC

;RETURN

CKMSG - COMPARE WRITE CHAR. MESSAGE BUFFERS

```

1327 .SBTTL CKMSG - COMPARE WRITE CHAR. MESSAGE BUFFERS
1328
1329
1330 ;ROUTINE TO COMPARE A WRITE CHARACTERISTICS EXPD AND RECV
1331 ;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
1332 ;ERROR PRINT ROUTINES.
1333
1334 ;INPUT:
1335
1336 ; R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS
1337 ; R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS
1338 ; R2 EXPD MESSAGE BUFFER ADDRESS
1339 ;OUTPUT:
1340
1341 ; CARRY SET - MESSAGE BUFFERS MATCH
1342 ; CLR -MESSAGE BUFFERS DON'T MATCH
1343
1344 ;IMPLICIT OUTPUT:
1345
1346 ; EXPMSG BUFFER IS SET TO EXPD DATA
1347 ; RECVMSG BUFFER IS SET TO RECV DATA
1348 ; RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
1349 ; RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
1350
1351
1352 CKMSG:;
1353 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1354 MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
1355 MOV R1,RCVLOADD ;SAVE RECV LOW ADDRESS
1356 TST KTENABLE ;TESTING ABOVE 28K?
1357 BEQ 10$ ;BR IF NO
1358 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
1359 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
1360 10$: CLR R4 ;WORD IN BUFFER
1361 CLR R3 ;CLEAR ERROR SEEN FLAG
1362 MOV R2,R5 ;GET EXPD BUFFER ADDRESS
1363 15$: MOV (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
1364 MOV (R1),RECVMSG(R4) ;SAVE RECV FOR ERROR REPORT
1365 CMP (R2)+,(R1)+ ;EXPD EQUAL RECV?
1366 BEQ 25$ ;BR IF YES
1367 INC R3 ;SET ERROR SEEN FLAG
1368 25$: ADD #2,R4 ;POINT TO NEXT WORD ADDRESS
1369 CMP R4,#14 ;DONE FIRST 7 WORDS?
1370 BLE 15$ ;BR IF NO
1371 BII #X2,EXTF,XST2(R5) ;IS EXTENDED FEATURES SET IN EXPD?
1372 BEQ 50$ ;BR IF NO
1373 CMP R4,#16 ;DONE EXTENDED FEATURES WORD?
1374 BLE 15$ ;BR IF NO
1375 50$: TST R3 ;ANY ERRORS SEEN?
1376 BEQ 55$ ;BR IF NO
1377 CLC ;SET FAILURE
1378 BR 60$
1379 55$: SEC ;SET SUCCESS
1380 60$: RTS PC ;RETURN
1381

```

CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

```

1383 .SBTTL CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS
1384
1385
1386 ;ROUTINE TO COMPARE AN EXPECTED AND RECEIVED MESSAGE
1387 ;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
1388 ;ERROR PRINT ROUTINES.
1389
1390 ;INPUT:
1391
1392 ; R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS
1393 ; R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS
1394 ; R2 EXPD MESSAGE BUFFER ADDRESS
1395 ; R3 NUMBER OF BYTES TO COMPARE
1396
1397 ;OUTPUT:
1398 ; CARRY SET - MESSAGE BUFFERS MATCH
1399 ; CLR - MESSAGE BUFFERS DON'T MATCH
1400
1401 ;IMPLICIT OUTPUT:
1402
1403 ; EXPMSG BUFFER IS SET TO EXPD DATA
1404 ; RECVMSG BUFFER IS SET TO RECV DATA
1405 ; RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
1406 ; RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
1407
1408
1409
1410 011470 CKMSG2:
1411 011470 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1412 011474 020327 000144 CMP R3, #RECVMSG-EXPMSG, #00 ;IS COUNT ABOVE MAX ALLOWED?
1413 011500 003412 BLE 5% ;NO BR IF NO
1414 011502 012703 000144 MOV #RECVMSG-EXPMSG, R3, #00
1415 011506 PRINTF #DEBUGMSG ;000
; 011506 012746 011622 MOV #DEBUGMSG, -(SP)
; 011512 012746 000001 MOV #1, -(SP)
; 011516 010600 MOV SP, R0
; 011520 104417 TRAP C#PNTF
; 011522 062706 000004 ADD #4, SP
1416 011526 010037 002302 5%: MOV R0, RCVHIADD ;SAVE RECV HIGH ADDRESS
1417 011532 010137 002304 MOV R1, RCVLOADD ;SAVE RECV LOW ADDRESS
1418 011536 005737 003130 TST #KTENABLE ;TESTING ABOVE 28K?
1419 011542 001403 BEQ 10% ;BR IF NO
1420 011544 004737 017306 JSR PC, SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
1421 011550 010001 MOV R0, R1 ;GET RETURNED ADDRESS BIASED TO PAR6
1422 011552 005004 10%: CLR R4 ;WORD IN BUFFER
1423 011554 005005 CLR R5 ;CLEAR ERROR SEEN FLAG
1424 011556 111264 002320 15%: MOVB (R2), EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
1425 011562 111164 002464 MOVB (R1), RECVMSG(R4) ;SAVE RECV FOR ERROR REPORT
1426 011566 122221 CMPB (R2)+, (R1)+ ;EXPD EQUAL RECV?
1427 011570 001401 BEQ 25% ;BR IF YES
1428 011572 005205 INC R5 ;SET ERROR SEEN FLAG
1429 011574 062704 000001 25%: ADD #1, R4 ;POINT TO NEXT BYTE
1430 011600 020403 CMP R4, R3 ;DONE ALL BYTES?
1431 011602 002001 BGE 50% ;BR IF YES
1432 011604 000764 BR 15% ;DO NEXT BYTE
1433 011606 005705 50%: TST R5 ;ANY ERRORS SEEN?
1434 011610 001402 BEQ 55% ;BR IF NO

```

K5

CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

```
1435 011612 000241                    CLC                    ;SET FAILURE
1436 011614 000401                    BR                    60#                    ;
1437 011616 000261                    55#: SEC                    ;SET SUCCESS
1438 011620 000207                    60#: RTS                    PC                    ;RETURN
1439
1440 011622        120        122        117 DEBUGMSG:        .ASCIZ 'PROGRAM INTERNA: ERROR -CKMSG2 MESSAGE BUFFER EXCEEDED-' ;@@D
1441 011712        045        116        045 FERCM:        .ASCII /#N#A ***/
1442 011723        040        040        124 ERCM:        .ASCIZ / TSSR ERROR CODE REC'D - /
1443 011756        056        056        056 SIMSG:        .ASCIZ /... AFTER DOING SOFT INIT/
1444 012011        124        105        123 TINERR:        .ASCIZ /TEST: .../
1445                    .EVEN
```

CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463 012024
012024
1464 012024 004737 006020
1465 012030 004737 017172
1466 012034
012034
012034 104423
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479 012036
012036
1480 012036 004737 006020
1481 012042 012700 000004
1482 012046 004737 007364
1483 012052
012052
012052 104423
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496 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  SFIMSG
SFIMSG:  JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
         JSR    PC,CKDROP     ;DROP UNIT, IF ALLOWED
         ENDMSG
L10003:  TRAP   C$MSG
;
;PRINT ROUTINE TO PRINT THE CONTENTS OF
;TSSR AND A COMMAND PACKET OTHER THAN GET STATUS COMMAND PACKET.
;INPUTS:
;      R1      TSSR CONTENTS
;      R4      ADDRESS OF COMMAND PACKET
;-
      BGNMSG  PKTSSR
PKTSSR:  JSR    PC,PRITSSR     ;PRINT THE CONTENTS OF TSSR REGISTER
         MOV    #4,R0         ;NO. OF WORDS IN PACKET
         JSR    PC,PRIPKT     ;PRINT THE CONTENTS OF COMMAND PACKET
         ENDMSG
L10004:  TRAP   C$MSG
;
;PRINT ROUTINE TO PRINT THE CONTENTS OF
;TSSR AND A GET STATUS COMMAND PACKET.
;INPUTS:
;      R1      TSSR CONTENTS
;      R4      ADDRESS OF COMMAND PACKET
;-
      BGNMSG  PKTGETS
PKTGETS:

```

CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS

```

1497 012054 004737 006020 JSR PC,PRITSSR ;PRINT THE CONTENTS OF TSSR REGISTER
1498 012060 012700 000002 MOV #2,R0 ;NO. OF WORDS IN GET STATUS PACKET
1499 012064 004737 007364 JSR PC,PRIPKT ;PRINT THE CONTENTS OF COMMAND PACKET
1500 012070 ENDMSG
      012070
      012070 104423 L10005: TRAP C$MSG
1501
1502
1503 ;*
1504 ;PRINT TSSR ERRORS FOR INITIALIZATION TESTS
1505 ;
1506 ;INPUTS:
1507 ;
1508 ; R1 TSSR CONTENTS
1509 ; R4 ADDRESS OF COMMAND PACKET
1510 ;
1511 ;-
1512 012072 BGNMSG SFFMSG
      012072 SFFMSG::
1513 012072 004737 006020 JSR PC,PRITSSR ;PRINT CONTENTS OF TSSR REGISTER
1514 012076 ENDMSG
      012076
      012076 104423 L10006: TRAP C$MSG
1515
1516
1517 ;SHTTL PKTMES - PRINT TSSR AND MESSAGE BUFFER
1518 ;*
1519 ;
1520 ;PRINT ROUTINE T) PRINT THE CONTENTS OF TSSR AND MESSAGE
1521 ;BUFFER FOR ERROR REPORTS
1522 ;
1523 ;INPUTS:
1524 ;
1525 ; R1 CONTENTS OF TSSR
1526 ; R2 LOW ORDER MESSAGE BUFFER
1527 ; R3 HIGH ORDER MESSAGE BUFFER ADDRESS
1528 ; NOTE: R3 IS IGNORED IF KTENABLE FLAG IS CLEAR
1529 ;-
1530 012100 BGNMSG PKTMES
      012100 PKTMES::
1531 012100 004737 006020 JSR PC,PRITSSR ;PRINT CONTENTS OF TSSR
1532 012104 010200 MOV R2,R0 ;LOW ORDER ADDRESS
1533 012106 010301 MOV R3,R1 ;HIGH ORDER ADDRESS
1534 012110 004737 014232 JSR PC,PRMESS ;PRINT THE MESSAGE BUFFER
1535 012114 ENDMSG
      012114
      012114 104423 L10007: TRAP C$MSG
1536

```


ADDSSR - PRINT TEST ADDRESS AND TSSR

```

1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550 012116
      012116
1551 012116 004737 010270
1552 012122 016501 000002
1553 012126 004737 006020
1554 012132
      012132
      012132 104423
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569 012134
      012134
1570 012134 012700 000007
1571 012140 005737 002224
1572 012144 001402
1573 012146 012700 000010
1574 012152 004737 014542
1575 012156
      012156
      012156 104423
1576
1577

```

```

      .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
      ENDMMSG

L10010:
      TRAP   C$MSG

      .SBTTL  MSGEXP - PRINT WRITE CHAR. EXPD-RECV MESSAGE BUFFERS
      ;+
      ;PRINT ROUTINE TO PRINT WRITE CHARACTERISTIC MESSAGE BUFFER
      ;
      ;IMPLICIT INPUTS:
      ;
      ;   EXPMSG  - EXPECTED MESSAGE BUFFER
      ;   RECMSG  - RECEIVED MESSAGE BUFFER
      ;   RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
      ;   RCVLOADU- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
      ;-

      BGNMSG  MSGEXP
MSGEXP::
      MCV     07,R0          ;ASSUME NO EXT FEATURES
      TST     EXTFEA        ;EXT FEATURES SET?
      BEQ     5$            ;BR IF NO
      MOV     08.,R0        ;EXT FEATURE BUFFER IS 8 WORDS
      JSR     PC,PRMSGEXP   ;PRINT EXPD/RECV MESSAGE BUFFERS
      ENDMMSG

L10011:
      TRAP   C$MSG

```

B6

FIFEXP - PRINT FIFO EXP/RECV DATA

```

1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591 012160
      012160
1592 012160
      012160 010146
      012162 012746 012232
      012166 012746 000002
      012172 010600
      012174 104415
      012176 062706 000006
1593 012202
      012202 012746 012 01
      012206 012746 0000 01
      012212 010600
      012214 104415
      012216 062706 000004
1594 012222 010100
1595 012224 004737 015112
1596 012230
      012230
      012230 104423
1597 012232 045 116
1598 012301 045 116
1599
1600

```

```

.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)
; RECVMSG - RECEIVED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
;
;
; BGNMSG FIFEXP
FIFEXP:
PRINTX #FIF MSG,R1 ;PRINT BYTES TRANSFERRED
MOV R1,-(SP)
MOV #FIF1MSG,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C#PNTX
ADD #6,SP
PRINTX #FIF2MSG ;PRINT HEADER MSG
MOV #FIF2MSG,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C#PNTX
ADD #4,SP
MOV R1,R0 ;GET BYTE COUNT
JSR PC,PRBYTEXP ;PRINT FIFO BYTES IN ERROR
ENDMSG

L10012:
TRAP C#MSG
;ASCIZ '#N#A NUMBER OF BYTES TRANSFERRED * #02'
;ASCIZ '#N#A FIFO DATA BYTES IN ERROR:'
.EVEN

```

MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS

```

1602                                     .SBTTL MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS
1603                                     |*
1604                                     |
1605                                     |PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
1606                                     |
1607                                     |IMPLICIT INPUTS:
1608                                     |
1609                                     |
1610                                     |   EXPMSG - EXPECTED MESSAGE BUFFER
1611                                     |   RECMSG - RECEIVED MESSAGE BUFFER
1612                                     |   RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1613                                     |   RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1614                                     |
1615                                     |   BGNMSG MSGSTAT
1616 012340 MSGSTAT::
1617 012340 012701 012402   MOV     @STATCOD,R1   |ASCII ADDRESS TABLE
1618 012344 012100   10*:  MOV     (R1)+,RO   |DONE ALL MSG LINES?
1619 012346 001410   BEQ     20*           |BR IF YES
1620 012350   PRINTX RO           |PRINT STATUS BIT NAMES
1621 012350 010046   MOV     RO,-(SP)
1622 012352 012746 000001  MOV     @1,-(SP)
1623 012356 010600   MOV     SP,RO
1624 012360 104415   TRAP   C#PRINTX
1625 012362 062706 000004  ADD     @4,SP
1626 012366 000766   BR      10*           |DO ANOTHER MSG LINE
1627 012370 012700 000012 20*:  MOV     @10,RO   |NUMBER OF WORDS IN A READ STATUS BUFFER
1628 012374 004737 014542  JSR    PC,PRMSGEXP  |PRINT EXPD/RCV MESSAGE BUFFERS
1629 012400   ENDMMSG
1630 012400
1631 012400 104423   L10013: TRAP   C#MSG
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648 013054   BGNMSG MSGLOOP
1649 013054 012701 013116 MSGLOOP::
1650 013054   MOV     @LOOPCOD,R1 |ASCII ADDRESS TABLE

```

```

STATCOD: .WORD 1#,2#,3#,4#,5#,6#,0
1#: .ASCIZ 'ANNA Tape Bus Signals in Word #8:'
2#: .ASCIZ 'ANNA PARERR<15> IEOT <12> IFMK <9> IRDY<6> IRWD<2>'
3#: .ASCIZ 'ANNA IRESV2<14> IIDENT<11> INER <8> IONL<5> IFBY<1>'
4#: .ASCIZ 'ANNA IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>'
5#: .ASCIZ 'ANNA Tape Bus Signals in Word #9:'
6#: .ASCIZ 'ANNA DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>'
.EVEN

```

.SBTTL MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS

D6

MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS

```

1650 013060 012100          10$:  MOV      (R1)+,R0          ;DONE ALL MSG LINES?
1651 013062 001410          BEQ      20$              ;BR IF YES
1652 013064          PRINTX  R0              ;PRINT STATUS BIT NAMES
      013064 010046          MOV      R0,-(SP)
      013066 012746 C00001    MOV      @1,-(SP)
      013072 010600          MOV      SP,R0
      013074 104415          TRAP    C#PNTX
      013076 062706 000004    ADD      @4,SP
1653 013102 000766          BR       10$              ;DO ANOTHER MSG LINE
1654 013104 012700 00001?   20$:  MOV      @10.,R0        ;NUMBER OF WORDS IN A READ STATUS BUFFER
1655 013110 004737 014542   JSR     PC,PRMSGEXP      ;PRINT EXPD/RECV MESSAGE BUFFERS
1656 013114          ENDMMSG
      013114          L10014:
      013114 104423          TRAP    C#MSG
1657
1658 013116 013136 013211 013310 LOOPCOD: .WORD 1$,2$,3$,4$,5$,6$,7$,0
1659 013136          045 116 045 1$:ASCIZ 'NMA Tape Bus Loopback Signals in Word #8:'
1660 013211          045 116 045 2$:ASCIZ 'NMA PARERR<15> IRESV2<14> IRESV1<13>'
1661 013310          045 116 045 3$:ASCIZ 'NMA IHISP=>IEOT<12> IWRT=>IIDENT<11> IREV =>ICER <10>'
1662 013407          045 116 045 4$:ASCIZ 'NMA IWMF =>IFMK<09> IEDIT=>IHER <08> IFAD =>ISPEED<07>'
1663 013506          045 116 045 5$:ASCIZ 'NMA ITADO=>IRDY<06> ITAD1=>IONL <05> IERASE=>ILDV <04>'
1664 013605          045 116 045 6$:ASCIZ 'NMA IREW =>IDBY<03> IRWU =>IRWD <02> IFEN =>IFBY <01>'
1665 013704          045 116 045 7$:ASCIZ 'NMA IGO =>IFPT<00>'
1666          .EVEN
1667

```

E6

MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER

```

1669          .SBTTL MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER
1670          ;*
1671          ;
1672          ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
1673          ;
1674          ;
1675          ;IMPLICIT INPUTS:
1676          ;
1677          ;     EXPMSG - EXPECTED MESSAGE BUFFER
1678          ;     RECMSG - RECEIVED MESSAGE BUFFER
1679          ;     RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1680          ;     RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1681          ;
1682          ;-
1682 013732      BGNMSG  MSGSUB
1683 013732      MSGSUB::
1683 013732 012700 000012      MOV     #10.,R0          ;SIZE OF WRITE SUBSYSTEM BUFFER
1684 013736 004737 014542      JSR     PC,PRMSGEXP      ;PRINT EXPD/RCV MESSAGE BUFFERS
1685 013742      ENDMMSG
1685 013742
1685 013742 104423
1686
1687
1688
1689
1690
1691          .SBTTL MEMADD - PRINT MEMORY ADDRESS DATA ERROR
1692          ;*
1693          ;
1694          ;PRINT ROUTINE TO PRINT MEMORY ADDRESS DATA COMPARE ERROR
1695          ;
1696          ;IMPLICIT INPUTS:
1697          ;
1698          ;     ERRHI - MEMORY ERROR HIGH ORDER ADDRESS
1699          ;     ERRLO - MEMORY ERROR LOW ORDER ADDRESS
1700          ;     EXP - EXPECTED DATA
1701          ;     RECV - RECEIVED DATA
1702          ;
1703          ;-
1703 013744      BGNMSG  MEMADD
1703 013744      MEMADD::
1704 013744 004737 010154      JSR     PC,PRIADD        ;PRINT MEMORY ADDRESS IN ERROR
1705 013750 013701 002230      MOV     EXPD,R1          ;GET EXPD DATA
1706 013754 013702 002232      MOV     RECV,R2          ;GET RECEIVED DATA
1707 013760 004737 007736      JSR     PC,PRI XOR       ;PRINT EXPD/RCV
1708 013764      ENDMMSG
1708 013764
1708 013764 104423
1709
1709

```

PRAMPKT - PRINT RAM AND PACKET DATA

```

1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732 013766
1733 013766
1734 013772 012701 002240
1735 013776 005002
1736 014000 122124
1737 014002 001005
1738 014004
1739 014014 000436
1740 014016 116105 177777
1741 014022 116403 177777
1742 014026
1743 014036 042703 177400
1744 014042 116137 177777 002232
1745 014050 116437 177777 002230
1746 014056
      014056 010346
      014060 013746 002230
      014064 013746 002232
      014070 010246
      014072 012746 014146
      014076 012746 000005
      014102 010600
      014104 104414
      014106 062706 000014
1747 014112 005202
1748 014114 005737 002300
1749 014120 001404
1750 014122 020237 002300
1751 014126 003724
1752 014130 000403
1753 014132 020227 000010
1754 014136 002720
1755 014140 005037 002300
1756 014144 000207
1757
1758 014146      045      116      045 RAMASC; .ASCIZ 'NNA BYTE; #D#A RAM; #03#A Packet; #03#A XOR;#03'
```

```

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

TSV3 - GLOBAL AREAS MACRO M1200 23-MAR-84 09:44 PAGE 40-1

G6

SEQ 0071

PRAMPKT - PRINT RAM AND PACKET DATA

1759
1760
1761

.EVEN

PRMESS - PRINT CONTENTS OF MESSAGE BUFFER

```

1763          .SBTTL  PRMESS - PRINT CONTENTS OF MESSAGE BUFFER
1764          ;+
1765          ;
1766          ; THIS ROUTINE PRINTS THE CONTENTS OF
1767          ; THE 7 OR 8 WORD MESSAGE BUFFER RETURNED BY THE
1768          ; TSV-05.
1769          ;
1770          ; INPUT:
1771          ;
1772          ;     R0     LOW ORDER ADDRESS OF MESSAGE BUFFER
1773          ;     R1     HIGH ORDER ADDRESS OF MESSAGE BUFFER
1774          ;     NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
1775          ;
1776          ; THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
1777          ;
1778          ;-
1779
1780 014232    PRMESS:
1781 014232    SAVREG                                ;SAVE THE REGISTERS
1782 014236    MOV     R0,R5                        ;SAVE LOW ORDER ADDRESS
1783 014240    TST     KTENABLE                    ;ADDRESS ABOVE 28K?
1784 014244    BNE     10$                          ;BR IF YES
1785 014246    CLR     R1                          ;SET HIGH ORDER ADDRESS TO 0
1786 014250    10$:  MOV     R1,R3                  ;SAVE HIGH ORDER ADDRESS
1787 014252    ROL     R0                          ;SHIFT BIT15 TO C BIT
1788 014254    ROL     R1                          ;SHIFT TO HIGH ORDER FOR PRINTOUT
1789 014256    PRINTX #PROASC,R1,R5                ;PRINT MESSAGE BUFFER ADDRESS
1790          014256    MOV     R5,-(SP)
1791          014260    MOV     R1,-(SP)
1792          014262    MOV     #PROASC,-(SP)
1793          014266    MOV     #3,-(SP)
1794          014272    MOV     SP,R0
1795          014274    TRAP   C:PNTX
1796          014276    ADD     #10,SP
1797          014302    PRINTX #PRIASC                ;PRINT HEADER FOR CONTENTS
1798          014302    MOV     #PRIASC,-(SP)
1799          014306    MOV     #1,-(SP)
1800          014312    MOV     SP,R0
1801          014314    TRAP   C:PNTX
1802          014316    ADD     #4,SP
1803          014322    CLR     R4                    ;NUMBER OF THE NEXT WORD
1804          014324    MOV     R5,R1                ;COPY LOW ORDER ADDRESS
1805          014326    MOV     R3,R0                ;COPY HIGH ORDER ADDRESS
1806          014330    BEQ     20$                    ;BR IF NOT ABOVE 28K
1807          014332    JSR     PC,SETMAP            ;SETUP PAR ADDRESS IN R0
1808          014336    MOV     R0,R5                ;GET PAR FORMAT ADDRESS ABOVE 28K
1809          014340    20$: PRINTX #PRASC,R4,(R5)+   ;PRINT THE CONTENTS OF MEMORY BUFFER
1810          014340    MOV     (R5)+,-(SP)
1811          014342    MOV     R4,-(SP)
1812          014344    MOV     #PRASC,-(SP)
1813          014350    MOV     #3,-(SP)
1814          014354    MOV     SP,R0
1815          014356    TRAP   C:PNTX
1816          014360    ADD     #10,SP
1817          014364    INC     R4                    ;NUMBER OF THE NEXT
1818          014366    CMP     R4,#7                ;DONE ALL YET ?
1819          014372    BGT     50$                    ;BRANCH IF ALL DONE
1800 014372    50$

```


PRMESS - PRINT CONTENTS OF MESSAGE BUFFER

```

1801 014374 002761                    BLT    20$                    ;PRINT FIRST 7 WORDS
1802 014376 032763 000200 000012      BIT    0X2.EXTF,XST2(R3);EXTENDED FEATUTES ON ?
1803 014404 001355                    BNE    20$                    ;PRINY EXTENDED STATUS WORD
1804 014406 000207                    50$:    RTS    PC                    ;RETURN
1805
1806 014410        045        116        045 PROASC: .ASCIZ 'N#A Message Buffer Address = #01#05'
1807 014455        045        116        045 PR1ASC: .ASCIZ 'N#A Message Buffer Contents:'
1808 014513        045        116        045 PRASC: .ASCIZ 'N#A    Word#D1#A; #0'
1809                                    .EVEN

```

PRMSGEXP - PRINT EXPD/RECV MESSAGE BUFFERS

```

1811          .SBTTL PRMSGEXP - PRINT EXPD/RECV MESSAGE BUFFERS
1812          ;+
1813          ;
1814          ;ROUTINE TO PRINT EXPECTED AND RECEIVED MESSAGE BUFFERS
1815          ;
1816          ;      RO      - NUMBER OF WORDS IN BUFFER
1817          ;
1818          ;IMPLICIT INPUTS:
1819          ;
1820          ;      EXPMSG - EXPECTED MESSAGE BUFFER
1821          ;      RECMMSG - RECEIVED MESSAGE BUFFER
1822          ;      RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
1823          ;      RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
1824          ;-
1825          PRMSGEXP::
1826          SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
1827          MOV            RO,R5          ;SAVE NUMBER OF WORDS
1828          MOV            RCVLOADD,RO    ;GET RECV LOW ADDRESS
1829          MOV            RO,R4          ;COPY LOW ADDRESS
1830          MOV            RCVHIADD,R1    ;GET RECV HIGH ADDRESS
1831          ROL            RO             ;SHIFT BIT15 TO C BIT
1832          ROL            R1             ;SHIFT TO HIGH ORDER FOR PRINTOUT
1833          PRINTX        @PRMSG0,R1,R4   ;PRINT MESSAGE BUFFER ADDRESS
1834          MOV            R4,-(SP)
1835          MOV            R1,-(SP)
1836          MOV            @PRMSG0,-(SP)
1837          MOV            @3,-(SP)
1838          MOV            SP,RO
1839          TRAP          C#PNTX
1840          ADD            @10,SP
1841          PRINTX        @PRMSG1          ;PRINT HEADER FOR CONTENTS
1842          MOV            @PRMSG1,-(SP)
1843          MOV            @1,-(SP)
1844          MOV            SP,RO
1845          TRAP          C#PNTX
1846          ADD            @4,SP
1847          CLR            R4              ;NUMBER OF THE CURRENT WORD
1848          MOV            @EXPMSG,R1      ;GET EXPD BUFFER ADDRESS
1849          MOV            @RECMMSG,R2     ;GET RECV BUFFER ADDRESS
20$:          MOV            (R1),RO       ;GET EXPD
1850          MOV            (R2),R3        ;GET RECV
1851          XOR            RO,R3          ;XOR EXPD/RECV
1852          PRINTX        @PRMSG2,R4,(R1)+,(R2)+,R3
1853          MOV            R3,-(SP)
1854          MOV            (R2)+,-(SP)
1855          MOV            (R1)+,-(SP)
1856          MOV            R4,-(SP)
1857          MOV            @PRMSG2,-(SP)
1858          MOV            @5,(SP)
1859          MOV            SP,RO
1860          TRAP          C#PNTX
1861          ADD            @14,SP
1862          INC            R4              ;NUMBER OF THE NEXT
1863          CMP            R4,R5          ;DONE ALL YET?
1864          BGE            50$           ;BR I: YES
1865          BR            20$           ;DO ANOTHER
1866          50$:          RTS            PC          ;RETURN

```

K6

PRMSGEXP - PRINT EXPD/RECV MESSAGE BUFFERS

```
1847
1848 014722    045    116    045 PRMSG0: .ASCIZ 'NNA Message Buffer Address = #01#05'
1849 014767    045    116    045 PRMSG1: .ASCIZ 'NNA Message Buffer Contents:'
1850 015025    045    116    045 PRMSG2: .ASCIZ 'NNA WORD #D2NA EXPD: #06NA RECV: #06NA XOR: #06'
1851          .EVEN
1852
```

PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER

```

1854          .SBTTL PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER
1855
1856          ;*
1857          ;ROUTINE TO PRINT ERROR BYTES IN MESSAGE BUFFERS
1858          ; ONLY THE FIRST 8 ERRORS ENCOUNTERED ARE PRINTED DUE TO SCREEN SPACE
1859
1860          ;
1861          ; RO - NUMBER OF BYTES IN BUFFER
1862
1863          ;IMPLICIT INPUTS:
1864          ;
1865          ; EXPMSG - EXPECTED MESSAGE BUFFER
1866          ; RECMMSG - RECEIVED MESSAGE BUFFER
1867          ;-
1867 015112 PRBYTEXP::
1868 015112          SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
1869 015116 010005          MOV      R0,R5          ;SAVE NUMBER OF BYTES
1870 015120 005037 002316          CLR      PRMNO          ;INIT ERROR COUNT
1871 015124 005004          CLR      R4          ;NUMBER OF THE CURRENT BYTE
1872 015126 012701 002320          MOV      @EXPMSG,R1          ;GET EXPD BUFFER ADDRESS
1873 015132 012702 002464          MOV      @RECMMSG,R2          ;GET RECV BUFFER ADDRESS
1874 015136 111100          20$: MOVVB  (R1),R0          ;GET EXPD BYTE
1875 015140 042700 177400          BIC      @+C<377>,R0          ;CLEAR UPPER BYTE
1876 015144 110037 015460          MOVVB  R0,PRBEXP          ;SAVE FOR ERROR REPORT
1877 015150 111203          MOVVB  (R2),R3          ;GET RECV BYTE
1878 015152 042703 177400          BIC      @+C<377>,R3          ;CLEAR UPPER BYTE
1879 015156 110337 015462          MOVVB  R3,PRBREC          ;FOR ERROR REPORT
1880 015162          XOR      R0,R3          ;XOR EXPD/RECV
1881 015172 122122          CMPB   (R1)+,(R2)+          ;EXPD = RECV?
1882 015174 001431          BEQ    30$          ;BR IF YES
1883 015176 005237 002316          INC      PRMNO          ;UPDATE ERROR COUNT
1884 015202 023727 002316 000010          CMP     PRMNO,@8          ;PRINTED 8?
1885 015210 101023          BHI    30$          ;BR IF YES
1886 015212          27$: PRINTX @PRBMSG,R4,PRBEXP,PRBREC,R3
1887 015246          MOV      R3,-(SP)
1888 015256 000404          MOV      PRBREC,-(SP)
1889 015260          MOV      PRBEXP,-(SP)
1890 015260          MOV      R4,-(SP)
1891 015270          MOV      @PRBMSG,-(SP)
1892 015270 005204          MOV      @5,-(SP)
1893 015272 020405          MOV      SP,R0
1894 015274 002001          TRAP   C:PNTX
1895 015276 000717          ADD     @14,SP
1896 015300          FORCEXIT 50$          ;000
1897 015300 013746 002316          BR     35$          ;000
1898 015304 012746 015413          30$: FORCERROR 27$,NOTSSR          ;000
1899 015310 012746 000002          35$:          ;000
1900 015314 010600          INC     R4          ;NUMBER OF THE NEXT
1901 015316 104415          CMP    R4,R5          ;DONE ALL YET?
1902          BGE    50$          ;BR IF YES
1903          BR     20$          ;DO ANOTHER
1904          50$: PRINTX @PRBTOT,PRMNO          ;PRINT TOTAL ERROR COUNT
1905          MOV      PRMNO,-(SP)
1906          MOV      @PRBTOT,-(SP)
1907          MOV      @2,-(SP)
1908          MOV      SP,R0
1909          TRAP   C:PNTX

