

TK25

TK25 FRT END FUNC #1
CZTKEAO

COPYRIGHT (c) 1984
AH-T726A-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA

This microfiche card contains 16 frames of data. The frames are arranged in a 4x4 grid. Each frame displays a small, high-contrast image, likely a scan of a document page. The text within these frames is too small and faded to be legible. The overall appearance is that of a standard microfiche card used for data storage and retrieval.

.REMN

IDENTIFICATION

PRODUCT ID: AC-T775A-MC
PRODUCT TITLE: CZTKEA TK25 FRT END FUNC #1
PRODUCT DATE: MARCH, 1984
DEPARTMENT: TAPE DIAGNOSTIC ENGINEERING
AUTHOR: DICE SYSTEMS, INC.

COPYRIGHT (C) 1984 BY
DIGITAL EQUIPMENT CORPORATION,
MAYNARD, MASSACHUSETTS.
ALL RIGHTS RESERVED.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

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

TABLE OF CONTENTS

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	HARDWARE REQUIREMENTS
2.2	SOFTWARE REQUIREMENTS
2.3	PREREQUISITES
3.0	OPERATING INSTRUCTIONS - OPERATOR COMMANDS
3.1	OPERATOR COMMANDS
3.2	HARDWARE PARAMETERS
3.3	SOFTWARE PARAMETERS
4.0	OPERATING INSTRUCTIONS - SAMPLE PRINTOUTS
4.1	SUCCESSFUL RUN EXAMPLES
4.2	ERROR MESSAGES
5.0	PROGRAM RUN TIMES
5.1	RUN TIME - CZTKE
6.0	TEST DESCRIPTIONS - CZTKE
6.1	TEST 1 - INITIALIZATION TEST 1
6.2	TEST 2 - RAM TEST
6.3	TEST 3 - COMMAND REJECT
6.4	TEST 4 - WRITE CHARACTERISTICS
6.5	TEST 5 - VOLUME CHECK
6.6	TEST 6 - COMPLETION INTERRUPT
6.7	TEST 7 - BASIC PACKET PROTOCOL
6.8	TEST 8 - NON-TAPE MOTION COMMANDS
6.9	TEST 9 - COMPLETION INTERRUPT
6.10	TEST 10 - MEMORY ADDRESSING
6.11	TEST 11 - BASIC WRITE SUBSYSTEM MEMORY TEST

ABSTRACT

1.0 ABSTRACT

THIS IS A PDP-11/LSI RESIDENT DIAGNOSTIC WHICH CHECKS THE FUNCTIONALITY OF AN IK25 MAGTAPE SUBSYSTEM WHILE CONNECTED TO A PDP-11 SYSTEM (Q-BUS OR UNIBUS). THE PROGRAM HAS BEEN DIVIDED INTO FOUR MAJOR PIECES: CZTKE, CZTKF, CZTKG, CZTKH. SUCCESSFUL RUN EXAMPLES, AND TEST DESCRIPTIONS HAVE BEEN PROVIDED FOR EACH PROGRAM.

THE PROGRAMS PROVIDE ERROR MESSAGES WHICH IDENTIFY FAILING FUNCTIONS, AND AID IN DEVICE REPAIR. REFERENCE THE FOLLOWING DIGITAL EQUIPMENT DOCUMENTS:

1. CIQPMAO XXDP+ PROGRAMMER'S MANUAL; DOCUMENT NUMBER AC-S296A-AC; DATE: 14 JULY 1980.

1.1 REVISION HISTORY

NEW RELEASE APRIL 1984

2.0 REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

PDP-11 FAMILY PROCESSOR WITH 32K WORDS OF MEMORY
TK25 MAGTAPE SUBSYSTEM (DRIVE AND CONTROLLER)
CAUTION:DIAGNOSTIC REQUIRES 32K WORDS OF MEMORY
(28K USEABLE I.E. 4K FOR I/O PAGE)

2.1.1 OPTIONAL HARDWARE -

FOUR TK25 CONTROLLERS PER PDP-11, ONE
DRIVE PER CONTROLLER

2.2 SOFTWARE REQUIREMENTS

PDP-11 DIAGNOSTIC SUPERVISOR (CIQPMAO VERSION 34 OR LATER)
PDP-11 DIAGNOSTIC LOADER/MONITOR (XXDP+)

2.3 PREREQUISITES

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

3.0 OPERATING INSTRUCTIONS - OPERATOR COMMANDS

3.1 OPERATOR COMMANDS

THE TK25 DIAGNOSTICS ARE PDP-11 DIAGNOSTIC SUPERVISOR COMPATIBLE PROGRAMS. ALL LOADING AND RUN TIME INSTRUCTIONS CAN BE REFERENCED IN THE PDP-11 PROGRAMMER'S MANUAL "CIQPMAO XXDP" PROGRAMMER'S MANUAL NUMBER AC-S296A-AC.

BOOT THE DIAGNOSTIC XXDP+ MEDIA (OPERATOR RESPONSES ARE UNDERLINED)

CHMDLEO XXDP+ DL MONITOR
BOOTED VIA UNIT 0
28K NON-UNIBUS SYSTEM

ENTER DATE <DD-MMM-YY>: 29-JAN-82

RESTART ADDRESS: 152010 -----
THIS IS XXDP+ TYPE "H" OR "H/L" FOR HELP.

.R CZTKEA

- -----

CZTKEA.BIC

DRS-E0
CZTKE-A-0
CZTKEA TK-25 FRT END FUNC #1 UNIT IS TK25
RSTRT ADR 147642
DR>START/FLAG:PNT:HOE

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".

3.2 HARDWARE PARAMETERS

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 QUESTION, THE PROGRAM WILL USE IT'S DEFAULT HARDWARE PARAMETER VALUES. IT WILL DEFAULT TO ONE UNIT SELECTED (UNIT 0). THE DEFAULT TSBA/TSDB WILL BE 172522 AND THE INTERRUPT VECTOR WILL BE 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 ONLY IF 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)" THAT A LOGICAL RESPONSE IS TO BE MADE: "Y" FOR YES, "N" FOR NO.

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

UNIT 0

DEVICE ADDRESS (O) 172522 ? <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 EIGHT UNITS CAN BE SELECTED FOR TESTING.

3.3 SOFTWARE PARAMETERS

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.>

ENABLE RAM DUMP ON ERROR (L) N? < TYPE "Y" TO DUMP
SELECTED RAM CONTENTS IN THE
CONTROLLER MODULE.>

4.0 OPERATING INSTRUCTIONS - SAMPLE PRINTOUTS

4.1 SUCCESSUL RUN EXAMPLES

4.1.1 SUCCESSFUL RUN EXAMPLE - CZTKE -

```
TST: 001 INITIALIZATION TEST
TST: 002 RAM TEST
TST: 003 COMMAND REJECT TEST
TST: 004 WRITE CHARACTERISTICS TEST
TST: 005 VOLUME CHECK TEST
TST: 006 COMPLETION INTERRUPT TEST
TST: 007 BASIC PACKET PROTOCAL TEST
TST: 008 NON-TAPE MOTION COMMANDS TEST
TST: 009 DMA MEMORY ADDRESSING TEST
TST: 010 INITIALIZATION AFTER WRITE CHARACTERISTICS TEST
TST: 011 BASIC WRITE SUBSYSTEM MEMORY TEST
CZTKE EOP 1
      0 TOTAL ERRS
```

NOTE: PROGRAM NOW STARTS OVER AGAIN AT TEST 1

4.2 OPERATING INSTRUCTIONS - SAMPLE ERROR MESSAGE

ERROR MESSAGE EXAMPLE

TST: 001
CZTKE DVC FTL ERR 00001 ON UNIT 00 TST 001 SUB 000 PC: 017300
NON-EXISTANT DEVICE REGISTER
ADDRESS: 172500

UNIT 0 DROPPED
PASS ABRTD THS UNIT
CZTKE EOP 1
1 TOTAL ERRS

5.0 PROGRAM RUN TIMES

THE AVERAGE RUN TIMES OF THE PROGRAMS ARE LISTED BELOW. THESE FIGURES ARE TO BE USED AS A GUIDE. THE TIMING WAS DONE ON A PDP-11/23 (LSI) PROCESSOR WITH A LA-120 CONSOLE.

THE PROGRAMS RUN IN NON-ITERATIVE MODE. EACH TEST IS RUN ONCE, WITH NO ITERATIONS. THEREFOR, THE DEFAULT MODE (NORMALLY ITERATIVE) AND THE NON-ITERATIVE MODE TIMES ARE IDENTICAL.

5.1 RUN TIMES - CZTKE

TEST NUMBER	N/I SECS.	DEF SECS.
1	8	8
2	11	11
3	9	9
4	10	10
5	12	12
6	10	10
7	2	2
8	8	8
9	5	5
10	8	8
11	13	13

THE TIMES REQUIRED TO RUN TESTS 1 THROUGH 11 IN ONE COMMAND:

Q.V.	1 MIN 46 SECONDS
DEFAULT	1 MIN 46 SECONDS

6.0 TEST DESCRIPTIONS - CZTKE

6.1 TEST 1 - INITIALIZATION TEST 1

* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *

THIS TEST VERIFIES THAT THE 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 WITH THE MODULE. AREAS OF LOGIC TESTED BY THE SELF-TEST SEQUENCER ARE AS FOLLOWS: ROM AND PIPELINE REGISTER, SEQUENCER, INTERNAL BUSES, 2901 MICROPROCESSOR, RAM AND TRANSPORT STATUS FLOPS. THIS TEST INITIALIZES THE CONTROLLER BY ISSUING THE BUS INIT SIGNAL VIA A RESET INSTRUCTION, OR BY WRITING INTO THE TSSR REGISTER, 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, A16, AND OFL, WHICH ARE IGNORED FOR THIS TEST) BEING CLEARED (0). IF THE CONTENTS OF THE TSSR ARE NOT AS EXPECTED, AN ERROR REPORT IS ISSUED LISTING THE EXPECTED DATA, ACTUAL DATA, AND THE DISCREPANCIES.

6.2 TEST 2 - RAM TEST

* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *

THIS TEST VERIFIES THAT ALL LOCATIONS OF THE RAM ON THE CONTROLLER CAN PROPERLY STORE AND READ BACK ALL DATA PATTERNS, AND THAT EACH RAM LOCATION IS UNIQUELY ADDRESSED (IE: THAT ONE AND ONLY ONE LOCATION IS ACCESSED BY ANY PARTICULAR ADDRESS). THESE TESTS ARE PERFORMED BY THREE SUBTESTS DESCRIBED BELOW.

6.2.1 TEST 2, SUBTEST 1: -

THIS SUBTEST VERIFIES EACH LOCATION BY PERFORMING THE FOLLOWING SEQUENCE FOR EACH ADDRESS 0-377 (OCTAL):

1. THE ADDRESS TO BE TESTED IS LOADED INTO THE TSDB+1 (VIA A HI-WRITE BYTE).
2. THE ADDRESSED RAM LOCATION IS READ, THEN WRITTEN INTO THE LOW BYTE OF THE TSBA.
3. THE LOW BYTE OF THE 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+1 (HI-BYTE 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.

6.2.2 TEST 2, SUBTEST 2: -

THIS SUBTEST USES THE SAME RAM READ/WRITE TECHNIQUES AS SUBTEST 1, EXCEPT THAT MEMORY IS FILLED WITH ZEROS AND A ONES WORD IS "WALKED" DOWN THROUGH. PRIOR TO THE ALL ONES WRITE TO MEMORY THE MEMORY IS CHECKED TO BE SURE THAT THE ZERO WORD HASN'T "PICKED" A BIT.

6.2.3 TEST 2, SUBTEST 3: -

THIS SUBTEST IS SIMILAR TO SUBTEST 2, EXCEPT THAT MEMORY IS FIRST SET TO ALL ONES AND A BYTE OF ZEROS IS "WALKED" DOWN THROUGH MEMORY BEGINNING AT LOCATION TOP-2.

6.3 TEST 3: COMMAND REJECT

 * NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
 * CONTROLLER *

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 THE TSSR REGISTERS ARE SET IN THE PROPER STATE AFTER EACH COMMAND IS REJECTED. THIS TEST CHECKS THE MICROPROCESSOR SEQUENCING, BASIC COMMAND DECODING, AND DATI DMA HANDLING. THE TEST CONTAINS TWO SUBTESTS: SUBTEST 1 SEQUENCES THROUGH ALL COMMAND WORDS (OTHER THAN WRITE CHARACTERISTICS) WITH THE INTERRUPT ENABLE BIT (IE) BIT CLEAR AND VERIFIES THAT AN INTERRUPT IS NOT GENERATED BY THE REJECTED COMMAND; SUBTEST TWO 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. THE 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 THE TSSR ARE CHECKED. TSSR IS CORRECT IF IT CONTAINS EITHER OCTAL 102206 OR 102306 (BIT 6 DEPENDS ON 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) IE: TSBA SHOULD POINT TO THE COMMAND PACKET.
6. USING THE MAINTENANCE MODE WRAP AROUND FUNCTIONS, THE COMMAND IMAGE BLOCK IN THE CONTROLLER'S RAM 20 - 27 (OCTAL)) ARE CHECKED; THE 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; E.G., THE REMAINING THREE WORDS OF THE COMMAND BUFFER ARE SEQUENCED WITH PSEUDO-RANDOM DATA. IF THE COMMAND WORD HAS NOT REACHED ITS MAXIMUM VALUE (17777+1) THE TEST SEQUENCE IS REPEATED.