```

PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER

```

      015320 062706 000006          ADD    #6,SP
1897 015324 000207          RTS     PC          ;RETURN
1898
1899 015326      045      116      045 PRBMSG: .ASCIZ 'N#A BYTE #D2#A EXPD: #03#A RECV: #03#A XOR: #03#
1900 015413      045      116      045 PRBTOT: .ASCIZ 'N#A NUMBER OF BYTES IN ERROR " #D2'
1901          .EVEN
1902 015460 000000          PRBEXP: .WORD 0          ;EXPD
1903 015462 000000          PRBREC: .WORD 0          ;RECV
1904

```

EXPREC - PRINT EXPD/RECV WORD DATA

1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922

015464
015464
015464 004737 007736
015470
015470
015470 104423

```

.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

```

EXPBREC - PRINT EXPD/RECV BYTE DATA

1924
 1925
 1926
 1927
 1928
 1929
 1930
 1931
 1932
 1933
 1934
 1935
 1936
 1937 015472
 015472
 1938 015472 004737 007606
 1939 015476
 015476
 015476 104423

```

      .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
      ENDMMSG
L10020: TRAP C#MSG
  
```

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 015500
 015500
 1965 015500 004737 013766
 1966 015504
 015504
 015504 104423

```

      .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
      ENDMMSG
L10021: TRAP C#MSG
  
```

1967
 1968
 1969
 1970
 1971
 1972
 1973
 1974

```

      .SBTTL RAHTADD - PRINT TEST ADDRESS, RAM AND PACKET DATA
      ;*
      ;PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
      ;
      ;INPUTS:
  
```

RAMTADD - PRINT TEST ADDRESS, RAM AND PACKET DATA

```

1975
1976          |          R4          POINTER TO COMMAND PACKET
1977          |
1978          |IMPLICIT INPUTS:
1979          |
1980          |          RAMDATA      DATA AS READ FROM THE RAM
1981          |          RAMSIZ      NUMBER OF BYTES IN PACKET
1982          |                      IF RAMSIZ=0 THEN DEFAULT TO 8.
1983          |          ERRHI      HIGH ORDER TEST ADDRESS
1984          |          ERRLO      LOW ORDER TEST ADDRESS
1985          |
1986          |IMPLICIT OUTPUTS:
1987          |
1988          |          RAMSIZ      SET TO 0
1989          |
1990          |
1991          |          BGNMSG      RAMTADD
1991 015506          |
1991 015506          |RAMTADD:
1992 015506 004737 010270          |          JSR          PC,PRITADD          ;PRINT TEST ADDRESS
1993 015512 004737 013766          |          JSR          PC,PRAMPKT          ;PRINT RAM/PACKET DATA
1994 015516          |          ENDMMSG
1994 015516          |L10022:
1994 015516 104423          |          TRAP          C#MSG
1995
1996
1997          |          .SBTTL      RAMEXP - PRINT RAM EXPD/RECV DATA
1998          |
1999          |
2000          |          PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
2001          |
2002          |INPUTS:
2003          |
2004          |          R1          RECEIVED DATA
2005          |          R2          EXPECTED DATA
2006          |          R4          CONTROLLER RAM ADDRESS
2007          |
2008          |
2009          |          BGNMSG      RAMEXP
2009 015520          |
2009 015520          |RAMEXP:
2010 015520 042702 177400          |          BIC          @*C<377>,R1          ;SAVE EXPD RAM DATA BYTE
2011 015524 042702 177400          |          BIC          @*C<377>,R2          ;SAVE EXPD RAM DATA BYTE
2012 015530 004737 010052          |          JSR          PC,PRIRAM          ;PRINT THE RAM ADDRESS
2013 015534 004737 007736          |          JSR          PC,PRIXOR          ;PRINT THE DATA
2014 015540          |          ENDMMSG
2014 015540          |L10023:
2014 015540 104423          |          TRAP          C#MSG
2015
2016          |          .SBTTL      TIMEXP - PRINT TIMER A,B AND EXP/REC
2017          |
2018          |
2019          |          PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
2020          |          AND TIMER A,B HEADER MESSAGE
2021          |
2022          |INPUTS:
2023          |
2024          |          R1          RECEIVED DATA
2025          |          R2          EXPECTED DATA

```


D7

TIMEXP PRINT TIMER A,B AND EXP/REC

```

2026
2027
2028 015542          BGNMSG  TIMEXP
      015542          TIMEXP::
2029 015542          PRINTX  @TIMSG0          ;PRINT HEADER
      015542 012746 015570          MOV      @TIMSG0,-(SP)
      015546 012746 000001          MOV      @1,-(SP)
      015552 010600          MOV      SP,RO
      015554 104415          TRAP    C@PNTX
      015556 062706 000004          ADD      @4,SP
2030 015562 004737 007736          JSR     PC,PRIXOR          ;PRINT THE DATA
2031 015566          ENDMSG
      015566          L10024:
      015566 104425          TRAP    C@MSG
2032
2033
2034 015570          045      116      045  TIMSG0: .ASCIZ  '***A TIMER A STATUS IS IN BIT 3***A TIMER B STATUS IS IN BIT 2'
2035          ,EVEN

```

BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS

```

2037                                    .SBTTL  BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS
2038
2039                                    ;*
2040                                    ;
2041                                    ;PRINT ROUTINE FOR TSSR ERRORS ON DATA TRANSFERS
2042                                    ;
2043                                    ;INPUTS:
2044                                    ;
2045                                    ;        R1        CONTENTS OF TSSR
2046                                    ;        R2        DATA WRITTEN (8 BITS)
2047                                    ;
2048                                    ;-
2049
2050  015670                            BGNMSG  BADSSR
                                  BADSSR:
2051  015670    010246                    MOV     R2,-(SP)                    ;SAVE DATA TRANSFERRED
2052  015672    042702    177400        BIC     #177400,R2                ;GET JUST ONE BYTE
2053  015676                            PRINTB  #XFERASC,R2
                                  MOV     R2,-(SP)
                                  MOV     #XFERASC,-(SP)
                                  MOV     #2,-(SP)
                                  MOV     SP,R0
                                  TRAP    C#PNTB
                                  ADD     #6,SP
2054  015720    012602                    MOV     (SP),R2                    ;RESTORE R2
2055  015722    004737    006020        JSR     PC,PRITSSR                ;DECODE TSSR CONTENTS
2056  015726                            ENDMSG
                                  L10025:
                                  TRAP    C#MSG
2057  015730    104423                    .ASCIZ  '#N#A Data Transferred = #03'
2058  015730    045                        116    045  XFERASC:

```

F7

GLOBAL SUBROUTINES SECTION

2060
2061
2062
2063
2064
2065
2066

.SBTTL GLOBAL SUBROUTINES SECTION

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

SOFINIT - SOFT INITIALIZE OF CONTROLLER

```

2068          .SBTTL  SOFINIT - SOFT INITIALIZE OF CONTROLLER
2069
2070          ;+
2071          ;
2072          ;ROUTINE TO DO A SOFT INITIALIZE OF THE CONTROLLER
2073          ;BY WRITING INTO THE TSSR REGISTER. AFTER THE INIT,
2074          ;THE TSSR REGISTER IS TESTED FOR ERRORS. ANY ERRORS
2075          ;DETECTED SHOULD BE TREATED AS DEVICE FATAL ERRORS.
2076          ;
2077          ;INPUTS:
2078          ;
2079          ;      R5      ADDRESS OF FIRST REGISTER
2080          ;
2081          ;OUTPUTS:
2082          ;
2083          ;      R0      CONTENTS OF TSSR, IF ERROR
2084          ;      CARRY   SET IF INIT WAS OKAY
2085          ;              CLEAR IF FATAL ERROR
2086          ;
2087          ;CALLING SEQUENCE:
2088          ;
2089          ;      MOV      #ADDRESS,R5
2090          ;      JSR      PC,SOFINIT
2091          ;      BCS      CONTINUE
2092          ;      ERRDF                    ;REPORT FATAL ERROR
2093          ;
2094          ;-
2095
2096 015764      SOFINIT::
2097 015764      SAVREG                    ; SAVE THE REGISTERS
2098 015770 012765 000000 000002      MOV      #0,TSSR(R5)      ; DO THE INIT.
2099 015776 004737 016240      JSR      PC,WAITF      ; WAIT FOR SSR
2100 016002 016500 000002      MOV      TSSR(R5),R0      ;GET THE TSSR REGISTER
2101 016006 010004      MOV      R0,R4      ;TSSR CONTENTS
2102 016010 042704 176277      BIC      #C<HIADDR!OFL>,R4
2103 016014 052704 002200      BIS      #SSR!NBA,R4      ;R4 HAS EXPECTED CONTENTS
2104 016020 020400      CMP      R4,R0      ;ONLY EXPECTED BITS SET ?
2105 016022 001402      BEQ      5$      ;BRANCH IF OKAY
2106 016024 000241      CLC      ;CLEAR THE CARRY FOR ERROR
2107 016026 000401      BR      10$      ;GO TO EXIT
2108 016030 000261      5$: SEC      ;SET THE CARRY BIT
2109 016032 000207      10$: RTS      PC      ;RETURN TO CALLER

```

CHKAMB - CHECK TSSR FOR AMBIGUITY

```

2111          .SBTTL  CHKAMB - CHECK TSSR FOR AMBIGUITY
2112
2113          ;*
2114          ;
2115          ;THIS ROUTINE TESTS THE CONTENTS OF THE TSSR REGISTER
2116          ;FOR AMBIGUITY
2117          ;
2118          ;INPUT:
2119          ;
2120          ;      RO      CONTENTS OF TSSR
2121          ;
2122          ;OUTPUT:
2123          ;
2124          ;      RO      CONTENTS OF TSSR
2125          ;
2126          ;      CARRY   SET - NO AMBIGUITY
2127          ;              CLR - AMBIGUOUS CONTENTS
2128          ;
2129          ;-
2130
2131  CHKAMB:
2132          SAVREG          ;SAVE THE GENERAL REGISTERS
2133          MOV             RO,R4          ;CONTENTS OF TSSR
2134          BIT             #SC,RO        ;IS BIT 15 SET ?
2135          BNE            5$           ;BRANCH IF YES
2136          BIT             #C<NBA!OFL!SSR!HIADDR>,RO ;ANY OTHER BITS SET ?
2137          BNE            40$          ;MUST BE AN ERROR
2138          BR             45$          ;RETURN WITH SUCCESS
2139          5$: BIT         #SSR,RO      ;IS READY BIT SET ?
2140          BNE            10$          ;BRANCH IF READY BIT IS SET.
2141          BIT             #BIT5,RO     ;IS FATAL ERROR BIT SET ?
2142          BEQ            40$          ;ERROR IF NOT
2143          BIC             #CTERCLS,R4   ;CLEAR ALL BUT TERMINATION CODE
2144          CMP             R4,#16      ;ALL THREE BITS MUST BE SET
2145          BNE            40$          ;ERROR IF NOT SET
2146          BR             45$          ;OK IF ALL ARE SET
2147          10$: BIT        #BIT5,RO     ;IS FATAL ERROR BIT SET ?
2148          BEQ            45$          ;ERROR IF BIT IS SET WITH SSR
2149          BIT             #BIT2!BIT1,RO ;IS THIS A FUNCTION REJECT
2150          BNE            45$          ;BR, IF TSSR IS OK
2151          40$: CLC              ;AMBIGUOUS CONTENTS
2152          BR             50$
2153          45$: SEC              ;SHOW SUCCESS - NO AMBIGUITY
2154          50$: RTS             ;RETURN TO CALLER
2155

```

ENAIN,DSBINT - ENABLE/DISABLE INTERRUPTS

```

2157          .SBTTL ENAIN,DSBINT - ENABLE/DISABLE INTERRUPTS
2158          ;
2159          ; DEFAULT DISPLAY INTERRUPT HANDLERS.
2160          ; IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
2161          ; OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
2162          ;
2163          ;
2164          ; BIT DEFINITIONS FOR "INTMASK" AND "INTFLAG" BYTES:
2165          ;
2166          ;         IOKCKIN=BIT7   ; DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
2167          ;         IOKSTP=BIT0   ; EXPECT "STOP" INTERRUPT.
2168          ;
2169          ; INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
2170 016134      000          INTMASK: .BYTE 0
2171          ; INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
2172 016135      000          INTFLAG: .BYTE 0
2173          ;
2174          ; SAVED INTERRUPT VECTOR:
2175 016136      000000      INTVEC: .WORD 0
2176          ; SAVE CPU PC
2177 016140      000000      INTCP: .WORD 0
2178          ;
2179          ; SUBROUTINE TO ENABLE INTERRUPTS:
2180 016142      010046      ENAIN: MOV    R0,-(SP)      ;SAVE R0
2181 016144      013700      002206      MOV    IVEC,R0      ;GET POINTER TO VECTORS
2182 016150      012720      016206      MOV    @INTR,(R0)+  ;SET UP INTERRUPT VECTOR
2183 016154      012720      000300      MOV    @PRI06,(R0)+
2184 016160      012600      MOV    (SP)+,R0      ;RESTORE R0
2185 016162      011646      MOV    (SP),-(SP)
2186 016164      012766      000000      000002      MOV    @0,2(SP)      ;SET CPU TO LEVEL 0
2187 016172      000002      RTI
2188          ;
2189          ; SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 6)
2190 016174      011646      DSBINT: MOV    (SP),-(SP)
2191 016176      012766      000300      000002      MOV    @PRI06,2(SP)
2192 016204      000002      RTI
2193

```

J7

INTR - INTERRUPT HANDLERS

```

2195          .SBTTL INTR - INTERRUPT HANDLERS
2196
2197 016206          BGNSRV INTR          ;DEFINE INTERRUPT ENTRY
          016206          INTR::
2198 016206 012737 000001 002222          MOV     #1,INTRECV          ;SET FLAG TO SHOW INTERRUPT RECEIVED
2199 016214 105037 016135          CLR#   INTFLAG          ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2200 016220 132737 000001 016134          BLTB   #IOKSTP,INTMASK ;EXPECTING STOP INTERRUPT?
2201 016226 001003          BNE     1$          ;BR IF YES
2202 016230 152737 000001 016135          BISB   #IOKSTP,INTFLAG ;NO. SET THE ERROR FLAG.
2203
2204          ;SAVE REGISTERS, MSG BUFFER, ETC.
2205 016236          1$:
2206 016236          ENDSRV
          016236          L10026:
          016236 000002          RTI
2207
2208

```

WAITF - WAIT FOR SUBSYSTEM READY

```

2210          .SBTTL WAITF - WAIT FOR SUBSYSTEM READY
2211          ;
2212          ; SUBROUTINE TO WAIT FOR THE SUBSYSTEM READY FLAG
2213          ;
2214          ; INPUTS:
2215          ;
2216          ; R5      ADDRESS OF FIRST DEVICE REGISTER
2217          ;
2218          ; OUTPUTS:
2219          ;
2220          ; R0      CONTENTS OF LAST TSSR READ
2221          ; CARRY  SET - READY BIT SET
2222          ;        CLR - TIMEOUT WAITING FOR READY
2223          ;
2224 016240 000401 WAITF:: BR      1$          ;NOP WHEN SUPER FIXED
2225 016242          BREAK          ; DO A SUPVSR BREAK FIRST.
          016242 104422          TRAP      C$BRK
2226 016244 012746 003000 1$: MOV      $3000,-(SP) ;300 MSEC TIMER
2227 016250 016500 000002 2$: MOV      TSSR(R5),R0 ;READ THE TSSR REGISTER
2228 016254 105700          TSTB     RO      ;TEST FOR READY BIT SET
2229          ;
2230 016256 100420          BMI      3$          ; EXIT ON STOP FLAG.
2231 016260          DELAY     1          ; WAIT 100 USEC
          016260 012727 000001          MOV      $1,(PC)+
          016264 000000          .WORD   0
          016266 013727 002116          MOV      L$DLY,(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
2232 016310 005316          DEC      (SP)          ;REDUCE DELAY COUNT
2233 016312 001356          BNE      2$          ;RETRY UNTIL TIMER EXPIRES
2234 016314 000241          CLC          ; C = 0, CONTROLLER STILL RUNNING...
2235 016316 000401          BR      4$          ;...OR HUNG-UP AFTER 300 MSEC.
2236 016320 000261          3$: SEC          ; C = 1, CONTROLLER IS STOPPED.
2237 016322 005326          4$: DEC      (SP)+
2238 016324 000207          RTS      PC          ;RESTORE STACK WITHOUT CHANGING CARRY BIT

```


CHKTSSR - CHECK TSSR FOR READY

2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258

.SBTTL CHKTSSR - CHECK TSSR FOR READY

```

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

2259 016326
2260 016326 004737 016240
2261 016332 103014
2262 016334 004737 016034
2263 016340 103006
2264 016342 032700 100000
2265 016346 001405
2266 016350 032700 074000
2267 016354 001402
2268 016356 000241
2269 016360 000401
2270 016362 000261
2271 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 0$SC, R0          ; SPECIAL CONDITION SET?
    BEQ 15$               ; BR IF NO
    BIT 0$SCE!BIE!RMR!NXM>, R0 ; ANY ERROR BITS SET?
    BEQ 15$               ; BR IF NO
    10$: CLC              ; SET FAILURE
    BR 20$                ;
    15$: SEC              ; SET SUCCESS
    20$: RTS              ; RETURN TO CALLER
    PC
```

XNXM - CHECK FOR NONEXISTENT MEMORY

```

2273          .SBTTL  XNXM  - CHECK FOR NONEXISTENT MEMORY
2274
2275          ; ROUTINE TO TEST FOR A NEXM IN THE RANGE (R1) THRU (R2).
2276          ; ON RETURN, IF "C" = 1, (R1) = NEXM ADDRESS.
2277          ; "C" = 0, ALL ADDRESSES OK.
2278
2279          ; CALL:  MOV ADR1,R1
2280          ;        MOV ADR2,R2
2281          ;        JSR PC,NXM
2282          ;        RETURN          ;TEST "C" AND PROCEED.
2283
2284 016366 012737 016420 000004 XNXM:  MOV    #2$,#0#4          ; SET BUSERR VECTOR.
2285 016374 012737 000200 000006      MOV    #PRIO4,#0#6
2286 016402 005003              CLR    R3          ; FLAG.
2287 016404 005711 1$:  TST    (R1)          ; TEST THE ADDRESS(ES).
2288
2289 016406 020102              CMP    R1,R2          ; IF ANY TRAP, CONTINUE AT 2$.
2290 016410 001407              BEQ    3$          ; OTHERWISE, CONTINUE HERE.
2291 016412 062701 000002      ADD    #2,R1          ; BR IF FINISHED (NO NEXM'S).
2292 016416 000772              BR    1$          ; SET NEXT ADDRESS...
2293
2294 016420 005103 2$:  COM    R3          ; ...AND CONTINUE.
2295 016422 012716 016430      MOV    #3$, (SP)          ; GOT ONE, SET FLAG...
2296 016426 000002              RTI
2297 016430 012700 000004      CLRVEC #4          ; ...AND DISMISS INTERRUPT...
2298 016434 104436              MOV    #4,R0          ; ...AND GIVE BACK THE VECTOR.
2299 016436 005703              TRAP  C#CVEC
2300 016440 001401              TST    R3          ; DID WE CATCH ONE ??
2301 016442 000261              BEQ    .+4          ; NO, "C" = 0, SKIP NEXT.
2302 016444 000207              SEC
2303
2304
2305
2306          .SBTTL  TSTLOOP - CHECK ITERATION COUNT
2307
2308          ; SUBROUTINE TO EXECUTE TEST ITERATIONS.
2309          ; EXIT WITH "C" SET IF LOOPS ALLOWED AND LOOP COUNT NON-ZERO.
2310          ; LOOP COUNTER IS SET BY "BEGIN,TEST" MACRO.
2311
2312          ; CALL:  LOOPTO  ARG
2313
2314          TSTLOOP:
2315 016446 005737 002166      TST    NOITS          ; ITERATIONS INHIBITED?
2316 016452 001006              BNE    1$          ; YES.
2317 016454 005737 002202      TST    QVP          ; NO.
2318 016460 100403              BMI    1$          ; LOOPS DISALLOWED IN QUICK PASS.
2319 016462 005337 002214      DEC    LOOPCNT          ; BUMP LOOP COUNTER.
2320 016466 001002              BNE    2$
2321 016470 000241 1$:  CLC          ; LOOP DISALLOWED, OR DONE.
2322 016472 000401              BR    3$
2323 016474 000261 2$:  SEC          ; LOOP ENABLED.
2324 016476 000207              RTS    PC 3$:

```

TSTLOOP - CHECK ITERATION COUNT

2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372

016500
016500 010046
016502 005037 003150
016506 005037 016746
016512 005037 005766
016516 105037 016134
016522 013700 002200
016526 006300
016530 005737 003110
016534 001430
016536 100010
016540 052760 160000 003172
016546
016546 104455
016550 000001
016552 003734
016554 005732
016556 000407
016560 052760 160001 003172 3:
016566
016566 104455
016570 000002
016572 004331
016574 000000
016576 013737 177777 003106 2:
016604
016604 013700 002200
016610 104451
016612

```

.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 RAISED TO LEVEL OF
;     THE DEVICE UNDER TEST
;
;-

TSTSETUP:
    MOV     R0, -(SP)           ; SAVE THE TEST ID MESSAGE
    CLR     SIFLAG             ; CLEAR "SOFT INIT" FLAG
    CLR     ERRK               ; CLEAR LOCAL ERROR COUNTER.
    CLR     EXTA              ; CLEAR ERROR EXTENSION FLAG.
    CLR     INTMASK           ; CLEAR INTERRUPT MASK (CHECK ERROR)
    MOV     UNITN, R0         ; GET THE UNIT NUMBER,
    ASL     R0                ; ... AND MAKE IT A WORD OFFSET.
    TST     NODEV             ; DID STARTUP FIND THE DEVICE?
    BEQ     4$                ; BR IF YES
    BPL     3$                ; BR IF NOT IDLE
    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
    ERDF   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

```

TSTSETUP - PRINT TEST NAME AND INIT ERROR COUNTS

```

016612 104444
2373 016614 000423
2374
2375 016616 4*: RFLAGS RO ; GET THE OPERATOR FLAGS.
016616 104421 TRAP C#FLA ; PRINT THE TEST NUMBERS?
2376 016620 032700 001000 BIT @PNT,RO ; BR IF NO
2377 016624 001412 BEQ 1# ; GET THE ID MESSAGE
2378 016626 011600 MOV (SP),RO ; DISPLAY THE TEST ID
2379 016630 PRINTF @TNAM,RO
016630 010046 MOV RO,-(SP)
016632 012746 016674 MOV @TNAM,-(SP)
016636 012746 000002 MOV @2,-(SP)
016642 010600 MOV SP,RO
016644 104417 TRAP C#PNTF
016646 062706 000006 ADD @6,SP
2380 016652 005237 002212 1*: INC TSTCNT ; BUMP TEST COUNTER.
2381 016656 SETPRI IPRI ; PRIORITY THAT OF DEVICE
016656 013700 002210 MOV IPRI,RO
016662 104441 TRAP C#SPRI
2382 016664 005726 5*: TST (SP), ; FIX UP THE STACK
2383 016666 013705 002204 MOV CSRADDR,R5 ; ADDRESS OF TSV REGISTERS ON UNIBUS
2384 016672 000207 RTS PC
2385 016674 045 123 045 TNAM: .ASCIZ 'SMT#A Test'
2386 .EVEN

```

TSTEND - PRINT ERRORS RECEIVED

```

2388
2389
2390
2391
2392
2393 016710
      016710 104421
2394 016712 030027 020000
2395 016716 001412
2396 016720
      016720 013746 016746
      016724 012746 016750
      016730 012746 000002
      016734 010600
      016736 104417
      016740 062706 000006
2397 016744 000207
2398
2399 016746 000000
2400 016750 045 101 040
2401 016767 105 122 122
2402
2403
2404
2405
2406
2407
2408 017034 005237 016746
2409 017040 010046
2410 017042 013700 002200
2411 017046 006300
2412 017050 062700 003172
2413 017054 005210
2414 017056 032710 007777
2415 017062 001001
2416 017064 005310
2417 017066 012600
2418 017070 000207
2419
2420 017072 010046
2421 017074 013700 002200
2422 017100 006300
2423 017102 016000 003172
2424 017106 042700 170000
2425 017112 020037 002172
2426 017116 103004
2427 017120 023737 016746 002170
2428 017126 103417
2429 017130
      017130 104421
2430 017132 032700 000040
2431 017136 001013
2432 017140 012737 177777 003106
2433 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 RO
        TRAP C1RFLA
        BIT RO, #IER
        BEQ 1$ ; BR IF "IER" NOT SET.
        PRINTF #ESUM,ERRK ; PRINT ERROR COUNT.
        MOV ERRK, -(SP)
        MOV #ESUM, -(SP)
        MOV #2, -(SP)
        MOV SP,RO
        TRAP C1PNTF
        ADD #6,SP
1$: RTS PC

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

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

CKEMAX: MOV RO, -(SP) ; SAVE RO
        MOV UNITN,RO ; GET UNIT NUMBER
        ASL RO ; ... AND MAKE IT A WORD OFFSET
        MOV ERTABL(RO),RO ; GET ERROR TABLE ENTRY
        BIC #170000,RO ; EXTRACT ERROR COUNT FIELD
        CMP RO,GERRMAX ; IS GLOBAL LIMIT EXCEEDED FOR THIS UNIT?
        BHS 1$ ; BR IF YES
        CMP ERRK,LERRMAX ; IS LOCAL LIMIT EXCEEDED FOR THIS TEST?
        BLO 2$ ; BR IF NO
1$: RFLAGS RO ; GET OPERATOR FLAGS
        TRAP C1RFLA
        BIT #IDU,RO ; IS DROPPING INHIBITED?
        BNE 2$ ; BR IF YES.
        MOV #-1,DUFLG ; NO -- DROP THE UNIT
        ERDF 4,EMAXDU
        TRAP C1ERDF
        .WORD 4
        .WORD EMAXDU
    
```

D8

INCERK - INCREMENT LOCAL ERROR COUNT

	017154	000000		.WORD	0	
2434	017156			DODU	UNITN	
	017156	013700	002200	MOV	UNITN,RO	
	017162	104451		TRAP	C#DODU	
2435	017164			DOCLN		
	017164	104444		TRAP	C#DCLN	
2436	017166	012600	24:	MOV	(SP)+,RO	; RESTORE RO
2437	017170	000207		RTS	PC	; RETURN TO CALLER
2438						

CKDROP - CHECK IF UNIT SHOULD BE DROPPED

```

2440          .SBTTL  CKDROP - CHECK IF UNIT SHOULD BE DROPPED
2441
2442          ; CHECK IF UNIT SHOULD BE DROPPED
2443          ;-
2444 017172 010046          CKDROP: MOV     RO, -(SP)
2445 017174          FORCERROR 1$,NOTSSR
2446 017204          RFLAGS  RO
2447 017206 032700 000040          TRAP   C$RFLA
2448 017212 001010          BIT     @IDU,RO
2449 017214 011600          BNE     1$
2450 017216 012737 177777 003106          MOV     (SP),RO
2451 017224          MOV     @-1,DUFLG
2452 017224 013700 002200          DODU   UNITN
2453 017230 104451          MOV     UNITN,RO
2454 017232 104444          TRAP   C$DODU          ;ABORT THE PASS
2455 017234 012600          DOCLN  TRAP   C$DOCLN
2456 017236 000207          1$:   MOV     (SP)+,RO
2457          RTS     PC
2458
2459          .SBTTL  CONFIG - DETERMINE CONFIGURATION OF SYSTEM
2460          ;
2461          ; SUBROUTINE - DETERMINE CONFIGURATION OF TSV05 SYSTEM.
2462          ;
2463 017240          CONFIG:
2464 017240 004737 015764          JSR     PC,SOFINIT
2465 017244 000207          RTS     PC
2466
2467
2468

```

KTON,KTOFF - ENABLE/DISABLE MEMORY MANAGEMENT

```

2470                                     .SBTTL KTON,KTOFF - ENABLE/DISABLE MEMORY MANAGEMENT
2471                                     ;
2472                                     ; SUBROUTINE - ENABLE MEM MGT.
2473                                     ;
2474 017246 005737 003126 KTON: TST KFLG ; GOT KT?
2475 017252 001403 BEQ 1$ ; NO.
2476 017254 012737 000001 177572 MOV #1,SRO ; YES. ENABLE KT11.
2477 017262 000207 1$: RTS PC
2478
2479
2480
2481                                     ;
2482                                     ; SUBROUTINE - DISABLE MEM MGT.
2483                                     ;
2484 017264 005737 003126 KTOFF: TST KFLG ; GOT KT11?
2485 017270 001405 BEQ 1$ ; NO.
2486 017272 000240 NOP
2487 017274 000240 NOP
2488 017276 012737 000000 177572 MOV #0,SRO ; DISABLE KT.
2489 017304 000207 1$: RTS PC
2490
2491

```


SETMAP - SETUP PAR6 MAPPING

```

2493                                   ,SBTTL SETMAP - SETUP PAR6 MAPPING
2494
2495                                   ;+
2496                                   ;
2497                                   ; THIS ROUTINE SETS UP KERNEL PAR6 TP HANDLE
2498                                   ; AN 18 BIT ADDRESS. THE OFFSET INTO THE PAGE
2499                                   ; IS RETURNED BIASED TO PAR6.
2500                                   ;
2501                                   ; INPUTS:
2502                                   ;
2503                                   ;       R0       HIGH ORDER ADDRESS BITS
2504                                   ;       R1       LOW ORDER ADDRESS BITS
2505                                   ;
2506                                   ; OUTPUTS:
2507                                   ;
2508                                   ;       R0       OFFSET INTO BLOCK WITH PAR6 BIAS (I.E. THE ADDRESS)
2509                                   ;       CARRY     SET IF SUCCESS
2510                                   ;               CLR IF ERROR
2511                                   ;-
2512                                   SETMAP:
2513                                   SAVREG                               ; SAVE R1-R4 UNTIL NEXT RETURN
2514                                   TST       KTFLG                     ; SYSTEM HAVE ABOVE 28K?
2515                                   BEQ       10$                     ; BR IF NO
2516                                   MOV       R1,R2                   ; SAVE LOW ORDER BITS
2517                                   .REPT     6
2518                                   ASR       R0                     ; CONVERT WORD ADDRESS TO 32W BLOCKS
2519                                   ROR       R1                     ; MAKE IT DOUBLE PRECISION
2520                                   .ENDR
2521                                   BIC       #177,R1                 ; ALINE FOR LOWER 4K BOUNDARY
2522                                   CMP       R1,KTFLG               ; HIGHER THAN EXISTING MEMORY?
2523                                   BHS       10$                     ; BR IF YES
2524                                   MOV       R1,#KIPAR6               ; SETUP MAPPING REGISTER PAR6
2525                                   BIC       #160000,R2             ; SETUP DISPLACEMENT IN PAGE
2526                                   ADD       #140000,R2             ; ADD IN PAR6 BIAS
2527                                   MOV       R2,R0                 ; RETURN IN R0
2528                                   SEC                               ; SET SUCCESS
2529                                   BR       15$
2530                                   10$:     CLC                     ; SET FAILURE
2531                                   15$:     RTS       PC               ; RETURN
2532

```

FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN

```

2534 .SBTTL FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN
2535 ;
2536 ; FILL MEMORY WITH BACKGROUND PATTERN
2537 ;
2538 ; INPUTS:
2539 ;
2540 ; R0 = BACKGROUND PATTERN
2541 ; FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
2542 ; KTFLG = SET TO MESHED MEMORY LOCATION IF > 28K.
2543 ;
2544 ; OUTPUTS:
2545 ; NONE
2546 ;
2547 FILLMEM:
2548 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
2549 JSR PC,KTOFF ;DISABLE KT.
2550 MOV RO,R3 ;COPY TEST PATTERN
2551 MOV FREE,R1 ;GET FIRST FREE LOCATION
2552 MOV FRESIZ,R2 ;SIZE OF FREE SPACE BELOW 28K.
2553 10$: MOV R3,(R1) ;STORE A BACKGROUND WORD
2554 DEC R2 ;DONE ALL MEMORY IN FREE SPACE?
2555 BGT 10$ ;BR IF NO
2556 TST KTFLG ; GOT KT?
2557 BEQ 55$ ; NO. GET OUT.
2558 JSR PC,KTON ; YES. ENABLE KT.
2559 CLR RO ;HIGH ORDER ADDRESS START
2560 MOV PST32W,R1 ;GET >28K START ADDRESS (IN 32W BLOCKS)
2561 .REPT 6
2562 CLC ;CLEAR C BIT
2563 ROL R1 ;CONVERT BLOCKS TO WORDS
2564 ROL RO ;MAK IT DOUBLE PRECISION
2565 .ENDR
2566 JSR PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
2567 30$: MOV R3,(RO) ;STORE TEST PATTERN IN >28K ADDRESS
2568 CMP RO,#160000 ;END OF PAR6 MAPPING AREA?
2569 BLO 30$ ;BR IF NO
2570 SUB #20000,RO ;BACKUP INTO PAR6 MAPPING BEGIN
2571 ADD #200,#KIPAR6 ;POINT TO NEXT 4K BLOCK >28K.
2572 CMP #KIPAR6,KTFLG ;END OF MEMORY?
2573 BEQ 50$ ;BR IF YES
2574 TST #23A ;11/23A?
2575 BEQ 35$ ;NO KEEP GOING
2576 MOV #0,R4 ;GET SRC CONTENTS
2577 BIC #7761,R4 ;CLEAR ALL BUT PAGE NUMBER
2578 CMP #5,R4 ;SEE IF PAGE 7
2579 BEQ 50$ ;EXIT IF THERE
2580 35$: TST #238 ;11/238?
2581 BEQ 45$ ;NO KEEP GOING
2582 CMP #KIPAR6,#7600 ;REACHED 18 BITS?
2583 BHIS 40$ ;YES
2584 BR 45$ ;NO KEEP GOING
2585 40$: MOV #20,SR3 ;SET 32 BIT RELOCATION
2586 45$: JMP 30$ ;KEEP GOING ON ETC.
2587 50$: JSR PC,KTOFF ;DISABLE KY.
2588 55$: RTS PC
2589

```

CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN

```

2591          .SBTTL  CMPMEM  - COMPARE MEMORY TO BACKGROUND PATTERN
2592          ;*
2593          ; COMPARE MEMORY WITH A BACKGROUND PATTERN
2594          ;
2595          ; INPUTS:
2596          ;
2597          ;     RO = BACKGROUND PATTERN
2598          ;     FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
2599          ;     KTF:G = SET TO HIGHEST MEMORY LOCATION IF > 28K.
2600          ;
2601          ; OUTPUTS:
2602          ;
2603          ;     CARRY = SET IF NO ERROR
2604          ;     CARRY = CLR IF ERROR
2605          ;
2606          ; IMPLICIT OUTPUTS:
2607          ;
2608          ;     ERRHI = ERROR HIGH ADDRESS
2609          ;     ERRLO = ERROR LOW ADDRESS
2610          ;     EXPD  = EXPECTED DATA
2611          ;     RECV  = RECEIVED DATA
2612          ;
2613          CMPMEM:
2614          SAVREC          ;SAVE R1-R5 UNTIL NEXT RETURN
2615          MOV            R0,R3          ;COPY TEST PATTERN
2616          JSR            PC,KTOFF       ;DISABLE KT.
2617          MOV            FREE,R1        ;GET FIRST FREE LOCATION
2618          MOV            FREE,R2        ;SIZE OF FREE SPACE BELOW 28K.
2619          10$:          CMP            R3,(R1)  ;FREE SPACE LOCATION EQUAL TO EXPD?
2620          BEQ            15$           ;BR IF YES
2621          MOV            R1,ERRLO       ;SAVE ADDRESS IN ERROR
2622          CLR            ERRHI          ;NO HIGH ADDRESS
2623          MOV            R3,EXPD        ;SAVE EXPD FOR ERROR REPORT
2624          MOV            (R1),RECV     ;SAVE RECV FOR ERROR REPORT
2625          BR            50$           ;
2626          15$:          TST            (R1)+  ;POINT TO NEXT ADDRESS
2627          DEC            R2            ;DONE ALL MEMORY IN FREE SPACE?
2628          BGT            10$           ;BR IF NO
2629          TST            KTF:G         ; GOT KT?
2630          BEQ            55$           ; NO. GET OUT.
2631          JSR            PC,KTON        ; YES. ENABLE KT.
2632          CLR            R0            ;HIGH ORDER ADDRESS START
2633          MOV            PST32W,R1     ;GET >28K START ADDRESS (IN 32W BLOCKS)
2634          .REPT          6
2635          ROL            R1            ;CONVERT BLOCKS TO WORDS
2636          ROL            R0            ;MAKE IT DOUBLE PRECISION
2637          .ENDR
2638          BIC            #177,R1       ;ALINE 4K BOUNDARY
2639          MOV            R0,-(SP)       ;SAVE HIGH ORDER
2640          MOV            R1,-(SP)       ;SAVE LOW ORDER
2641          JSR            PC,SETMAP      ;SETUP PAR6 MAPPING REGISTER
2642          MOV            R0,R4         ;COPY ADDRESS BIASED TO PAR6
2643          MOV            (SP)+,R1      ;RESTORE LOW ORDER IN NON PAR6 FORMAT
2644          MOV            (SP),R0       ;RESTORE HIGH ORDER IN NON PAR6 FORMAT
2645          30$:          CMP            R3,(R4)  ;ABOVE 28K LOCATION EQUAL EXPD?
2646          BEQ            32$           ;BR IF YES
2647          MOV            R0,ERRHI      ;SAVE HIGH ORDER IN ERROR

```

J8

CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN

2648	020030	010137	002236		MOV	R1,ERRLO	;	SAVE LOW ORDER IN ERROR
2649	020034	010337	002230		MOV	R3,EXPD	;	SAVE EXPD FOR ERROR REPORT
2650	020040	011437	002232		MOV	(R4),RECV	;	SAVE RECV FOR ERROR REPORT
2651	020044	000421			BR	50\$;	
2652	020046	062701	000002	32\$:	ADD	#2,R1	;	UPDATE NON PAR6 ADDRESS
2653	020052	005500			ADC	R0	;	MAKE IT DOUBLE PRECISION ADD
2654	020054	062704	000002		ADD	#2,R4	;	UPDATE PAR FORMAT ADDRESS
2655	020060	020427	160000		CMP	R4,#160000	;	END OF PAR6 MAPPING AREA?
2656	020064	103755			BLO	30\$;	BR IF NO
2657	020066	162704	020000		SUB	#20000,R4	;	BACKUP INTO PAR6 MAPPING BEGIN
2658	020072	062737	000200	172354	ADD	#200,#KIPAR6	;	POINT TO NEXT 4K BLOCK >28K.
2659	020100	023737	172354	003126	CMP	#KIPAR6,KTFLG	;	END OF MEMORY?
2660	020106	101744			BLOS	30\$;	BR IF NO
2661	020110	004737	017264	50\$:	JSR	PC,KTOFF	;	TURN OFF MEMORY MAPPING
2662	020114	000241			CLC		;	SET FAILURE
2663	020116	000403			BR	60\$;	
2664	020120	004737	017264	55\$:	JSR	PC,KTOFF	;	TURN OFF MEMORY MAPPING
2665	020124	000261			SEC		;	SET SUCCESS
2666	020126	000207		60\$:	RTS	PC		
2667								

REGSAV - SAVE R1-R5 ON STACK

2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703

020130
020130 010446
020132 010346
020134 010246
020136 010146
020140 010546
020142 016605
020146 004736
020150 012601
020152 012602
020154 012603
020156 012604
020160 012605
020162 000207

000012

```

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

```

GETPAT - GET 8 BIT PATTERN FROM OPERATOR

```

2705          .SBTTL  GETPAT  - GET 8 BIT PATTERN FROM OPERATOR
2706          ;+
2707          ;
2708          ;ROUTINE TO REQUEST AN 8 BIT DATA PATTERN FROM THE OPERATOR
2709          ;
2710          ;INPUTS:
2711          ;
2712          ;      NONE.
2713          ;
2714          ;OUTPUTS:
2715          ;
2716          ;      RO      OCTAL NUMBER FROM THE OPERATOR
2717          ;
2718          ;CALLING SEQUENCE:
2719          ;
2720          ;      JSR      PC,GETPAT
2721          ;
2722          ;-
2723
2724          GETPAT::
2725          SAVREG          ;SAVE THE GENERAL REGISTERS
2726          1$:  GMANID  DATASC,PATDAT,0,377,0,377,NO
                TRAP    C#GMAN
                BR      10000$
                .WORD  PATDAT
                .WORD  T#CODE
                .WORD  DATASC
                .WORD  377
                .WORD  T#LOLIM
                .WORD  T#HILIM
2727          10000$:  BNCOMPLETE  1$      ;RETRY IF ERROR
                BCC    1$
2728          MOV      PATDAT,RO      ;DATA PATTERN FROM OPERATOR
2729          RTS      PC            ;RETURN TO CALLER
2730
2731          ;+
2732          ;LOCAL DATA AREA
2733          ;-
2734
2735          PATDAT: .WORD  0          ;TEMPORARY STORAGE FOR DATA
2736          DATASC: .ASCIZ 'ENTER DATA PATTERN'
2737          .EVEN

```

GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE

```

2739          .SBTTL  GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE
2740          ;+
2741          ;
2742          ;ROUTINE TO ISSUE A MENU AND GET
2743          ;THE OPERATOR'S RESPONSE.
2744          ;
2745          ;INPUTS:
2746          ;
2747          ;      R0      ADDRESS OF ASCIZ STRING OF MENU
2748          ;      R1      MAXIMUM ALLOWABLE OPERATOR RESPONSE
2749          ;
2750          ;OUTPUTS:
2751          ;
2752          ;      R0      NUMBER OF THE OPERATOR'S SELECTION
2753          ;
2754          ;-
2755
2756          GETSEL::
2757          SAVREG          ;SAVE GENERAL REGISTERS
2758          MOV            R0,R2          ;SAVE THE MENU ADDRESS
2759          1$: MOV        R2,R3          ;START OF MENU STRING
2760          2$: TST        (R3)          ;END OF ASCII ?
2761          BEQ           3$            ;BRANCH IF ALL LINES DISPLAYED
2762          PRINTF        @SELASC,(R3)+ ;DISPLAY THE MENU
2763          MOV            (R3)+,-(SP)
2764          MOV            @SELASC,-(SP)
2765          MOV            @2,-(SP)
2766          MOV            SP,R0
2767          TRAP           C@PNTF
2768          ADD            @6,SP
2769          BR            2$
2770          3$: GMANID     MENASC,MENRES,D,-1,0,-1,NO
2771          TRAP           C@GMAN
2772          BR            10001$
2773          .WORD         MENRES
2774          .WORD         T@CODE
2775          .WORD         MENASC
2776          .WORD         -1
2777          .WORD         T@LOLIM
2778          .WORD         T@HILIM
2779          10001$: BNCOMplete      1$          ;RETRY IF ERROR
2780          BCC            1$
2781          MOV            MENRES,R0      ;GET THE OPERATOR'S REPLY
2782          CMP            R0,R1          ;COMPARE TO MAXIMUM ALLOWED
2783          BLOS           5$            ;BRANCH IF OK
2784          PRINTF        @MENERR        ;DISPLAY ERROR MESSAGE
2785          MOV            @MENERR,-(SP)
2786          MOV            @1,-(SP)
2787          MOV            SP,R0
2788          TRAP           C@PNTF
2789          ADD            @4,SP
2790          BR            1$              ;RETRY
2791          5$: RTS        PC            ;RETURN TO CALLER
2792          MENERR: .ASCIZ  '###NMA *** Menu Selection Too Large ***'
2793          SELASC: .ASCIZ  '###N#T'
2794          MENASC: .ASCIZ  'Enter Menu Selection: '

```

N8

TSV3 - GLOBAL AREAS MACRO M1200 23-MAR-84 09:44 PAGE 64-1

SEQ 0104

GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE

2775
2776 020466 000000

MENRES: .EVEN .WORD 0

CHKMAN - CHECK MANUAL INTERVENTION LEGALITY

```

2778          .SBTTL  CHKMAN  - CHECK MANUAL INTERVENTION LEGALITY
2779          ;*
2780          ;ROUTINE TO TEST FOR MANUAL INTERVENTION LEGALITY.
2781          ;
2782          ;INPUT:
2783          ;
2784          ;      NONE.
2785          ;
2786          ;OUTPUT:
2787          ;
2788          ;      CARRY  0      MANUAL INTERVENTION NOT ALLOWED
2789          ;              1      MANUAL INTERVENTION IS OK
2790          ;
2791          ;SIDE EFFECTS:
2792          ;
2793          ;      A MESSAGE IS DISPLAYED WARNING THAT TEST IS
2794          ;      NOT EXECUTED IF MANUAL INTERVENTION IS NOT
2795          ;      ALLOWED.
2796          ;
2797          ;
2798          ;
2799          ;
2800          020470          CHKMAN:  SAVREG          ;SAVE THE REGISTERS
2801          020470          MANUAL          ;SEE IF MANUAL INTERVENTION OK
2802          020474          104450          TRAP  C#MANI
2803          020476          103411          BCOMPLETE 1#          ;BRANCH IF ALLOWED
2804          020500          020524          PRINTF #NOMAN          ;PRINT THE WARNING MESSAGE
2805          020504          012746          000001          MOV  #NOMAN, -(SP)
2806          020510          010600          MOV  #1, -(SP)
2807          020512          104417          MOV  SP,RO
2808          020514          062706          000004          TRAP  C#PNTF
2809          020520          000241          ADD  #4,SP
2810          020522          000207          CLC          ;CLEAR CARRY FOR ERROR
2811          ;
2812          ;      1#:  RTS  PC          ;RETURN
2813          ;
2814          020524          045          116          045  NOMAN:  .ASCIZ  '### Manual Intervention not Allowed - Test Aborted ###'
2815          ;
2816          ;      .even

```

ENVIRN - SETUP FREE DIAGNOSTIC SPACE

```

2811          .SBTTL  ENVIRN - SETUP FREE DIAGNOSTIC SPACE
2812
2813          ; SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
2814          ;
2815          ENVIRN: MEMORY  R0
                TRAP      C#MEM
2816 020620 104431          MOV      R0,FREE          ; GET 1ST FREE ADDRESS...
2817 020626 062737 000002 003120      ADD      #2,FREE
2818 020634 011037 003122          MOV      (R0),FRESIZ      ; ...AND WORD COUNT.
2819 020640 162737 000004 003122      SUB      #4,FRESIZ
2820 020646 013702 002012          MOV      L#UNIT,R2      ; GET NUMBER OF UNITS
2821 020652 162737 000007 003122 10#;  SUB      #7,FRESIZ      ; TAKE AWAY 7 WORDS PER UNIT
2822 020660 005302          DEC      R2
2823 020662 001373          BNE     10#
2824 020664 013700 003120          MOV      FREE,R0      ;GET FIRST FREE ADDRESS
2825 020670 063700 003122          ADD      FRESIZ,R0    ;POINT TO LAST FREE ADDRESS
2826 020674 162700 000002          SUB      #2,R0      ;BACKUP 1 WORD
2827 020700 010037 003124          MOV      R0,FREEHI   ;STORE LAST FREE ADDRESS
2828 020704 000240          NOP
2829 020706 012701 177520          MOV      #BDVPCR,R1  ;GET BDV11 PCR ADDRESS
2830 020712 010102          MOV      R1,R2      ;COPY TO R2
2831 020714 062702 000002          ADD      #2,R2      ;SET THE RANGE
2832 020720 004737 016366          JSR     PC,XNXM     ;SEE IF WE HAVE ONE
2833 020724 103001          BCC     15#        ;OK TO SET FLAGS
2834 020726 000445          BR     40#        ;RETURN WITH FLAGS CLEAR
2835 020730 013701 177520 15#;  MOV      BDVPCR,R1  ;SAVE PCR CONTENTS
2836 020734 062701 000001          ADD      #1,R1      ;ADD ONE TO IT
2837 020740 012702 177520          MOV      #BDVPCR,R2 ;GET BDV11 PCR ADDRESS
2838 020744 005212          INC      (R2)      ;TRY TO WRITE TO IT
2839 020746 013703 177520          MOV      BDVPCR,R3  ;GET RESULTS
2840 020752 020103          CMP      R1,R3     ;DID IT CHANGE?
2841 020754 001017          BNE     20#        ;NO, MUST BE 11/23B
2842 020756 005237 003140          INC      T23A      ;SET THE FLAG
2843 020762 042737 170000 002120      RIC     #170000,L#HIME ;SUPERVISOR COULD BE WRONG
2844 020770 000240          NOP
2845 020772          PRINTF #M8186      ;TELL THE SYSTEM TYPE
                MOV      #M8186,-(SP)
                MOV      #1,-(SP)
                MOV      SP,R0
                TRAP     C#PNTF
                ADD      #4,SP
2846 021012 000413          BR     40#        ;RETURN
2847 021014 005237 003142 20#;  INC      T23B      ;SET THE FLAG
2848 021020 000240          NOP
2849 021022          PRINTF #M8189      ;TELL THE SYSTEM TYPE
                MOV      #M8189,-(SP)
                MOV      #1,-(SP)
                MOV      SP,R0
                TRAP     C#PNTF
                ADD      #4,SP
2850 021042 000207 40#;  RTS     PC          ;RETURN
2851

```

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```

2853          .SBTTL  KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS
2854          ;+
2855          ;
2856          ;ROUTINE TO INIT KT-11
2857          ;
2858          ;-
2859
2860          KTINIT:
2861 021044 005037 003126          CLR      KTFLG          ; INIT >28K MEMORY FLAG
2862 021050 005037 003130          CLR      KTENABLE      ; INIT TEST >28K FLAG
2863 021054 023727 002120 001577  CMP      L#HIME,#1577    ; GOT ENOUGH MEMORY (>28K)?
2864 021062 101444          BLOS     9#             ; NO.
2865 021064 013700 000004          MOV      @#ERRVEC,R0    ; SAVE OLD ERR VEC PTR,
2866 021070 012737 021162 000004  MOV      #2#,@#ERRVEC   ; SET ERR VEC PTR.
2867 021076 005737 177572          TST     @#SRO          ; GOT KT11?
2868 021102 000240          NOP                    ; (TRAP IF NO).
2869 021104 013737 002120 003126  MOV      L#HIME,KTFLG   ; YES. SET KT FLAG.
2870 021112 042737 0C0177 003126  BIC      #177,KTFLG     ;
2871 021120 010037 000004          MOV      R0,@#ERRVEC   ; RESTORE OLD ERR VEC PTR.
2872 021124 005000          CLR      R0            ; R0 = AR DATA.
2873 021126 012701 172340          MOV      #KIPAR0,R1    ; R1 = KI REGS PTR.
2874 021132 012761 077406 177740 1# : MOV      #77406,-40(R1) ; SET DESCRIPTOR REG.
2875 021140 010021          MOV      R0,(R1)+      ; SET KIPAR REG.
2876 021142 062700 000200          ADD     #200,R0        ; BUMP AR DATA BY "4K".
2877 021146 020027 002000          CMP     R0,#2000       ; AT "I/O"?
2878 021152 001367          BNE     1#             ; NO.
2879 021154 012741 177600          MOV     #177600,-(R1)  ; YES. SET KTPAR7 FOR I/O.
2880 021160 000405          BR      9#             ;
2881
2882 021162 012716 021170          2# : MOV     #6#,(SP)    ; SET UP RETURN
2883 021166 000002          RTI                    ; RTI TO NEXT LOCATION
2884
2885 021170 010037 000004          6# : MOV     R0,@#ERRVEC ; RESTORE OLD ERR VEC PTR.
2886
2887 021174 000207          9# : RTS     PC
2888

```

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```

2890      ;*
2891      ;   SUBROUTINE TO SET EXTENDED FEATURES SWITCH
2892      ;
2893      ;   Requires that SOFINIT and WRTCHR have been done previous to call.
2894      ;
2895      ;
2896      ;INPUTS:
2897      ;   R5      CURRENT UNIT NUMBER
2898      ;OUTPUTS:
2899      ;   The Extended Features Switch is set.
2900      ;
2901      ;-
2902
2903      INVERT::
2904
2905      021176 005737 002224      TST      EXTFEA      ; IS SWITCH SET?
2906      021202 001020      BNE      1$          ; YES,EXIT STAGE RIGHT!(or the next one outa town!)
2907      021204 012737 100206 021250  MOV      #100206,CMDPKT ; WRT SUB-SYS MEM CMD
2908      021212 012737 021260 021252  MOV      #WSMBK,CMDPKT+2 ; MSG BUF ADDR
2909      021220 012737 000006 021256  MOV      #6,CMDPKT+6    ; BYTE COUNT
2910      021226 012737 100010 021260  MOV      #100010,WSMBK  ; INVERT THE SWITCH
2911      021234 012704 021250      MOV      #CMDPKT,R4     ; SET CMDPKT INTO R4
2912      021240 004737 010652      JSR      PC,WRTCHR     ; DO IT
2913      021244 000207      1$:     RTS      PC                ; RETURN
2914
2915
2916      ;   COMMAND PACKET.
2917
2918      ;   "      <.,+3>&177774      ;MUST BE ON MOD 4 BOUNDRY.
2919
2920      CMDPKT:: 0          ;1ST WORD IS TS05 COMMAND.
2921      0          ;2ND WORD IS THE BUFFER LOW ADDRESS.
2922      0          ;3RD WORD IS THE BUFFER HIGH ADDRESS.
2923      0          ;4TH WORD IS THE BYTE/RECORD/FILE COUNT.
2924
2925
2926      ;   WRITE SUB-SYSTEM MEMORY CHARACTERISTIC BLOCK.
2927
2928      WSMBK:: 0          ;1ST WORD:: SEL 0
2929      0          ;2ND WORD:: SEL 2
2930      0          ;3RD WORD:: SEL 4
2931      .EVEN
2932

```

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```

2934
2935      ;+
2936      ; SUBROUTINE TO CHECK WETHER OR NOT WE'LL TEST NXM
2937      ;
2938      ;
2939      ; INPUTS:
2940      ; OUTPUTS:
2941      ; The NXMFLG is set if we can test.
2942      ; The NXMLO and NXMHI addresses are setup.
2943      ;-
2944      MEMCK::
2945
2946      SAVREG
2947      CLR NXMFLG ;SAVE THE REGISTERS
2948      CLR NXMLO ;CLEAR THE FLAG
2949      CLR NXMHI ;CLEAR THE TEST ADDRESS LO
2950      TST T23B ;CLEAR THE TEST ADDRESS HI
2951      BEQ 1$ ;IS IT A 11/23B?
2952      CMP L$HIME,07777 ;NO
2953      BLO 2$ ;GREATER THAN 128K
2954      JSR PC,NXMTST ;NO
2955      BR 13$ ;SETUP THE ADDRESS
2956      TST T23A ;SET THE FLAG AND EXIT
2957      BEQ 4$ ;IS IT A 11/23A?
2958      CMP L$HIME,05777 ;NO
2959      BHI 14$ ;GREATER THAN 96K
2960      CMP L$HIME,03777 ;YES,23A/23B WITH 128K MEMORY
2961      BLO 4$ ;GREATER THAN 64K BUT LESS THAN 92K?
2962      JSR PC,NXMTST ;NO, CHECK 24K
2963      BR 13$ ;SETUP THE ADDRESS
2964      CMP L$HIME,01577 ;SET THE FLAG AND EXIT
2965      BLO 14$ ;GREATER THAN 24K BUT LESS THAN 64K?
2966      JSR PC,NXMTST ;NO, TELL THEM AND EXIT WITH FLAG CLEAR
2967      ADD 077,NXMHI ;SETUP THE ADDRESS
2968      INC NXMFLG ;FOOL THE 11/02 & 11/03
2969      BR 15$ ;SET THE FLAG
2970      BR 15$ ;EXIT
2971      PRINTF 0NOMEM ;NOP FOR PRINTOUT
2972      MOV 0NOMEM,-(SP) ;TELL THEM & EXIT ***NO PRINT****
2973      MOV 01,-(SP)
2974      MOV SP,RO
2975      TRAP C$PNTF
2976      ADD 04,SP
2977      RTS PC ;RETURN
2978
2979      ;+
2980      ; SUBROUTINE TO SETUP THE NXM ADDRESS FOR TESTING
2981      ;
2982      ;
2983      ; OUTPUTS:NXMLO,NXMHI ;SETUP WITH NXM ADDRESS
2984      ;
2985      ;-
2986      NXMTST: MOV L$HIME,R1 ;GET TOP OF MEMORY
2987      ADD 0200,R1 ;MAKE IT I/O BLOCK OR OTHER NXM
2988      BIC 0177,R1
2989      MOV R1,R2 ;RESAVE RESULTS

```

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```
2986          000006          .REPT      6
2987          .ASL          R1          ;PUT IN PLACE FOR XFER
2988          .ENDR
2989 021474 010137 003134     MOV      R1,NXML0          ;SAVE TEST ADDRESS LOW
2990          000012          .REPT     10
2991          .ASR          R2          ;PUT IN PLACE FOR XFER
2992          .ENDR
2993 021524 042702 177700     BIC      #177700,R2        ;DON'T WANT ILA!
2994 021530 010237 003136     MOV      R2,NXMHI          ;SAVE TEST ADDRESS HIGH
2995 021534 000207          RTS      PC                ;RETURN
2996
2997
2998
2999
3000 021536          ENDMOD
```

KTINIT - SETUP KT11 MEMORY MANAGEMENT REGISTERS

```
7  
8  
9 021536 .TITLE TSV4 - MISCELLANEOUS SECTIONS  
021536 BGNMOD TSV4  
10 TSV4::  
16
```

PROTECTION TABLE

```

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

```

NO DEVICE PROTECTION REQUIRED.

J9

INITIALIZE SECTION

```

24          .SBTTL  INITIALIZE SECTION
25
26          ;**
27          ;THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
28          ;AT THE BEGINNING OF EACH PASS.
29
30          ;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
38 021546          L$INIT::
39 021546          SETVEC  #140,#170000,#340          ;ODT ROM ADDRESS          ;JB REV A-0
021546          MOV      #340,-(SP)
021552          MOV      #170000,-(SP)
021556          MOV      #140,-(SP)
021562          MOV      #3,-(SP)
021566          TRAP    C$SVEC
021570          ADD     #10,SP
40
41 021574          005037 002224          40$: CLR      EXTFL
42 021600          005037 003132          CLR      NXMFLG
43 021604          012737 006354          002176  MOV     #EPRT1,EPRTSW          ;SET UP PRIMARY MESSAGE FOR REPLACEMENT
44 021612          005037 003150          CLR      SIFLAG          ;CLEAR "SOFT INIT" FLAG
45 021616          005037 003130          CLR      KTENABLE          ;CLEAR TEST ABOVE 28K FLAG
46 021622          005037 002300          CLR      RAMSIZ          ;CLEAR RAM SIZE FOR RAMERR ROUTINE
47 021626          READEF #EF,CONTINUE
021626          MOV     #EF,CONTINUE,RO
021632          TRAP    C$REFG
48 021634          BNCOMPLETE 1$
021634          BCC     1$
49 021636          023737 002200          002012  CMP     UNITN,L$UNIT          ;UNIT IN RANGE?
50 021644          103070          BHIS    4$          ;BR IF NO.
51 021646          005737 003106          TST     DUFLG          ;DROPPED UNIT?
52 021652          100472          BMI     NXTU          ;BR IF YES
53 021654          013701 002200          MOV     UNITN,R1
54 021660          006301          ASL     R1
55 021662          005761 003172          TST     ERTABL(R1)
56 021666          001516          BEQ     SETU
57 021670          032761 040000          003172  BIT     #BIT14,ERTABL(R1)          ;DROPPED?
58 021676          001060          BNE     NXTU
59 021700          EXIT    INIT          ;DO NOTHING IF "CONTINUE".
021700          TRAP    C$EXIT
021702          .WORD  L10030-
60 021704          012700 000035          1$: READEF #EF,NEW
021710          TRAP    C$REFG
61 021712          BNCOMPLETE NXTU          ;TAKE NEXT UNIT IF NOT NEW PASS.
021712          BCC     NXTU
62 021714          READEF #EF,START
021714          MOV     #EF,START,RO
021720          TRAP    C$REFG
63 021722          BCMPLETE 2$

```

INITIALIZE SECTION

```

021722 103404      BCS      2$
64 021724      READEF  #EF.RESTART
021724 012700 000037  MOV      #EF.RESTART,RO
021730 104447      TRAP     C#REFG
65 021732      BNCOMPLETE 31$
021732 103031      ACC      31$
66 021734      2$:          ;1ST PASS, BUS-INIT...
67 021734      BRESET          ;BUS RESET.
021734 104433      TRAP     C#RESET
68 021736 005037 002212  CLR      TSTCNT          ;NUMBER OF TESTS RUN IN PASS
69 021742 005037 002220  CLR      FATFLG         ;CLEAR FATAL ERROR COUNT
70 021746 005037 003140  CLR      T23A          ;CLEAR 11/23A FLAG
71 021752 005037 003142  CLR      T23B          ;CLEAR 11/23B FLAG
72          ; MOV      #340,-(SP)
73          ; MOV      #20$,-(SP)          ;RETURN TO DEBUGGER
74          ; JMP      0,ODT          ;;ENTER THE DEBUGGER
75 021756 005037 003374  CLR      SKIPT          ;CLEAR THE SUBTEST "SKIPPER"
76 021762      20$:
77 021762 012737 177777 002202  MOV      # -1,QVP          ;...QUICK VERIFY...
78 021770 004737 020620  JSR      PC,ENVIRN        ;SET ENVIRONMENT.
79 021774 004737 021044  JSR      PC,KTINIT        ;INITIALIZE KT MEMORY MANAGEMENT
80 022000 012700 003172  MOV      #ERTABL,RO
81 022004 005020 30$:          CLR      (RO)+          ;CLEAR THE ERROR TABLE
82 022006 020027 003372  CMP      RO,#ERTABE
83 022012 103774      BLO      30$
84 022014 000404      BR      4$
85 022016 005037 002202 31$:          CLR      QVP
86 022022 000137 022072  JMP      PASRPT          ;GO REPORT THE STATUS
87
88 022026      4$:
89 022026 012737 177777 002200  NEWPAS: MOV      # -1,UNITN          ;INIT UNIT NUMBER...
90 022034 005037 002216  CLR      DEVCNT          ;CLEAR COUNT OF DEVICES RUNNING
91 022040      NXTU:
022040 104422      TRAP     C#BRK
92 022042 005237 002200  INC      UNITN          ;...AND SET NEXT UNIT NUMBER.
93 022046 023737 002200 002012  CMP      UNITN,L$UNIT
94 022054 103423      BLO      SETU
95 022056 012737 177777 003106  MOV      # -1,DUFLG
96 022064 000401      BR      11$
97 022066      DOCLN
022066 104444      TRAP     C#DCLN          ;ABORT, NO MORE UNITS.
98 022070 000240      11$:
99 022072      PASRPT:
100 022072 023727 002012 000001  CMP      L$UNIT,#1          ;HOW MANY UNITS SELECTED?
101 022100 101752      BLOS      NEWPAS          ;BR IF ONLY 1
102 022102 005737 002216  TST      DEVCNT          ;ARE ANY STILL RUNNING?
103 022106 001747      BEQ      NEWPAS          ;BR IF NO
104 022110      RFLAGS
022110 104421      TRAP     C#RFLA
105 022112 032700 000100  BIT      #ISR,RO          ;SHOULD WE PRINT STATISTICS
106 022116 001343      BNE      NEWPAS          ;BR IF NO
107
108 022120      DORPT
022120 104424      TRAP     C#DRPT
109 022122 000741      BR      NEWPAS
110 022124      10$:
111