6.4 TEST 4 - WRITE CHARACTERISTICS

* NOTE: IF THIS TEST DETECTS A FAILURE REPLACE THE TK25'S *
* CONTROLLER *

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 THE 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 CHARACTERISTICS 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 PROCEESOR PRIORITY ZERO, 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 TESTS VARIOUS MICROPROGRAM SEQUENCES, COMMAND DECODING, DMA LOGIC, AND BASIC PACKET PROTOCOL HANDLING. THIS IS THE FIRST TEST IN WHICH DATA DMA CYCLES (FOR STORING THE MESSAGE PACKET) ARE PERFORMED. ANY ERRORS IN THE BODY OF THE TEST (IE: ERRORS OTHER THAN INITIALIZATION ERRORS RELATED TO THE TRANSPORT BUS) DEFINITELY INDICATES A BAD CONTROLLER MODULE.

6.4.1 TEST 4, SUBTEST 1: -

VERIFIES BASIC STANDARD OPERATION (USING PROPER MESSAGE BUFFER AND LENGTH DATA IN AN INCREMENTING SERIES OF VALUES FOR THE FOURTH CHARACTERISTICS DATA IN THE CHARACTERISTICS DATA BLOCK.). AFTER THE COMMAND IS EXECUTED FOR EACH VALUE OF THE FOURTH CHARACTERISTICS DATA WORD, THE PROGRAM VERIFIES THAT:

1. THE TSSR IS CORRECT, INCLUDING A CHECK THAT THE NBA BIT IS CLEARED AND THAT NORMAL TERMINATION WAS ACCOMPLISHED.
2. THAT A PROPER MESSAGE PACKET IS STORED.
3. THAT THE COMMAND PACKET CHARACTERISTIC DATA, AND MESSAGE PACKET IMAGE BLOCKS IN CONTROLLER RAM ARE CORRECT.

6.4.2 TEST 4, SUBTEST 2: -

VERIFIES THAT THE COMMAND IS REJECTED AND THAT THE NBA BIT DOES NOT GET CLEARED IF NONZERO BITS ARE SET INTO ANY RESERVED OR UNUSED FIELD WITHIN THE FIRST THREE COMMAND PACKET WORDS.

6.4.3 TEST 4, SUBTEST 3: -

VERIFIES THAT THE COMMAND IS REJECTED AND THAT THE NBA BIT DOES NOT GET CLEARED IF THE MESSAGE BUFFER ADDRESS SPECIFIED IN THE CHARACTERISTICS DATA BLOCK DOES NOT SPECIFY A LEGAL ADDRESS.

6.5 TEST 5: VOLUME CHECK

* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *

THIS TEST VERIFIES THAT THE VOLUME CHECK (VCK) BIT, A FLAG HELD WITHIN THE CONTROLLER AND APPEARING IN XSTO, IS SET BY INITAILIZE AND CLEARED BY EXECUTING A WRITE CHARACTERISTICS COMMAND WITH THE CVC SET. IT IS ALSO VERIFIED THAT A WRITE CHARACTERISTICS 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 MOTIN COMMAND DEPENDING ON 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)
3. THE PREVIOUS STEP IS REPEATED TO VERIFY THAT VCK DOES NOT CHANGE (REMAINS AT 0).
4. A WRITE CHARACTERISTICS COMMAND IS ISUED 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 ISUED WITH CVC=0 AND THE VCK BIT IN XSTO IN THE MESSAGE BUFFER IS EXAMINED; THE VCK BIT SHOULD REMAIN CLEAR (0). THIS 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 IS VERIFIED BY LOW BYTE OF TSBA WHICH SHOULD CONTAIN ALL 1'S.
6. 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.

6.6 TEST 6 - COMPLETION INTERRUPT

```

*****
* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *
*****

```

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 STATUS BIT IN XSTO OF ANY MESSAGE PACKET IS SET AND THAT A COMPLETION INTERRUPT IS GENERATED. FINALLY A SEQUENCE OF TWO COMMANDS IS 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.

6.7 TEST 7 - BASIC PACKET PROTOCOL

* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *

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.

6.7.1 TEST 7, SUBTEST 1: -

VERIFIES THAT THE BASIC MESSAGE BUFFER RELEASE COMMAND OPERATES PROPERLY
WHEN MESSAGE BUFFER RELEASE INTERRUPTS ARE DISABLED (ERI=0 ON PREVIOUS
WRITE CHARACTERISTICS COMMAND). CHECKS THAT TSSR IS UPDATED PROPERLY
AND THAT NO INTERRUPT IS GENERATED (EVEN IF THE IE BIT IN THE COMMAND
WORD IS SET) AND THAT NO MESSAGE PACKET IS STORED.

6.7.2 TEST 7, SUBTEST 2: -

VERIFIES THAT THE BASIC MESSAGE BUFFER RELEASE COMMAND OPERATES PROPERLY
WHEN MESSAGE BUFFER RELEASE INTERRUPTS ARE ENABLED (ERI=1 ON PREVIOUS
WRITE CHARACTERISTICS COMMAND). CHECKS THAT TSSR IS UPDATED PROPERLY
AND THAT AN INTERRUPT IS GENERATED (IF THE IE BIT IN THE COMMAND WORD IS
SET) BUT THAT NO MESSAGE PACKET IS STORED.

6.7.3 TEST 7, SUBTEST 3: -

VERIFIES THAT AFTER THE CPU GIVES UP OWNERSHIP OF A MESSAGE BUFFER (VIA
THE MESSAGE BUFFER RELEASE COMMAND), THAT A NEW COMMAND (E.G., WRITE
CHARACTERISTICS) IS PROPERLY EXECUTED WHEN ISSUED WITH THE ACK BIT IN
THE COMMAND HEADER EITHER SET OR CLEAR.

6.7.4 TEST 7, SUBTEST 4: -

VERIFIES THAT THE REGISTER VERIFICATION REFUSED (RMR) BIT IN TSSR
OPERATES PROPERLY WHEN A COMMAND (WRITE CHARACTERISTICS) IS BEING
EXECUTED. THE PROGRAM ISSUES THE WRITE CHARACTERISTICS COMMAND (FROM
ONE COMMAND BUFFER) THEN IMMEDIATELY WRITES THE ADDRESS OF ANOTHER
COMMAND BUFFER (CONTAINING ANOTHER WRITE CHARACTERISTICS COMMAND, BUT
WITH THIS ONE SPECIFYING DIFFERENT CHARACTERISTICS DATA). WHEN SSR
SETS, THE PROGRAM VERIFIES THAT THE FIRST COMMAND COMPLETED PROPERLY,
THAT RMR IS SET, AND THAT THE SECOND COMMAND IS IGNORED.

6.8 TEST 8 - NON-TAPE MOTION COMMANDS

```

*****
* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S       *
* CONTROLLER                                                       *
*****

```

THIS TEST VERIFIES PROPER OPERATION OF THE GET STATUS AND INITIALIZE
 COMMANDS. THREE SUBTESTS ARE USED. THE FIRST TWO VERIFY THAT THE
 RESPECTIVE COMMANDS RUN TO COMPLETION AND STORE A VALID MESSAGE PACKET.

6.9 TEST 9 - MEMORY ADDRESSING TEST

```

*****
* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *
*****

```

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

6.9.1 TEST 9, SUBTEST 1: -

THIS SUBTEST VERIFIES THAT THE CONTROLLER CAN FETCH A GET STATUS COMMAND FROM ALL AVAILABLE MEMORY LOCATIONS. TWO WORD BLOCKS ARE TESTED ONE AT A TIME BY FIRST SETTING ALL AVAILABLE MEMORY TO A BACKGROUND PATTERN OF 12252. A GET STATUS COMMAND IS THEN EXECUTED TO VARIOUS ADDRESSES IN EACH AVAILABLE MEMORY 4K BLOCK. THE VARIOUS ADDRESSES ARE DETERMINED BY FLOATING FIRST A (1) THEN A (0) THROUGH THE ADDRESS BITS.

6.9.2 TEST 9, SUBTEST 2: -

THIS SUBTEST VERIFIES THAT THE CONTROLLER CAN DEPOSIT MESSAGE PACKETS TO ALL AVAILABLE MEMORY LOCATIONS. FIRST ALL AVAILABLE MEMORY IS SET TO A BACKGROUND PATTERN OF 125252. WRITE CHARACTERISTICS COMMANDS ARE THEN EXECUTED WITH MESSAGE BUFFER ADDRESSES SET TO VARIOUS ADDRESSES IN EACH AVAILABLE MEMORY LOCATION. THE VARIOUS ADDRESSES ARE DETERMINED BY FLOATING FIRST A (1) THEN A (0) THROUGH THE ADDRESS BITS.

6.9.3 TEST 9, SUBTEST 3: -

THIS SUBTEST VERIFIES THAT A CONTROLLER CAN FETCH A WRITE CHARACTERISTICS DATA BLOCK FROM ALL AVAILABLE MEMORY LOCATIONS. FIRST ALL AVAILABLE MEMORY IS SET TO A BACKGROUND PATTERN OF 125252. THE WRITE CHARACTERISTICS COMMANDS ARE EXECUTED WITH CHARACTERISTIC DATA BLOCKS AT VARIOUS MEMORY ADDRESSES. THE VARIOUS MEMORY ADDRESSES ARE DETERMINED BY FLOATING FIRST A (1) THEN A (0) THROUGH THE ADDRESS BITS.

6.9.4 TEST 9, SUBTEST 4: -

THIS SUBTEST VERIFIES THAT THE NXM ERROR BIT IN THE TSSR REGISTER IS SET WHEN ATTEMPTING TO FETCH DATA (A CHARACTERISTIC DATA BLOCK) FROM SELECTED NONEXISTANT LOCATIONS. IF NXM FAILS TO SET IT IS LIKELY THAT

AN LSI-11 BUS DRIVER IS FAILING TO ASSERT AN ADDRESS LINE. ADDRESSES TESTED INCLUDE ALL COMBINATIONS OF HIGH ORDER ADDRESS BITS (IE: BITS 16-21).

6.10 TEST 10 - INITIALIZE AFTER WRITE CHARACTERISTICS

* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *

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

6.11 TEST 11 - BASIC WRITE SUBSYSTEM MEMORY COMMAND

* NOTE: IF THIS TEST DETECTS AN ERROR REPLACE THE TK25'S *
* CONTROLLER *

THIS TEST VERIFIES THAT THE WRITE SUBSYSTEM MEMORY COMMAND WITH A BSELO
SELECT CODE OF 0 (NO-OP) EXECUTES CORRECTLY. THE TEST FURTHER VERIFIES
MICROPROGRAM COMMAND DECODING AND HANDLING SEQUENCES.

```

743
744 .SBTTL PROGRAM HEADER
750 .MCALL SVC
751 000000 SVC ; INITIALIZE SUPERVISOR MACROS
752 .ENABLE LC
753 .NLIST BEX,CND
759 000000 .ENABL AMA,ABS
760 002000 002000 . = 2000
761 002000 BGNMOD TUV2A
002000 TUV2A::
762
763 ;**
764 ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
765 ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
766 ;--
767
768
769 002000 POINTER BGNSW,BGNSFT,BGNAU,BGNDU,BGNRPT,BGNSETUP
770 002000 HEADER CZTKE,A,0,655.,0
002000 L$NAME:: ;DIAGNOSTIC NAME
002000 103 .ASCII /C/
002001 132 .ASCII /Z/
002002 124 .ASCII /T/
002003 113 .ASCII /K/
002004 105 .ASCII /E/
002005 000 .BYTE 0
002006 000 .BYTE 0
002007 000 .BYTE 0
002010 L$REV:: ;REVISION LEVEL
002010 101 .ASCII /A/
002011 L$DEPO:: ;0
002011 060 .ASCII /0/
002012 L$UNIT:: ;NUMBER OF UNITS
002012 000001 .WORD T$PTHV
002014 L$TIML:: ;LONGEST TEST TIME
002014 001217 .WORD 655.
002016 L$HPCP:: ;POINTER TO H.W. QUES.
002016 046052 .WORD L$HARD
002020 L$SPCP:: ;POINTER TO S.W. QUES.
002020 046212 .WORD L$SOFT
002022 L$HPTP:: ;PTR. TO DEF. H.W. PTABLE
002022 002124 .WORD L$HW
002024 L$SPTP:: ;PTR. TO S.W. PTABLE
002024 002134 .WORD L$SW
002026 L$LADP:: ;DIAG. END ADDRESS
002026 046436 .WORD L$LAST
002030 L$STA:: ;RESERVED FOR APT STATS
002030 000000 .WORD 0
002032 L$CO::
002032 000000 .WORD 0
002034 L$DTYP:: ;DIAGNOSTIC TYPE
002034 000000 .WORD 0
002036 L$APT:: ;APT EXPANSION
002036 000000 .WORD 0
002040 L$DTP:: ;PTR. TO DISPATCH TABLE
002040 046404 .WORD L$DISPATCH
  
```


002042		L\$PRIO::		;DIAGNOSTIC RUN PRIORITY
002042	000000	.WORD	0	
002044		L\$ENVI::		;FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000	.WORD	0	
002046		L\$EXP1::		;EXPANSION WORD
002046	000000	.WORD	0	
002050		L\$MREV::		;SVC REV AND EDIT #
002050	003	.BYTE	C\$REVISION	
002051	003	.BYTE	C\$EDIT	
002052		L\$EF::		;DIAG. EVENT FLAGS
002052	000000	.WORD	0	
002054	000000	.WORD	0	
002056		L\$SPC::		
002056	000000	.WORD	0	
002060		L\$DEVP::		; POINTER TO DEVICE TYPE LIST
002060	003334	.WORD	L\$DVTYP	
002062		L\$REPP::		;PTR. TO REPORT CODE
002062	022674	.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	022366	.WORD	L\$AU	
002072		L\$DUT::		;PTR. TO DROP UNIT CODE
002072	022464	.WORD	L\$DU	
002074		L\$LUN::		;LUN FOR EXERCISERS TO FILL
002074	000000	.WORD	0	
002076		L\$DESP::		;POINTER TO DIAG. DESCRIPTION
002076	003342	.WORD	L\$DESC	
002100		L\$LOAD::		;GENERATE SPECIAL AUTOLOAD EMT
002100	104035	EMT	E\$LOAD	
002102		L\$ETP::		;POINTER TO ERR_TBL
002102	000000	.WORD	0	
002104		L\$ICP::		;PTR. TO INIT CODE
002104	021606	.WORD	L\$INIT	
002106		L\$CCP::		;PTR. TO CLEAN-UP CODE
002106	022646	.WORD	L\$CLEAN	
002110		L\$ACP::		;PTR. TO AUTO CODE
002110	022572	.WORD	L\$AUTO	
002112		L\$PRT::		;PTR. TO PROTECT TABLE
002112	021576	.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	

```

772          .SBTTL  DEFAULT HARDWARE P-TABLE
773          ;**
774          ; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
775          ; THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
776          ; IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
777          ;--
778          BGNHW   DFPTBL      ;DEFAULT HARD-P-TABLE
          .WORD    L10000-L$HW/2
779          L$HW::
          DFPTBL::
          .WORD    172522      ; 2ND (OF 2) REGISTERS.
          .WORD    224        ; INTERRUPT VECTOR
          .WORD    PRI05      ; INTERRUPT PRIORITY.
          ENDMW
          L10000:
002122      000003
002124
002124
002124
002124
002124      172522
002126      000224
002130      000240
002132
002132

```



```

785          .SBTTL  SOFTWARE P-TABLE
786
787          ;**
788          ; THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
789          ; PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
790          ;--
791          BGNSW  SFPTBL
          002132  000004  .WORD  L10001-L#SW/2
          002134
          002134
792
793          002134  000000  TRANSTST::  .WORD  0          ;ENABLE RAM DUMP IF =1
794          002136  000000  NOITS::    .WORD  0          ; INHIBIT ITERATION OPTION.
795
796
797          002140  000031  LERRMAX:: .WORD  25.     ; ... 0 = ITERATE.
798          002142  000310  GERRMAX:: .WORD  200.    ; ...NZ = INHIBIT ITERATE.
799          002144          ENDSW  ; LOCAL (PER TEST) ERROR LIMIT
          002144          ; GLOBAL (PER UNIT) ERROR LIMIT
800          L10001:

```

802
809
814
820
821
822
823
824
825
826
827
828
829
833 002144

.SBTTL GLOBAL EQUATES SECTION

.SBTTL GLOBAL EQUATES SECTION

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

EQUALS ; GET STANDARD EQUATES.

:
: BIT DIFINITIONS

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

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

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

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

:
: PRIORITY LEVEL DEFINITIONS
:

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

; OPERATOR FLAG BITS

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

834
835 002144

KT11 .. ; DEFINE MEMORY MANAGEMENT REGISTERS

.SBTTL MEMORY MANAGEMENT DEFINITIONS

000250	; *KT11 VECTOR ADDRESS
	MMVEC= 250
	; *KT11 STATUS REGISTER ADDRESSES
177572	SR0= 177572
177574	SR1= 177574
177576	SR2= 177576
172516	SR3= 172516

.IF NB ; *USER "I" PAGE DESCRIPTOR REGISTERS

UIPDR0=	177600
UIPDR1=	177602
UIPDR2=	177604
UIPDR3=	177606
UIPDR4=	177610
UIPDR5=	177612
UIPDR6=	177614
UIPDR7=	177616

.IF NB ; *USER "D" PAGE DESCRIPTOR REGISTERS

UDPDR0=	177620
UDPDR1=	177622
UDPDR2=	177624
UDPDR3=	177626
UDPDR4=	177630
UDPDR5=	177632
UDPDR6=	177634
UDPDR7=	177636

.ENDC ; *USER "I" PAGE ADDRESS REGISTERS

```
UIPAR0= 177640
UIPAR1= 177642
UIPAR2= 177644
UIPAR3= 177646
UIPAR4= 177650
UIPAR5= 177652
UIPAR6= 177654
UIPAR7= 177656
  .IF NB
  ;*USER "D" PAGE ADDRESS REGISTERS
  UDPAR0= 177660
  UDPAR1= 177662
  UDPAR2= 177664
  UDPAR3= 177666
  UDPAR4= 177670
  UDPAR5= 177672
  UDPAR6= 177674
  UDPAR7= 177676
  .ENDC
  .ENDC
  .IF NB
  ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
  SIPDR0= 172200
  SIPDR1= 172202
  SIPDR2= 172204
  SIPDR3= 172206
  SIPDR4= 172210
  SIPDR5= 172212
  SIPDR6= 172214
  SIPDR7= 172216
  .IF NB
  ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
  SDPDR0= 172220
  SDPDR1= 172222
  SDPDR2= 172224
  SDPDR3= 172226
  SDPDR4= 172230
  SDPDR5= 172232
  SDPDR6= 172234
  SDPDR7= 172236
  .ENDC
  ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS
  SIPAR0= 172240
  SIPAR1= 172242
  SIPAR2= 172244
  SIPAR3= 172246
  SIPAR4= 172250
  SIPAR5= 172252
  SIPAR6= 172254
  SIPAR7= 172256
  .IF NB
  ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS
  SDPAR0= 172260
  SDPAR1= 172262
  SDPAR2= 172264
  SDPAR3= 172266
  SDPAR4= 172270
```



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

```

840          .SBTTL TK-25 REGISTER AND PACKET DEFINITIONS
841
842          ;
843          ; SOME GENERAL EQUATES.
844          ;
845
846          000004      ERRVEC==      4          ; POINTER TO ERROR VECTOR FOR BUS TIME OUT.
847          000060      TTIVEC==     60          ; INTERRUPT VECTOR FOR CONSOLE INPUT
848          177560      TTICSR==    177560       ; BUS ADDRESS OF CONSOLE INPUT
849          177562      TTIBFR==    177562       ; CONSOLE INPUT DATA BUFFER
850
851          ;+
852          ;BIT DEFINITIONS FOR TSSR REGISTER
853          ;-
854
855          100000      SC=      BIT15          ;SPECIAL CONDITION
856          040000      BIE=     BIT14          ;BUS INTERFACE ERROR
857          020000      SCE=     BIT13          ;SANITY CHECK ERROR
858          010000      RMR=     BIT12          ;MODIFICATION REFUSED
859          004000      NXM=     BIT11          ;NONEXISTANT MEMORY ERROR
860          002000      NBA=     BIT10          ;NEED BUFFER ADDRESS
861          001400      HIADDR=  BIT9!BIT8      ;EXTENDED ADDRESS BITS
862          000200      SSR=     BIT7           ;SUB SYSTEM READY
863          000100      OFL=     BIT6           ;OFF LINE BIT
864          000060      FATERR=  BIT4!BIT5      ;FATAL TERMINATION ERROR CODES
865          000016      TERCLS=  BIT3!BIT2!BIT1 ;TERMINATION CODES
866
867
868          ;+
869          ;
870          ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 0
871          ;(XST0)
872          ;
873          ;-
874
875          100000      XSOTMK=  BIT15          ;TAPE MARK DETECTED
876          040000      XSORLS=  BIT14          ;RECORD LENGTH SHORT
877          020000      XSOLET=  BIT13          ;LOGICAL END OF TAPE
878          010000      XSORLL=  BIT12          ;RECORD LENGTH LONG
879          004000      XSOWLE=  BIT11          ;WRITE LOCK ERROR
880          002000      XSONEF=  BIT10          ;NON EXECUTABLE FUNCTION
881          001000      XSOILC=  BIT9           ;ILLEGAL COMMAND
882          000400      XSOILA=  BIT8           ;ILLEGAL ADDRESS
883          000200      XSOMOT=  BIT7           ;TAPE IN MOTION
884          000100      XSOONL=  BIT6           ;TRANSPORT ON LINE
885          000040      XSOIE=   BIT5           ;INTERRUPT ENABLE
886          000020      XSOVCK=  BIT4           ;VOLUME CHECK BIT
887          000010      XSOPED=  BIT3           ;PHASE ENCODED DRIVE
888          000004      XSOWLK=  BIT2           ;WRITE LOCKED
889          000002      XSOBOT=  BIT1           ;BEGINNING OF TAPE
890          000001      XSOEOT=  BIT0           ;END OF TAPE
891
892
893          ;+
894          ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 1
895          ;(XST1)
896          ;-

```



```

897      100000      X1.DLT = BIT15      ;DATA LATE
898      040000      X1.SPARE= BIT14      ;NOT USED
899      020000      X1.COR = BIT13      ;CORRECTABLE DATA ERROR
900      017375      X1.MBZ = BIT12+BIT11+BIT10+BIT9+BIT8+BIT7+BIT6+BIT5+BIT4+BIT3+BIT2+BIT0 ;ALWAYS 0
901      000400      X1.RBP = BIT8      ;READ BUS PARITY ERROR
902      000002      X1.UNC = BIT1      ;UNCORRECTABLE DATA OR HARD ERROR
903
904      ;+
905      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 2
906      ;(XST2)
907      ;-
908      100000      X2.OPM = BIT15      ;OPERATION IN PROGRESS (TAPE MOVING)
909      040000      X2.RCE = BIT14      ;RAM CHECKSUM ERROR
910      035400      X2.SPARE= BIT13+BIT12+BIT11+BIT9+BIT8 ;NOT USED BY TK-25 (ALWAYS=0)
911      002000      X2.WCF = BIT10      ;WRITE CLOCK FAILURE (FIFO NOT EMPTIED BY TRANSPORT)
912      000200      X2.EXTF = BIT7      ;IF WRITE CHAR CMD THEN = EXTENDED FEATURES ENABLED
913      000100      X2.BUFE = BIT6      ;IF WRITE CHAR CMD THEN = BUFFERING ENABLED
914      000077      X2.REV = 000077      ;IF WRITE CHAR CMD THEN = MICROCODE REVISION LEVEL
915      000007      X2.UNIT = BIT2+BIT1+BIT0 ;IF GET STATUS THEN = CURRENTLY SELECTED UNIT NO.
916
917      ;+
918      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 3
919      ;(XST3)
920      ;-
921      177400      X3.MDE = 177400      ;MICRO-DIAGNOSTIC ERROR CODE
922      000200      X3.SPARE= BIT7      ;NOT USED BY TK-25
923      000100      X3.OPI = BIT6      ;OPERATION INCOMPLETE
924      000040      X3.REV = BIT5      ;REVERSE
925      000020      X3.TRF = BIT4      ;TRANSPORT RESPONSE FAILURE
926      000010      X3.DCK = BIT3      ;DENSITY CHECK
927      000006      X3.MBZ =BIT2+BIT1      ;NOT USED ALWAYS 0
928      000001      X3.RIB = BIT0      ;REVERSE INTO BOT
929
930      ;+
931      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 4
932      ;(XST4)
933      ;-
934      100000      X4.HSP = BIT15      ;HIGH SPEED
935      040000      X4.RCE = BIT14      ;RETRY COUNT EXCEEDED
936      020000      X4.TSM = BIT13      ;TRANSPORT SPECIAL MODE
937      017400      X4.MBZ = BIT12+BIT11+BIT10+BIT9+BIT8 ;NOT USED ALWAYS 0
938      000377      X4.WRC = 000377      ;WRITE RETRY COUNT FIELD
939
940
941      ;+
942      ;
943      ;TSSR TERMINATION CODES (BIT 0-2)
944      ;
945      ;-
946
947      000006      TSREJ= 3+2      ;COMMAND REJECTED
948      000006      UNREC= 6      ;UNRECOVERABLE ERROR
949
950      ;+
951      ;
952      ;DEVICE REGISTER OFFSETS
953      ;