```

INITIALIZE SECTION

```

112 022124          SETU:  GPWARD  UNITN,R0          ;GET UNIT N P-TABLE POINTER.
      022124 013700 002200  MOV      UNITN,R0
      022130 104442  TRAP     C#GPHRD
113 022132          BNCOMPLET NXTU          ;BR IF UNIT NOT AVAILABLE.
      022132 103342  BCC     NXTU
114 022134 005037 003106  CLR     DUFLG          ;CLEAR "DROPPED" FLAG.
115 022140 005237 002216  INC     DEVCNT
116 022144 012001  MOV     (R0)+,R1          ;GET 1ST REGISTER ADDRESS.
117 022146 010137 002204  MOV     R1,CSRADDR      ;ADDRESS OF REGISTERS OF UNIT UNDER TEST
118
119 022152 012001  MOV     (R0)+,R1          ;GET VECTOR ADDRESS.
120      ;MOV    (R0),R2          ;GET INTERRUPT PRIORITY
121      ;MOV    R2,IPRI          ;SET INTERRUPT PRIORITY.
122 022154 010137 002206  MOV     R1,IVEC          ;SET INTERRUPT VECTOR POINTER...
123 022160 012721 016206  MOV     @INTR,(R1)+     ;...VECTOR...
124 022164 013721 002210  MOV     IPRI,(R1)+     ;...AND PRIORITY.
125
126 022170          1$:
127      ; TST     QVP          ;1ST PASS ??
128      ; BEQ     5$          ;NO, SKIP THE PASS 1 STUFF.
129
130
131      ;
132      ;1ST PASS, CHECK THAT DEVICE ADDRESSES ARE VALID, AND
133      ;THAT THE DISPLAY STATUS IS PROPERLY INITIALIZED.
134
134 022170 013701 002200  MOV     UNITN,R1
135 022174 006301  ASL     R1
136 022176 052761 100000 003172  BIS     @BIT15,ERYABL(R1) ;SAY DEVICE RUNNING
137 022204 005037 005766  CLR     EXTA          ;CLEAR ERROR EXTENSION FLAG.
138 022210 023727 002012 000001  CMP     L#UNIT,#1      ;ARE WE TESTING MULTIPLE UNITS?
139 022216 101416  BLOS    10$          ;BR IF NO.
140 022220  RFLAGS  RO          ;YES -- GET OPERATOR FLAGS.
      022220 104421  TRAP     C#RFLA
141 022222 032700 001000  BIT     @PNT,RO          ;SHOULD WE PRINT UNIT #?
142 022226 001412  BEQ     10$          ;BR IF NOT.
143 022230  PRINTF  @PUNIT,UNITN ;PRINT THE UNIT #
      022230 013746 002200  MOV     UNITN,-(SP)
      022234 012746 022322  MOV     @PUNIT,-(SP)
      022240 012746 000002  MOV     @2,-(SP)
      022244 010600  MOV     SP,RO
      022246 104417  TRAP     C#PNTF
      022250 062706 000006  ADD     @6,SP
144 022254          10$:
145 022254 005037 003110  CLR     NODEV
146 022260 013701 002204  MOV     CSRADDR,R1      ;ADDRESS OF FIRST REGISTER
147 022264 010102  MOV     R1,R2          ;START OF REGISTERS
148 022266 062702 000002  ADD     @TSSK,R2        ;ADDRESS OF TSSR REGISTER
149 022272 004737 016366  JSR     PC,XNXM        ;TEST BOTH CONTROLLER REGISTERS...
150 022276 103005  BCC     2$          ;...AND BR IF A.L OK.
151 022300 010137 003110  MOV     R1,NODEV        ;FLAG DEVICE AS NON-EXISTENT
152 022304 012737 177777 003106  MOV     @-1,DUFLG      ;DROP THIS UNIT.
153 022312          2$:
154
155      ;
156      ;FINALLY, SET CPU PRIORITY AND WE'RE DONE.
157 022312          5$:
      022312 012700 000000  SETPRI  @PRI00          ;ENABLE INTERRUPTS.
      MOV     @PRI00,RO

```

INITIALIZE SECTION

```

158 022316 10444: TRAP C$SPRI
022320 ENDINIT
022320 L10030:
159 022320 104411 TRAP C$INIT
160 022322 045 116 045 PUNIT: .ASCIZ /#N#N#A***** TESTING UNIT *D2#A *****/
161 .EVEN

```

ADD AND DROP UNITS SECTIONS

```

163                                     .SBTTL  ADD AND DROP UNITS SECTIONS
164
165
166                                     ;++
167                                     ; THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
168                                     ; TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
169                                     ; OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
170                                     ;--
170 022370                                     BGNUA
171 022370                                     L$AU::
171 022370 010001                               MOV     R0,R1                               ; GET UNIT TO BE ADDED (R0)
172 022372 006301                               ASL     R1                                 ; MAKE IT A WORD INDEX
173 022374 052761 100000 003172                BIS     #100000,ERTABL(R1)                 ; SET THE "ACTIVE" BIT
174 022402 042761 040000 003172                BIC     #40000,ERTABL(R1)                 ; CLEAR THE "DROPPED" BIT
175 022410                                     PRINTF  #1$,R0
175 022410 010046                               MOV     R0,-(SP)
175 022412 012746 022436                               MOV     #1$,-(SP)
175 022416 012746 000002                               MOV     #2,-(SP)
175 022422 010600                               MOV     SP,R0
175 022424 104417                               TRAP   C$PNTF
175 022426 062706 000006                               ADD     #6,SP
176 022432                                     EXIT   AU
176 022432 000167                               .WORD  J$JMP
176 022434 000026                               .WORD  L10031-2-.
177 022436 045 116 045 1$:                   .ASCIZ  /#N#A UNIT #D#A ADDED/
178                                     .EVEN
179
180 022464                                     ENDAU                                     ; UNUSED.
180 022464                                     L10031:
180 022464 104452                               TRAP   C$AU
181
182                                     ;++
183                                     ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
184                                     ; TO BE REMOVED FROM THE TEST LIST.
185                                     ;
186                                     ; SUPVSR DOES THE "DROPPING". THIS IS JUST TO TELL THE MAN,
187                                     ; "DROPPED" UNITS ARE RE-SELECTED ON OPERATOR "STA" OR "ADD"
188                                     ; COMMAND, OTHERWISE REMAIN INACTIVE. THE "DISPLAY" COMMAND
189                                     ; WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
190                                     ; WHICH ARE STILL ACTIVE.
191                                     ; UPON ENTRY, R0 CONTAINS THE UNIT TO BE DROPPED.
192
192 022466                                     BGNDU
193 022466                                     L$DU::
193 022466 012737 177777 003106                MOV     #-1,DUFLG
194 022474 010001                               MOV     R0,R1
195 022476 006301                               ASL     R1
196 022500 052761 140000 003172                BIS     #140000,ERTABL(R1)                 ; SAY DROPPED
197 022506 0C0240 000240 000240                240,240,240                               ; ??????????
198 022514                                     PRINTF  #1$,R0
198 022514 010046                               MOV     R0,-(SP)
198 022516 012746 022542                               MOV     #1$, (SP)
198 022522 012746 000002                               MOV     #2,-(SP)
198 022526 010600                               MOV     SP,R0
198 022530 104417                               TRAP   C$PNTF
198 022532 062706 000006                               ADD     #6,SP
199 022536                                     EXIT   DU
199 022536 000167                               .WORD  J$JMP
199 022540 000030                               .WORD  L10032 2-.

```

ADD AND DROP UNITS SECTIONS

```

200 022542      045      116      045 10:  .ASCIZ  /MNA UNIT #D#A DROPPED/
201                .EVEN
202 022572                ENDDU
      022572                L10032:
      022572      104453          TRAP  C#DU
203
204                ;**
205                ; AUTO-DROP CODE SECTION.
206                ;--
      022574                BGNAUTO
      022574                L#AUTO:
207 022574      013705      002204          MOV  CSRADDR,R5          ;POINT TO DEVICE REGISTER
208 022600      012703      000550          MOV  #360.,R3          ;ENOUGH TIME FOR 2400' REEL TO REWIND
209 022604      004737      016240          10:  JSR  PC,WAITF          ;WAIT FOR SSR TO SET
210 022610      103420                BCS  20#              ;LEAVE WHEN SSR IS SET
211 022612                DELAY  250.          ;WAIT FOR .25 SECONDS
      022612      012727      000372          MOV  #250.,(PC)+
      022616      000000                .WORD  0
      022620      013727      002116          MOV  L#DLY,(PC)+
      022624      000000                .WORD  0
      022626      005367      177772          DEC  -6(PC)
      022632      001375                BNE  .-4
      022634      005367      177756          DEC  -22(PC)
      022640      001367                BNE  .-20
212 022642      005303                DEC  R3              ;BUMP COUNTER DOWN
213 022644      001357                BNE  10#             ;KEEP GOING
214 022646      004737      017172          JSR  PC,CKDROP        ;TRY AND DROP UNIT
215 022652                20:
216 022652                ENDAUTO          ; UNUSED.
      022652                L10033:
      022652      104461          TRAP  C#AUTO

```


CLEAN-UP AND REPORT CODING SECTIONS

```

023020 012746 000002      MOV      #2,-(SP)
023024 010600             MOV      SP,R0
023026 104416             TRAP    C#PNTS
023030 062706 000006      ADD      #6,SP
257 023034 000431             BR      4#
258 023036 020227 160001      3#:    CMP      R2,#160001      ; WAS UNIT NOT READY AT STARTUP?
259 023042 001012             BNE     30#              ; BR IF NO.
260 023044             PRINTS  #DEVNRD,R3
023044 010346             MOV      R3,-(SP)
023046 012746 023333      MOV      #DEVNRD,-(SP)
023052 012746 000002      MOV      #2,-(SP)
023056 010600             MOV      SP,R0
023060 104416             TRAP    C#PNTS
023062 062706 000006      ADD      #6,SP
261 023066 000414             BR      4#
262 023070 042702 170000      30#:   BIC      #C7777,R2
263 023074             PRINTS  #DEVDR0,R3,R2
023074 010246             MOV      R2,-(SP)
023076 010346             MOV      R3,-(SP)
023100 012746 023414      MOV      #DEVDR0,-(SP)
023104 012746 000003      MOV      #3,-(SP)
023110 010600             MOV      SP,R0
023112 104416             TRAP    C#PNTS
023114 062706 000010      ADD      #10,SP
264 023120 062704 000002      4#:    ADD      #2,R4
265 023124 005203             INC      R3
266 023126 020427 003372      CMP      R4,#ERTABE
267 023132 103701             BLO     1#
268 023134 012604             MOV      (SP)+,R4
269 023136 012603             MOV      (SP)+,R3
270 023140 012602             MOV      (SP)+,R2
271 023142             ENDRPT      ; UNUSED.
023142             L10035:
023142 104425             TRAP    C#RPT

272
273
274 023144      045      116      045  DEVSUM: .ASCIZ /#N#ADEVICE STATUS SUMMARY:#N/
275 023201      045      101      040  DEVONL: .ASCIZ /#A UNIT #D3#A ONLINE. ERRORS = #D#N/
276 023251      045      101      040  DEVNXR: .ASCIZ /#A UNIT #D3#A DROPPED, NON-EXISTENT REGISTER#N/
277 023333      045      101      040  DEVNRD: .ASCIZ /#A UNIT #D3#A DROPPED, NOT READY AT STARTUP#N/
278 023414      045      101      040  DEVDR0: .ASCIZ /#A UNIT #D3#A DROPPED, ERRORS = #D#N/
279             .EVEN
280
281 023464             ENDMOD
282
283

```


E10

TSV5A - HARDWARE TESTS MACRO M1200 23-MAR-84 09:44 PAGE 77

SEQ 0121

CLEAN-UP AND REPORT CODING SECTIONS

1
2
9
10 023464
023464
16
24

.TITLE TSV5A - HARDWARE TESTS
BGNMOD TSV5
TSV5::

TEST 1: BUS RESET TEST

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 023464          BGNTST
    023464
70 023464 012700 023662          MOV      #TST1ID,R0          ;ASCII MESSAGE TO IDENTIFY TEST
71 023470 004737 016500          JSR      PC,TSTSETUP        ;DO INITIAL TEST SETUP
72 023474 012737 000024 002214  MOV      #20.,LOOPCNT      ;PERFORM 20 ITERATIONS
73 023502          T1LOOP:
74 023502 005003          CLR      R3                ;USE R3 AS FATAL ERROR FLAG
75
76 023504          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
    023504
    023504 104402          T1.1:          TRAP      C$BSUB
77
78 023506          BRESET          ;ISSUE A BUS RESET
    023506 104433          TRAP      C$RESET
79 023510 004737 016240          JSR      PC,WAITF          ;WAIT FOR READY
80 023514 016501 00000?          MOV      TSSR(R5),R1       ;GET THE CONTENTS OF TSSR
81 023520 010102          MOV      R1,R2            ;CONTENTS OF TSSR
82 023522 042702 176277          BIC      #+C<HIADDR!OFL>,R2 ;THESE BITS MAY BE SET
```

TEST 1: BUS RESET TEST

```

83 023526 052702 002200      BIS      #SSR!NBA,R2      ;READY AND NEW DATA SHOULD BE SET
84 023532 020102             CMP      R1,R2          ;COMPARE EXPECTED TO RECEIVED
85 023534 001405             BEQ      10$            ;BRANCH IF COMPARE
89 023536             ERRDF  ERRNO,SFHERR,SFFMSG ;REPORT A FATAL ERROR
    023536 104455             TRAP      C$ERDF
    023540 000145             .WORD   101
    023542 003701             .WORD   SFHERR
    023544 012072             .WORD   SFFMSG
90 023546 005203             INC      R3              ;SET THE FATAL ERROR FLAG
91 023550             10$:  ENDSUB          ;////////////////// END SUBTEST ////////////////////
92 023550             L10037:
    023550 104403             TRAP      C$ESUB
93

```

TEST 1: BUS RESET TEST

```

95 023552 005703          TST      R3          ;DID WE HAVE FATAL ERROR ?
96 023554 001402          BEQ      20$          ;BRANCH IF NOT
97 023556 004737 017172   JSR      PC,CKDROP    ;GO DROP THIS UNIT, IF ALLOWED
98 023562 005003          CLR      R3          ;RESET FATAL ERROR FLAG
99
100
101 023564          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
    023564          T1.2:          TRAP      C$BSUB
    023564 104402
102
103 023566 005065 000002   CLR      TSSR(R5)    ;WRITE TO ISSUE A SOFT RESET
104 023572 004737 016240   JSR      PC,WAITF    ;WAIT FOR READY TO SET
105 023576 016501 000002   MOV      TSSR(R5),R1 ;GET REGISTER TSSR DATA
106 023602 010102          MOV      R1,R2       ;CONTENTS OF TSSR
107 023604 042702 176277   BIC      #C<HIADDR!OFL>,R2 ;THESE BITS MAY BE SET
108 023610 052702 002200   BIS      #SSR!NBA,R2 ;READY AND NEW DATA SHOULD BE SET
109 023614 020102          CMP      R1,R2       ;COMPARE EXPECTED TO RECEIVED
110 023616 001405          BEQ      10$          ;BRANCH IF COMPARE
114 023620          ERRDF      ERRNO,SFIERR,SFFMSG ;REPORT A FATAL ERROR
    023620 104455          TRAP      C$ERDF
    023622 000146          .WORD    102
    023624 003646          .WORD    SFIERR
    023626 012072          .WORD    SFFMSG
115 023630 005203          INC      R3          ;SET THE ERROR FLAG
116 023632          10$:
117 023632          ENDSUB          ;//////////////// END SUBTEST //////////////////
    023632 104403          L10040:          TRAP      C$ESUB
118
119
120 023634 005703          TST      R3          ;FATAL ERROR DETECTED ?
121 023636 001402          BEQ      20$          ;BRANCH IF NOT
122 023640 004737 017172   JSR      PC,CKDROP    ;SEE IF TIME TO DROP UNIT
123 023644 004737 016446   JSR      PC,TSTLOOP   ;SHOULD WE DO ITERATIONS ?
124 023650 103002          BCC      40$          ;BRANCH IF NOT
125 023652 000137 023502   JMP      T1LOOP       ;LOOP UNTIL COUNT EXPIRED
126 023656          40$:          TST
    023656 104432          TRAP      C$EXIT
    023660 000022          .WORD    L10036-.
127
128
129          ;*
130          ;LOCAL TEXT MESSAGES FOR TEST
131          ;-
132 023662          111      156      151 TST1ID: .ASCIZ 'Initialization'
133
134 023702          .EVEN
    023702          ENDTST
    023702 104401          L10036:          TRAP      C$ETST
135
136

```


TEST 3: WRAP DATA - LOW BYTE

264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290

.SBTTL 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 TRANSCIEVER 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).

291 024402
024402
296 024402 012700 025045
297 024406 004737 016500
298 024412 012737 000024 002214
299 024420 005004
300 024422 012703 000001
301 024426 005703
302 024430 001412
303 024432 005003
304 024434 004737 015764
305 024440 103406
309 024442 010001
310 024444
024444 104455
024446 000455
024450 003646
024452 012024
311 024454 005203
312 024456 005037 002220
313
314 024462
024462 104404
315
316 024464 110465 000000
317 024470 004737 016240
318 024474 103411
319 024476 010001
320 024500 010402
324 024502

```

BGNTST
T3LOOP:
5$:
10$:
BGNSEG

```

MOV #TST3ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
JSR PC,TS1SETUP ;DO INITIAL TEST SETUP
MOV #20,LOOPCNT ;PERFORM 20 ITERATIONS
CLR R4 ;STARTING DATA PATTERN
MOV #1,R3 ;FORCE SOFT INIT ON FIRST PASS
TST R3 ;SET IF INIT IS REQUIRED
BEQ 10\$;BRANCH IF NO INIT NEEDED
CLR R3 ;ASSUME NO INIT NEEDED ON NEXT LOOP
JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
BCS 10\$;BR IF SOFT INIT = OK
MOV R0,R1 ;SAVE CONTENTS OF TSSR
ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
TRAP C\$ERDF
WORD 301
WORD SFIERR
WORD SFIMSG
INC R3 ;FORCE INIT ON NEXT PASS
CLR FATFLG ;CLEAR FATAL ERROR FLAG
TRAP C\$BSEG
MOV R4,TSDB(R5) ;SET MAINT MODE + WRITE DATA
JSR PC,WAITF ;WAIT FOR SSR TO SET
BCS 15\$;BR IF CARRY SET (GOOD RETURN)
MOV R0,R1 ;SAVE CONTENTS OF TSSR
MOV R4,R2 ;DATA THAT WAS WRITTEN
ERRDF ERRNO,T3SSR,EXPREC ;DEVICE FATAL SSR FAILED TO SET

TEST 3: WRAP DATA - LOW BYTE

024502	104455						TRAP	C\$ERDF
024504	000456						.WORD	302
024506	024774						.WORD	T3SSR
024510	015464						.WORD	EXPREC
325 024512	005203			INC	R3			
326 024514	005237	002220		INC	FATFLG			
327 024520			15\$:	CKLOOP				
024520	104406							
328 024522	005737	002220		TST	FATFLG		TRAP	C\$CLP1
329 024526	001402			BEQ	20\$			
330 024530	004737	017172		JSR	PC,CKDROP			
331 024534	010402		20\$:	MOV	R4,R2			
332 024536	042702	177774		BIC	#C<BIT0!BIT1>,R2			
333 024542	000302			SWAB	R2			
334 024544	052702	002200		BIS	#SSR!NBA,R2			
335 024550	016501	000002		MOV	TSSR(R5),R1			
336 024554	032701	000100		BIT	#OFL,R1			
337 024560	001402			BEQ	25\$			
338 024562	052702	000100		BIS	#OFL,R2			
339 024566	020201		25\$:	CMP	R2,R1			
340 024570	001405			BEQ	30\$			
344 024572				ERRHRD	ERRNO,T3TSSR,EXPREC			
024572	104456						TRAP	C\$ERHRD
024574	000457						.WORD	303
024576	024730						.WORD	T3TSSR
024600	015464						.WORD	EXPREC
345 024602	005203			INC	R3			
346 024604			30\$:	CKLOOP				
024604	104406							
347 024606	016501	000000		MOV	TSBA(R5),R1		TRAP	C\$CLP1
348 024612	005002			CLR	R2			
349 024614	150402			BISB	R4,R2			
350 024616	000302			SWAB	R2			
351 024620	150402			BISB	R4,R2			
352 024622	020102			CMP	R1,R2			
353 024624	001405			BEQ	35\$			
357 024626				ERRHRD	ERRNO,T3TSBA,EXPREC			
024626	104456						TRAP	C\$ERHRD
024630	000460						.WORD	304
024632	024664						.WORD	T3TSBA
024634	015464						.WORD	EXPREC
358 024636	005203			INC	R3			
359								
360 024640			35\$:	ENDSEG				
024640								
024640	104405							
361							TRAP	C\$ESEG
362 024642	105204			INCB	R4			
363 024644	001270			BNE	5\$			
364 024646	004737	016446		JSR	PC,TSTLOOP			
365 024652	103002			BCC	40\$			
366 024654	000137	024420		JMP	T3LOUP			
367 024660			40\$:	EXIT	TST			
024660	104432							
024662	000210						TRAP	C\$EXIT
368							.WORD	L10042-
369			1+					

TEST 5: WRAP DATA - LOW BYTE

```

370          ;LOCAL TEXT MESSAGES FOR TEST
371          ;
372
373 024664      124      123      102  T3TSBA: .ASCIZ  'TSBA Incorrect After TSDB Low Write'
374 024730      124      123      123  T3TSSR: .ASCIZ  'TSSR Incorrect After TSDB Low Write'
375 024774      116      157      040  T3SSR:  .ASCIZ  'No Sub-System Ready After TSDB Low Write'
376 025045      127      162      141  TST3ID: .ASCIZ  'Wrap Data - Low Byte'
377          ;EVEN
378 025072          ;
   025072          ;
   025072 104401          ;
379
380

```

L10042: TRAP C\$ETST

TEST 4: RAM TEST

```

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 025074
      025074
430
431 025074
      025074
      025074 104402
432
437 025076 012700 026404
438 025102 004737 016500
439 025106 012737 000005 002214

```

.SBITL 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.

RGNTST

T4:;

RGNSUB

```

////////// BEGIN SUBTEST //////////
T4.1: TRAP C#BSUB

```

```

MOV #TST4ID,RO ;ASCII MESSAGE TO IDENTIFY TEST
JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
MOV #5,LOOPCNT ;PERFORM 5 ITERATIONS

```

TEST 4: RAM TEST

```

440 025114           T4LOOP:
441 025114 004737 015764   JSR    PC,SOFINIT       ;DO INITIALIZE ON CONTROLLER
442 025120 103405           BCS    20$             ;BR IF INIT WAS OK
446 025122 010001           MOV    R0,R1          ;CONTENTS OF TSSR REGISTER
447 025124           ERRDF  ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
      025124 104455           TRAP   C1ERDF
      025126 000621           .WORD 401
      025130 003646           .WORD SFIERR
      025132 012024           .WORD SFIMSG
448 025134 005004           20$: CLR    R4             ;SET RAM ADDRESS AT ZERO
449 025136 004737 016326   JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
450 025142 105065 000000   CLRB  TSDB(R5)       ;SET INTO MAINTENANCE MODE
451 025146           25$:
452 025146 104404           BGNSEG
      025146 104404           ;>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>
      025146 104404           TRAP   C1BSEG
453
454 025150 110402           MOVB  R4,R2          ;EXPECTED DATA FROM WRAP-AROUND
455 025152 004737 016326   JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
456 025156 010465 000000   MOV    R4,TSDB(R5)   ;LOAD ADDRESS INTO TSDB
457 025162 004737 016326   JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
458 025166 110265 000000   MOVB  R2,TSDB(R5)   ;LOADS DATA INTO RAM LOCATION
459 025172 004737 016326   JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
460 025176 016501 000000   MOV    TSBA(R5),R1   ;READS WRAP DATA
461 025202 120102           CMPB  R1,R2          ;DOES WRITTEN(WRAP) = READ
462 025204 001404           BEQ   30$             ;BR IF OK, THEY ARE EQUAL
466 025206           ERRHRD ERRNO,TSBAM2,EXPREC ;DATA NOT WRAPPED CORRECTLY
      025206 104456           TRAP   C1ERHRD
      025210 000622           .WORD 402
      025212 026242           .WORD TSBAM2
      025214 015464           .WORD EXPREC
467 025216           30$:
468 025216           ENDSEG
      025216 104405           ;<<<<<<<<<<<<<<<<<< END SEGMENT <<<<<<<<<<<<<<<<<<<
      025216 104405           10000$; TRAP   C1ESEG
469
470 025220 005204           INC   R4             ;NEXT ADDRESS
471 025222 020427 010000   CMP   R4,#10000     ;END OF RAM MEMORY CHECK
472 025226 001347           BNE   25$             ;LOOP TILL ALL RAM WRITTEN
473 025230           ESCAPE SUB           ;DON'T CONTINUE IF ERROR ON WRITE
      025230 104410           TRAP   C1ESCAPE
      025232 000102           .WORD L10044-.
474
475 025234 005002           CLR   R2             ;CLEAR OUT R2 HIGH BITS
476 025236 005304           DEC   R4             ;SET BACK TO 7777
477 025240 110402           MOVB  R4,R2          ;GET DATA PATTERN BACK IN SHAPE
478 025242 004737 016326   JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
479 025246 010465 000000   MOV    R4,TSDB(R5)   ;LOAD UP RAM ADDRESS POINTER
480 025252 004737 016326   JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
481 025256 016501 000000   MOV    TSBA(R5),R1   ;READ RAM CONTENTS BACK
482 025262 120102           CMPB  R1,R2          ;CHECK WITH DATA WRITTEN
483 025264 001404           BEQ   45$             ;BR IF OK, DATA IN = DATA OUT
487 025266           ERRHRD ERRNO,TSBAM2,EXPREC ;WRITTEN DATA NOT = TO READ
      025266 104456           TRAP   C1ERHRD
      025270 000623           .WORD 403
      025272 026242           .WORD TSBAM2
      025274 015464           .WORD EXPREC
488 025276           45$: CKLOOP       ;SCOPE LOOP

```

D11

TEST 4: RAM TEST

```

025276 104406
489 025300 116501 000001          MOVB   TSBAH(R5),R1          ;HIGH BYTE READ OF TSBA      TRAP   C$CLP1
490 025304 010402                MOV    R4,R2                ;DATA PATTERN WRITTEN
491 025306 000302                SWAB  R2                    ;HIGH TO LOW
492 025310 060402                ADD   R4,R2                ;TOTAL OF BYTES IN LOW BYTE
493 025312 120102                CMPB  R1,R2                ;SUM OF BYTES WRITTEN TO TSDB = TSBAH
494 025314 001404                BEQ   50$                  ;BR IF OK, THEY SHOULD BE
498 025316                ERRHRD ERRNO,M2901,EXPREC ;2901 PROBLEM ADDER
                                TRAP   C$ERHRD
                                .WORD  404
                                .WORD  M2901
                                .WORD  EXPREC
025316 104456
025320 000624
025322 026152
025324 015464
499 025326                50$: CKLOOP                ;SCOPE LOOP
                                TRAP   C$CLP1
025326 104406                DEC   R4                    ;DROP DATA COUNTER (PATTERN)
500 025330 005304                BGE  40$                   ;NOT AT LOC. ZERO YET
501 025332 002342
502
503 025334                ENDSUB                      ;////////////////// END SUBTEST ////////////////////
                                L10044:
025334 104403                TRAP   C$ESUB
504

```


TEST 4: RAM TEST

```

559 025506          ERRHRD  ERRNO,TSBAM3,EXPREC      ;CHARACTERISTICS DATA NOT CORRECT
      025506 104456          TRAP          C$ERHRD
      025510 000627          .WORD        407
      025512 026324          .WORD        TSBAM3
      025514 015464          .WORD        EXPREC
560 025516 012702 000377    43$:  MOV      #000377,R2      ;SET ALL ONES WORD
561 025522 010465 000000    MOV      R4,TSDB(R5)    ;LOAD UP RAM ADDRESS POINTER
562 025526 004737 016326    JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
563 025532 110265 000000    MOVB     R2,TSDB(R5)    ;WRITE DATA INTO RAM
564 025536 004737 016326    JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
565 025542 016501 000000    MOV      TSBA(R5),R1    ;READ RAM CONTENTS BACK
566 025546 120102          CMPB     R1,R2          ;CHECK WITH DATA WRITTEN
567 025550 001404          BEQ      45$           ;SR IF OK, DATA IN = DATA OUT
571 025552          ERRHRD  ERRNO,TSBAM2,EXPREC      ;WRITTEN DATA NOT = TO READ
      025552 104456          TRAP          C$ERHRD
      025554 000630          .WORD        408
      025556 026242          .WORD        TSBAM2
      025560 015464          .WORD        EXPREC
572 025562          45$:  CKLOOP      ;SCOPE LOOP
      025562 104406          TRAP          C$CLP1
573 025564 004737 016326    JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
574 025570 010465 000000    MOV      R4,TSDB(R5)    ;WORD WRITE TO SET UP ADDRESS
575 025574 004737 016326    JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
576 025600 116501 000001    MOVB     TSBAH(R5),R1   ;HIGH BYTE READ OF TSBA
577 025604 010403          MOV      R4,R3          ;DATA PATTERN WRITTEN
578 025606 000303          SWAB     R3            ;HIGH TO LOW
579 025610 060403          ADD      R4,R3          ;TOTAL OF BYTES IN LOW BYTE
580 025612 120103          CMPB     R1,R3          ;SUM OF BYTES WRITTEN TO TSDB = TSBAH
581 025614 001404          BEQ      50$           ;BR IF OK, THEY SHOULD BE
585 025616          ERRHRD  ERRNO,M2901,EXPREC      ;2901 PROBLEM ADDER
      025616 104456          TRAP          C$ERHRD
      025620 000631          .WORD        409
      025622 026152          .WORD        M2901
      025624 015464          .WORD        EXPREC
586 025626          50$:  CKLOOP      ;SCOPE LOOP
      025626 104406          TRAP          C$CLP1
587 025630 005304          DEC      R4            ;DROP RAM ADDRESS POINTER
588 025632 002312          BGE      40$           ;NOT AT LOC. ZERO YET
589
590 025634          ENDSUB      ;////////////////// END SUBTEST ////////////////////
      025634          L10045:
      025634 104403          TRAP          C$ESUB
591

```

TEST 4: RAM TEST