```

```

954          ; -
955
956          177776      TSBA== -2
957          177776      TSBAL== -2
958          177776      TSDB== -2          ;TSDB/TSBA REGISTER
959          177776      TSDBL== -2        ;TSDB/TSBA REGISTER
960          177777      TSBAH== -1
961          177777      TSDBH== -1        ;TSDB/TSBA REGISTER HIGH BYTE
962          000000      TSSR== 0          ;TSSR REGISTER
963          000001      TSSRH== 1         ;TSSR REGISTER HIGH BYTE
964
965          ;+
966          ; TSDB ADDRESS BIT DEFINITIONS
967          ; -
968          000003      A1716 = BIT1+BIT0   ;ADDRESS BITS 17;16 ARE IN 1:0
969
970          ;+
971          ; COMMAND DEFINITIONS
972          ; -
973          000017      P.GETSTAT = 17      ;GET STATUS
974          000013      P.INIT = 13         ;INITIALIZE
975          000012      P.CONTROL = 12      ;CONTROL COMMANDS
976          000011      P.FORMAT = 11       ;FORMAT
977          000010      P.POSITION = 10     ;POSITION
978          000006      P.WRTSUB = 6        ;SUBSYSTEM WRITE
979          000005      P.WRITE = 5         ;WRITE
980          000004      P.WRTCHAR = 4       ;WRITE CHARACTERISTICS
981          000001      P.READ = 1         ;READ
982
983          ;+
984          ; COMMAND PACKET HEADER WORD BIT DEFINITIONS
985          ; -
986          100000      P.ACK = BIT15       ;BUFFER AVAIL FOR CONTROLLER
987          040000      P.CVC = BIT14      ;CLEAR VOLUME CHECK
988          020000      P.OPP = BIT13      ;REVERSE SEQUENCE OF DATA BITS
989          010000      P.SWB = BIT12      ;SWAP BYTES IN MEMORY
990          007400      P.MODE = BIT11!BIT10!BIT9!BIT8 ;EXTENDED COMMAND MODE FIELD
991          000200      P.IE = BIT7        ;INTERRUPT ENABLE
992          000140      P.FMT = BIT6!BIT5   ;PACKET HEADER TYPE (ALWAYS=0)
993          000037      P.CMD = 37         ;MAJOR COMMAND FIELD
994
995          ;+
996          ; CONTROL COMMAND MODE CODES
997          ; -
997          000000      PC.RELEASE = 0*256. ;RELEASE BUFFER
998          000400      PC.REWIND = 1*256. ;REWIND
999          001000      PC.NOOP = 2*256.   ;NO-OP
1000         002000      PC.IEREW = 4*256.  ;REWIND IMMEDIATE INTERRUPT
1001         002400      PC.ERASE = 5*256.  ;SECURITY ERASE
1002
1003          ;+
1004          ; CONTROLLER RAM DEFINITIONS
1005          ; -
1006          000167      RMCHBEG = 167       ;CHARACTERISTICS IO DATA BEGIN RAM ADDRESS
1007          000200      RMCHEND = 200      ;CHARACTERISTICS IO DATA END RAM ADDRESS
1008          000020      RMPKTBEG = 20       ;COMMAND PACKET BEGIN RAM ADDRESS
1009          000027      RMPKTEND = 27      ;COMMAND PACKET END RAM ADDRESS
1010          000104      RMMMSGBEG = 104     ;MESSAGE BUFFER BEGIN RAM ADDRESS

```



```

1011      000117      RMMSGEND= 117      ;MESSAGE BUFFER END RAM ADDRESS
1012      ;*
1013      ;
1014      ;REGISTER DEFINITIONS IN THE MESSAGE BUFFER
1015      ;
1016      ;-
1017
1018      000006      XST0== 6      ;EXTENDED STATUS REGISTER 0 (WORD 4)
1019      000010      XST1== 8.      ;EXTENDED STATUS REGISTER 1 (WORD 5)
1020      000012      XST2== 10.     ;EXTENDED STATUS REGISTER 2 (WORD 6)
1021      000014      XST3== 12.     ;EXTENDED STATUS REGISTER 3 (WORD 7)
1022      000016      XST4== 14.     ;EXTENDED STATUS REGISTER 4 (WORD 8)
1023
1024
1025      ;*
1026      ;
1027      ;OFFSETS TO WORD LOCATIONS IN PACKET DEFINITIONS
1028      ;
1029      ;-
1030
1031      000002      PKLOW  = 2      ;LOW ORDER CHARACTERISTIC DATA POINTER
1032      000004      PKHI   = 4      ;HIGH ORDER CHARACTERISTIC DATA POINTER
1033      000006      PKBCNT = 6      ;NUMBER OF BYTES IN DATA PACKET
1034
1035      000010      EXBCNT=10      ;NUMBER OF BYTES IN EXTENDED DATA PACKET
1036
1037      ;*
1038      ;DATA PACKET OFFSETS FOR WRITE SUBSYSTEM COMMAND
1039      ;-
1040      000000      BSELO  = 0      ;BYTE 0
1041      000001      BSEL1  = 1      ;BYTE 1
1042      000002      SEL2   = 2      ;WORD 2
1043      000004      SELDATA = 4      ;WORD 3
1044
1045      ;*
1046      ;BSELO SELECT CODES FOR WRITE SUBSYSTEM COMMAND
1047      ;-
1048      000000      PW.NOP   = 0      ;NO-OP
1049      000001      PW.RDRAM = 1      ;READ RAM
1050      000002      PW.WTRAM = 2      ;WRITE RAM
1051      000003      PW.RFIFO = 3      ;READ FIFO
1052      000004      PW.WFIFO = 4      ;WRITE FIFO
1053      000005      PW.RDSTAT = 5     ;READ STATUS
1054      000006      PW.WCTL  = 6      ;WRITE TAPE CONTROL
1055      000007      PW.WFMT  = 7      ;WRITE TAPE FORMAT
1056      000010      PW.WMISC = 10     ;WRITE MISCELLANEOUS
1057      000011      PW.WNPR  = 11     ;WRITE NPR CONTROL
1058      000020      PW.D22   = 20     ;DO MICROTEST 22
1059      000021      PW.D11   = 21     ;DO MICROTEST 11
1060      000022      PW.D13   = 22     ;DO MICROTEST 13
1061      000023      PW.NO1311 = 23    ;DISABLE MICROTEST 11 AND 13
1062      000024      PW.RDXT  = 24     ;READ EXT. TAPE STATUS (NOT SUPPORTED BY ALL TRANSP
RTS
1063
1064      ;*
1065      ;BSEL1 CODES FOR WRITE TAPE CONTROL
1066      ;-
1067      000200      WC.IFAD  = BIT7    ;IFAD - FORMATTER ADDRESS

```

```

1068      000100      WC.IOTAD      = BIT6      ;ITADO - TRANSPORT ADDRESS BIT 0
1069      000040      WC.I1TAD      = BIT5      ;ITAD1 - TRANSPORT ADDRESS BIT 1
1070      000020      WC.ISRESV     = BIT4      ;IRESV5 - RESERVED #5
1071      000010      WC.IREW       = BIT3      ;IREW - REWIND
1072      000004      WC.IRWU        = BIT2      ;IRWU - REWIND AND UNLOAD
1073      000002      WC.IFEN        = BIT1      ;IFEN - FORMATTER ENABLE
1074      000001      WC.IGO         = BIT0      ;GO
1075
1076      ;+
1077      ;BSEL1 CODES FOR WRITE FORMAT
1078      ;-
1079      000200      WF.IHISP       = BIT7      ;IHISP - HIGH SPEED
1080      000100      WF.IWRT        = BIT6      ;IWRT - WRITE
1081      000040      WF.IREV        = BIT5      ;IREV - REVERSE
1082      000020      WF.IWFM        = BIT4      ;IWFM - WRITE FILE MARK
1083      000010      WF.IEDIT       = BIT3      ;IEDIT - EDIT
1084      000004      WF.IERASE      = BIT2      ;IERASE - ERASE
1085      000002      WF.I3RESV     = BIT1      ;IRESV3 - RESERVED #3
1086      000001      WF.I4RESV     = BIT0      ;IRESV4 - RESERVED #4
1087
1088
1089      ;+
1090      ;BSEL1 CODES FOR WRITE MISCELLANEOUS SUBCOMMAND
1091      ;-
1092      000200      MS.EXT          = BIT7      ;INVERT SENSE OF EXTENDED FEATURES SWITCH
1093      000020      MS.RSFIFO      = BIT4      ;RESET FIFO AND INPUT PARITY ERRORR
1094      000010      MS.RSTAPE      = BIT3      ;RESET TAPE STATUS IN 2 FLIP-FLOPS
1095      000006      MS.ATTN        = BIT2!BIT1 ;ATTENTION TRIGGER FIELD
1096      000001      MS.RSD         = BIT0      ;RESET TIMER A,B THEN DELAY TIMES IN SEL2
1097
1098      ;+
1099      ; MS.ATTN SUBCODES
1100      ;-
1100      000000      MSA.NOP        = 0*2      ;NO-OP (NOTHING TRIGGERED)
1101      000002      MSA.VOL        = 1*2      ;SIMULATE ON-LINE/OFF-LINE TRANSISTION
1102      000004      MSA.NRAM       = 2*2      ;FORCE NON-FATAL RAM ERROR (FORCES ERRCODE 54)
1103      000006      MSA.FRAME      = 3*2      ;FORCE FATAL RAM ERROR (CAUSES SCE TO SET)
1104
1105      ;+
1106      ; WRITE SUBSYSTEM WRITE NPR BSEL1 BIT DEFINITIONS
1107      ;-
1107      000200      NP.IR           = BIT7      ;INTERRUPT REQUEST (0-1 TRANSITION)
1108      000100      NP.OUT          = BIT6      ;TAPE DATA DIRECTION OUT (0= IN)
1109      000040      NP.LOOP        = BIT5      ;ENABLE TRANSPORT LOOPBACK
1110      000020      NP.WRP         = BIT4      ;WRITE CORRECT PARITY (SET=0 TO WRITE WRONG)
1111
1112      ;+
1113      ; READ STATUS MESSAGE BUFFER BIT DEFINITIONS
1114      ;-
1115      000200      S2.DIM           = BIT7      ;WORD #9 BYTE 2 DATA IN MISS
1116      000100      S2.ILW         = BIT6      ;
1117      000040      S2.OUTRDY       = BIT5      ;
1118      000020      S2.INRDY        = BIT4      ;
1119      000010      S2.ATIMR        = BIT3      ;
1120      000004      S2.BTIMR        = BIT2      ;
1121      000003      S2.UNDEF        = BIT1+BIT0 ;(UNDEFINED)
1122      100000      S1.PARIN        = BIT15     ;WORD #8 BYTE 1 PARIN H
1123      040000      S1.I2RESV     = BIT14     ;
1124      020000      S1.I1RESV     = BIT13     ;

```


1125	010000	S1.IEOT	= BIT12	:	IEOT L
1126	004000	S1.IIDENT	= BIT11	:	IIDENT H
1127	002000	S1.ICER	= BIT10	:	ICER H
1128	001000	S1.IFMK	= BIT9	:	IFMK H
1129	000400	S1.IHER	= BIT8	:	IHER H
1130	000200	SO.ISPEED	= BIT7	;	WORD #8 BYTE 0 ISPEED H
1131	000100	SO.IRDY	= BIT6	:	IRDY L
1132	000040	SO.IONL	= BIT5	:	IONL L
1133	000020	SO.ILDP	= BIT4	:	ILDP L
1134	000010	SO.IDBY	= BIT3	:	IDBY L
1135	000004	SO.IRWD	= BIT2	:	IRWD L
1136	000002	SO.IFBY	= BIT1	:	IFBY L
1137	000001	SO.IFPT	= BIT0	:	IFPT L
1138		:		:	
1139		:		:	
1140	177560	TKS	=177560	;	KEYBOARD STATUS REGISTER
1141	177562	TKB	=177562	;	KEYBOARD DATA REGISTER
1142	177564	TPS	=177564	;	CONSOLE PRINTER STATUS REGISTER
1143	177566	TPB	=177566	;	CONSOLE PRINTER DATA REGISTER
1144	007776	HIMEM	=007776	;	HIGH MEMORY MASK VALUE
1145		:		:	
1146		:		:	
1147		:		:	
1148	174400	CSR	=174400	;	STATUS AND CONTROL REGISTER
1149	174402	BAR	=174402	;	DL ADDRESS REGISTER
1150	174404	DAR	=174404	;	PLATTER ADDRESS
1151	174406	MPR	=174406	;	MULTIPURPOSE REGISTER
1152		:		:	
1153		:		:	
1154		:		:	
1155		:		:	
1156		:		:	
1157		:		:	
1158		:		:	
1159		:		:	
1160	000004	DLGETS	=4	;	GET STATUS COMMAND
1161	000006	SEEK	=6	;	SEEK TRACK AND HEAD SELECT
1162	000010	DLRDHD	=10	;	READ SECTOR HEADER
1163	000014	READ	=14	;	READ COMMAND
1164	000016	DLRDNH	=16	;	READ SECTOR NO HEADER CHECK
1165		:		:	
1166		:		:	
1167		:		:	
1168		:		:	
1169		:		:	
1170		:		:	
1171	000001	READY	=1	;	DRIVE READY BIT IN STATUS REG.
1172	000013	DLSR	=13	;	STATUS AND RESET
1173	177730	DLERR	=177730	;	MASK FOR COVER OPEN
1174	000006	DLUN	=6	;	HEADS UNLOADED
1175	000177	DLCYL	=000177	;	MASK FOR CYLINDER ADDRESS
1176	100200	DLDNER	=100200	;	DONE SET OR ERROR SET BITS
1177		:		:	
1178		:		:	
1179		:		:	
1180		:		:	
1181	177560	TTICSR	= 177560	;	KEYBOARD INPUT STATUS