```
593
594 025636      BGNSUB              ;///////////////// BEGIN SUBTEST ///////////////////
      025636                                T4.3:
      025636 104402                      TRAP      C$BSUB
595
596            ; TEST 4, SUBTEST 3
597            ;
598            ; THIS SUBTEST WRITES RAM WITH ALL ONES
599            ; THEN WALKS A ZERO WORD DOWN THROUGH MEMORY
600            ;
601 025640 004737 015764  JSR      PC,SOFINIT      ;DO INITIALIZE ON CONTROLLER
602 025644 103405          BCS      20$            ;BR IF INIT WAS OK
606 025646 010001          MOV      R0,R1          ;CONTENTS OF TSSR REGISTER
607 025650          ERRDF  ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
      025650 104455                      TRAP      C$ERDF
      025652 000632                      .WORD   410
      025654 003646                      .WORD   SFIERR
      025656 012024                      .WORD   SFIMSG
608 025660 012702 177777 20$:  MOV      #177777,R2      ;SET DATA AT ALL ONES
609 025664 005004          CLR      R4             ;SET RAM ADDRESS AT ZERO
610 025666 004737 016326  JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
611 025672 105065 000000  CLR     TSDB(R5)      ;SET INTO MAINTENANCE MODE
612 025676          25$:
613 025676          BGNSEG              ;>>>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>
      025676 104404                      TRAP      C$BSEG
614
615 025700 004737 016326  JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
616 025704 010465 000000  MOV      R4,TSDB(R5)   ;LOAD ADDRESS INTO TSDB
617 025710 004737 016326  JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
618 025714 110265 000000  MOV     R2,TSDB(R5)   ;LOADS DATA INTO RAM LOCATION
619 025720 004737 016326  JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
620 025724 016501 000000  MOV     TSBA(R5),R1   ;READS WRAP DATA
621 025730 120102          CMP     R1,R2          ;DOES WRITTEN(WRAP) = READ ?
622 025732 001404          BEQ     30$            ;BR IF OK, THEY ARE EQUAL
626 025734          ERRHRD  ERRNO,TSBAM2,EXPREC ;DATA NOT WRAPPED CORRECTLY
      025734 104456                      TRAP      C$ERHRD
      025736 000633                      .WORD   411
      025740 026242                      .WORD   TSBAM2
      025742 015464                      .WORD   EXPREC
627 025744          30$:
628 025744          ENDSEG             ;<<<<<<<<<<<<<<< END SEGMENT <<<<<<<<<<<<<<<<<<<<<<<<<
      025744 104405                      10000$:
      025744          TRAP      C$ESEG
629
630 025746 005204          INC      R4             ;NEXT ADDRESS
631 025750 020427 010000  CMP     R4,#10000     ;END OF RAM MEMORY CHECK
632 025754 001350          BNE     25$            ;BR, MORE RAM TO GO
633
634 025756          ESCAPE  SUB          ;NO CHECK IF WRITTEN INCORRECTLY
      025756 104410                      TRAP      C$ESCAPE
      025760 000152                      .WORD   L10046-
635 025762 005304          35$:  DEC      R4             ;SET BACK TO 777
636 025764 004737 016326  40$:  JSR      PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
637 025770 012702 000377  MOV     #000377,R2   ;SET UP EXPECTED DATA REGISTER
638 025774 005001          CLR     R1             ;CLEAN OUT REGISTER
639 025776 010465 000000  MOV     R4,TSDB(R5)   ;SELECT ADDRESS IN RAM
640 026002 004737 016326  JSR      PC,CHKTSSR     ;WAIT FOR READY (SSR)
```


TEST 4: RAM TEST

```

641 026006 016501 000000      MOV     TSBA(R5),R1      ;PICK UP RAM CONTENTS
642 026012 120102             CMPB   R1,R2           ;IS MEMORY STILL ALL ONES
643 026014 001404             BEQ    43$            ;BR, IF OK (ALL ONES)
647 026016             ERRHRD  ERRNO,TSBAM3,EXPREC ;MEMORY CHANGED AFTER ALL ONES WRITE
                                TRAP    C$ERHRD
                                .WORD  412
                                .WORD  TSBAM3
                                .WORD  EXPREC
        026016 104456
        026020 000634
        026022 026324
        026024 015464
648 026026 005002             43$:  CLR     R2           ;SET UP NEW EXPECTED
649 026030 010465 000000      MOV     R4,TSDB(R5)    ;LOAD UP RAM ADDRESS POINTER
650 026034 004737 016326      JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
651 026040 110265 000000      MOVB   R2,TSDB(R5)    ;WRITE DATA INTO RAM
652 026044 004737 016326      JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
653 026050 016501 000000      MOV     TSBA(R5),R1    ;READ RAM CONTENTS BACK
654 026054 120102             CMPB   R1,R2           ;CHECK WITH DATA WRITTEN
655 026056 001404             BEQ    45$            ;BR IF OK, DATA IN = DATA OUT
659 026060             ERRHRD  ERRNO,TSBAM2,EXPREC ;WRITTEN DATA NOT = TO READ
                                TRAP    C$ERHRD
                                .WORD  413
                                .WORD  TSBAM2
                                .WORD  EXPREC
        026060 104456
        026062 000635
        026064 026242
        026066 015464
660 026070             45$:  CKLOOP          ;SCOPE LOOP
        026070 104406                                TRAP    C$CLP1
661 026072 004737 016326      JSR    PC,CHKTSSR     ;WAIT FOR READY, NON-AMBIGUOUS
662 026076 116501 000001      MOVB   TSBAM(R5),R1   ;HIGH BYTE READ OF TSBA
663 026102 010203             MOV     R2,R3          ;DATA PATTERN WRITTEN
664 026104 000303             SWAB   R3              ;HIGH TO LOW
665 026106 060203             ADD    R2,R3           ;TOTAL OF BYTES IN LOW BYTE
666 026110 120103             CMPB   R1,R3           ;SUM OF BYTES WRITTEN TO TSDB = TSBAM
667 026112 001404             BEQ    50$            ;BR IF OK, THEY SHOULD BE
671 026114             ERRHRD  ERRNO,M2901,EXPREC ;2901 PROBLEM ADDER
                                TRAP    C$ERHRD
                                .WORD  414
                                .WORD  M2901
                                .WORD  EXPREC
        026114 104456
        026116 000636
        026120 026152
        026122 015464
672 026124             50$:  CKLOOP          ;SCOPE LOOP
        026124 104406                                TRAP    C$CLP1
673 026126 005304             DEC    R4              ;DROP RAM ADDRESS POINTER
674 026130 001315             BNE   40$            ;NOT AT LOC. ZERO YET
675
676 026132             ENDSUB          ;////////////////// END SUBTEST ////////////////////
        026132                                L10046:
        026132 104403                                TRAP    C$ESUB
677
678 026134 004737 016446      JSR    PC,TSTLOOP     ;DO WE NEED TO ITERATE TEST ?
679 026140 103002             BCC   63$            ;BRANCH IF NOT
680 026142 000137 025114      JMP    T4LOOP         ;EXECUTE AGAIN
681 026146             63$:  EXIT     TST          ;ALL DONE THIS TEST
        026146 104432                                TRAP    C$EXIT
        026150 000256                                .WORD  L10043-
682
683 ;*
684 ;LOCAL TEXT MESSAGES FOR TEST
685 ;-
686 026152 040 124 123 M2901: .ASCIZ ' TSBA High Byte Not Sum of Last TSDB Write (2901 Error)'
687 026242 040 127 162 TSBAM2: .ASCIZ ' Write to TSDD Not Equal to Read of TSBA Low Byte'
688 026324 127 162 151 TSBAM3: .ASCIZ ' Write To RAM Location Modified Another Location'

```

TEST 4: RAM TEST

689 026404 122 101 115 TST4ID: .ASCIZ 'RAM Verification'
690 .EVEN
691 026426 .ENDTST
026426
026426 104401
692

L10043: TRAP C\$ETST

TEST 5: SECOND INITIALIZATION TEST

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 026430          BGNTST
      026430
736 026430 012700 027402      MOV    #TST5ID,R0      ;ASCII MESSAGE TO IDENTIFY TEST
737 026434 004737 016500      JSR    PC,TSTSETUP    ;DO INITIAL TEST SETUP
738 026440 012737 000024 002214  MOV    #20,,LOOPCNT  ;PERFORM 20 ITERATIONS
739 026446          T5LOOP:
740 026446 005037 002220      CLR    FATFLG        ;CLEAR THE FATAL ERROR FLAG
741
742 026452          BGNSUB
      026452
      026452 104402          ;//////////////// BEGIN SUBTEST //////////////////
                          T5.1:
743                                TRAP    C#BSUB
744 026454 004737 015764      JSR    PC,SOFINIT    ;DO A SOFT TO START
745 026460 103404          BCS    10#           ;BRANCH IF O.K.
749 026462          ERRDF  ERRNO,SFIERR,SFIMSG  ;REPORT ERROR AND DROP DRIVE
                          TRAP    C#ERDF
                          .WORD   501
                          .WORD   SFIERR
                          .WORD   SFIMSG
750 026472 012702 177777      10#:  MOV    #-1,R2      ;ALL ONE DATA PATTERN

```

TEST 5: SECOND INITIALIZATION TEST

```

751 026476 005004          CLR      R4          ;STARTING RAM ADDRESS
752 026500 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
753 026504 105065 000000   CLRB    TSDB(R5)    ;SET MAINTENANCE MODE
15$: 754 026510 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
755 026514 010465 000000   MOV     R4,TSDB(R5) ;SET THE NEXT RAM ADDRESS
756 026520 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
757 026524 110265 000000   MOVB   R2,TSDB(R5) ;LOAD TEST DATA
758 026530 005204          INC     R4          ;NEXT ADDRESS TO TEST
759 026532 020427 007777   CMP     R4,#7777    ;COMPARE TO LAST ADDRESS
760 026536 003762          BLE     15$        ;BRANCH TILL ALL DATA WRITTEN
761 026540          BRESET          ;ISSUE A BUS RESET
          TRAP      C$RESET
762 026542 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
763 026546 016501 000002   MOV     TSSR(R5),R1 ;GET THE CONTENTS OF TSSR
764 026552 010102          MOV     R1,R2       ;CONTENTS OF TSSR
765 026554 042702 176277   BIC     #C<HIADDR!OFL>,R2 ;THESE BITS MAY BE SET
766 026560 052702 002200   BTS    #SSR!NBA,R2 ;READY AND NEW DATA SHOULD BE SET
767 026564 020102          CMP     R1,R2       ;COMPARE EXPECTED TO RECEIVED
768 026566 001406          BRQ    20$        ;BRANCH IF COMPARE
772 026570          ERDF   ERRNO,SFHERR,SFFMSG ;REPORT A FATAL ERROR
          TRAP      C$ERDF
          .WORD    502
          .WORD    SFHERR
          .WORD    SFFMSG
773 026600 005237 002220   INC     FATFLG      ;SET FATAL ERROR FLAG
774 026604          CKLOOP          ;LOOP ON ERROR IF FLAG SET
20$: 775 026606          ESCAPE  SUB        ;EXIT IF FATAL ERROR DETECTED
          TRAP      C$CLP1
          TRAP      C$ESCAPE
          .WORD    '10050-'
776 026612 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR SSR TO SET
777 026616 105065 000000   CLRB    TSDB(R5)    ;PUT BACK INTO MAINTENANCE MODE
778 026622 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
779 026626 005065 000000   CLR     TSDB(R5)    ;SET ADDRESS BACK TO 0000
780 026632 012702 000377   MOV     #377,R2     ;
781 026636 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
782 026642 110265 000000   MOVB   R2,TSDB(R5) ;SHOULD POINT TO RAM 0
783 026646 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
784 026652 005065 000000   CLR     TSDB(R5)    ;SELECT LOCATION 0
785 026656 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
786 026662 116501 000000   MOVB   TSBA(R5),R1 ;READ RAM LOCATION SPECIFIED
787 026666 120102          CMPB   R1,R2       ;LOCATION SHOULD BE 377 OCTAL
788 026670 001406          BEQ    25$        ;BR IF OK
789 026672          ERDF   ERRNO,TSADDR,EXPREC ;WASN'T POINTING TO CORRECT LOC.
          TRAP      C$ERDF
          .WORD    502
          .WORD    TSADDR
          .WORD    EXPREC
790 026702 005237 002220   INC     FATFLG      ;SET THE FATAL ERROR FLAG
791 026706          CKLOOP          ;SCOPE LOOP
25$: 792 026710          ESCAPE  SUB        ;NO MORE CHECKS IF FATAL ERROR
          TRAP      C$CLP1
          TRAP      C$ESCAPE
          .WORD    '10050-'
793 026714 012704 000310   MOV     #310,R4     ;START WITH LOC 310
794 026720 005002          CLR     R2         ;MEMORY EXPECTED SHOULD BE 000000
795 026722 004737 016326   JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS

```

TEST 5: SECOND INITIALIZATION TEST

```

796 026726 010465 000000      30$:  MOV    R4,TSDB(R5)      ;SELECT LOCATION SPECIFIED
797 026732 004737 016326      JSR    PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
798 026736 116501 000000      MOVB   TSBA(R5),R1    ;READ LOC CONTENTS
799 026742 120102      CMPB   R1,R2          ;CHECK MEMORY FOR 000000
800 026744 001406      BEQ    40$           ;BRANCH IF DATA OKAY
801 026746      ERRDF  ERRNO,TSMEM,SFFMSG ;MEMORY NOT ZERO AFTER INIT.
      026746 104455      TRAP   C$ERDF
      026750 000766      .WORD 502
      026752 027432      .WORD TSMEM
      026754 012072      .WORD SFFMSG
802 026756 005237 002220      40$:  INC    FATFLG      ;SET THE FATAL ERROR FLAG
803 026762      CKLOOP
      026762 104406      TRAP   C$CLP1
804 026764      ESCAPE  SUB      ;EXIT ON FATAL ERROR
      026764 104410      TRAP   C$ESCAPE
      026766 000012      .WORD L10050-.
805 026770 005204      INC    R4            ;LOOK AT NEXT RAM LOC.
806 026772 020427 000400      CMP    R4,#400      ;AT TOP OF RAM ADDRESS SPACE
807 026776 001353      BNE   30$           ;BRANCH TILL ALL MEMORY TESTED
808
809 027000      ENDSUB           ;////////////////// END SUBTEST ////////////////////
      027000      L10050:
      027000 104403      TRAP   C$ESUB
810
811 027002 005737 002220      TST    FATFLG      ;IS FATAL ERROR FLAG SET ?
812 027006 001404      BEQ    50$         ;BRANCH IF NOT
813 027010 004737 017172      JSR    PC,CKDROP    ;NO LOOP, TRY TO DROP DEVICE
814 027014 005037 002220      CLR    FATFLG      ;CLEAR THE FATAL ERROR FLAG
815 027020      50$:

```

TEST 5: SECOND INITIALIZATION TEST

```

817 027020          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      027020          T5.2:
      027020 104402          TRAP          C$BSUB
818
819 027022 004737 015764      JSR      PC,SOFINIT      ;DO A SOFT TO START
820 027026 103404          BCS      10$          ;BRANCH IF O.K.
824 027030          ERRDF      ERRNO,SFIERR,SFIMSG      ;REPORT ERROR AND DROP DRIVE
      027030 104455          TRAP          C$ERDF
      027032 000767          .WORD      503
      027034 003646          .WORD      SFIERR
      027036 012024          .WORD      SFIMSG
825 027040 012702 177777      10$:  MOV      #-1,R2          ;ALL ONE DATA PATTERN
826 027044 005004          CLR      R4          ;STARTING RAM ADDRESS
827 027046 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
828 027052 105065 000000      15$:  CLRB     TSDB(R5)      ;SET MAINTENANCE MODE
829 027056 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
830 027062 010465 000000      MOV      R4,TSDB(R5)      ;SET THE NEXT RAM ADDRESS
831 027066 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
832 027072 110265 000000      MOVB    R2,TSDB(R5)      ;LOAD TEST DATA
833 027076 005204          INC      R4          ;NEXT ADDRESS TO TEST
834 027100 020427 007777      CMP      R4,#7777      ;COMPARE TO LAST ADDRESS
835 027104 003762          BLE      15$          ;BRANCH TILL ALL DATA WRITTEN
836 027106 005065 000002      CLR      TSSR(R5)      ;ISSUE A SOFT RESET
837 027112 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
838 027116 016501 000002      MOV      TSSR(R5),R1    ;GET THE CONTENTS OF TSSR
839 027122 010102          MOV      R1,R2          ;CONTENTS OF TSSR
840 027124 042702 176277      BIC     #C<#IADDR!OFL>,R2 ;THESE BITS MAY BE SET
841 027130 052702 002200      BIS     #SSR!NBA,R2     ;READY AND NEW DATA SHOULD BE SET
842 027134 020102          CMP      R1,R2          ;COMPARE EXPECTED TO RECEIVED
843 027136 001406          BEQ     20$          ;BRANCH IF COMPARE
847 027140          ERRDF      ERRNO,SFHERR,SFFMSG      ;REPORT A FATAL ERROR
      027140 104455          TRAP          C$ERDF
      027142 000770          .WORD      504
      027144 003701          .WORD      SFHERR
      027146 012072          .WORD      SFFMSG
848 027150 005237 002220      INC      FATFLG        ;SET FATAL ERROR FLAG
849 027154          20$:  CKLOOP     ;LOOP ON ERROR IF FLAG SET
      027154 104406          TRAP          C$CLP1
850 027156          ESCAPE    SUB          ;EXIT IF FATAL ERROR DETECTED
      027156 104410          TRAP          C$ESCAPE
      027160 000170          .WORD      L10051-.
851 027162 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
852 027166 105065 000000      CLRB     TSDB(R5)      ;PUT BACK INTO MAINTENANCE MODE
853 027172 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
854 027176 005065 000000      CLR      TSDB(R5)      ;SET ADDRESS BACK TO 0000
855 027202 012702 000377      MOV      #377,R2      ;
856 027206 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
857 027212 110265 000000      MOVB    R2,TSDB(R5)      ;SHOULD POINT TO RAM 0
858 027216 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
859 027222 005065 000000      CLR      TSDB(R5)      ;SELECT LOCATION 0
860 027226 004737 016326      JSR      PC,CHKTSSR      ;WAIT FOR READY, NON-AMBIGUOUS
861 027232 116501 000000      MOVB    TSBA(R5),R1    ;READ RAM LOCATION SPECIFIED
862 027236 120102          CMPB    R1,R2          ;LOCATION SHOULD BE 377 OCTAL
863 027240 001406          BEQ     25$          ;BR IF OK
864 027242          ERRDF      ERRNO,T5ADDR,EXPREC      ;WASN'T POINTING TO CORRECT LOC.
      027242 104455          TRAP          C$ERDF
      027244 000770          .WORD      504

```

TEST 5: SECOND INITIALIZATION TEST

```

      027246 027470                                .WORD  T5ADDR
      027250 015464                                .WORD  EXPREC
865 027252 005237 002220          INC    FATFLG          ;SET THE FATAL ERROR FLAG
866 027256                104406          CKLOOP          ;SCOPE LOOP
867 027260                104410          ESCAPE  SUB      ;NO MORE CHECKS IF FATAL ERROR
      027262 000066                                .WORD  L10051-.
868 027264 012704 000310          MOV     #310,R4    ;START WITH LOC 310
869 027270 005002                CLR     R2         ;MEMORY EXPECTED SHOULD BE 000000
870 027272 004737 016326          JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
871 027276 010465 000000          JSR    R4,TSDB(R5) ;SELECT LOCATION SPECIFIED
872 027302 004737 016326          JSR    PC,CHKTSSR ;WAIT FOR READY, NON-AMBIGUOUS
873 027306 116501 000000          MOVB  TSBA(R5),R1 ;READ LOC CONTENTS
874 027312 120102                CMPB  R1,R2        ;CHECK MEMORY FOR 000000
875 027314 001406                BEQ   40$         ;BRANCH IF DATA OKAY
876 027316                ERRDF  ERRNO,T5MEM,SFFMSG ;MEMORY NOT ZERO AFTER INIT.
      027316 104455                                TRAP   C$ERDF
      027320 000770                                .WORD  504
      027322 027432                                .WORD  T5MEM
      027324 012072                                .WORD  SFFMSG
877 027326 005237 002220          INC    FATFLG          ;SET THE FATAL ERROR FLAG
878 027332                104406          CKLOOP          ;SCOPE LOOP
879 027334                104410          ESCAPE  SUB      ;EXIT ON FATAL ERROR
      027336 000012                                .WORD  L10051-.
880 027340 005204                INC    R4         ;LOOK AT NEXT RAM LOC.
881 027342 020427 000400          CMP   R4,#400     ;AT TOP OF RAM ADDRESS SPACE
882 027346 001353                BNE   30$         ;BRANCH TILL ALL MEMORY TESTED
883
884 027350                ENDSUB          ;////////////////// END SUBTEST ////////////////////
      027350                L10051:
      027350 104403                                TRAP   C$ESUB
885
886 027352 005737 002220          TST   FATFLG       ;IS FATAL ERROR FLAG SET ?
887 027356 001402                BEQ   50$         ;BRANCH IF NOT
888 027360 004737 017172          JSR   PC,CKDROP    ;NO LOOP, TRY TO DROP DEVICE
889 027364 004737 016446          JSR   PC,TSTLOOP   ;SHOULD WE DO ITERATIONS ?
890 027370 103002                BCC   60$         ;BRANCH IF NOT
891 027372 000137 026446          JMP   T5LOOP       ;LOOP UNTIL COUNT EXPIRED
892 027376                EXIT   TST         ;ALL DONE THIS TEST
      027376 104432                                TRAP   C$EXIT
      027400 000132                                .WORD  L10047-.
893
894
895          ;*
896          ;LOCAL TEXT MESSAGES FOR TEST
897          ;-
898 027402                105      170      164  TST5ID: .ASCIIZ 'Extended Initialization'
899 027432                111      156      143  T5MEM:  .ASCII  'Incorrect RAM Data After Init'
900 027470                111      156      143  T5ADDR: .ASCIIZ 'Incorrect RAM Address After Init'
901
902 027532                ENDTST
      027532                L10047:
903 027532 104401                                TRAP   C$ETST

```

TEST 6: COMMAND REJECT

.SBITL TEST 6: COMMAND REJECT

905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961

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

TEST 6: COMMAND REJECT

```

962          ;      CAUSES THE IE BIT TO BE SET IN EACH COMMAND WORD AND THEN
963          ;      VERIFIES THAT AN INTERRUPT OCCURS.
964          ;
965          ;      BGNTST
966          ;
970          ;      T6.:
971          ;      ASCII MESSAGE TO IDENTIFY TEST
972          ;      DO INITIAL TEST SETUP
973          ;      PERFORM 20 ITERATIONS
974          ;      T6LOOP:
975          ;      BGNSUB
976          ;      T6.1:
977          ;      TRAP    C#BSUB
978          ;
979          ;      SETPRI  #PRIO0
980          ;      LOWER PRIORITY TO ALLOW INTERRUPTS
981          ;      MOV      #PRIO0,R0
982          ;      TRAP    C#SPRI
983          ;
984          ;      MOV      #T6PACKET,R4
985          ;      GET THE ADDRESS OF COMMAND PACKET
986          ;      MOV      #TSTBLK,R3
987          ;      BLOCK OF TEST DATA
988          ;      MOV      (R3)+,(R4)
989          ;      INSERT THE NEXT TEST DATA WORD
990          ;      BGNSEG
991          ;      BEGIN SEGMENT
992          ;      TRAP    C#BSEG
993          ;
994          ;      JSR      PC,SOFINIT
995          ;      DO SOFT INIT OF CONTROLLER
996          ;      BCS     10#
997          ;      BR IF SOFT INIT - OK
998          ;      MOV      R0,R1
999          ;      SAVE CONTENTS OF TSSR
1000          ;      ERDF   ERRNO,SFIERR,SFIMSG
1001          ;      DEVICE FATAL ERROR DURING INIT
1002          ;      TRAP    C#ERDF
1003          ;      .WORD   601
1004          ;      .WORD   SFIERR
1005          ;      .WORD   SFIMSG
1006          ;
1007          ;      10#:
1008          ;      CLR     FATFLG
1009          ;      CLEAR FATAL ERROR FLAG
1010          ;      CLR     INTRECV
1011          ;      CLEAR INTERRUPT RECEIVED FLAG
1012          ;      JSR     PC,CHKTSSR
1013          ;      WAIT FOR READY, NON-AMBIGUOUS
1014          ;      BIC     #BIT7,(R4)
1015          ;      DISABLE INTERRUPTS
1016          ;      MOV      R4,TSDB(R5)
1017          ;      SET THE PACKET ADDRESS
1018          ;      JSR     PC,WAITF
1019          ;      WAIT FOR SSR TO SET
1020          ;      BCS     15#
1021          ;      BR IF CARRY SET (GOOD RETURN)
1022          ;      MOV      R0,R1
1023          ;      SAVE CONTENTS OF TSSR
1024          ;      ERDF   ERRNO,T6SSR,PKTSSR
1025          ;      DEVICE FATAL SSR FAILED TO SET
1026          ;      TRAP    C#ERDF
1027          ;      .WORD   602
1028          ;      .WORD   T6SSR
1029          ;      .WORD   PKTSSR
1030          ;
1031          ;      15#:
1032          ;      INC     FATFLG
1033          ;      SET FATAL ERROR FLAG
1034          ;      CKLOOP
1035          ;      LOOP ON ERROR, IF FLAG SET
1036          ;      ESCAPE  SUB
1037          ;      BY-PASS SUBTEST IF FATAL ERROR
1038          ;      TRAP    C#CLP1
1039          ;      .WORD   C#ESCAPE
1040          ;      .WORD   L10053-
1041          ;
1042          ;      TST     INTRECV
1043          ;      DID AN INTERRUPT OCCUR ?
1044          ;      BEQ     22#
1045          ;      BRANCH IF NOT
1046          ;      ERHRD   ERRNO,T6INT,PKTSSR
1047          ;      TRAP    C#ERHRD
1048          ;      .WORD   603
1049          ;      .WORD   T6INT
1050          ;      .WORD   PKTSSR
1051          ;
1052          ;      22#:
1053          ;      MOV     #SCINBA:SSR:TSREJ,R2
1054          ;      EXPECTED CONTENTS OF TSSR
1055          ;      JSR     PC,CHKTSSR
1056          ;      WAIT FOR READY, NON-AMBIGUOUS

```


TEST 6: COMMAND REJECT

1152 030446 052525 .WORD 052525

1153

1154

1155

;*
;LOCAL TEXT MESSAGES FOR TEST

1156

1157

1158

1159	030450	103	157	155	T6NBA:	.ASCIZ	'Command Not Rejected'
1160	030475	103	157	156	T6SSR:	.ASCIZ	'Contents of TSSR Incorrect After Write Packet'
1161	030553	125	156	145	T6INT:	.ASCIZ	'Unexpected Interrupt Received On Write Packet'
1162	030631	105	170	160	T6NINT:	.ASCIZ	'Expected Interrupt Not Received On Write Packet'
1163	030711	111	156	143	T6TSBA:	.ASCIZ	'Incorrect TSBA Address After Packet Write'
1164	030763	103	157	155	TST6ID:	.ASCIZ	'Command Reject'

1165

1166 031002 .EVEN
031002 ENDTST

031002 104401

L10052: TRAP C\$ETST

TEST 6: COMMAND REJECT

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

.SBTTL TEST 7: WRITE CHARACTERISTICS

```

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

```

```

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

```

1196	031004					BGNTST		
	031004						T7::	
1201	031004	012700	034273			MOV	0TST7ID,RO	;ASCII MESSAGE TO IDENTIFY TEST
1202	031010	004737	016500			JSR	PC,TSTSETUP	;DO INITIAL TEST SETUP
1203	031014	012737	000024	002214		MOV	020.,LOOPCNT	;PERFORM 20 ITERATIONS
1204	031022				T7LOOP:			
1205	031022					BGNSSUB		;////////// BEGIN SUBTEST //////////
	031022						T7.1:	
	031022	104402					TRAP	C\$BSUB
1206	031024	004737	034322			JSR	PC,T7KEST	;SET PACKET TO START-UP VALUES
1207								
1208	031030					SETPRI	0PRI00	;LOWER PRIORITY TO ALLOW INTERRUPTS
	031030	012700	000000				MOV	0PRI00,RO
	031034	104441					TRAP	C\$SPRI
1209	031036	012703	002762			MOV	0TSTBLK*10.,R3	;START OF TEST DATA
1210	031042	012704	033140			MOV	0T7PACKET,R4	;GET THE ADDRESS OF COMMAND PACKET
1211	031046	012764	000010	000006		MOV	08.,PKBCNT(R4)	;START WITH MINIMUM ALLOWABLE VALUE
1212	031054				5\$:			
1213	031054					BGNSEEG		;>>>>>>>>> BEGIN SEGMENT >>>>>>>>>
	031054	104404					TRAP	C\$BSEEG
1214								
1215	031056	004737	015764			JSR	PC,SOFINIT	;DO SOFT INIT OF CONTROLLER
1216	031062	103405				BCS	10\$;BR IF SOFT INIT = OK
1220	031064	010001				MOV	RO,R1	;SAVE CONTENTS OF TSSR
1221	031066					ERRDF	ERRNO,SFIERR,SFIMSG	;DEVICE FATAL ERROR DURING INIT
	031066	104455					TRAP	C\$ERDF
	031070	001275					.WORD	701
	031072	003646					.WORD	SFIERR
	031074	012024					.WORD	SFIMSG

TEST 7: WRITE CHARACTERISTICS

```

1222 031076 005037 002220      10$: CLR      FATFLG      ;CLEAR FATAL ERROR FLAG
1223 031102 005037 002222      CLR      INTRECV     ;CLEAR INTERRUPT RECEIVED FLAG
1224 031106 010465 000000      MOV      R4,TSDB(R5) ;SET THE PACKET ADDRESS
1225 031112 004737 016326      JSR      PC,CHKTSSR  ;WAIT FOR SSR TO SET
1226 031116 103407      BCS      15$        ;BR IF CARRY SET (GOOD RETURN)
1227 031120 010001      MOV      R0,R1      ;SAVE CONTENTS OF TSSR
1231 031122      ERRDF  ERRNO,T7SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      031122 104455      TRAP     C$ERDF
      031124 001276      .WORD   702
      031126 033741      .WORD   T7SSR
      031130 012036      .WORD   PKTSSR
1232 031132 005237 002220      15$: INC      FATFLG      ;SET FATAL ERROR FLAG
1233 031136      CKLOOP  ;LOOP ON ERROR, IF FLAG SET
      031136 104406      TRAP     C$CLP1
1234 031140      ESCAPE  SEG      ;BY-PASS SUBTEST IF FATAL ERROR
      031140 104410      TRAP     C$ESCAPE
      031142 000152      .WORD   10000$-.
1235 031144 005737 002222      TST      INTRECV     ;DID AN INTERRUPT OCCUR ?
1236 031150 001404      BEQ      22$        ;BRANCH IF NOT
1240 031152      ERRHRD  ERRNO,T7INT,PKTSSR ;
      031152 104456      TRAP     C$ERHRD
      031154 001277      .WORD   703
      031156 034121      .WORD   T7INT
      031160 012036      .WORD   PKTSSR
1241 031162 016501 000002      22$: MOV      TSSR(R5),R1 ;GET THE CONTENTS OF TSSR
1242 031166 012702 000200      MOV      #SSR,R2    ;EXPECTED CONTENTS OF TSSR
1243 031172 032701 000100      BIT      #OFL,R1    ;IS OFF-LINE BIT SET ?
1244 031176 001402      BEQ      25$        ;BRANCH IF NOT OFF-LINE
1245 031200 052702 000100      BIS      #OFL,R2    ;SET OFF-LINE IN EXPECTED DATA
1246 031204 020201      25$: CMP      R2,R1    ;DOES EXPECTED MATCH RECEIVED ?
1247 031206 001404      BEQ      30$        ;OKAY IF MATCH
1251 031210      ERRHRD  ERRNO,T7NBA,PKTSSR ;NBA NOT ZERO
      031210 104456      TRAP     C$ERHRD
      031212 001300      .WORD   704
      031214 033300      .WORD   T7NBA
      031216 012036      .WORD   PKTSSR
1252 031220      30$: CKLOOP  ;LOOP ON ERROR ?
      031220 104406      TRAP     C$CLP1
1253 031222 004737 016326      JSR      PC,CHKTSSR  ;WAIT FOR READY, NON-AMBIGUOUS
1254 031226 016501 000000      MOV      TSBA(R5),R1 ;GET TSBA REGISTER CONTENTS
1255 031232 012702 033166      MOV      #T7BFR,R2  ;START OF THE DATA BUFFER
1256 031236 032762 000200 000012 BIT      #BIT7,XST2(R2) ;IS EXTENDED FEATURES BIT SET ?
1257 031244 001404      BEQ      32$        ;BRANCH IF EXTENDED FEATURES OFF
1258 031246 005237 002224      INC      EXTFEA     ;SET EXTENDED FEATURES FOR SUBTEST 6
1259 031252 062702 000002      ADD      #2,R2      ;EXTRA WORD IF SPECIAL FEATURES
1260 031256 062702 000016      32$: ADD      #14,,R2  ;EXPECTED CONTENTS OF TSDA
1261 031262 020102      CMP      R1,R2      ;COMPARE EXPECTED TO RECEIVED
1262 031264 001404      BEQ      35$        ;ERROR IF NOT EQUAL
1266 031266      ERRHRD  ERRNO,T7TSBA,EXPREC ;PRINT THE ERROR & EXPD/RCV
      031266 104456      TRAP     C$ERHRD
      031270 001301      .WORD   705
      031272 034210      .WORD   T7TSBA
      031274 015464      .WORD   EXPREC
1267
1268
1269 031276 004737 011104      35$: JSR      PC,CKRAM  ;SEE IF DATA IN RAM IS CORRECT
1270 031302 103404      BCS      40$        ;BRANCH IF PACKET IN RAM IS CORRECT

```


TEST 7: WRITE CHARACTERISTICS

```

1442      |
1443      |
1444      | TEST 7, SUBTEST 4
1445      |
1446      | SUBTEST TO VERIFY THAT A WRITE CHARACTERISTICS COMMAND IS
1447      | REJECTED IF AN ILLEGAL DATA BLOCK ADDRESS IS ISSUED.
1448      |
1449      |
1450      |
1451 032066      BGNSUB      |///////////////// BEGIN SUBTEST ///////////////////
      032066      |                                T7.4:
      032066 104402      |                                TRAP      C#BSUB
1452      |
1453 032070      SETPRI  #PRI00      |LOWER PRIORITY TO ALLOW INTERRUPTS
      032070 012700 000600      |                                MOV      #PRI00,R0
      032074 104441      |                                TRAP      C#SPRI
1454 032076 012703 033206      |START OF TEST DATA FOR SUBTEST
1455 032102 012704 033140      5#:  MOV      #T7PACKET,R4      |GET THE ADDRESS OF COMMAND PACKET
1456 032106 004737 034322      |JSR      PC,T7REST      |RESTORE PACKET TO STARTING VALUES
1457      |
1458      |
1459 032112 004737 015764      |JSR      PC,SOFINIT      |DO SOFT INIT OF CONTROLLER
1460 032116 103405      |BCS      10#      |BR IF SOFT INIT = OK
1464 032120 010001      |MOV      R0,R1      |SAVE CONTENTS OF TSSR
1465 032122      ERROF      ERRNO,SFIERR,SFIMSG      |DEVICE FATAL ERROR DURING INIT
      032122 104455      |                                TRAP      C#ERDF
      032124 001315      |                                .WORD      717
      032126 003646      |                                .WORD      SFIERR
      032130 012024      |                                .WORD      SFIMSG
1466 032132 005037 002222      10#: CLR      INTRECV      |CLEAR INTERRUPT RECEIVED FLAG
1467 032136 052737 000001 033150      |BIS      #1,T7DATA      |MAKE ADDRESS ODD
1468 032144 010465 000000      |MOV      R4,T5DB(R5)      |SET THE PACKET ADDRESS
1469 032150 004737 016240      |JSR      PC,WAITF      |WAIT FOR SSR TO SET
1470 032154 103405      |BCS      15#      |BR IF CARRY SET (GOOD RETURN)
1471 032156 010001      |MOV      R0,R1      |SAVE CONTENTS OF TSSR
1475 032160      ERROF      ERRNO,T7SSR,PKTSSR      |DEVICE FATAL SSR FAILED TO SET
      032160 104455      |                                TRAP      C#ERDF
      032162 001316      |                                .WORD      718
      032164 033741      |                                .WORD      T7SSR
      032166 012036      |                                .WORD      PKTSSR
1476 032170      15#:  CKLOOP      |LOOP ON ERROR, IF FLAG SET
      032170 104406      |                                TRAP      C#CLP1
1477 032172      ESCAPE  SUB      |BY-PASS SUBTEST IF FATAL ERROR
      032172 104410      |                                TRAP      C#ESCAPE
      032174 000116      |                                .WORD      L10061..
1478 032176 005737 002222      |TST      INTRECV      |DID AN INTERRUPT OCCUR ?
1479 032202 001404      |BEQ      22#      |BRANCH IF NOT
1483 032204      ERROF      ERRNO,T7INT,PKTSSR      |
      032204 104456      |                                TRAP      C#ERROF
      032206 001317      |                                .WORD      719
      032210 034121      |                                .WORD      T7INT
      032212 012036      |                                .WORD      PKTSSR
1484 032214 016501 000002      22#: MOV      TSSR(R5),R1      |GET THE CONTENTS OF TSSR
1485 032220 012702 102206      |MOV      #SC1SSR:T5REJINBA,R2      |EXPECTED CONTENTS OF TSSR
1486 032224 032701 000100      |BIT      #OFL,R1      |IS OFF LINE BIT SET ?
1487 032230 001402      |BEQ      25#      |BRANCH IF NOT OFF LINE
1488 032232 052702 000100      |BIS      #OFL,R2      |SET OFF-LINE IN EXPECTED DATA

```

TEST 7: WRITE CHARACTERISTICS

```

1489 032236 020201          254:  CMP      R2,R1          ;DOES EXPECTED MATCH RECEIVED ?
1490 032240 001414          BEQ      304          ;OKAY IF MATCH
1491 032242 010100          MOV      R1,R0          ;DATA FROM TSSR
1492 032244                XOR      R2,R0          ;FIND BITS IN ERROR
1493 032254 020027 002000      CMP      C0,#NBA       ;IS NBA ONLY BIT IN ERROR ?
1494 032260 001404          BEQ      304          ;DON'T PRINT ERROR IF NBA ONLY BAD BIT
1498 032262                ERRHRD  ERRNO,T74REJ,PKTSSR ;COMMAND NOT REJECTED
      032262 104456                TRAP    C#ERHRD
      032264 001320                .WORD  720
      032266 033545                .WORD  T74REJ
      032270 012036                .WORD  PKTSSR
1499 032272                304:  CKLOOP          ;LOOP ON ERROR ?
      032272 104406                TRAP    C#CLP1
1500 032274 032701 002000      BIT      #NBA,R1       ;IS NBA BIT SET ?
1501 032300 001004          BNE      354          ;OKAY IF NBA SET
1505 032302                ERRHRD  ERRNO,T72NBA,PKTSSR ;NBA NOT SET
      032302 104456                TRAP    C#ERHRD
      032304 001321                .WORD  721
      032306 033222                .WORD  T72NBA
      032310 012036                .WORD  PKTSSR
1506
1507 032312                354:  ENDSUB          ;////////////////// END SUBTEST ////////////////////
      032312                L10061:
1508 032312 104403                TRAP    C#ESUB

```


TEST 7: WRITE CHARACTERISTICS

```

1556 032456 001402          BEQ      25$
1557 032460 052702 000100   BIS      @OFL,R2
1558 032464 020201          CMP      R2,R1
1559 032466 001414          BEQ      30$
1560 032470 010100          MOV      R1,R0
1561 032472          XOR      R2,R0
1562 032502 020027 002000   CMP      R0,@NBA
1563 032506 001404          BEQ      30$
1567 032510          ERRHRD  ERRNO,T75REJ,PKTSSR
      032510 104456
      032512 001325
      032514 033643
      032516 012036
1568 032520          30$:  CKLOOP
      032520 104406
1569 032522 032701 002000   BIT      @NBA,R1
1570 032526 001004          BNE      35$
1574 032530          ERRHRD  ERRNO,T72NBA,PKTSSR
      032530 104456
      032532 001326
      032534 033222
      032536 012036
1575 032540          35$:
1576 032540          ENDSEG
      032540
      032540 104405
1577
1578 032542 005203          INC      R3
1579 032544 020327 000016   CMP      R3,@14.
1580 032550 002002          BGE      57$
1581 032552 000137 032330   JMP      5$
1582
1583 032556          57$:  ENDSUB
      032556
      032556 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$ERHRD
      .WORD    725
      .WORD    T75REJ
      .WORD    PKTSSR
;LOOP ON ERROR ?
      TRAP      C$CLP1
;IS NBA BIT SET ?
;OKAY IF NBA SET
;NBA NOT SET
      TRAP      C$ERHRD
      .WORD    726
      .WORD    T72NBA
      .WORD    PKTSSR
;<<<<<<<<<<<<<<<< END SEGMENT 10000$:
      TRAP      C$ESEG
;NEXT BUFFER LENGTH
;HAVE ALL BAD VALUES BEEN TESTED ?
;BRANCH IF ALL TESTED
;BRANCH TILL BACK TO ZERO
;////////////////// END SUBTEST L10062:
      TRAP      C$ESUB

```


TEST 7: WRITE CHARACTERISTICS

```

1680 033116          55$:  ENDSUB          ;////////////////// END SUBTEST ////////////////////
      033116          L10063:          TRAP      C$ESUB
      033116 104403
1681
1682 033120 005737 002220          TST      FATFLG          ;ANY FATAL ERRORS ?
1683 033124 001402          BEQ      60$          ;BRANCH IF NOT
1684 033126 004737 017172          JSR      PC,CKDROP      ;TRY TO DROP THE UNIT
1685 033132          60$:  EXIT      TST          ;ALL DONE THIS TEST
1686 033132 104432          TRAP      C$EXIT
      033132 001234          .WORD    L10055-.
1687
1688
1689          ;+
1690          ;LOCAL STORAGE FOR THIS TEST
1691          ;-
1693          .=<.+10>&177770
1695 033140          T7PACKET:          ;COMMAND PACKET FOR TEST
1696 033140 100004          .WORD    100004          ;WRITE CHARACTERISTICS COMMAND, WITH ACK
1697 033142 033150          .WORD    T7DATA          ;ADDRESS OF CHARACTERISTICS BLOCK
1698 033144 000000          .WORD    0
1699 033146 000010          .WORD    8.             ;STARTING VALUE OF BLOCK SIZE
1700
1701 033150          T7DATA:          ;CHARACTERISTICS DATA BLOCK
1702 033150 033166          .WORD    T7BFR          ;ADDRESS OF MESSAGE BUFFER
1703 033152 000000          .WORD    0
1704 033154 000016          .WORD    14.           ;LENGTH OF MESSAGE BUFFER
1705 033156 000000          .WORD    0
1706 033160 000000          T7SP:   .WORD    0          ;EXTFEA EXTRA WORD
1707
1708 033162 000000 000000          .WORD    0,0          ;SPACE
1709 033166          T7BFR:   .BLKW   8.             ;MESSAGE BUFFER
1710
1711          ;+
1712          ;
1713          ;TEST DATA FOR SUBTEST TWO
1714          ;
1715          ;DATA HAS FORMAT:
1716          ;
1717          ;          1ST WORD      OFFSET TO TEST WORD IN PACKET
1718          ;          2ND WORD      BITS TO SET FOR TEST
1719          ;
1720          ;-
1721
1722 033206          Y72DATA:          ;
1723 033206 000000 037140          .WORD    0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
1724 033212 000002 000001          .WORD    2,BIT0
1725 033216 000004 100100          .WORD    4,BIT6!BIT15
1726          T72DONE=.
1727
1728
1729          ;+
1730          ;LOCAL TEXT MESSAGES FOR TEST
1731          ;-
1733 033222          116      102      101  T72NBA: .ASCIZ 'NBA Not Set On Rejected WRI. CHARACTERISTICS'
1734 033300          127      122      111  T7NBA:  .ASCIZ 'WRITE CHARACTERISTICS Command Not Accepted'

```

TEST 7: WRITE CHARACTERISTICS

1735	033353	127	122	111	T72REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
1736	033452	127	122	111	T73REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
1737	033545	127	122	111	T74REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
1738	033643	127	122	111	T75REJ: .ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
1739	033741	103	157	156	T7SSR: .ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
1740	034030	105	170	160	T7NINT: .ASCIZ	'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
1741	034121	125	156	145	T7INT: .ASCIZ	'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
1742	034210	111	156	143	T7TSBA: .ASCIZ	'Incorrect TSBA Address After WRITE CHARACTERISTICS'
1743	034273	127	162	151	TST7ID: .ASCIZ	'Write Characteristics'
1744					.EVEN	
1745						

TEST 7: WRITE CHARACTERISTICS

1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768

034322
034322 012701 033140
034326 012721 100004
034332 012721 033150
034342 005021
034344 012721 000010
034350 012721 033166
034354 005021
034356 012721 000020
034362 005021
034364 005011
034366 000207
034370
034370 104401

```

;+
;
;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 C\$ETST

TEST 8: VOLUME CHECK

1770
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 034372
034372
1809 034372 012700 035257
1810 034376 004737 016500
1811 034402 012737 000024 002214
1812 034410
1813
1814 034410 012704 035000
1815 034414 004737 015764
1816 034420 103405
1820 034422 010001
1821 034424
034424 104455
034426 001441
034430 003646
034432 012024
1822 034434 042714 040000
1823 034440 010465 000000
1824 034444 004737 016326
1825 034450 103405
1826 034452 010001
1830 034454
034454 104455

.SBTTL TEST 8: VOLUME CHECK

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

THE TEST PROCEEDS AS FOLLOWS:

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

BGNTST

```

T8:
;ASCII MESSAGE TO IDENTIFY TEST
;DO INITIAL TEST SETUP
;PERFORM 20 ITERATIONS
MOV #T8ID,RO
JSR PC,TSTSETUP
MOV #20,LOOPCNT
T8LOOP:
MOV #T8PACKET,R4
;PACKET FOR WRITE CHARACTERISTICS
JSR PC,SOFINIT
;DO SOFT INIT OF CONTROLLER
BCS 10$
;BR IF SOFT INIT = OK
MOV R0,R1
;SAVE CONTENTS OF TSSR
ERRDF ERRNO,SFIERR,SFIMSG
;DEVICE FATAL ERROR DURING INIT
TRAP C$ERDF
;WORD 801
;WORD SFIERR
;WORD SFIMSG
10$:
BIC #BIT14,(R4)
;CLEAR THE CVC BIT
MOV R4,TSD8(R5)
;SET THE PACKET ADDRESS FOR WRITE CHAR
JSR PC,CHKTSSR
;WAIT FOR SSR TO SET
BCS 15$
;BR IF CARRY SET (GOOD RETURN)
MOV R0,R1
;SAVE CONTENTS OF TSSR
ERRDF ERRNO,T8SSR,PKTSSR
;DEVICE FATAL SSR FAILED TO SET
TRAP C$ERDF

```

TEST 8: VOLUME CHECK

Line	Address	Hex	Hex	Hex	Hex	Label	Code	Comment	Trap
	034456	001442						.WORD 802	
	034460	035170						.WORD T8SSR	
	034462	012036						.WORD PKTSSR	
1831	034464		15:			CKLOOP		;LOOP ON ERROR, IF FLAG SET	
	034464	104406						TRAP C\$CLP1	
1832	034466					ESCAPE TST		;EXIT IF FATAL ERROR	
	034466	104410						TRAP C\$ESCAPE	
	034470	000604						.WORD L10064-	
1833	034472	012702	035022			MOV	#T8BFR,R2	;ADDRESS OF THE MESSAGE BUFFER	
1834	034476	032762	000020	000006		BIT	#XSOVCK,XSTO(R2)	;IS VOLUME CHECK SET IN XSTO ?	
1835	034504	001406				BEQ	20:	;OKAY IF VOLUME CHECK IS CLEAR	
1839	034506	016501	000002			MOV	TSSR(R5),R1	;CONTENTS OF TSSR FOR ERROR REPORT	
1840	034512					ERRHRD	ERRNO,T8NVCK,PKTMES	;VOLUME CHECK NOT CLEAR	
	034512	104456						TRAP C\$ERHRD	
	034514	001443						.WORD 803	
	034516	035077						.WORD T8NVCK	
	034520	012100						.WORD PKTMES	
1841	034522		20:			CKLOOP		;LOOP ON ERROR ?	
	034522	104406						TRAP C\$CLP1	
1842	034524	010465	000000			MOV	R4,T8DB(R5)	;SET THE PACKET ADDRESS FOR WRITE CHAR	
1843	034530	004737	016326			JSR	PC,CHKTSSR	;WAIT FOR SSR TO SET	
1844	034534	103405				BCS	25:	;BR IF CARRY SET (GOOD RETURN)	
1845	034536	010001				MOV	R0,R1	;SAVE CONTENTS OF TSSR	
1849	034540					ERRDF	ERRNO,T8SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	034540	104455						TRAP C\$ERDF	
	034542	001444						.WORD 804	
	034544	035170						.WORD T8SSR	
	034546	012036						.WORD PKTSSR	
1850	034550		25:			CKLOOP		;LOOP ON ERROR, IF FLAG SET	
	034550	104406						TRAP C\$CLP1	
1851	034552					ESCAPE TST		;EXIT IF FATAL ERROR	
	034552	104410						TRAP C\$ESCAPE	
	034554	000520						.WORD L10064-	
1852	034556	032762	000020	000006		BIT	#XSOVCK,XSTO(R2)	;IS VOLUME CHECK SET IN XSTO ?	
1853	034564	001406				BEQ	30:	;OKAY IF VOLUME CHECK IS SET	
1857	034566	016501	000002			MOV	TSSR(R5),R1	;CONTENTS OF TSSR FOR ERROR REPORT	
1858	034572					ERRHRD	ERRNO,T8NVCK,PKTMES	;VOLUME CHECK NOT SET	
	034572	104456						TRAP C\$ERHRD	
	034574	001445						.WORD 805	
	034576	035077						.WORD T8NVCK	
	034600	012100						.WORD PKTMES	
1859	034602		30:			CKLOOP		;LOOP ON ERROR ?	
	034602	104406						TRAP C\$CLP1	
1860	034604	052714	040000			BIS	#BIT14,(R4)	;SET THE CVC BIT	
1861	034610	010465	000000			MOV	R4,T8DB(R5)	;SET THE PACKET ADDRESS FOR WRITE CHAR	
1862	034614	004737	016326			JSR	PC,CHKTSSR	;WAIT FOR SSR TO SET	
1863	034620	103405				BCS	35:	;BR IF CARRY SET (GOOD RETURN)	
1864	034622	010001				MOV	R0,R1	;SAVE CONTENTS OF TSSR	
1868	034624					ERRDF	ERRNO,T8SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET	
	034624	104455						TRAP C\$ERDF	
	034626	001446						.WORD 806	
	034630	035170						.WORD T8SSR	
	034632	012036						.WORD PKTSSR	
1869	034634		35:			CKLOOP		;LOOP ON ERROR, IF FLAG SET	
	034634	104406						TRAP C\$CLP1	
1870	034636					ESCAPE TST		;EXIT IF FATAL ERROR	
	034636	104410						TRAP C\$ESCAPE	

TEST 8: VOLUME CHECK

```

034640 000434
1871 034642 032762 000020 000006      BIT    #XSOVCK,XST0(R2)      ;IS VOLUME CHECK CLEAR IN XST0 ?
1872 034650 001406                      BEQ    40$                  ;OKAY IF VOLUME CHECK IS CLEARED
1876 034652 016501 000002              MOV    TSSR(R5),R1         ;CONTENTS OF TSSR FOR ERROR REPORT
1877 034656                      ERRHRD  ERRNO,T8VCK,PKTMES ;VOLUME CHECK NOT CLEARED
                                034656 104456                      TRAP   C$ERHRD
                                034660 001447                      .WORD  807
                                034662 035042                      .WORD  T8VCK
                                034664 012100                      .WORD  PKTMES
1878 034666                      40$:   CKLOOP             ;LOOP ON ERROR ?
                                034666 104406                      TRAP   C$CLP1
1879 034670 042714 040000              BIC    #BIT14,(R4)        ;CLEAR THE CVC BIT
1880 034674 010465 000000              MOV    R4,TSD8(R5)        ;SET THE PACKET ADDRESS FOR WRITE CHAR
1881 034700 004737 016326              JSR    PC,CHKTSSR         ;WAIT FOR SSR TO SET
1882 034704 103405                      BCS    45$                ;BR IF CARRY SET (GOOD RETURN)
1883 034706 010001                      MOV    R0,R1              ;SAVE CONTENTS OF TSSR
1887 034710                      ERRDF  ERRNO,T8SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                034710 104455                      TRAP   C$ERDF
                                034712 001450                      .WORD  808
                                034714 035170                      .WORD  T8SSR
                                034716 012036                      .WORD  PKTSSR
1888 034720                      45$:   CKLOOP             ;LOOP ON ERROR, IF FLAG SET
                                034720 104406                      TRAP   C$CLP1
1889 034722                      ESCAPE TST                ;EXIT IF FATAL ERROR
                                034722 104410                      TRAP   C$ESCAPE
                                034724 000350                      .WORD  L10064-.
1890 034726 032762 000020 000006      BIT    #XSOVCK,XST0(R2)      ;IS VOLUME CHECK CLEAR IN XST0 ?
1891 034734 001406                      BEQ    50$                  ;OKAY IF VOLUME CHECK IS CLEARED
1895 034736 016501 000002              MOV    TSSR(R5),R1         ;CONTENTS OF TSSR FOR ERROR REPORT
1896 034742                      ERRHRD  ERRNO,T8VCK,PKTMES ;VOLUME CHECK NOT CLEARED
                                034742 104456                      TRAP   C$ERHRD
                                034744 001451                      .WORD  809
                                034746 035042                      .WORD  T8VCK
                                034750 012100                      .WORD  PKTMES
1897 034752                      50$:   CKLOOP             ;LOOP ON ERROR ?
                                034752 104406                      TRAP   C$CLP1
1898 034754 004737 016446              JSR    PC,TSTLOOP         ;SHOULD WE DO ITERATIONS ?
1899 034760 103002                      BCC    62$                ;BRANCH IF NOT
1900 034762 000137 034410              JMP    T8LOOP             ;LOOP UNTIL COUNT EXPIRED
1901 034766                      62$:   EXIT    TST        ;ALL DONE THIS TEST
                                034766 104432                      TRAP   C$EXIT
                                034770 000304                      .WORD  L10064-.
1902
1903
1904      ;*
1905      ;LOCAL STORAGE FOR THIS TEST
1906      ;-
1908
1910 035000 035000      T8PACKET: .<<. *10>E177770
                                .WORD  100004
1911 035000 100004      .WORD  T8DATA
1912 035002 035010      .WORD  0
1913 035004 000000      .WORD  10
1914 035006 000010      .WORD  10
1915
1916 035010      T8DATA:
1917 035010 035022      .WORD  T8BFR
1918 035012 000000      .WORD  0
;COMMAND PACKET FOR TEST
;WRITE CHARACTERISTICS COMMAND
;ADDRESS OF CHARACTERISTICS BLOCK
;STARTING VALUE OF COUNTER
;CHARACTERISTICS DATA BLOCK
;ADDRESS OF MESSAGE BUFFER

```


TEST 8: VOLUME CHECK

```

1919 035014 000020          .WORD 16.          ;LENGTH OF MESSAGE BUFFER
1920 035016 000000 000000  .WORD 0,0
1921
1922 035022          T8BFR: .BLKW 8.          ;MESSAGE BUFFER
1923
1924
1925          ;*
1926          ;LOCAL TEXT MESSAGES FOR TEST
1927          ;*
1928
1929 035042          126      157      154  T8VCK: .ASCIZ 'Volume Check Bit Not Cleared'
1930 035077          126      157      154  T8NVCK: .ASCIZ 'Volume Check Bit (VCK) Not Clear After Initialize (XST0)'
1931 035170          103      157      156  T8SSR: .ASCIZ 'Contents of TSSR Incorrect After Write Characteristics'
1932 035257          126      157      154  T8TID: .ASCIZ 'Volume Check'
1933
1934 035274          .EVEN
          035274          .EVEN
          035274          .EVEN
          104401          .EVEN
          L10064:          TRAP          C$ETST

```


TEST 9: COMPLETION INTERRUPT

```

2030           |*
2031           |
2032           |TEST 9, SUBTEST 2
2033           |
2034           |CHECK THAT UNUSED BITS BEING SET CAUSES
2035           |WRITE CHARACTERISTICS COMMAND TO BE REJECTED
2036           |
2037           |-
2038
2039 035566             BGNSUB             ;////////// BEGIN SUBTEST ///////////
           035566             T9.2: TRAP C#BSUB
           035566 104402
2040
2041 035570             SETPRI  #PRI00             ;LOWER PRIORITY TO ALLOW INTERRUPTS
           035570 012700 000000             MOV #PRI00,R0
           035574 104441             TRAP C#SPRI
2042 035576 012703 037302             ;START OF TEST DATA FOR SUBTEST
2043 035602 012704 037240             ;GET THE ADDRESS OF COMMAND PACKET
2044 035606 004737 040336             ;RESTORE PACKET TO STARTING VALUES
2045
2046 035612             BGNSEG             ;>>>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>
           035612 104404             TRAP C#BSEG
2047
2048 035614 004737 015764             JSR PC,SOFINIT             ;DO SOFT INIT OF CONTROLLER
2049 035620 103405             BCS 10#             ;BR IF SOFT INIT = OK
2053 035622 010001             MOV R0,R1             ;SAVE CONTENTS OF TSSR
2054 035624             ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
           035624 104455             TRAP C#ERDF
           035626 001611             .WORD 905
           035630 003646             .WORD SFIERR
           035632 012024             .WORD SFIMSG
2055 035634 005037 002222             10#: CLR INTRECV             ;CLEAR INTERRUPT RECEIVED FLAG
2056 035640 010400             MOV R4,R0             ;START OF THE COMMAND PACKET
2057 035642 061300             ADD (R3),R0             ;OFFSET TO THE DATA WORD TO TEST
2058 035644 056310 000002             BIS 2(R3),(R0)             ;SET THE DATA BITS TO BE TESTED
2059 035650 010465 000000             MOV R4,TSD8(R5)             ;SET THE PACKET ADDRESS
2060 035654 004737 016240             JSR PC,WAITF             ;WAIT FOR SSR TO SET
2061 035660 103405             RCS 15#             ;BR IF CARRY SET (GOOD RETURN)
2062 035662 010001             MOV R0,R1             ;SAVE CONTENTS OF TSSR
2066 035664             ERRDF  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
           035664 104455             TRAP C#ERDF
           035666 001612             .WORD 906
           035670 037757             .WORD T9SSR
           035672 012036             .WORD PKTSSR
2067 035674             15#: CKLOOP             ;LOOP ON ERROR, IF FLAG SET
           035674 104406             TRAP C#CLP1
2068 035676             ESCAPE SEG             ;BYPASS CHECKS IF FATAL ERROR
           035676 104410             TRAP C#ESCAPE
           035700 000056             .WORD 10000#..
2069 035702 005737 002222             ;DID AN INTERRUPT OCCUR ?
2070 035706 001004             BNE 22#             ;BRANCH IF YES
2074 035710             ERRHRD  ERRNO,T9NINT,PKTSSR
           035710 104456             TRAP C#ERHRD
           035712 001613             .WORD 907
           035714 040046             .WORD T9NINT
           035716 012036             .WORD PKTSSR
2075 035720 016501 000002             22#: MOV TSSR(R5),R1             ;GET THE CONTENTS OF TSSR

```


TEST 9: COMPLETION INTERRUPT

```

2096 ;*
2097 ;
2098 ;TEST 9, SUBTEST 3
2099 ;
2100 ;CHECK THE WRITE CHARACTERISTICS COMMAND IS REJECTED
2101 ;IF ISSUED WITH AN INVALID DATA BLOCK BYTE COUNT
2102 ;
2103 ;
2104 ;-
2105 036000          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
        036000                    ;T9.3:
        036000 104402                TRAP      C#BSUB
2106
2107 036002          SETPRI  #PRI00      ;LOWER PRIORITY TO ALLOW INTERRUPTS
        036002 012700 000000          ;
        036006 104441                TRAP      #PRI00,R0
        036006 104441                TRAP      C#SPRI
2108 036010 012703 000001              ;STARTING BYTE COUNT
2109 036014 012704 037240              ;GET THE ADDRESS OF COMMAND PACKET
2110 036020 004737 040336              ;RESTORE PACKET TO STARTING VALUES
2111
2112 036024          BGNSEG          ;>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>
        036024 104404                TRAP      C#BSEG
2113
2114 036026 004737 015764              ;DO SOFT INIT OF CONTROLLER
2115 036032 103405              ;BR IF SOFT INIT = OK
2119 036034 010001              ;SAVE CONTENTS OF TSSR
2120 036036          ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
        036036 104455                TRAP      C#ERRDF
        036040 001615                .WORD   909
        036042 003646                .WORD   SFIERR
        036044 012024                .WORD   SFIMSG
2121 036046 005037 002222              ;CLEAR INTERRUPT RECEIVED FLAG
2122 036052 010364 000006              ;INSERT THE BYTE COUNT FOR TEST
2123 036056 010465 000000              ;SET THE PACKET ADDRESS
2124 036062 004737 016240              ;WAIT FOR SSR TO SET
2125 036066 103405              ;BR IF CARRY SET (GOOD RETURN)
2126 036070 010001              ;SAVE CONTENTS OF TSSR
2130 036072          ERRDF  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
        036072 104455                TRAP      C#ERRDF
        036074 001616                .WORD   910
        036076 037757                .WORD   T9SSR
        036100 012036                .WORD   PKTSSR
2131 036102          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
        036102 104406                TRAP      C#CLP1
2132 036104          ESCAPE  SEG       ;BY-PASS SUBTEST IF FATAL ERROR
        036104 104410                TRAP      C#ESCAPE
        036106 000056                .WORD   100000-
2133 036110 005737 002222              ;DID AN INTERRUPT OCCUR ?
2134 036114 001004              ;BRANCH IF YES
2138 036116          ERRHRD ERRNO,T9NINT,PKTSSR ;
        036116 104456                TRAP      C#ERRHRD
        036120 001617                .WORD   911
        036122 040046                .WORD   T9NINT
        036124 012036                .WORD   PKTSSR
2139 036126 016501 000002              ;GET THE CONTENTS OF TSSR
2140 036132 012702 102206              ;EXPECTED CONTENTS OF TSSR
2141 036136 032701 000100              ;IS OFF-LINE BIT SET ?

```


TEST 9: COMPLETION INTERRUPT