1182	177562	TTIBFR =	177562	;KEYBOARD DATA REGISTER
1183	177564	TTOCSR =	177564	;CONSOLE PRINTER STATUS REGISTER
1184	177566	TTOBFR =	177566	;CONSOLE PRINTER DATA REGISTER
1185				


```

1187           .SBTTL  SPECIAL MACROS AND OPDEFS.
1188
1189
1190           ;+
1191           ;SAVE GENERAL REGS 1 TO 5
1192           ;-
1193
1194           .MACRO  SAVREG
1195           JSR    R5,REGSAV
1196           .ENDM
1197
1198           ;+
1199           ; MACRO TO FORCE AN ERROR
1200           ;-
1201           .MACRO  FORCERROR      TAG,NOTSSR
1202           .NLIST
1203           .IF NDF LISTALL, .NLIST
1204           .LIST
1205           .IF B NOTSSR
1206           MOV    TSSR(R5),R1           ;READ TSSR
1207           .ENDC
1208           MOV    FORCER,FORCER       ;IS FORCER SET? (LEAVE C BIT ALONE)
1209           BNE   TAG                 ;BR IF YES
1210           .NLIST
1211           .IF NDF LISTALL, .LIST
1212           .LIST
1213           .ENDM
1214
1215           ;+
1216           ; MACRO TO FORCE AN EXIT TO AVOID SECTION ITERATIONS
1217           ; WILL EXIT TO A LABEL IF FORCER IS NEGATIVE
1218           ; SO TO FORCE ERRORS AND EXIT ON 1 ERROR SET
1219           ; FORCER TO 177777
1220           ; TO FORCE ERRORS AND ITERATIONS SET FORCER TO 1.
1221           ;-
1222           .MACRO  FORCEEXIT      TAG
1223           .NLIST
1224           .IF NDF LISTALL, .NLIST
1225           .LIST
1226           MOV    FORCER,FORCER       ;IS FORCER NEGATIVE?
1227           BMI   TAG                 ;BR IF YES
1228           .NLIST
1229           .IF NDF LISTALL, .LIST
1230           .LIST
1231           .ENDM
1232           ;+
1233           ; MACRO TO INCREMENT ERROR COUNTS
1234           ;-
1235           .MACRO  NEXT.ERRNO
1236           .NLIST
1237           ;;;.IF NDF LISTALL, .NLIST
1238           ERRNO=ERRNO+1
1239           ;;;.IF NDF LISTALL, .LIST
1240           .LIST
1241           .ENDM
1242
1243           ;+

```

1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267

MACRO TO PERFORM XOR

;-

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

000000

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

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

002:44 000000

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


.SBTTL GLOBAL DATA SECTION

```

1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280 002146 000000
1281 002150 000000
1282 002152 000000
1283 002154 000000
1284 002156 000224
1285 002160 000200
1286 002162 000000
1287 002164 000000
1288 002166 000000
1289 002170 000000
1290 002172 000000
1291 002174 000000
1292 002176 000000
1293 002200 000000
1294 002202 000000
1295 002204 000000
1296 002206
1297 002246 000000
1298 002250 000000
1299 002252 000000
1300 002254 000000
1301 002256 000000
1302 002260 000000
1303 002262 000000
1304 002264 000000
1305 002266
1306 002432
1307 002576
1308 002716 000000

```

```

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

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

```

1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326 002720
1327 002720 000000
1328 002722 177777
1329 002724 000001
1330 002726 000002
1331 002730 000004
1332 002732 000010
1333 002734 000020
1334 002736 000040
1335 002740 000100
1336 002742 000200
1337 002744 000400
1338 002746 001000
1339 002750 002000
1340 002752 004000
1341 002754 010000
1342 002756 020000
1343 002760 040000
1344 002762 100000
1345 002764 177776
1346 002766 177775
1347 002770 177773
1348 002772 177767
1349 002774 177757
1350 002776 177737
1351 003000 177677
1352 003002 177577
1353 003004 177377
1354 003006 176777
1355 003010 175777
1356 003012 173777
1357 003014 167777
1358 003016 157777
1359 003020 137777
1360 003022 077777
1361 003024 125252
1362 003026 052525
1363 003030

.SBTTL TSTBLK - TEST DATA TABLE

```

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

```

```

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

TBLEND**

```



```

1365                .SBTTL GLOBAL ENVIRONMENT STORAGE
1366                ;
1367                ; STORAGE FOR DEVICE REGISTERS
1368                ;
1369 003030 000000 100000 000000 DUMMY: 0,100000,0,0          ; DUMMY DEVICE REGISTERS...
1370 003040 000000 000000 000000      0,0,0,0,0,0,0,0,0    ; ...FOR MULTI-UNIT CHECKOUT.
1371
1372
1373
1374 003060 000000          DUFLG::      .WORD 0          ; "DROPPED UNIT" FLAG.
1375                                     ; INHIBITS CODE IN "CLEAN-UP".
1376 003062 000000          NODEV::      .WORD 0          ; FLAG TO SAY NO DEVICE.
1377
1378 003064 000000          TEMP1::       .WORD 0          ; SOME TEMP LOCATIONS.
1379 003066 000000          TEMP2::       .WORD 0
1380 003070 000000          XXCOMM::      .WORD 0          ; XXDP, COMM BLOCK POINTER.
1381 003072 000000          FREE::        .WORD 0          ; 1ST FREE MEMORY ADDRESS...
1382 003074 000000          FRESIZ::      .WORD 0          ; ...AND SIZE (IN WORDS).
1383 003076 000000          FREEHI::      .WORD 0          ; LAST WORD IN FREE SPACE
1384 003100 000000          KTFLG::       .WORD 0          ; KT11, MEM AVAIL FLAG -
1385                                     ; - .WORD 0 = <24K OR NO KT -
1386                                     ; - NZ = >24K AND KT.
1387 003102 000000          KTENABLE::     .WORD 0          ; SET BY TEST ROUTINES TO FLAG >28K UNDER TEST
1388 003104 002000          PST32W::      .WORD 2000       ; 32W BLOCK ADDRESS FOR 32K START
1389 003106 000000          SIFLAG::      .WORD 0
1390 003110 000000          BADDAT::      .WORD 0          ; ACTUAL DATA
1391 003112 000000          GDDAT::      .WORD 0          ; EXPECTED DATA
1392 003114 000000          LOOPFL::      .WORD 0
1393 003116
1394 003116 000000          CTABM::       .WORD 0          ; CONFIGURATION TABLES.
1395 003120 000000          .WORD 0          ; CONFIG WORK.
1396 003122 000000          .WORD 0
1397 003124 000000          .WORD 0
1398 003126 177777          .WORD -1          ; END OF MEM TABLE.
1399 003130
1400          CTABE::
1401          ; ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:
1402          ;
1403          ; 0 = UNIT NOT TESTED
1404          ; 100000 = UNIT ONLINE, NO ERRORS
1405          ; 10XXXX = UNIT ONLINE, ENCOUNTERED XXXX ERRORS
1406          ; 160000 = UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
1407          ; 160001 = UNIT DROPPED, NOT IDLE AT START
1408          ; 14XXXX = UNIT DROPPED, ENCOUNTERED XXXX ERRORS
1409 003130          ERTABL:      .BLKW 64.
1410 003330 000000          ERTABE:      .WORD 0
1411
1412 003332 000000          SKIPT:      .WORD 0          ; 1=SKIP SUBTEST 0=NO SKIP OF SUBTEST

```

```

1414 .SBTTL . GLOBAL TEXT MESSAGES
1415 ;**
1416 ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
1417 ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
1418 ; MORE THAN ONE TEST.
1419 ;--
1420
1421
1422
1423 ;*
1424 ;NAMES OF DEVICES SUPPORTED
1425 ;-
1426
1427 003334          DEVTYP <TK-25>
003334          L$DVTYP::
003334          .ASCIZ /TK-25/
          124      113      055      .EVEN
1428
1429
1430
1431 ;*
1432 ;TEST DESCRIPTION
1433 ;-
003342          DESCRIPT <CZTKEA TK-25 FRT END FUNC #1>
003342          L$DESC::
003342          .ASCIZ /CZTKEA TK-25 FRT END FUNC #1/
          103      132      124      .EVEN
1434
1435
1436 ;*
1437 ;BIT TO ASCII CONVERSION FOR TSSR REGISTER
1438 ;-
1439 003400 003440 003443 003447 TSSRBIT:: .WORD 1$,2$,3$,4$,5$,6$,7$,8$
1440 003420 003501 003505 003511 .WORD 9$,10$,11$,12$,13$,14$,15$,16$
1441 003440          123      103      000      1$: .ASCIZ 'SC'
1442 003443          102      111      105      2$: .ASCIZ 'BIE'
1443 003447          123      103      105      3$: .ASCIZ 'SCE'
1444 003453          122      115      122      4$: .ASCIZ 'RMR'
1445 003457          116      130      115      5$: .ASCIZ 'NXM'
1446 003463          116      102      101      6$: .ASCIZ 'NBA'
1447 003467          102      111      124      7$: .ASCIZ 'BIT9'
1448 003474          102      111      124      8$: .ASCIZ 'BIT8'
1449 003501          123      123      122      9$: .ASCIZ 'SSR'
1450 003505          117      106      114     10$: .ASCIZ 'OFL'
1451 003511          102      111      124     11$: .ASCIZ 'BIT5'
1452 003516          102      111      124     12$: .ASCIZ 'BIT4'
1453 003523          102      111      124     13$: .ASCIZ 'BIT3'
1454 003530          102      111      124     14$: .ASCIZ 'BIT2'
1455 003535          102      111      124     15$: .ASCIZ 'BIT1'
1456 003542          102      111      124     16$: .ASCIZ 'BIT0'
1457          .EVEN
1458 003550          124      123      123     SFIERR: .ASCIZ 'TSSR ERROR AFTER SOFT INIT'
1459 003603          124      123      123     SFHERR: .ASCIZ 'TSSR ERROR AFTER BUS RESET'
1460 003636          040      040      116     NXR: .ASCIZ / NON-EXISTANT DEVICE REGISTER/
1461 003675          045      101      040     NXR: .ASCIZ /#A ADDRESS: #06/
1462 003716          045      101      040     TSSX: .ASCII /#A TSBA,TSSR EXP'D: #06#A,#06#N/
1463 003756          045      101      040     TSSX: .ASCII /#A TSBA,TSSR REC'D: #06#A,#06/
1464 004015          045      116      045     FUSI: .ASCII /#N#A/

```



```

1465 004021    040    040    125  USI:    .ASCIZ  / UNEXPECTED INTERRUPT/
1466 004050    040    040    111  NSI:    .ASCIZ  / INTERRUPT EXPECTED, NOT RECEIVED/
1467 004113    045    116    045  FNOINTR: .ASCIZ  / ASCII /#N#A/
1468 004117    040    040    116  NOINTR: .ASCIZ  / NO INTERRUPT WAS GENERATED/
1469 004154    040    040    111  IFAULT: .ASCIZ  / INTERRUPT FAULT/
1470 004176    045    101    040  INTX:   .ASCIZ  /#A CPU PC: #06#A TSBA: #06/
1471 004233    040    040    042  NOINIT: .ASCIZ  / "BUS-INIT" DIDN'T INITIALIZE CONTROLLER/
1472 004305    040    040    042  NSINIT: .ASCIZ  / "SOFT-INIT" DIDN'T INITIALIZE THE DPU/
1473 004355    040    040    042  BRINIT: .ASCIZ  / "BUS-RESET" DIDN'T INITIALIZE THE DPU/
1474
1475 004425    000
1476 004426    045    116    000  NULCR:  .ASCIZ  //
1477 004431    045    101    040  EXPGOT: .ASCIZ  /#A EXP'D: #06#A, REC'D: #06/
1478 004465    045    116    045  EXPGT2: .ASCIZ  /#N#A EXP'D: #06#A, #06#N#A REC'D: #0#A, #06/
1479 004541    045    101    040  DUAD12: .ASCIZ  /#A REG(W) WRITTEN TO: #06#A REG(R) READ; EXP'D: #06#A, REC'D: #06/
1480 004643    122    101    115  PKTRAM: .ASCIZ  'RAM Contents Do Not Match Packet Sent'
1481 004711    040    040    103  SCME:   .ASCIZ  / CONFIG DOESN'T MATCH MFG. MASTER/
1482 004754    127    122    111  WRMSG:  .ASCIZ  'WRITE CHARACTERISTICS Failed'
1483 005011    124    123    123  WRTERR: .ASCIZ  'TSSR Incorrect After WRITE Command, More Bits Set Than SSR'
1484 005104    124    123    123  RDERR:  .ASCIZ  'TSSR Incorrect After READ Command, More Bits Set Than SSR'
1485
1486
1487
1488

```

```

1490
1491
1492
1493
1494
1495
1496
1497
1498 005176
      005176
1499 005176
      005176 013746 003062
      005202 012746 003675
      005206 012746 000002
      005212 010600
      005214 104415
      005216 062706 000006
1500 005222 004737 005230
1501 005226
      005226
      005226 104423
1502
1503
1504
1505
1506
1507
1508 005230 005727
1509 005232 000000
1510 005234 001402
1511 005236 004777 177770
1512 005242
      005242 012746 004426
      005246 012746 000001
      005252 010600
      005254 104415
      005256 062706 000004
1513 005262 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.
          PRINTX #NULCR         ; PRINT A BLANK LINE
          MOV   #NULCR,-(SP)
          MOV   #1,-(SP)
          MOV   SP,R0
          TRAP  C#PNTX
          ADD   #4,SP
          RTS   PC
  
```



```

1516          .SBTTL  PRITSSR - PRINT TSSR CONTENTS
1517
1518          ;+
1519          ;
1520          ;ROUTINE TO DISPLAY THE CONTENTS, AND BIT DEFINITIONS, OF
1521          ;THE TSSR REGISTER. THIS ROUTINE IS NORMALLY CALLED ONLY
1522          ;BY A MESSAGE PRINTING ROUTINE
1523          ;
1524          ;INPUTS:
1525          ;
1526          ;       R1       CONTENTS OF TSSR
1527          ;
1528          ;SUBORDINATE ROUTINES:
1529          ;
1530          ;       CHKAMB  CHECK FOR AMBIGUOUS CONTENTS
1531          ;
1532          ;-
1533
1534 005264      PRITSSR:
1535 005264      SAVREG                               ;SAVE GENERAL REGISTERS
1536 005270 010104      MOV      R1,R4                ;SAVE THE TSSR CONTENTS
1537 005272      PRINTB  @TSSRFOR,R4                ;PRINT THE CONTENTS OF TSSR
1538          005272 010446      MOV      R4,-(SP)
1539          005274 012746 005736      MOV      @TSSRFOR,-(SP)
1540          005300 012746 000002      MOV      @2,-(SP)
1541          005304 010600      MOV      SP,R0
1542          005306 104414      TRAP     C:PNTB
1543          005310 062706 000006      ADD      @6,SP
1544          005314 010400      MOV      R4,R0                ;GET TSSR BACK FOR CHKAMB
1545          005316 004737 016540      JSR     PC,CHKAMB        ;ARE CONTENTS AMBIGUOUS ?
1546          005322 103410      BCS     5$                ;BRANCH IF NOT
1547          005324      PRINTX @AMBTSSR            ;SHOW CONTENTS ARE AMBIGUOUS
1548          005324 012746 006156      MOV      @AMBTSSR,-(SP)
1549          005330 012746 000001      MOV      @1,-(SP)
1550          005334 010600      MOV      SP,R0
1551          005336 104415      TRAP     C:PNTX
1552          005340 062706 000004      ADD      @4,SP
1553          005344 010403      5$:  MOV      R4,R3                ;CONTENTS OF TSSR
1554          005346 042703 001476      BIC     @HIADDR!FATERR!TERCLS,R3        ;CLEAR ALL MULTIPLE BIT FIELDS
1555          005352 001434      BEQ     20$                ;NO BITS ARE SET
1556          005354 012702 002576      MOV      @TMPBFR,R2        ;TEMPORARY ASCII BUFFER
1557          005360 012701 003400      MOV      @TSSRBIT,R1      ;ASCII EQUIVALENT OF BITS
1558          005364 005703      10$: TST     R3                ;REMAINING BITS TO CONVERT
1559          005366 001413      BEQ     15$                ;BRANCH WHEN ALL ARE DONE
1560          005370 000241      CLC                    ;CLEAR CARRY FOR SHIFT
1561          005372 006103      ROL     R3                ;SHIFT NEXT BIT TO CARRY
1562          005374 103006      BCC     13$                ;BRANCH IF BIT NOT SET
1563          005376 011100      MOV      (R1),R0          ;POINTER TO BIT DEFINITION
1564          005400 112022      11$: MOVB   (R0)+,(R2)+        ;MOVE ASCII TO BUFFER
1565          005402 001376      BNE                    ;MOVE ALL BITS
1566          005404 112762 000054 177777      MOVB   @',,-1(R2)        ;INSERT A COMMA TO TERMINATE
1567          005412 005721      13$: TST     (R1)+        ;POINT TO NEXT DESCRIPTION
1568          005414 000763      BR     10$                ;GET THE REMAINING BITS
1569          005416 105042      15$: CLRB   -(R2)          ;TERMINATE THE LINE
1570          005420      PRINTX @TSSDEF,@TMPBFR      ;PRINT THE BIT DEFINITIONS
1571          005420 012746 002576      MOV      @TMPBFR,-(SP)
1572          005424 012746 006127      MOV      @TSSDEF,-(SP)

```

005430	012746	000002		MOV	#2,-(SP)	
005434	010600			MOV	SP,R0	
005436	104415			TRAP	C#PNTX	
005440	062706	000006		ADD	#6,SP	
1560						
1561	005444	010403	20\$:	MOV	R4,R3	;GET THE TSSR CONTENTS
1562	005446	042703	177761	BIC	#+CTERCLS,R3	;CLEAR ALL BUT TERMINATION
1563	005452	016303	006220	MOV	TCOCOD(R3),R3	;GET THE TERMINATION CODE MEANING
1564	005456			PRINTX	#TCOASC,R3	;PRINT THE TERMINATION CODE
	005456	010346		MOV	R3,-(SP)	
	005460	012746	006017	MOV	#TCOASC,-(SP)	
	005464	012746	000002	MOV	#2,-(SP)	
	005470	010600		MOV	SP,R0	
	005472	104415		TRAP	C#PNTX	
	005474	062706	000006	ADD	#6,SP	
1565	005500	010403		MOV	R4,R3	;TSSR CONTENTS AGAIN
1566	005502	042703	177717	BIC	#+CFATERR,R3	;CLEAR ALL BUT FATAL TERMINATION
1567	005506	001421		BEQ	25\$;DON'T PRINT IF ZERO
1568	005510	006203		ASR	R3	
1569	005512	006203		ASR	R3	
1570	005514	006203		ASR	R3	;ALINE TERMINATION CODE FOR INDEX
1571	005516	016303	006560	MOV	TSFCOD(R3),R3	;GET THE FATAL TERMINATION CODE
1572	005522			PRINTX	#TFCASC,R3	;PRINT THE FATAL TERMINATION CODE
	005522	010346		MOV	R3,-(SP)	
	005524	012746	006060	MOV	#TFCASC,-(SP)	
	005530	012746	000002	MOV	#2,-(SP)	
	005534	010600		MOV	SP,R0	
	005536	104415		TRAP	C#PNTX	
	005540	062706	000006	ADD	#6,SP	
1573	005544	012737	000031	MOV	#25.,FATFLG	;DROP UNIT AFTER THIS ERROR
1574	005552	010403		MOV	R4,R3	;GET TSSR CONTENTS
1575	005554	042703	176377	BIC	#+CHIADDR,R3	;CLEAR ALL BUT EXTENDED ADDRESS
1576	005560	001411		BEQ	30\$;DON'T PRINT IF ZERO
1577	005562			PRINTX	#TEXASC,R3	;PRINT THE EXTENDED ADDRESS BITS
	005562	010346		MOV	R3,-(SP)	
	005564	012746	005756	MOV	#TEXASC,-(SP)	
	005570	012746	000002	MOV	#2,-(SP)	
	005574	010600		MOV	SP,R0	
	005576	104415		TRAP	C#PNTX	
	005600	062706	000006	ADD	#6,SP	
1578	005604	022704	100210	30\$:	CMP	#100210,R4
1579	005610	001003		BNE	31\$;BR, IF PROBABLY NOT TAPE ERROR
1580	005612	012737	005672	002146	MOV	#EPRT3,EPRTSW
1581	005620	005737	002146	31\$:	TST	EPRTSW
1582	005624	001003		BNE	310\$;CHECK FOR THE SWITCH EMPTY
1583	005626	012737	005672	002146	MOV	#EPRT1,EPRTSW
1584	005634	013737	002146	005644	310\$:	MOV
1585	005642			32\$:	PRINTB	#EPRT1
	005642	012746	005672	MOV	#EPRT1,-(SP)	
	005646	012746	000001	MOV	#1,-(SP)	
	005652	010600		MOV	SP,R0	
	005654	104414		TRAP	C#PNTB	
	005656	062706	000004	ADD	#4,SP	
1586	005662	012737	005672	002146	MOV	#EPRT1,EPRTSW
1587	005670	000207		RTS	PC	;RESET TO NORMAL ERROR POINTER
1588						;RETURN TO CALLER
1589	005672			EPRT2:		


```

1590 005672          EPRT3:
1591 005672      045    116    045 EPRT1: .ASCIZ 'NWA *****REPLACE CONTROLLER*****S'
1592 005736      045    116    045 TSSR^OR: .ASCIZ 'NWA TSSR = #06'
1593 005756      045    116    045 TEXASC: .ASCIZ 'NWA Extended Address Bits = #06'
1594 006017      045    116    045 TCOASC: .ASCIZ 'NWA Termination Class Code = #T'
1595 006060      045    116    045 TFCASC: .ASCIZ 'NWA Fatal Termination Class Code = #T'
1596 006127      045    116    045 TSSDEF: .ASCIZ 'NWA TSSR Bits Set: #T'
1597 006156      045    116    045 AMBTSSR: .ASCIZ 'NWA TSSR Contents Are Ambiguous'
1598
1599 006220      006240 006263 006311 TCOCOD: .EVEN
1600 006240          116    157    162 1$: .WORD 1$,2$,3$,4$,5$,6$,7$,8$
1601 006263          124    145    162 2$: .ASCIZ 'Normal Termination'
1602 006311          124    141    160 3$: .ASCIZ 'Termination Condition'
1603 006333          106    165    156 4$: .ASCIZ 'Tape Status Alert'
1604 006353          122    145    143 5$: .ASCIZ 'Function Reject'
1605 006435          122    145    143 6$: .ASCIZ 'Recoverable Error - Tape Position One Record Down'
1606 006504          125    156    162 7$: .ASCIZ 'Recoverable Error - Tape Was Not Moved'
1607 006530          106    141    164 8$: .ASCIZ 'Unrecoverable Error'
1608                                     .ASCIZ 'Fatal Controller Error'
1609                                     .EVEN
1610 006560      006570 006624 006635 TSFCOD: .WORD 1$,2$,3$,4$
1611 006570          111    156    164 1$: .ASCIZ 'Internal Diagnostic Failure'
1612 006624          122    145    163 2$: .ASCIZ 'Reserved'
1613 006635          102    165    163 3$: .ASCIZ 'Bus Interface or Sanity Check Error'
1614 006701          122    145    163 4$: .ASCIZ 'Reserved'
1615                                     .EVEN

```

```

1617                                     .SBTTL PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET
1618
1619
1620                                     ;*
1621                                     ;THIS ROUTINE PRINTS THE ADDRESS AND CONTENTS OF A COMMAND PACKET.
1622                                     ;THIS ROUTINE IS NORMALLY ONLY CALLED FROM A PRINT ROUTINE.
1623                                     ;
1624                                     ;INPUT:
1625                                     ;
1626                                     ;       R0      NUMBER OF WORDS IN PACKET
1627                                     ;       R3      HIGH ORDER COMMAND PACKET ADDRESS
1628                                     ;       R4      ADDRESS OF COMMAND PACKET
1629                                     ;
1630                                     ;       NOTE:   R3 IS IGNORED IF THE KTENABLE FLAG IS CLEAR.
1631                                     ;
1632 006712 PRIPKT::
1633 006712 SAVREG                                ;SAVE THE REGISTERS
1634 006716 010005 MOV R0,R5                    ;SAVE NO. OF WORDS IN PACKET
1635 006720 005737 003102 TST KTENABLE          ;ABOVE 28K UNDER TEST?
1636 006724 001001 BNE 10$                      ;BR IF YES
1637 006726 005003 CLR R3                      ;SET HIGH ORDER ADDRESS TO 0
1638 006730 010301 10$: MOV R3,R1              ;COPY HIGH ORDER ADDRESS
1639 006732 010400 MOV R4,R0              ;GET LOWER ADDRESS
1640 006734 006100 ROL R0                      ;SHIFT BIT 15 INTO C BIT
1641 006736 006101 ROL R1                      ;AND INTO HIGH ORDER.
1642 006740 PRINTB #PKTADD,R1,R4              ;PRINT PACKET ADDRESS
1643 006742 010446 MOV R4,-(SP)
1644 006744 010146 MOV R1,-(SP)
1645 006746 012746 007116 MOV #PKTADD,-(SP)
1646 006750 012746 000003 MOV #3,-(SP)
1647 006754 010600 MOV SP,R0
1648 006756 104414 TRAP C#PNTB
1649 006760 062706 000010 ADD #10,SP
1650 006764 010300 15$: MOV R3,R0              ;GET HIGH ORDER ADDRESS
1651 006766 001404 BEQ 20$                      ;BR IF NOT ABOVE 28K.
1652 006770 010401 MOV R4,R1              ;GET LOW ORDER ADDRESS
1653 006772 004737 020112 JSR PC,SETMAP        ;SETUP PAR6 MAPPING FOR 18 BIT ADDRESS
1654 006776 010004 MOV R0,R4              ;GET RETURNED PAR6 ADDRESS BIAS
1655 006780 005001 20$: CLR R1                  ;SAVE WORD NUMBER
1656 006784 012402 25$: MOV (R4)+,R2          ;GET PACKET CONTENTS
1657 006788 010246 PRINTB #PKTFRM,R1,R2        ;PRINT THE DATA
1658 006792 010146 MOV R2,-(SP)
1659 006796 012746 007060 MOV R1,-(SP)
1660 006800 012746 000003 MOV #PKTFRM,-(SP)
1661 006804 010600 MOV #3,-(SP)
1662 006808 010414 MOV SP,R0
1663 006812 062706 000010 TRAP C#PNTB
1664 006816 005201 ADD #10,SP
1665 006820 020105 INC R1                  ;NEXT WORD NUMBER
1666 006824 002762 CMP R1,R5              ;DONE ALL PACKET WORDS?
1667 006828 007036 BLT 25$                      ;LOOP TILL ALL DONE
1668 006832 012746 007153 PRINTB #PKTNEW
1669 006836 012746 000001 MOV #PKTNEW,-(SP)
1670 006840 010600 MOV #1,-(SP)
1671 006844 104414 MOV SP,R0
1672 006848 062706 000004 TRAP C#PNTB
1673 006852 007052 ADD #4,SP