```

2160          ;*
2161          ;
2162          ;TEST 9, SUBTEST 4
2163          ;
2164          ;SUBTEST TO VERIFY THAT A WRITE CHARACTERISTICS COMMAND IS
2165          ;REJECTED IF AN ILLEGAL DATA BLOCK ADDRESS IS ISSUED.
2166          ;
2167          ;-
2168
2169 036204          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
          036204          T9.4:          TRAP          C#RSUB
          036204 104402
2170
2171 036206          SETPRI  #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
          036206 012700 000000          MOV          #PRI00,R0
          036212 1044*1          TRAP          C#SPRI
2172 036214 012703 037302          MOV          #T92DATA,R3          ;START OF TEST DATA FOR SUBTEST
2173 036220 012704 037240          MOV          #T9PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
2174 036224 004737 040336          JSR          PC,T9REST          ;RESTORE PACKET TO STARTING VALUES
2175
2176
2177 036230 004737 015764          JSR          PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
2178 036234 103405          BCS          10#          ;BR IF SOFT INIT = OK
2182 036236 010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
2183 036240          ERRDF          ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL ERROR DURING INIT
          036240 104455          TRAP          C#ERDF
          036242 001621          .WORD          913
          036244 003646          .WORD          SFIERR
          036246 012024          .WORD          SFIMSG
2184 036250 005037 002222          CLR          INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
2185 036254 052737 000001 037250          BIS          #1,T9DATA          ;MAKE ADDRESS ODD
2186 036262 010463 000000          MOV          R4,TSD8(R5)          ;SET THE PACKET ADDRESS
2187 036266 004737 016240          JSR          PC,WAITF          ;WAIT FOR SSR TO SET
2188 036272 103405          BCS          15#          ;BR IF CARRY SET (GOOD RETURN)
2189 036274 010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
2193 036276          ERRDF          ERRNO,T9SSR,PKTSSR          ;DEVICE FATAL SSR FAILED TO SET
          036276 104455          TRAP          C#ERDF
          036300 001622          .WORD          914
          036302 037757          .WORD          T9SSR
          036304 012036          .WORD          PKTSSR
2194 036306          15#:          CKLOOP          ;LOOP ON ERROR, IF FLAG SET
          036306 104406          TRAP          C#CLP1
2195 036310          ESCAPE          SUB          ;BY-PASS SUBTEST IF FATAL ERROR
          036310 104410          TRAP          C#ESCAPE
          036312 000056          .WORD          L10071
2196 036314 005737 002222          TST          INTRECV          ;DID AN INTERRUPT OCCUR ?
2197 036320 001004          BNE          22#          ;BRANCH IF YES
2201 036322          ERRHRD          ERRNO,T9NINT,PKTSSR
          036322 104456          TRAP          C#ERHRD
          036324 001623          .WORD          915
          036326 040046          .WORD          T9NINT
          036330 012036          .WORD          PKTSSR
2202 036332 016501 000002          22#:          MOV          TSSR(R5),R1          ;GET THE CONTENTS OF TSSR
2203 036336 012702 102206          MOV          #SC!SSR!TSREJ!NBA,R2          ;EXPECTED CONTENTS OF TSSR
2204 036342 012701 000100          BIT          #OFL,R1          ;IS OFF-LINE BIT SET ?
2205 036346 001402          BEQ          25#          ;BRANCH IF NOT OFF-LINE
2206 036350 052702 000100          BIS          #OFL,R2          ;SET OFF-LINE IN EXPECTED DATA

```


TEST 9: COMPLETION INTERRUPT

```

2207 036354 020201          25$:  CMP      R2,R1          ;DOES EXPECTED MATCH RECEIVED ?
2208 036356 001404          BEQ      30$          ;OKAY IF MATCH
2212 036360          ERRHRD  ERRNO,T94REJ,PKTSSR ;COMMAND NOT REJECTED
      036360 104456          TRAP     C$ERHRD
      036362 001624          .WORD   916
      036364 037563          .WORD   T94REJ
      036366 012036          .WORD   PKTSSR
2213 036370          30$:
2214
2215 036370          ENDSUB          ;////////////////// END SUBTEST ////////////////////
      036370          L10071:
2216 036370 104403          TRAP     C$ESUB

```


TEST 9: COMPLETION INTERRUPT

```

2354
2355
2356
2357
2358
2359
2360
2361
2362
2363 037030          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      037030          ;              T9.7:
      037030 104402          TRAP      C$BSUB
2364
2365 037032          SETPRI  #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
      037032 012700 000000          MOV      #PRI00,RC
      037036 104441          TRAP      C$SPRI
2366 037040          MOV      #T9PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
2367 037044          JSR      PC,T9REST          ;SET UP A VALID PACKET
2368 037050          JSR      PC,$OFINIT          ;DO SOFT INIT OF CONTROLLER
2369 037054          BCS      10$          ;BR IF SOFT INIT = OK
2373 037056          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
2374 037060          ERRDF   ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      037060 104455          TRAP      C$ERDF
      037062 001635          .WORD    925
      037064 003646          .WORD    SFIERR
      037066 012024          .WORD    SFIMSG
2375 037070          10$: CLR      INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
2376 037074          BIS      #BIT7,(R4)          ;ENABLE INTERRUPTS
2377 037100          MOV      R4,TSDB(R5)          ;SET THE PACKET ADDRESS
2378 037104          JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
2379 037110          BCS      15$          ;BR IF CARRY SET (GOOD RETURN)
2380 037112          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
2384 037114          ERRDF   ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      037114 104455          TRAP      C$ERDF
      037116 001636          .WORD    926
      037120 037757          .WORD    T9SSR
      037122 012036          .WORD    PKTSSR
2385 037124          15$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      037124 104406          TRAP      C$CLP1
2386 037126          ESCAPE SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      037126 104410          TRAP      C$ESCAPE
      037130 000102          .WORD    L10074-
2387 037132          TST      INTRECV          ;DID AN INTERRUPT OCCUR ?
2388 037136          BNE      22$          ;BRANCH IF YES
2392 037140          ERRHRD  ERRNO,T9NINT,PKTSSR
      037140 104456          TRAP      C$ERHRD
      037142 001637          .WORD    927
      037144 040046          .WORD    T9NINT
      037146 012036          .WORD    PKTSSR
2393 037150          22$: CKLOOP          ;LOOP ON ERROR ?
      037150 104406          TRAP      C$CLP1
2394
2395 037152          CLR      INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
2396 037156          BIC      #BIT7,(R4)          ;DISABLE INTERRUPTS
2397 037162          MOV      R4,TSDB(R5)          ;SET THE PACKET ADDRESS
2398 037166          JSR      PC,CHKTSSR          ;WAIT FOR SSR TO SET
2399 037172          BCS      25$          ;BR IF CARRY SET (GOOD RETURN)

```

B15

TEST 9: COMPLETION INTERRUPT

```

2400 037174 010001      MOV      R0,R1          ;SAVE CONTENTS OF TSSR
2404 037176          ERRDF  ERRNO,T9SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP  C1FRDF
                                .WORD 928
                                .WORD T9SSR
                                .WORD PKTSSR
                                TRAP  C1CLP1
2405 037206          251:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP  C1ESCAPE
2406 037210          ESCAPE  SUB          ;BY-PASS SUBTEST IF FATAL ERROR
                                TRAP  C1ESCAPE
                                .WORD  L10074
2407 037214          00222  TST      INTRECV        ;DID AN INTERRUPT OCCUR ?
2408 037220          BEQ      301          ;BRANCH IF NOT
2412 037222          ERRHRD  ERRNO,T9INT,PKTSSR
                                TRAP  C1ERHRD
                                .WORD  929
                                .WORD  T9INT
                                .WORD  PKTSSR
2414 037232          301:  ENDSUB          ;////////////////////// END SUBTEST ////////////////////////
2415 037232          037232  104403          ;L10074:
                                TRAP  C1ESUB
2416 037234          EXIT      TST          ;ALL DONE THIS TEST
                                TRAP  C1EXIT
                                .WORD  L10065
2417
2418
2419
2420
2421
2425 037240          T9PACKET:
2426 037240          .WORD  100204          ;COMMAND PACKET FOR TEST
2427 037242          .WORD  T9DATA          ;WRITE CHAR COMMAND, WITH IE, ACK
2428 037244          .WORD  0              ;ADDRESS OF CHARACTERISTICS BLOCK
2429 037246          .WORD  8              ;STARTING VALUE OF BLOCK SIZE
2430
2431 037250          T9DATA:
2432 037250          .WORD  T9BFR          ;CHARACTERISTICS DATA BLOCK
2433 037252          .WORD  0              ;ADDRESS OF MESSAGE BUFFER
2434 037254          .WORD  14             ;LENGTH OF MESSAGE BUFFER
2435 037256          .WORD  0,0          ;MESSAGE BUFFER
2436
2437 037262          T9BFR:  .BLKW  8
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450 037302          T92DATA:

```

TEST 9: COMPLETION INTERRUPT

2451	037302	000000	037140	.WORD	0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
2452	037306	000002	000001	.WORD	2,BIT0
2453	037312	000004	100100	.WORD	4,BIT6!BIT15
2454		037316			T92DONE=.

2455
2456
2457
2458
2459
2460

```

|
|LOCAL TEXT MESSAGES FOR TEST
|

```

2461	037316	127	122	111	T9NBA:	.ASCIZ	'WRITE CHARACTERISTICS Command Not Accepted'
2462	037371	127	122	111	T92REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
2463	037470	127	122	111	T93REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
2464	037563	127	122	111	T94REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
2465	037661	127	122	111	T95REJ:	.ASCIZ	'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
2466	037757	103	157	156	T9SSR:	.ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
2467	040046	105	170	160	T9NINT:	.ASCIZ	'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
2468	040137	125	156	145	T9INT:	.ASCIZ	'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
2469	040226	111	156	143	T9TSBA:	.ASCIZ	'Incorrect TSBA Address After WRITE CHARACTERISTICS'
2470	040311	103	157	155	T9ID:	.ASCIZ	'Completion Interrupt'
2471						.EVEN	
2472							

TEST 9: COMPLETION INTERRUPT

2474
 2475
 2476
 2477
 2478
 2479
 2480
 2481 040336
 2482 040336
 2483 040342 012701 037240
 2484 040346 012721 100204
 2485 040352 012721 037250
 2486 040356 005021
 2487 040360 012721 000010
 2488 040364 012721 037262
 2489 040370 005021
 2490 040372 012721 000016
 2491 040376 005021
 2492 040400 005011
 2493 040402 005037 037262
 2494 040406 000207
 2495 040410
 040410
 040410 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 C\$ETST

TEST 10: BASIC PACKET PROTOCOL

	041752	002000					.WORD	1024
	041754	042634					.WORD	T10MBF
	041756	015464					.WORD	EXPREC
2825								
2826	041760		70\$:	CKLOOP		;LOOP ON ERROR IF FLAG SET		
	041760	104406					TRAP	C\$CLP1
2827								
2828	041762	005037		CLR	INTRECV	;CLEAR INTERRUPT RECEIVED FLAG		
2829	041766	004737		JSR	PC,T10RST	;RESET THE PACKETS AND COMMANDS		
2830	041772	042714		BIC	*100000,(R4)	;CLEAR THE ACK BIT		
2831	041776	010465		MOV	R4,TSD8(R5)	;SET THE PACKET ADDRESS		
2832	042002	004737		JSR	PC,CHKTSSR	;WAIT FOR SSR TO SET		
2833	042006	103407		BCS	75\$;BR IF CARRY SET (GOOD RETURN)		
2834	042010	010001		MOV	R0,R1	;SAVE CONTENTS OF TSSR		
2838	042012			ERRDF	ERRNO,T10SSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET		
	042012	104455					TRAP	C\$ERDF
	042014	002001					.WORD	1025
	042016	043070					.WORD	T10SSR
	042020	012036					.WORD	PKTSSR
2839	042022	005237		INC	FATFLG	;SET FATAL ERROR FLAG		
2840	042026		75\$:	CKLOOP		;LOOP ON ERROR, IF FLAG SET		
	042026	104406					TRAP	C\$CLP1
2841	042030			ESCAPE	SEG	;BY-PASS SUBTEST IF FATAL ERROR		
	042030	104410					TRAP	C\$ESCAPE
	042032	000062					.WORD	10001\$-.
2842	042034	005737		TST	INTRECV	;DID AN INTERRUPT OCCUR ?		
2843	042040	001006		BNE	82\$;BRANCH IF YES		
2847	042042	016500		MOV	TSSR(R5),R0	;GET TSSR FOR ERROR REPORT		
2848	042046			ERRHRD	ERRNO,T10NINT,PKTSSR			
	042046	104456					TRAP	C\$ERHRD
	042050	002002					.WORD	1026
	042052	043157					.WORD	T10NINT
	042054	012036					.WORD	PKTSSR
2849	042056	016501		MOV	TSSR(R5),R1	;GET THE CONTENTS OF TSSR		
2850	042062	012702		MOV	*SSR,R2	;EXPECTED CONTENTS OF TSSR		
2851	042066	032701		BIT	*OFL,R1	;IS OFF-LINE BIT SET ?		
2852	042072	001402		BEQ	85\$;BRANCH IF NOT OFF-LINE		
2853	042074	052702		BIS	*OFL,R2	;SET OFF-LINE IN EXPECTED DATA		
2854	042100	020201		CMP	R2,R1	;DOES EXPECTED MATCH RECEIVED ?		
2855	042102	001404		BEQ	90\$;OKAY IF MATCH		
2859	042104			ERRHRD	ERRNO,T10SSR,PKTSSR	;NBA NOT ZERO		
	042104	104456					TRAP	C\$ERHRD
	042106	002003					.WORD	1027
	042110	043070					.WORD	T10SSR
	042112	012036					.WORD	PKTSSR
2860	042114							
2861	042114		90\$:	ENDSEG		;***** END SEGMENT *****		
	042114							
	042114	104405					TRAP	C\$ESEG
2862	042116	005737		TST	FATFLG	;ANY FATAL ERRORS		
2863	042122	001403		BEQ	95\$;BR, IF NO FATAL ERRORS		
2864	042124	004737		JSR	PC,CKDROP	;TRY TO DROP THE UNIT		
2865								
2866	042130			BGNSEG		;***** BGN SEGMENT *****		
	042130	104404					TRAP	C\$BSEG
2867	042132	005037		CLR	INTRECV	;CLEAR INTERRUPT RECEIVED FLAG		
2868	042136	004737		JSR	PC,T10RST	;RESET THE PACKETS AND COMMANDS		

TEST 10: BASIC PACKET PROTOCOL

042452	104456						TRAP	C#ERHRD
012454	002011						.WORD	1033
042456	043157						.WORD	T10NINT
042460	012036						.WORD	PKTSSR
2956	042462	016501	000002	22:	MOV	TSSR(R5),R1		
2957	042466	012702	110200		MOV	#SSR!RMR!SC,R2		
2958	042472	032701	000100		BIT	#OFL,R1		
2959	042476	001402			BEQ	25:		
2960	042500	052702	000100		BIS	#OFL,R2		
2961	042504	020201		25:	CMP	R2,R1		
2962	042506	001404			BEQ	30:		
2966	042510				ERRHRD	ERRNO,T10SSR,PKTSSR		
	042510	104456					TRAP	C#ERHRD
	042512	002012					.WORD	1034
	042514	043070					.WORD	T10SSR
	042516	012036					.WORD	PKTSSR
2967	042520			30:				
2968	042520				ENDSEG			
	042520							
	042520	104405						
2969	042522				ENDSUB			
	042522							
	042522	104403						
2970	042524				EXIT	TST		
	042524	104432						
	042526	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 ?
;OKAY IF MATCH
;NBA NOT ZERO

;***** END SEGMENT *****
10000:

;//////////////// END SUBTEST //////////////////
L10101:

;ALL DONE WITH THIS TEST

TRAP C#ESEG
TRAP C#ESUB
TRAP C#EXIT
.WORD L10075-.

TEST 10: BASIC PACKET PROTOCOL

```

2972
2973
2974
2975
2979 042530
2980 042530 100204
2981 042532 042540
2982 042534 000000
2983 042536 000010
2984
2985 042540
2986 042540 042552
2987 042542 000000
2988 042544 000016
2989 042546 000000 000000
2990
2991 042552
2992
2993
2994
2995
2996
2997 042572
2998 042572 100204
2999 042574 042602
3000 042576 000000
3001 042600 000010
3002
3003 042602
3004 042602 042614
3005 042604 000000
3006 042606 000016
3007 042610 000000 000000
3008
3009 042614
3010
3011
3012
3013
3014
3015
3016 042634 115 145 163
3017 042731 116 102 101
3018 043013 116 102 101
3019 043070 103 157 156
3020 043157 105 170 160
3021 043250 125 156 145
3022 043337 102 141 163
3023
3024

```

```

; LOCAL STORAGE FOR THIS TEST
;
; T1OPACKET:
; .WORD 100204 ; COMMAND PACKET FOR TEST
; .WORD T10DATA ; WRITE CHAR COMMAND, WITH IE, ACK
; .WORD 0 ; ADDRESS OF CHARACTERISTICS BLOCK
; .WORD 8. ; STARTING VALUE OF BLOCK SIZE
;
; T10DATA:
; .WORD T10BFR ; CHARACTERISTICS DATA BLOCK
; .WORD 0 ; ADDRESS OF MESSAGE BUFFER
; .WORD 14. ; LENGTH OF MESSAGE BUFFER
; .WORD 0,0
;
; T10BFR: .BLKW 8. ; MESSAGE BUFFER
;
; TEST DATA FOR SUBTEST FOUR
;
; T10PKT:
; .WORD 100204 ; COMMAND PACKET FOR TEST
; .WORD T10DATA ; WRITE CHAR COMMAND, WITH IE, ACK
; .WORD 0 ; ADDRESS OF CHARACTERISTICS BLOCK
; .WORD 8. ; STARTING VALUE OF BLOCK SIZE
;
; T10DATA:
; .WORD T10BFR ; CHARACTERISTICS DATA BLOCK
; .WORD 0 ; ADDRESS OF MESSAGE BUFFER
; .WORD 14. ; LENGTH OF MESSAGE BUFFER
; .WORD 0,0
;
; T10BFR: .BLKW 8. ; MESSAGE BUFFER
;
; LOCAL TEXT MESSAGES FOR TEST
;
; T10MBF: .ASCIIZ 'Message Buffer Modified after MESSAGE BUFFER RELEASE Command'
; T10NBA: .ASCIIZ 'NBA Not Clear After WRITE CHARACTERISTICS Command'
; T10NBA: .ASCIIZ 'NBA Set After MESSAGE BUFFER RELEASE Command'
; T10SSR: .ASCIIZ 'Contents of ISSR Incorrect After WRITE CHARACTERISTICS'
; T10INT: .ASCIIZ 'Expected Interrupt Not Received On WRITE CHARACTERIS. CS'
; T10INT: .ASCIIZ 'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
; T10ID: .ASCIIZ 'Basic Packet Protocol'
; .EVEN

```

TEST 10: BASIC PACKET PROTOCOL

3026
 3027
 3028
 3029
 3030
 3031
 3032
 3033 043366
 3034 043366
 3035 043372 012701 042530
 3036 043376 012721 100204
 3037 043402 012721 042540
 3038 043406 005021
 3039 043410 012721 000010
 3040 043414 012721 042552
 3041 043420 005021
 3042 043422 012721 000016
 3043 043426 005021
 3044 043430 005011
 3045 043432 005037 042552
 3046 043436 000207
 3047
 3048
 3049
 3050
 3051
 3052
 3053 043440
 3054 043440
 3055 043444 012701 042572
 3056 043450 012721 100204
 3057 043454 012721 042602
 3058 043460 005021
 3059 043462 012721 000010
 3060 043466 012721 042614
 3061 043472 005021
 3062 043474 012721 000016
 3063 043500 005021
 3064 043502 005011
 3065 043504 005037 042614
 3066 043510 000207
 3067 043512
 043512
 043512 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 DATA 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    #T10DATA,(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 C\$ETST

TEST 11: NON-TAPE MOTION COMMANDS

```

3070                                .SBTTL TEST 11: NON-TAPE MOTION COMMANDS
3071
3072                                ;*
3073                                ;
3074                                ; THIS TEST VERIFIES PROPER OPERATION OF THE INITIALIZE
3075                                ; COMMAND. TWO SUBTESTS ARE USED. THE FIRST VERIFIES THAT
3076                                ; THE COMMAND RUNS TO COMPLETION AND STORES A VALID
3077                                ; MESSAGE PACKET. THE SECOND VERIFIES THAT NON-ZERO
3078                                ; VALUES IN THE COMMAND MCDE FIELD CAUSES COMMAND REJECT.
3079                                ;
3080                                ;*
3081
3082 043514                                BGNTEST
3083 043514
3087 043514 012700 045512                                MOV     #TST1:ID,RO      ;ASCII MESSAGE TO IDENTIFY TEST
3088 043520 004737 016500                                JSR     PC,TSTSETUP    ;DO INITIAL TEST SETUP
3089 043524 012737 000024 002214                                MOV     #20.,LOOPCNT  ;PERFORM 20 ITERATIONS
3090 043532                                T11LOOP:
3091 043532                                BGNSUB                 ;//////////////////// BEGIN SUBTEST //////////////////////
3092 043532 104402                                T11.1:
3093                                TRAP     C$BSUB
3094 043534                                SETPRI  #PRI00        ;LOWER PRIORITY TO ALLOW INTERRUPTS
3095 043534 012700 000000                                MOV     #PRI00,RO
3096 043540 104441                                TRAP     C$SPRI
3097 043542 004737 015764                                JSR     PC,SOFINIT    ;DO SOFT INIT OF CONTROLLER
3098 043546 103405                                BCS     3#            ;BR IF SOFT INIT = OK
3099 043550 010001                                MOV     RO,R1         ;SAVE CONTENTS OF TSSR
3100 043552                                ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
3101 043552 104455                                TRAP     C$ERDF
3102 043554 002115                                .WORD   1101
3103 043556 003646                                .WORD   SFIERR
3104 043560 012024                                .WORD   SFIMSG
3105 043562                                3#:
3106 043562 012704 044740                                MOV     #T11PK2,R4   ;WRITE CHARACTERISTICS PACKET
3107 043566 004737 010652                                JSR     PC,WRTCHR    ;ISSUE WRITE CHARACTERISTICS
3108 043572 103404                                BCS     4#            ;BR, IF COMMAND ISSUED OK
3109 043574                                ERRHRD  ERRNO,WRTMSG,SFIMSG ;WRITE CHARACTERISTIC FAILED
3110 043574 104456                                TRAP     C$ERHRD
3111 043576 002116                                .WORD   1102
3112 043600 005052                                .WORD   WRTMSG
3113 043602 012024                                .WORD   SFIMSG
3114 043604                                4#:
3115 043604 004737 045544                                JSR     PC,T11REST   ;SET UP PACKET FOR COMMAND
3116 043610 012704 044670                                MOV     #T11PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
3117 043614                                5#:
3118 043614                                BGNSEG                ;>>>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>>>>
3119 043614 104404                                TRAP     C$BSEG
3120 043616 005037 002220                                10#:
3121 043622 005037 002222                                CLR     FATALG       ;CLEAR FATAL ERROR FLAG
3122 043626 010465 000000                                CLR     INTRECV      ;CLEAR INTERRUPT RECEIVED FLAG
3123 043632 004737 016326                                MOV     R4,TSDB(R5)  ;SET THE PACKET ADDRESS
3124 043636 103407                                JSR     PC,CHKTSSR   ;WAIT FOR SSR TO SET
3125 043640 010001                                BCS     15#          ;BR IF CARRY SET (GOOD RETURN)
3126 043642                                MOV     RO,R1         ;SAVE CONTENTS OF TSSR
3127 043642 104455                                ERRDF  ERRNO,T11SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
3128                                TRAP     C$ERDF

```


TEST 11: NON-TAPE MOTION COMMANDS

	043644	002117					.WORD	1103
	043646	045234					.WORD	T11SSR
	043650	012036					.WORD	PKTSSR
3125	043652	005237	002220		INC	FATFLG		
3126	043656			15\$:	CKLOOP			
	043656	104406						
3127	043660				ESCAPE	SEG		
	043660	104410						
	043662	000074						
3128	043664	005737	002222		TST	INTRECV		
3129	043670	001004			BNE	22\$		
3133	043672				ERRHRD	ERRNO,T11NINT,PKTSSR		
	043672	104456					TRAP	C\$ERHRD
	043674	002120					.WORD	1104
	043676	045364					.WORD	T11NINT
	043700	012036					.WORD	PKTSSR
3134	043702	016501	000002	22\$:	MOV	TSSR(R5),R1		
3135	043706	012702	000200		MOV	#SSR,R2		
3136	043712	032701	000100		BIT	#OFL,R1		
3137	043716	001402			BEQ	25\$		
3138	043720	052702	000100		BIS	#OFL,R2		
3139	043724	020201		25\$:	CMP	R2,R1		
3140	043726	001404			BEQ	30\$		
3144	043730				ERRHRD	ERRNO,T11NBA,PKTSSR		
	043730	104456					TRAP	C\$ERHRD
	043732	002121					.WORD	1105
	043734	045002					.WORD	T11NBA
	043736	012036					.WORD	PKTSSR
3145	043740			30\$:				
3146	043740	004737	011104	35\$:	JSR	PC,CKRAM		
3147	043744	103405			BCS	59\$		
3151	043746				ERRHRD	ERRNO,PKTRAM,RAMERR		
	043746	104456					TRAP	C\$ERHRD
	043750	002122					.WORD	1106
	043752	004741					.WORD	PKTRAM
	043754	015500					.WORD	RAMERR
3152	043756				ENDSEG			
	043756							
	043756	104405					TRAP	C\$ESEG
3153								
3154								
3155	043760			59\$:	ENDSUB			
	043760							
	043760	104403					TRAP	C\$ESUB
3156								
3157	043762	005737	002220		TST	FATFLG		
3158	043766	001402			BEQ	60\$		
3159	043770	004737	017172		JSR	PC,CKDROP		
3160	043774			60\$:				

```

;SET FATAL ERROR FLAG
;LOOP ON ERROR, IF FLAG SET
;BY-PASS SUBTEST IF FATAL ERROR
;DID AN INTERRUPT OCCUR ?
;BRANCH IF YES
;GET THE CONTENTS OF TSSR
;EXPECTED CONTENTS OF TSSR
;IS OFF-LINE BIT SET ?
;BRANCH IF NOT OFF-LINE
;SET OFF-LINE IN EXPECTED DATA
;DOES EXPECTED MATCH RECEIVED ?
;OKAY IF MATCH
;NBA NOT ZERO
;CHECK RAM TO MEMORY
;RAM OK GO ON
;THEY DON'T MATCH
;<<<<<<<<<<<<<<<< END SEGMENT<<<<<<<<<<<<<<<<
1000$:
;//////////////// END SUBTEST \\\\\\\\\\\\\\\\\\\\\\\
L10103:
;ANY FATAL ERRORS ?
;BRANCH IF NOT
;TRY TO DROP THE UNIT

```


TEST 11: NON-TAPE MOTIUN COMMANDS

```

3208 044122 005737 002222          TST      INTRECV          ;DID AN INTERRUPT OCCUR ?
3209 044126 001004          BNE      22$              ;BRANCH IF YES
3213 044130          ERRHRD  ERRNO,T11N1NT,PKTSSR
      044130 104456
      044132 002126
      044134 045364
      044136 012036
3214 044140 016501 000002      22$:   MOV      TSSR(R5),R1      ;GET THE CONTENTS OF TSSR
3215 044144 012702 100206      MOV      0SC!SSR!TSREJ,R2  ;EXPECTED CONTENTS OF TSSR
3216 044150 032701 000100      BIT      0OFL,R1          ;IS OFF-LINE BIT SET ?
3217 044154 001402          BEQ      25$              ;BRANCH IF NOT OFF-LINE
3218 044156 052702 000100      BIS      0OFL,R2          ;SET OFF-LINE IN EXPECTED DATA
3219 044162 020201          25$:   CMP      R2,R1          ;DOES EXPECTED MATCH RECEIVED ?
3220 044164 001404          BEQ      30$              ;OKAY IF MATCH
3224 044166          ERRHRD  ERRNO,T112REJ,PKTSSR ;COMMAND NOT REJECTED
      044166 104456
      044170 002127
      044172 045042
      044174 012036
3225 044176          30$:
3226 044176 004737 011104      35$:   JSR      PC,CKRAM        ;CHECK RAM TO MEMORY
3227 044202 103405          BCS      59$              ;RAM OK GO ON
3231 044204          ERRHRD  ERRNO,PKTRAM,RAMERR ;THEY DON'T MATCH
      044204 104456
      044206 002130
      044210 004741
      044212 015500
3232 044214          ENDSEG
      044214
      044214 104405
3233
3234 044216          59$:   ENDSUB
      044216
      044216 104403

```


TEST 11: NON-TAPE MOTION COMMANDS

```

3356 044574         ERRHRD  ERRNO,T11NINT,PKTSSR
      044574 104456
      044576 002142                               TRAP  C$E 0
      044600 045364                               .WORD 112
      044602 012036                               .WORD T11 NINT
      044602 012036                               .WORD PKTSSR ?
3357 044604 016501 000002      22$: MOV    TSSR(R5),R1      ;GET THE CONTENTS OF TSSR
3358 044610 012702 100206      MOV    #SC!SSR!TSREJ,R2      ;EXPECTED CONTENTS OF TSSR
3359 044614 032701 000100      BIT    #OFL,R1             ;IS OFF-LINE BIT SET ?
3360 044620 001402             BEQ    25$                  ;BRANCH IF NOT OFF-LINE
3361 044622 052702 000100      BIS    #OFL,R2             ;SET OFF-LINE IN EXPECTED DATA
3362 044626 020201      25$: CMP    R2,R1           ;DOES EXPECTED MATCH RECEIVED ?
3363 044630 001404             BEQ    30$                  ;OKAY IF MATCH
3367 044632         ERRHRD  ERRNO,T114REJ,PKTSSR      ;COMMAND NOT REJECTED
      044632 104456                               TRAP  C$ERHRD
      044634 002143                               .WORD 1123
      044636 045153                               .WORD T114REJ
      044640 012036                               .WORD PKTSSR
3368 044642      30$:
3369
3370 044642 004737 011104      35$: JSR    PC,CKRAM           ;CHECK RAM TO MEMORY
3371 044646 103404             BCS    59$                  ;RAM OK GO ON
3375 044650         ERRHRD  ERRNO,PKTRAM,RAMERR      ;THEY DON'T MATCH
      044650 104456                               TRAP  C$ERHRD
      044652 002144                               .WORD 1124
      044654 004741                               .WORD PKTRAM
      044656 015500                               .WORD RAMERR
3376 044660      59$:
3377 044660             ENDSEG      ;<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
      044660 104405                               10000$:
3378 044662         ENDSUB      ;////////////////// END SUBTEST //////////////////////
      044662 104403                               L10106:
3379                             TRAP  C$ESUB
    
```

IV WTU V7

....B1
....C1
....D1
....E1
....F1
....G1
....H1
....I1
....J1
....K1
....L1
....M1
....N1

....B5
....C5
....D5
....E5
....F5
....G5
....H5
....I5
....J5
....K5
....L5
....M5
....N5

....B9
....C9
....D9
....E9
....F9
....G9
....H9
....I9
....J9
....K9
....L9
....M9
....N9

....B13
....C13
....D13
....E13
....F13
....G13
....H13
....I13
....J13
....K13
....L13
....M13
....N13

....B2
....C2
....D2
....E2
....F2
....G2
....H2
....I2
....J2
....K2
....L2
....M2
....N2

....B6
....C6
....D6
....E6
....F6
....G6
....H6
....I6
....J6
....K6
....L6
....M6
....N6

....B10
....C10
....D10
....E10
....F10
....G10
....H10
....I10
....J10
....K10
....L10
....M10
....N10

....B14
....C14
....D14
....E14
....F14
....G14
....H14
....I14
....J14
....K14
....L14
....M14
....N14

....B3
....C3
....D3
....E3
....F3
....G3
....H3
....I3
....J3
....K3
....L3
....M3
....N3

....B7
....C7
....D7
....E7
....F7
....G7
....H7
....I7
....J7
....K7
....L7
....M7
....N7

....B11
....C11
....D11
....E11
....F11
....G11
....H11
....I11
....J11
....K11
....L11
....M11
....N11

....B15
....C15
....D15
....E15
....F15
....G15
....H15
....I15
....J15
....K15
....L15
....M15
....N15

....B4
....C4
....D4
....E4
....F4
....G4
....H4
....I4
....J4
....K4
....L4
....M4
....N4

....B8
....C8
....D8
....E8
....F8
....G8
....H8
....I8
....J8
....K8
....L8
....M8
....N8

....B12
....C12
....D12
....E12
....F12
....G12
....H12
....I12
....J12
....K12
....L12
....M12
....N12

....B16
....C16
....D16
....E16
....F16
....G16
....H16
....I16
....J16
....K16
....L16
....M16

TEST 11: NON-TAPE MOTION COMMANDS