```


M4

CZTKEA TK25 FRT END FUNC #1 MACRO M1200 20-APR-84 08:12 PAGE 38-1
PRIPKT - PRINT THE ADDRESS/CONTENTS OF COMMAND PACKET

SEQ 51

					RTS	PC	;RETURN
1655	007056	000207					
1656							
1657	007060	045	116	045	PKTFRM:	.ASCIZ	'#N#A Packet Word #D1#A = #06'
1658	007116	045	116	045	PKTADD:	.ASCIZ	'#N#A Packet Address = #01#05'
1659							
1660	007153	045	116	045	PKTNEW:	.ASCIZ	'#N#N#A '
1661						.EVEN	
1662							

```

1664 .SBTTL PRIBXOR - PRINT EXPD, RECV AND XOR BYTE
1665
1666
1667 ;*
1668 ;PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE DATA BYTE
1669 ;THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
1670 ;
1671 ;INPUTS:
1672 ;
1673 ; R1 RECEIVED DATA
1674 ; R2 EXPECTED DATA
1675 ;
1676 ;OUTPUT:
1677 ;
1678 ; R0 XOR OF EXPECTED/RECEIVED DATA
1679 ;
1680 ;-
1681
1682 007164 PRIBXOR::
1683 007164 SAVREG ;SAVE THE REGISTERS
1684 007170 010203 MOV R2,R3 ;EXPECTED DATA
1685 007172 XOR R1,R3 ;FORM THE EXCLUSIVE OR
1686 007202 012700 177400 MOV #C<377>,R0 ;BYTE MASK
1687 007206 040001 BIC R0,R1 ;SAVE LOW BYTE RECV
1688 007210 040002 BIC R0,R2 ;SAVE LOW BYTE EXPD
1689 007212 040003 BIC R0,R3 ;SAVE LOW BYTE XOR
1690 007214 PRINTB #XORBFOR,R2,R1,R3 ;PRINT THE MESSAGE
1691 007214 010346 MOV R3,-(SP)
1692 007216 010146 MOV R1,-(SP)
1693 007220 010246 MOV R2,-(SP)
1694 007222 012746 007246 MOV #XORBFOR,-(SP)
1695 007226 012746 000004 MOV #4,-(SP)
1696 007232 010600 MOV SP,R0
1697 007234 104414 TRAP C#PNTB
1698 007236 062706 000012 ADD #12,SP
1699 007242 010300 MOV R3,R0 ;R0 HAS XOR ON RETURN
1700 007244 000207 RTS PC ;RETURN TO CALLER
1701
1702 045 116 045 XORBFOR: .ASCIZ '#N#A EXPD: #03#A RECV: #03#A XOR: #03#A'
1703 .EVEN
  
```



```

1698 .SBTTL PRI XOR - PRINT EXPD, RECV AND XOR
1699
1700 ;*
1701 ;PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE TWO
1702 ;THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
1703 ;
1704 ;INPUTS:
1705 ;
1706 ; R1 RECEIVED DATA
1707 ; R2 EXPECTED DATA
1708 ;
1709 ;OUTPUT:
1710 ;
1711 ; R0 XOR OF EXPECTED/RECEIVED DATA
1712 ;
1713 ;-
1714
1715
1716 007314 SAVREG ;SAVE THE REGISTERS
1717 007314 MOV R2,R3 ;EXPECTED DATA
1718 007320 010203 XOR R1,R3 ;FORM THE EXCLUSIVE OR
1719 007322 PRINTB @XORFOR,R2,R1,R3 ;PRINT THE MESSAGE
1720 007332 MOV R3,-(SP)
007332 010346 MOV R1,-(SP)
007334 010146 MOV R2,-(SP)
007336 010246 MOV @XORFOR,-(SP)
007340 012746 007364 MOV #4,-(SP)
007344 012746 000004 MOV SP,R0
007350 010600 TRAP C:PNTB
007352 104414 ADD #12,SP
007354 062706 000012 MOV R3,R0 ;R0 HAS XOR ON RETURN
1721 007360 010300 RTS ;RETURN TO CALLER
1722 007362 000207
1723
1724 007364 045 116 045 XORFOR: .ASCIZ 'N#A EXPD: #06#A RECV: #06#A XOR: #06#
1725 .EVEN

```



```

1778
1779
1780 007532
1781 007532
1782 007536 013700 002202
1783 007542 013701 002204
1784 007546 010102
1785 007550 006101
1786 007552 006100
1787 007554
    007554 010246
    007556 010046
    007560 012746 007602
    007564 012746 000003
    007570 010600
    007572 104414
    007574 062706 000010
1788 007600 000207
1789
1790 007602 045 116
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806 007646
1807 007646
1808 007652 013700 002202
1809 007656 013701 002204
1810 007662 010102
1811 007664 006101
1812 007666 006100
1813 007670
    007670 010246
    007672 010046
    007674 012746 007716
    007700 012746 000003
    007704 010600
    007706 104414
    007710 062706 000010
1814 007714 000207
1815
1816 007716 045 116
1817
1818
1819
1820

```

```

;
; -
; 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

```

```

045 PRIA0: .ASCIZ '#N#A MEMORY ERROR ADDRESS = #01#05'
          .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,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 @PRITO,R0,R2 ;SHIFT INTO HIGH ORDER
    MOV R2,-(SP) ;PRINT MEMORY ADDRESS IN ERROR
    MOV R0,-(SP)
    MOV @PRITO,-(SP)
    MOV @3,-(SP)
    MOV SP,R0
    TRAP C#PNTB
    ADD @10,SP
    RTS PC ;RETURN

```

```

045 PRITO: .ASCIZ '#N#A MEMORY TEST ADDRESS = #01#05'
          .EVEN

```

```

1822 .SBTTL SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND
1823
1824 ;*
1825 ;
1826 ;ROUTINE TO ISSUE A SPACE RECORDS
1827 ;COMMAND (FORWARD OR REVERSE)
1828 ;
1829 ;INPUT:
1830 ;
1831 ; R3 NUMBER OF RECORDS TO BE SPACED OVER
1832 ; BIT15 CONTROLS DIRECTION
1833 ; BIT15 = 0 IS FORWARD
1834 ; BIT15 = 1 IS REVERSE
1835 ; R5 FIRST DEVICE UNIBUS ADDRESS
1836 ;
1837 ; REQUIRES A WRITE CHARACTERISTICS DONE PREVIOUSLY
1838 ;
1839 ;OUTPUT:
1840 ;
1841 ; CARRY SET - SPACE RECORDS COMMAND OK
1842 ; CLR - SPACE RECORDS FAILED
1843 ;
1844 ;
1845 ; R0 THE CONTENTS OF R4 IS MOVED TO R0
1846 ;
1847 ;
1848 ;IMPLICIT OUTPUT:
1849 ;
1850 ; TAPE HAS BEEN MOVED
1851 ;
1852 ;SIDE EFFECTS:
1853 ;
1854 ;
1855 ;-
1856 ;
1857 SPACE::
1858 SAVREG ;SAVE THE GENERAL REGISTERS
1859 MOV #500.,SDELAY ;SET UP DELAY
1860 MOV #140010,80$ ;SET UP COMMAND, SPACE FORWARD
1861 T R3 ;CHECK FOR DIRECTION
1862 BR 5$ ;BR, IF REVERSE INDICATED
1863 MOV R3,90$ ;LOAD UP NUMBER OF RECORDS TO SPACE
1864 BR 10$ ;GO DO COMMAND
1865 5$: BIC #BIT15,R3 ;CLEAR DIRECTION BIT
1866 MOV R3,90$ ;LOAD UP NUMBER OF RECORDS TO SPACE
1867 BIS #BIT8,80$ ;SET REVERSE BIT IN COMMAND PACKET
1868 10$: MOV #80$,R4 ;SET UP R4 WITH PACKET ADDRESS
1869 MOV R4,TSD8(R5) ;SEND OUT COMMAND
1870 15$: JSR PC,WAITF ;WAIT FOR SSR
1871 BCS 20$ ;BR, IF SSR IS SET AND OK
1872 DELAY 250 ;DELAY ABOUT .25 SECONDS
1873 MOV #250,(PC)+
1874 .WORD 0
1875 MOV L$DLY,(PC)+
1876 .WORD 0
1877 DEC -6(PC)
1878 BNE .-4

```

1857	007760								
1858	007760								
1859	007764	012737	000764	010150					
1860	007772	012737	140010	010140					
1861	010000	005703							
1862	010002	100403							
1863	010004	010337	010142						
1864	010010	000407							
1865	010012	042703	100000		5\$:				
1866	010016	010337	010142						
1867	010022	052737	000400	010140					
1868	010030	012704	010140		10\$:				
1869	010034	010465	177776						
1870	010040	004737	016744		15\$:				
1871	010044	103420							
1872	010046								
	010046	012727	000250						
	010052	000000							
	010054	013727	002116						
	010060	000000							
	010062	005367	177772						
	010066	001375							

	010070	005367	177756		DEC	-22(PC)	
	010074	001367			BNE	.-20	
1873	010076	005337	010150		DEC	SDELAY	;BUMP DELAY COUNTER DOWN
1874	010102	001356			BNE	15\$;BR, IF MORE DELAY
1875	010104	000411			BR	60\$;BR IF TROUBLE CARRY = CLEAR
1876	010106	016501	000000	20\$:	MOV	TSSR(R5),R1	;READ TSSR
1877	010112	012702	000200		MOV	#SSR,R2	;SET UP EXPECTED
1878	010116	020201		25\$:	CMP	R2,R1	;ARE THEY OK
1879	010120	001401			BEG	40\$;BR, IF EQUAL = OK
1880	010122	000402			BR	60\$;TROUBLE EXIT
1881	010124	000261		40\$:	SEC		;SET CARRY NO TROUBLE
1882	010126	000401			BR		;EXIT
1883	010130	000241		60\$:	CLC	70\$;CARRY CLEAR = ERROR
1884	010132			70\$:			
1885	010132	010400			MOV	R4,R0	;PASS PACKET ADDRESS
1886	010134	000207			RTS	PC	;RETURN

```

1888      ;
1889      ;
1890      ;
1891      ;PACKET FOR SPACE COMMAND
1892      ;
1894 010136      .BLKB  10-<.-TUV2A&7>
1896      ;
1897      ;COMMAND WORD
1898 010140 000000 80$: .WORD
1899      ;NUMBER OF RECORDS TO BE SPACED OVER WORD
1900 010142 000000 90$: .WORD
1901 010144 000000      .WORD
1902 010146 000000      .WORD
1903 010150 000000 SDELAY: .WORD 0      ;DELAY COUNTER
1904      .EVEN

```


1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937 010152
1938 010152
1939 010156 005037 002174
1940 010162 010465 177776
1941 010166 004737 017060
1942 010172 103401
1943 010174 000423
1944 010176 016501 000000
1945 010202 012702 000200
1946 010206 032701 000100
1947 010212 001402
1948 010214 052702 000100
1949 010220 020201
1950 010222 001401
1951 010224 000407
1952 010226 062704 000010
1953 010232 011403
1954 010234 010337 002716
1955 010240 000261
1956 010242 000401
1957 010244 000241
1958 010246 016500 000000
1959 010252 000207
1960
1961

.SBTTL WRTCHR - WRITE CHARACTERISTICS COMMAND

```

; *
;
; ROUTINE TO ISSUE A WRITE CHARACTERISTICS
; COMMAND SO THAT OTHER COMMANDS WILL BE ACCEPTED
;
; INPUT:
;
; R4 ADDRESS OF PACKET FROM TEST
; R5 FIRST DEVICE UNIBUS ADDRESS
; REQUIRES A CALL TO SOFINIT BE DONE PREVIOUSLY
;
; OUTPUT:
;
; R0 TSSR CONTENTS
; CARRY SET - WRITE CHARACTERISTICS COMMAND OK
; CLR - WRITE CHARACTERISTICS FAILED
;
; IMPLICIT OUTPUT:
;
; MESSAGE BUFFER AND OTHER BUFFERS ALL SET UP
; SOFTWARE SWITCHES SET AS FOLLOWS:
; BENBSW = BUFFER ENABLE SWITCH ON OR OFF
;
; SIDE EFFECTS:
;
; -
  