```

3381 044664          EXIT   TST          |ALL DONE THIS TEST
      044664 104432          TRAP      CEXIT
      044666 000762          .WORD    L10102-.
3382
3383
3384      |*
3385      |LOCAL STORAGE FOR THIS TEST
3386      |-
3390 044670          T11PACKET:      |COMMAND PACKET FOR TEST
3391 044670 100204          .WORD    100204      |WRITE CHAR COMMAND, WITH IE, ACK
3392 044672 044700          .WORD    T11DATA    |ADDRESS OF CHARACTERISTICS BLOCK
3393 044674 000000          .WORD    0
3394 044676 000010          .WORD    8.         |STARTING VALUE OF BLOCK SIZE
3395
3396 044700          T11DATA:          |CHARACTERISTICS DATA BLOCK
3397 044700 044712          .WORD    T11BFR     |ADDRESS OF MESSAGE BUFFER
3398 044702 000000          .WORD    0
3399 044704 000016          .WORD    14.        |LENGTH OF MESSAGE BUFFER
3400 044706 000000 000000  .WORD    0.0
3401
3402 044712          T11BFR: .BLKW 8.   |MESSAGE BUFFER
3403
3404
3406          044740
3408 044740          T11PK2:      .=<. +10>E177770
3409 044740 100204          .WORD    100204    |COMMAND PACKET FOR TEST
3410 044742 044750          .WORD    T11DTA    |WRITE CHAR COMMAND, WITH IE, ACK
3411 044744 000000          .WORD    0         |ADDRESS OF CHARACTERISTICS BLOCK
3412 044746 000010          .WORD    8.         |STARTING VALUE OF BLOCK SIZE
3413
3414 044750          T11DTA:          |CHARACTERISTICS DATA BLOCK
3415 044750 044762          .WORD    T11BF2    |ADDRESS OF MESSAGE BUFFER
3416 044752 000000          .WORD    0
3417 044754 000016          .WORD    14.        |LENGTH OF MESSAGE BUFFER
3418 044756 000000 000000  .WORD    0.0
3419
3420 044762          T11BF2: .BLKW 8.   |MESSAGE BUFFER
3421
3422
3423
3424      |*
3425      |LOCAL TEXT MESSAGES FOR TEST
3426      |-
3427
3428 045002 111 116 111 T11NBA: .ASCIZ 'INITIALIZE Command Not Accepted'
3429 045042 111 116 111 T112REJ: .ASCIZ 'INITIALIZE Not Rejected With Non-Zero Mode Field'
3430 045123 107 105 124 T113REJ: .ASCIZ 'GET STATUS Not Accepted'
3431 045153 107 105 124 T114REJ: .ASCIZ 'GET STATUS Not Rejected With Non-Zero Mode Field'
3432 045234 103 157 156 T115SR: .ASCIZ 'Contents of TSSR Incorrect After INITIALIZE'
3433 045310 103 157 156 T11SR2: .ASCIZ 'Contents of TSSR Incorrect After GET STATUS'
3434 045364 105 170 160 T11NINT: .ASCIZ 'Expected Interrupt Not Received On INITIALIZE'
3435 045442 111 156 143 T11TSBA: .ASCIZ 'Incorrect TSBA Address After INITIALIZE'
3436 045512 116 157 156 TST11ID: .ASCIZ 'Non-Tape Motion Commands'
3437
3438          .EVEN

```

TEST 11: NON-TAPE MOTION COMMANDS

```

3440
3441
3442
3443 ;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
3444 ;INITIALIZE COMMAND
3445
3446
3447
3448 045544 T11REST:
3449 045544 SAVREG ;SAVE THE REGISTERS
3450 045550 012701 044670 MOV #T11PACKET,R1 ;START OF THE PACKET
3451 045554 012721 100213 MOV #100213,(R1)+ ;INITIALIZE WITH ACK, IE
3452 045560 005021 CLR (R1)+ ;ADDRESS OF CHAR DATA BLOCK
3453 045562 005021 CLR (R1)+ ;EXTENDED ADDRESS
3454 045564 005021 CLR (R1)+ ;SIZE OF DATA BLOCK IN BYTES
3455 045566 005021 CLR (R1)+ ;ADDRESS OF MESSAGE BUFFER
3456 045570 005021 CLR (R1)+
3457 045572 005021 CLR (R1)+ ;LENGTH OF MESSAGE BUFFER
3458 045574 005021 CLR (R1)+
3459 045576 005011 CLR (R1)
3460 045600 005037 044712 CLR T11BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
3461 045604 000207 RTS PC ;RETURN
3462
3463
3464 ;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
3465 ;GET STATUS COMMAND
3466
3467
3468
3469 045606 T11RT2:
3470 045606 SAVREG ;SAVE THE REGISTERS
3471 045612 012701 044670 MOV #T11PACKET,R1 ;START OF THE PACKET
3472 045616 012721 100217 MOV #100217,(R1)+ ;GET STATUS WITH ACK, IE
3473 045622 005021 CLR (R1)+ ;ADDRESS OF CHAR DATA BLOCK
3474 045624 005021 CLR (R1)+ ;EXTENDED ADDRESS
3475 045626 005021 CLR (R1)+ ;SIZE OF DATA BLOCK IN BYTES
3476 045630 005021 CLR (R1)+ ;ADDRESS OF MESSAGE BUFFER
3477 045632 005021 CLR (R1)+
3478 045634 005021 CLR (R1)+ ;LENGTH OF MESSAGE BUFFER
3479 045636 005021 CLR (R1)+
3480 045640 005011 CLR (R1)
3481 045642 005037 044712 CLR T11BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
3482 045646 000207 RTS PC ;RETURN
3483 045650 ENDTST
3484 045652 ENDMOD

```

L10102: TRAP C\$ETST

TEST 11: NON-TAPE MOTION COMMANDS

```

1          .TITLE  TSV6 - PARAMETER CODING
7
12
18
19 045652          BGNMOD  TSV6
045652          TSV6::
20
21          .SBTTL  HARDWARE PARAMETER CODING SECTION
22
23          ;**
24          ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
25          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
26          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
27          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
28          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
29          ; WITH THE OPERATOR.
30          ;**
31 045652          BGNHRD
045652          .WORD  L10107-L#HARD/2
045654          L#HARD::
32
33 045654          GPRMA  HPM1,0,0,160010,177776,YES          ;GET TSBA/TSDB REGISTER ADDRESS.
045654          .WORD  T#CODE
045656          .WORD  HPM1
045660          .WORD  T#LOLIM
045662          .WORD  T#HILIM
34 045664          GPRMA  HPM2,2,0,0,776,YES          ;GET VECTOR ADDRESS.
045664          .WORD  T#CODE
045666          .WORD  HPM2
045670          .WORD  T#LOLIM
045672          .WORD  T#HILIM
35          ;GPRMD  HPM3,4,0,340,0,7,YES          ;GET INTERRUPT PRIORITY.
36 045674          ENDMRD
          .EVEN
          045674          L10107:
37 045674          104          105          126          HPM1:  .ASCIZ  'DEVICE ADDRESS (TSBA/TSDB) '
38 045730          111          116          124          HPM2:  .ASCIZ  'INTERRUPT VECTOR '
39 045754          111          116          124          HPM3:  .ASCIZ  'INTERRUPT PRIORITY '
40          .EVEN

```

SOFTWARE PARAMETER CODING SECTION

```

42          .SBTTL  SOFTWARE PARAMETER CODING SECTION
43
44          ;**
45          ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
46          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
47          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
48          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
49          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
50          ; WITH THE OPERATOR.
51          ;**
52 046004          BGNSFT
53 046004          .WORD L10110-L#SOFT/2
54 046006          L#SOFT::
55          ; GPRML  SPM1,0,-1,YES          ; GET TRANSPORT TEST FLAG.
56          ; GPRML  SPM4,2,-1,YES          ; GET ITERATION CONTROL.
57 046006          .WORD  T#CODE
58 046010          .WORD  SPM4
59 046012          .WORD  -1
60          ; GPRMD  SPM6,4,D,7777,0,7777,YES          ; GET LOCAL ERROR LIMIT
61          ; GPRMD  SPM7,6,D,7777,0,7777,YES          ; GET GLOBAL ERROR LIMIT
62 046014          ENDSFT
63          .EVEN
64          L10110:
65 046014          SPM1:  .ASCIZ  'ENABLE TRANSPORT TESTS '
66 046014          SPM4:  .ASCIZ  'INHIBIT ITERATIONS '
67 046014          SPM6:  .ASCIZ  'PER TEST ERROR LIMIT '
68 046014          SPM7:  .ASCIZ  'PER UNIT ERROR LIMIT '
69          .SBTTL  PATCH AREA
70
71          ;
72          ; FINALLY A GENEROUS PATCH AREA.
73          ;
74          ; AND AN ADJUSTMENT TO ACCOUNT FOR THE "LASTAD BIT7" HACK
75          ; DESCRIBED IN "SUPPRG.MEM" (FOR REV C).
76          ;
77          PATCH::
78          .BLKW  32.
79          . = !377*1
80          LASTAD          ;SET LAST USED ADDRESS.
81          .EVEN
82          .WORD  0
83          .WORD  0
84          L#LAST::
85          ENDMOD
86          .END

```

SYMBOL TABLE

ADSSR	012116	G	C#AU	000052	DEVDR0	023414	FRESIZ	003122	G	INTFLA	016135	
ADR	000020	G	C#AUTO	000061	DEVNRD	023333	FUSI	004113		INTMAS	016134	
AMBTSS	006631		C#BRK	000022	DEVNXR	023251	F#AU	000015		INTR	016206	G
ASSEMB	000010		C#BSEG	000004	DEVONL	023201	F#AUTO	000020		INTREC	002222	G
A1716	000003		C#RSUB	000002	DEVSUM	023144	F#BGN	000040		INTVEC	016136	
BADDAT	003152	G	C#CEFG	000045	DFPTBL	002154	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	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	000036	F#MOD	000000		IXE	000000	G
BIT02	000004	G	C#DRPT	000024	EF.NEW	000035	F#MSG	000011		I#AU	000041	
BIT03	000010	G	C#DU	000053	EF.PWR	000034	F#PROT	000021		I#AUTO	000041	
BIT04	000020	G	C#EDIT	000003	EF.RES	000037	F#PWR	000017		I#CLN	000041	
BIT05	000040	G	C#ERDF	000055	EF.STA	000040	F#RPT	000012		I#DU	000041	
BIT06	000100	G	C#ERHR	000056	EMAXDU	016767	F#SEG	000003		I#HRD	000041	
BIT07	000200	G	C#ERRO	000060	EN	000000	F#SOFT	000005		I#INIT	000041	
BIT08	000400	G	C#ERSF	000054	ENAINI	016142	F#SRV	000010		I#MOD	000041	
BIT09	001000	G	C#ERSO	000057	ENVIRN	020620	F#SUB	000002		I#MSG	000041	
BIT1	000002	G	C#ESCA	000010	EPRTSW	002176	F#SW	000014		I#PROT	000040	
BIT10	002000	G	C#ESGC	000005	EPRT1	006354	F#TEST	000001		I#PTAB	000041	
BIT11	004000	G	C#ESUB	000003	EPRT2	006354	GDDAT	003154	G	I#PWR	000041	
BIT12	010000	G	C#ETST	000001	ERCM	011723	GERRMA	002172	G	I#RPT	000041	
BIT13	020000	G	C#EXIT	000032	ERRHI	002234	GETPAT	020164	G	I#SEG	000041	
BIT14	040000	G	C#GETB	000026	ERRK	016746	GETSEL	020246	G	I#SETU	000041	
BIT15	100000	G	C#GETW	000027	ERRLO	002236	G#CNT0	000200		I#SFT	000041	
BIT2	000004	G	C#GMAN	000043	ERRNO	002144	G#DELM	000372		I#SRV	000041	
BIT3	000010	G	C#GPHR	000042	ERRVEC	000000	G#DISP	000003		I#SUB	000041	
BIT4	000020	G	C#GPL0	000030	ERTABE	003372	G#EXCP	000400		I#TST	000041	
BIT5	000040	G	C#GPRI	000040	ERTABL	003172	G#HILI	000002		J#JMP	000167	
BIT6	000100	G	C#GPRI	000040	ESUM	016750	G#LOLI	000001		KIPAR0	172340	
BIT7	000200	G	C#INIT	000011	EVL	000004	G#NO	000000		KIPAR1	172342	
BIT8	000400	G	C#INLP	000020	EXBCNT	000010	G#OFFS	000400		KIPAR2	172344	
BIT9	001000	G	C#MANI	000050	EXBRE	015472	G#OFSI	000376		KIPAR3	172346	
BOE	000400	G	C#MEM	000031	EXPD	002230	G#PRMA	000001		KIPAR4	172350	
BRINIT	004453		C#MSG	000023	EXPGT	004527	G#PRMD	000002		KIPAR5	172352	
BSELO	000000		C#OPEN	000034	EXPGT2	004563	G#PRML	000000		KIPAR6	172354	
BSEL1	000001		C#PNTB	000014	EXPMSG	002320	G#RADA	000140		KIPAR7	172356	
CHKAMB	016034		C#PNTF	000017	EXPREC	015464	G#RADB	000000		KIPDR0	172300	
CHKMAN	020470	G	C#PNTS	000016	EXTA	005766	G#RADD	000040		KIPDR1	172302	
CHKTSS	016326		C#PNTX	000015	EXTEND	005764	G#RADL	000120		KIPDR2	172304	
CKDROP	017172		C#QIO	000377	EXTFEA	002224	G#RADO	000020		KIPDR3	172306	
CKEMAX	017072		C#RDBU	000007	E#END	002100	G#XFER	000004		KIPDR4	172310	
CKMSG	011350	G	C#REFG	000047	E#LOAD	000035	G#YES	000010		KIPDR5	172312	
CKMSG2	011470	G	C#RESE	000033	FATERR	000060	HIADDR	001400		KIPDR6	172314	
CKRAM	011104	G	C#REVI	000003	FATFLG	002220	HOE	100000	G	KIPDR7	172316	
CKRAM2	011214	G	C#RFLA	000021	FERCM	011712	HPM1	045674		KTENAB	003130	G
CMDPKT	021250	G	C#RPT	000025	FIFEXP	012160	HPM2	045730		KTFLG	003126	G
CMEM	017650		C#SEFG	000046	FIF1MS	012232	HPM3	045754		KTINIT	021044	
CONFIG	017240		C#SPRI	000041	FIF2MS	012301	IBE	010000	G	KTOFF	017264	
COUNT	002306	G	C#SVEC	000037	FILLME	017412	IDU	000040	G	KTON	017246	
CSRADD	002204	G	C#TPRI	000013	FNOINT	004211	IER	020000	G	LEHRMA	002170	G
CTAB	003160	G	DATA	002310	FORCER	002174	IFAUT	004252		LISTAL	000001	
CTABE	003172	G	DATASC	020222	FREE	003120	INCERK	017034		LOE	040000	G
CTABM	003160	G	DEBUGM	011632	FREEHI	003124	INTCPC	016140		LOOPCN	002214	G
			DEVCNT	002216	G							

SYMBOL TABLE

LOOPCO	013116	L10001	002174	L10073	037026	NXR	003734	PRI05	= 000240	G
LOOPFL	003156	L10002	005762	L10074	037232	NXRERR	005732	PRI06	= 000300	G
LOT	= 000010	L10003	012034	L10075	043512	NXRX	003773	PRI07	= 000340	G
L\$ACP	002110	L10004	012052	L100'6	041012	NXTU	022040	PRMESS	014232	
L\$APT	002036	L10005	012070	L100'7	041404	OFL	= 000100	PRMNO	002316	G
L\$AU	022370	L10006	012076	L10100	042274	ONEFIL	= 000000	PRMSG	014542	G
L\$AUT	002070	L10007	012114	L10101	042522	O#APTS	= 000000	PRMSG0	014722	
L\$AUTO	022574	L10010	012132	L10102	045650	O\$AU	= 000001	PRMSG1	014767	
L\$CCP	002106	L10011	012156	L10'03	043760	O#BGNR	= 000001	PRMSG2	015025	
L\$CLEA	022654	L10012	012230	L10104	044216	O#BGNS	= 000001	PROASC	014410	
L\$CO	002032	L10013	012400	L10105	044436	O#DU	= 000001	PR1ASC	014455	
L\$DEPO	002011	L10014	013114	L10106	044662	O#ERRT	= 000000	PST32W	003146	G
L\$DESC	003404	L10015	013742	L10107	045674	O#GNSW	= 000001	PUNIT	022322	
L\$DESP	002076	L10016	013764	L10110	046014	O#POIN	= 000001	PW.D11	= 000021	
L\$DEVP	002060	L10017	015470	MEMADD	013744	O#SETU	= 000000	PW.D13	= 000022	
L\$DISP	002124	L10020	015476	MEMCK	021266	PASRPT	022072	PW.D22	= 000020	
L\$DLY	002116	L10021	015504	MENASC	020437	PATCH	046154	PW.NOP	= 000000	
L\$DTP	002040	L10022	015516	MENERR	020364	PATDAT	020220	PW.N01	= 000023	
L\$DTYP	002034	L10023	015540	MENRES	020466	PC.ERA	= 002400	PW.RDE	= 000024	
L\$DU	022466	L10024	015566	MMVEC	= 000250	PC.IER	= 002000	PW.RDR	= 000001	
L\$DUT	002072	L10025	015726	MSA.FR	= 000006	PC.N00	= 001000	PW.RDS	= 000005	
L\$DVTY	003376	L10026	016236	MSA.NO	= 000000	PC.REL	= 000000	PW.RFI	= 000003	
L\$EF	002052	L10030	022320	MSA.NR	= 000004	PC.REW	= 000400	PW.WCT	= 000006	
L\$ENVI	002044	L10031	022464	MSA.VO	= 000002	PKBCNT	= 000006	PW.WFI	= 000004	
L\$ETP	002102	L10032	022572	MSGEXP	012134	PKHI	= 000004	PW.WFM	= 000007	
L\$EXP1	002046	L10033	022652	MSGLOO	013054	PKLOW	= 000002	PW.WMI	= 000010	
L\$EXP4	002064	L10034	022700	MSGSTA	012340	PKTADD	007550	PW.WNP	= 000011	
L\$EXP5	002066	L10035	023142	MSGSUB	013732	PKTFRM	007512	PW.WTR	= 000002	
L\$HARD	045654	L10036	023702	MS.ATT	= 000006	PKTGET	012054	P.ACK	= 100000	
L\$HIME	002120	L10037	023550	MS.EXT	= 000200	PKTMES	012100	P.CMD	= 000037	
L\$HPCP	002016	L10040	023632	MS.RSD	= 000001	PKTRAM	004741	P.CONT	= 000012	
L\$HPTP	002022	L10041	024400	MS.RSF	= 000020	PKTSSR	012036	P.CVC	= 040000	
L\$HW	002154	L10042	025072	MS.RST	= 000010	PNT	= 001000	P.FMT	= 000140	
L\$ICP	002104	L10043	026426	M2901	026152	PRAMPK	013766	P.FORM	= 000011	
L\$INIT	021546	L10044	025334	M8186	005550	PRASC	014513	P.GETS	= 000017	
L\$LADP	002026	L10045	025634	M8189	005641	PRBEXP	015460	P.IE	= 000200	
L\$LAST	046404	L10046	026132	NBA	= 002000	PRBMSG	015326	P.INIT	= 000013	
L\$LOAD	002100	L10047	027532	NEWPAS	022026	PRBREC	015462	P.MODE	= 007400	
L\$LUN	002074	L10050	027000	NODEV	003110	PRBTOT	015413	P.OPP	= 020000	
L\$MREV	002050	L10051	027350	NOINIT	004331	PRBYTE	015112	P.POSI	= 000010	
L\$NAME	002000	L10052	031002	NOINTR	004215	PRI	= 002000	P.READ	= 000001	
L\$PRIO	002042	L10053	030062	NOITS	002166	PRIADD	010154	P.SWB	= 010000	
L\$PROT	021536	L10054	030406	NOMAN	020524	PRIAO	010224	P.WRIT	= 000005	
L\$PRT	002112	L10055	034370	NOMEM	005454	PRI BXO	007606	P.WRTC	= 000004	
L\$REPP	002062	L10056	031334	NP.IR	= 000200	PRIEQU	010054	P.WRTS	= 000006	
L\$REV	002010	L10057	031620	NP.LOO	= 000040	PRIPKT	007364	QVP	002202	G
L\$RPT	022702	L10060	032064	NP.OUT	= 000100	PRIPAM	010062	RAMASC	014146	
L\$SOFT	046006	L10061	032312	NP.WRP	= 000020	PRITAD	010270	RAMDAT	002240	G
L\$SPC	002056	L10062	032556	NSI	004146	PRITSS	006020	RAMERR	015500	G
L\$SPCP	002020	L10063	033116	NSINIT	004403	PRITO	010352	RAMEXP	015520	G
L\$SPTP	002024	L10064	035274	NUL	004523	PRIT1	010415	RAMFOR	010112	
L\$STA	002030	L10065	040410	NULCR	004524	PRI XOR	007736	RAMSIZ	002300	G
L\$SW	002164	L10066	035536	NXM	= 004000	PRI00	= 000000	RAMTAD	015506	G
L\$TEST	002114	L10067	035776	NXMFLG	003132	PRI01	= 000040	RCVHIA	002302	G
L\$TIML	002014	L10070	036202	NXMH1	003136	PRI02	= 000100	RCVLOA	002304	G
L\$UNIT	002012	L10071	036370	NXMLO	003134	PRI03	= 000140	RDERR	005202	
L10000	002162	L10072	036574	NXMTST	021442	PRI04	= 000200	RECM SG	002464	G

SYMBOL TABLE

RECV	002232	G	S1.IEO	010000	TST6ID	030763	T1.1	023504	T4	025074	G
REGSAV	020130		S1.IFM	001000	TST7ID	034273	T1.2	023564	T4LOOP	025114	
RETErr	005366		S1.IHE	000400	TST8ID	035257	T10	040412	T4.1	025074	
REWIND	011004	G	S1.IID	004000	TST9ID	040311	T10BFR	042552	T4.2	025336	
RMCHBE	000167		S1.IIR	020000	TSV2	002000	T10BUF	042614	T4.3	025636	
RMCHEN	000200		S1.IZR	040000	TSV3	002174	T10DAT	042540	T5	026430	G
RMMSGB	000215		S1.PAR	100000	TSV4	021536	T10DTA	042602	TSADDR	027470	
RMMSGE	000234		S2.ATI	000010	TSV5	023464	T10INT	043250	TSLOOP	026446	
RMPKTB	000201		S2.BTI	000004	TSV6	045652	T10LOO	040430	T5MEM	027432	
RMPKTE	000210		S2.DJM	000200	TTIBFR	177562	T10MBF	042634	T5.1	026452	
RMR	010000		S2.ILW	000100	TTICSR	177560	T10NBA	042731	T5.2	027020	
RWPACK	011100		S2.INR	000020	TTIVEC	000060	T10NIN	043157	T6	027534	G
SC	100000		S2.OUT	000040	T#ARGC	000003	T10NNB	043013	T6INT	030553	
SCE	020000		S2.UND	000003	T#CODE	001130	T10PAC	042530	T6LOOP	027552	
SCHErr	005274		TBLEND	003056	T#ERRN	002144	T10PKT	042572	T6NBA	030450	
SCME	005007		TCOASC	006472	T#EXCP	000000	T10RST	043366	T6NINT	030631	
SDELAY	010650		TCOCOD	006672	T#FLAG	000040	T10RT2	043440	T6PACK	030440	
SELASC	020432		TEMP1	003112	T#GMAN	000000	T10SSR	043070	T6SSR	030475	
SELDAT	000004		TEMP2	003114	T#HILI	000776	T10.1	040430	T6TSBA	030711	
SEL2	000002		TERCLS	000016	T#LAST	000001	T10.2	041014	T6.1	027552	
SETMAP	017306		TESTNO	000013	T#LOLI	000000	T10.3	041406	T6.2	030076	
SETU	022124		TEXASC	006431	T#LSYM	010000	T10.4	042276	T7	031004	G
SFFMSG	012072	G	TFCASC	006533	T#LTNO	000013	T11	043514	T7BFR	033166	
SFHERR	003701		TIMEXP	015542	T#NEST	177777	T11BFR	044712	T7DATA	033150	
SFIERR	003646		TIMSGO	015570	T#NSO	000000	T11BF2	044762	T7INT	034121	
SFIMSG	012024	G	TINERR	012011	T#NS1	000005	T11DAT	044700	T7LOOP	031022	
SFPTBL	002164	G	TMPBFR	002630	T#NS2	000002	T11DTA	044750	T7NBA	033300	
SIFLAG	003150	G	TNAM	016674	T#NS3	000003	T11LOO	043532	T7NINT	034030	
SIMSG	011756		TRANST	002164	T#PTNU	000000	T11NBA	045002	T7PACK	033140	
SKIPT	003374		TSBA	000000	T#SAVL	177777	T11NIN	045364	T7REST	034322	
SOFINI	015764	G	TSBAH	000001	T#SEGL	177777	T11PAC	044670	T7SP	033160	
SPACE	010462	G	TSBAM2	026242	T#SEKO	010000	T11PK2	044740	T7SSR	033741	
SPM1	046014		TSBAM3	026324	T#SUBN	000004	T11RES	045544	T7TSBA	034210	
SPM4	046044		TSDB	000000	T#TAGL	177777	T11RT2	045606	T7.1	031022	
SPM6	046074		TSDBH	000001	T#TAGN	010111	T11SR2	045310	T7.2	031350	
SPM7	046124		TSFCOD	007232	T#TEMP	000000	T11SSR	045234	T7.3	031622	
SRO	177572		TSREJ	000006	T#TEST	000013	T11TSB	045442	T7.4	032066	
SR1	177574		TSSDEF	006602	T#TSTM	177777	T11.1	043532	T7.5	032314	
SR2	177576		TSSR	000002	T#TSTS	000001	T11.2	043774	T7.6	032560	
SR3	172516		TSSRBI	003476	T#AU	010031	T11.3	044220	T72DAT	033206	
SSR	000200		TSSRFO	006411	T#AUT	010033	T11.4	044440	T72DON	033222	
STATCO	012402		TSSRH	000003	T#CLE	010034	T112RE	045042	T72NBA	033222	
SVCGBL	000000		TSSX	004014	T#DU	010032	T113RE	045123	T72REJ	033353	
SVCINS	000000		TSTBLK	002750	T#HAR	010107	T114RE	045153	T73REJ	033452	
SVCSUB	000001		TSTCNT	002212	T#HW	010000	T2	023704	T74REJ	033545	
SVCTAG	000000		TSTEND	016710	T#INI	010030	T2LOOP	023722	T75REJ	033643	
SVCTST	000001		TSTFLA	002312	T#MSG	010025	T2SSR	024300	T8	034372	G
S#LSYM	010000		TSTLOO	016446	T#PRO	010027	T2TSBA	024166	T8BFR	035022	
S#.IDB	000010		TSTPTR	002314	T#RPT	010035	T2TSSR	024233	T8DATA	035010	
SO.IFB	000002		TSTSET	016500	T#SEG	010000	T23A	003140	T8LOOP	034410	
SO.IFP	000001		TST1ID	023662	T#SOF	010110	T23B	003142	T8NVCK	035077	
SO.ILD	000020		TST1OI	043337	T#SMV	010026	T3	024402	T8PACK	035000	
SO.ION	000040		TST1II	045512	T#SUB	010106	T3BFLG	003144	T8SSR	035170	
SO.IRD	000100		TST2ID	024352	T#SW	010001	T3LOOP	024420	T8VCK	035042	
SO.IRW	000004		TST3ID	025045	T#TES	010102	T3SSR	024774	T9	035276	G
SO.ISP	000200		TST4ID	026404	T1	023464	T3TSBA	024664	T9BFR	037262	
S1.ICB	002000		TST5ID	027402	T1LOOP	023502	T3TSSR	024730	T9DATA	037250	

SYMBOL TABLE

T9INY	040137	UAM	=	000200	G	WRTCHR	010652	G	XSOONL	=	000100	X2.EXT	=	000200	
T9LOOP	035320	UNITN		002200	G	WRTERR	005107		XSOPED	=	000010	X2.OPM	=	100000	
T9NBA	037316	UNREC	=	000006		WRTMSG	005052		XSORLL	=	010000	X2.RCE	=	040000	
T9NINT	040046	USI		004117		WSMBK	021260	G	XSORLS	=	040000	X2.REV	=	000077	
T9PCK	037240	WAITF		016240	G	XFERAS	015730		XSOTMK	=	100000	X2.SPA	=	035400	
T9REST	040336	WC.IFA	=	000200		XNXM	016366		XSOVCK	=	000020	X2.UNI	=	000007	
T9SSR	037757	WC.IFE	=	000002		XORBFO	007670		XSOWLE	=	004000	X2.WCF	=	002000	
T9TSBA	040226	WC.IGO	=	000001		XORFOR	010006		XSOWLK	=	000004	X3.DCK	=	000010	
T9.1	035320	WC.IRE	=	000010		XST0	=	000006	G	XXCOMM	003116	G	X3.MBZ	=	000006
T9.2	035566	WC.IRW	=	000004		XST1	=	000010	G	X#ALWA	=	000000	X3.MDE	=	177400
T9.3	036000	WC.IGT	=	000100		XST2	=	000012	G	X#FALS	=	000040	X3.OPI	=	000100
T9.4	036204	WC.IIT	=	000040		XST3	=	000014	G	X#OFFS	=	000400	X3.REV	=	000040
T9.5	036372	WC.I5R	=	000020		XST4	=	000016	G	X#TRUE	=	000020	X3.RIB	=	000001
T9.6	036576	WF.IED	=	000010		XSOBOT	=	000002		X1.COR	=	020000	X3.SPA	=	000200
T9.7	037030	WF.IER	=	000004		XSOEOT	=	000001		X1.DLT	=	100000	X3.TRF	=	000020
T92DAT	037302	WF.IHI	=	000200		XSOIE	=	000040		X1.MBZ	=	017375	X4.HSP	=	100000
T92DON	037316	WF.IRE	=	000040		XSOILA	=	000400		X1.RBP	=	000400	X4.MBZ	=	017400
T92REJ	037371	WF.IWF	=	000020		XSOILC	=	001000		X1.SPA	=	040000	X4.RCE	=	040000
T93REJ	037470	WF.IWR	=	000100		XSOLET	=	020000		X1.UNC	=	000002	X4.TSM	=	020000
T94REJ	037563	WF.I3R	=	000002		XSOMOT	=	000200		X2.BUF	=	000100	X4.WRC	=	000377
T95REJ	037661	WF.I4R	=	000001		XSONEF	=	002000							

. ABS. 046404 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 28880 WORDS (113 PAGES)

DYNAMIC MEMORY: 20060 WORDS (77 PAGES)

ELAPSED TIME: 00:37:00

CNTSAA0,BIC,CNTSAA0,SEQ/-SP*SVC34/ML,TSV1A,TSV22A,TSV3A,TSV4,TSV5A,TSV6

....B1
....C1
....D1
....E1
....F1
....G1
....H1
....I1