```

```

WRTCHR:
  SAVREG                                ;SAVE THE GENERAL REGISTERS
  CLR BENBSW                            ;CLEAR BUFFER ENABLE SWITCH
10$: MOV R4,TSDB(R5)                    ;SEND OUT COMMAND
      JSR PC,CHKTSSR                    ;WAIT FOR SSR
      BCS 20$                           ;BR, IF SSR IS SET AND OK
      BR 60$                            ;BR IF TROUBLE CARRY = CLEAR
20$: MOV TSSR(R5),R1                    ;READ TSSR
      MOV #SSR,R2                        ;SET UP EXPECTED
      BIT #OFL,R1                        ;WAS OFF LINE SET IN TSSR
      BEQ 25$                            ;BR, IF NO OFL SET
      BIS #OFL,R2                        ;MAKE THEM LOOK ALIKE
25$: CMP R2,R1                          ;ARE THEY OK
      BEQ 40$                            ;BR, IF EQUAL = OK
      BR 60$                            ;TROUBLE EXIT
40$: ADD #8.,R4                          ;POINT TO WRT CHARA DATA PACKET
      MOV (R4),R3                         ;GET ADDRESS OF MESSAGE BUFFER
      MOV R3,MESBFA                       ;STORE FOR PRINT ROUTINES
      SEC                                ;SET CARRY NO TROUBLE
      BR 70$                             ;EXIT
60$: CLC                                ;CARRY CLEAR = ERROR
70$: MOV TSSR(R5),R0                    ;RETURN TSSR CONTENTS
      RTS PC                              ;RETURN
  
```

1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990

.SBTTL REWIND - POSITION TAPE (REWIND) COMMAND

```

;+
; THIS ROUTINE WILL REWIND THE SELECTED TAPE.
;
; CAUTION: THE ROUTINE DOES NOT WAIT FOR BOT
; TO ARRIVE. ALSO THE CALLER MUST CHECK FOR
; SSR TO SET IN THE TSSR
;
; CALLING SEQUENCE:
;
; DO A SOFT INIT
; DO A WRITE CHARACTERISTICS
; JSR PC,REWIND
;
; INPUT:
;
; R5 FIRST DEVICE UNIBUS ADDRESS
;
; OUTPUT
;
; R0 THE CONTENTS OF R4 IS PASSED TO R0
;
; -
REWIND::
  SAVREG                                ;SAVE R1-R5 UNTIL NEXT RETURN
  MOV #RWPACK,R4                        ;GET PACKET ADDRESS
  MOV R4,TSDB(R5)                       ;SEND PACKET ADDRESS TO EXECUTE
  MOV #360.,R3                          ;ENOUGH TIME FOR 2400' REEL TO REWIND
10$: JSR PC,WAITF                         ;WAIT FOR SSR TO SET
      BCS 20$                            ;LEAVE WHEN SSR IS SET
      DELAY 250.                         ;WAIT FOR .25 SECONDS
      MOV #250.,(PC)+
      .WORD 0
      MOV L$DLY,(PC)+
      .WORD 0
      DEC -6(PC)
      BNE -.4
      DEC -22(PC)
      BNE -.20
      DEC R3                              ;BUMP COUNTER DOWN
      BNE 10$                             ;KEEP GOING
      CLC                                 ;CLEAR CARRY TO SET ERROR
20$: MOV R4,R0                          ;PASS THE PACKET ADDRESS
      RTS PC                             ;RETURN

RWPACK: .BLKB 10-<.-TUV2A&7>
        .WORD 102010                    ;POSITION COMMAND (REWIND)
        .WORD 0                        ;NOT USED
  
```

1991 010254
1992 010254
1993 010260 012704 010350
1994 010264 010465 177776
1995 010270 012703 000550
1996 010274 004737 016744
1997 010300 103417
1998 010302
010302 012727 000372
010306 000000
010310 013727 002116
010314 000000
010316 005367 177772
010322 001375
010324 005367 177756
010330 001367
1999 010332 005303
2000 010334 001357
2001 010336 000241
2002 010340 010400
2003 010342 000207
2004
2005
2007 010344
2009 010350
2010 010350 102010
2011 010352 000000

2013 .SBTTL CKRAM - COMPARE RAM TO I/O PACKET

```

2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040

```

```

;+
;
;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:
;
;
;-

```

```

2041 010354
2042 010354
2043 010360 012701 002206
2044 010364 012702 000020
2045 010370 005003
2046 010372 004737 017060
2047 010376 004737 017060
2048 010402 110265 177777
2049 010406 004737 017060
2050 010412 116511 177776
2051 010416 122124
2052 010420 001401
2053 010422 005203
2054 010424 005202
2055 010426 020227 000027
2056 010432 003761
2057 010434 005703
2058 010436 001402
2059 010440 000241
2060 010442 000401
2061 010444 000261
2062 010446 012737 000010 002246
2063 010454 000207
2064

```

```

CKRAM::
  SAVREG
  MOV @RAMDATA,R1 ;SAVE THE GENERAL REGISTERS
  MOV @RMPKTBEGR2 ;ADDRESS TO SAVE THE RAM DATA
  CLR R3 ;BYTE ADDRESS OF FIRST RAM DATA
  JSR PC,CHKTSSR ;CLEAR THE ERROR FLAG
  JSR PC,CHKTSSR ;WAIT FOR SSR
  MOV R2,TSDBH(R5) ;WAIT FOR SSR TO SET
  JSR PC,CHKTSSR ;SELECT NEXT RAM ADDRESS
  MOV B(TSBAL(R5),(R1) ;WAIT FOR SSR TO SET
  CMPB (R1)*,(R4)* ;READ THE RAM DATA
  BEQ 20$ ;COMPARE TO EXPECTED
  INC R3 ;BRANCH IF OK
  INC R2 ;SET ERROR FLAG
  CMP R2,@RMPKTEND ;ADDRESS OF NEXT RAM LOCATION
  BLE 10$ ;REACHED END YET ?
  TST R3 ;BRANCH TILL ALL READ
  BEQ 30$ ;WAS AN ERROR FOUND ?
  CLC ;BRANCH IF NOT
  BR 50$ ;CLEAR CARRY TO SHOW ERROR
  SEC ;AND EXIT
  MOV @8.,RAMSIZ ;SHOW GOOD COMPARE
  RTS PC ;SETUP RAMSIZ FOR PRAMPKT ROUTINE

```

```

2066 .SBTTL RAMER - READ AND DISPLAY SELECTED RAM
2067 ;*
2068 ;
2069 ;ROUTINE TO READ THE SELECTED RAM LOCATIONS
2070 ;
2071 ;INPUT:
2072 ;
2073 ; R5 FIRST DEVICE UNIBUS ADDRESS
2074 ; CONSOLE WILL ALSO BE PRINTED TO
2075 ;
2076 ;IMPLICIT OUTPUT:
2077 ;
2078 ; THE TABLE RAMDATA IS FILLED WITH THE
2079 ; DATA HELD IN RAM.
2080 ;
2081 ;SIDE EFFECTS:
2082 ;
2083 ;
2084 ;-
2085
2086 010456 RAMER:: SAVREG ;SAVE THE GENERAL REGISTERS
2087 010456 MOV RAMR5H,R5 ;RESET R5 TO FIRST DEVICE REGISTER
2088 010462 013705 010642 MOV @RAMDATA,R1 ;ADDRESS TO SAVE THE RAM DATA
2089 010466 012701 002206 MOV RAMHLD,R2 ;BYTE ADDRESS OF THE FIRST RAM DATA
2090 010472 013702 010640 MOV RAMSIZ,R3 ;SET THE SIZE OF THE READ UP
2091 010476 013703 002246 MOV RAMSIZ,R3 ;SET THE SIZE OF THE READ UP
2092 010502 004737 017060 10$: JSR PC,CHKTSSR ;WAIT FOR THE SSR TO SET
2093 010506 110265 177777 MOV R2,TSDBH(R5) ;SELECT NEXT RAM ADDRESS
2094 010512 004737 017060 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
2095 010516 116521 177776 MOV R2,TSDBH(R5),(R1)+ ;READ THE RAM DATA
2096 010522 062702 000001 20$: ADD #1,R2 ;ADDRESS OF THE NEXT RAM LOCATION
2097 010526 077313 SOB R3,10$ ;NUMBER OF LOCATIONS COUNTER
2098 010530 013704 002246 MOV RAMSIZ,R4 ;GET THE RAM SIZE
2099 010534 013702 010640 MOV RAMHLD,R2 ;GET THE STARTING RAM ADDRESS
2100 010540 060204 ADD R2,R4 ;CALCULATE THE END ADDRESS
2101 010542 162704 000001 SUB #1,R4 ;CORRECT VALUE OF PRINTOUT
2102 010546 PRINTX @RAMIOP,R2,R4 ;RAM ADDRESS = 10 - 17, ETC.
    010546 010446 MOV R4,-(SP)
    010550 010246 MOV R2,-(SP)
    010552 012746 010644 MOV @RAMIOP,-(SP)
    010556 012746 000003 MOV #3,-(SP)
    010562 010600 MOV SP,R0
    010564 104415 TRAP C#PNTX
    010566 062706 000010 ADD #10,SP
2103 010572 012701 002206 MOV @RAMDATA,R1 ;ADDRESS OF WHERE RAM DATA IS
2104 010576 013703 002246 MOV RAMSIZ,R3 ;THE SIZE OF THE RAM FIELD READ
2105 010602 005004 30$: CLR R4 ;NO EXTRA DATA LEFT OVER
2106 010604 112104 MOV R4,(R1)+,R4 ;PICK UP BYTE OF RAM DATA
2107 010606 042704 177400 BIC #177400,R4 ;GET RID OF SIGN EXTEND
2108 010612 PRINTX @RAMPD,R4 ;"010 211 111 222 377 000 123 134 ETC."
    010612 010446 MOV R4,-(SP)
    010614 012746 010715 MOV @RAMPD,-(SP)
    010620 012746 000002 MOV #2,-(SP)
    010624 010600 MOV SP,R0
    010626 104415 TRAP C#PNTX
    010630 062706 000006 ADD #6,SP
2109 010634 077316 SOB R3,30$ ;LOOP UNTIL ALL PRINTED

```



```

2119 .SBTTL CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA
2120 ;*
2121 ;
2122 ;ROUTINE TO READ THE FIRST 8 OR 10 BYTES FROM RAM
2123 ;MEMORY AND COMPARE THIS DATA TO A CHARACTERISTICS DATA BLOCK.
2124 ;
2125 ;INPUT:
2126 ;
2127 ; R4 ADDRESS OF THE CHARACTERISTICS DATA
2128 ; R5 FIRST DEVICE UNIBUS ADDRESS
2129 ;
2130 ;OUTPUT:
2131 ;
2132 ; CARRY SET - RAM MATCHES PACKET
2133 ; CLR - RAM DOES NOT MATCH PACKET
2134 ;
2135 ;IMPLICIT OUTPUT:
2136 ;
2137 ; THE TABLE RAMDATA IS FILLED WITH THE
2138 ; DATA HELD IN RAM.
2139 ; RAMSIZ IS SET TO 8. OR 10. FOR PRAMPKT ROUTINE
2140 ;
2141 ;SIDE EFFECTS:
2142 ;
2143 ;
2144 ;-
2145
2146 010730 CKRAM2::
2147 010730 SAVREG ;SAVE THE GENERAL REGISTERS
2148 010734 012701 002206 MOV #RAMDATA,R1 ;ADDRESS TO SAVE THE RAM DATA
2149 010740 012702 000167 MOV #RMCHBEG,R2 ;BYTE ADDRESS OF FIRST RAM DATA
2150 010744 005003 CLR R3 ;CLEAR THE ERROR FLAG
2151 010746 004737 017060 JSR PC,CHKTSSR ;WAIT FOR SSR
2152 010752 004737 017060 10$: JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
2153 010756 110265 177777 MOV# R2,TSDBH(R5) ;SELECT NEXT RAM ADDRESS
2154 010762 004737 017060 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
2155 010766 116511 177776 MOV# TSBAL(R5),(R1) ;READ THE RAM DATA
2156 010772 122124 CMP# (R1),.(R4) ;COMPARE TO EXPECTED
2157 010774 001401 BEQ 20$ ;BRANCH IF OK
2158 010776 005203 INC R3 ;SET ERROR FLAG
2159 011000 005202 20$: INC R2 ;ADDRESS OF NEXT RAM LOCATION
2160 011002 012737 000010 002246 MOV #8.,RAMSIZ ;ASSUME NORMAL NOT SET
2161 011010 020227 000176 CMP R2,#RMCHEND-2 ;REACHED END YET ?
2162 011014 003756 BLE 10$ ;BRANCH TILL ALL READ
2163 011016 005703 27$: TST R3 ;WAS AN ERROR FOUND ?
2164 011020 001402 BEQ 30$ ;BRANCH IF NOT
2165 011022 000241 CLC ;CLEAR CARRY TO SHOW ERROR
2166 011024 000401 BR 50$ ;AND EXIT
2167 011026 000261 30$: SEC ;SHOW GOOD COMPARE
2168 011030 000207 50$: RTS PC ;RETURN
2169

```



```

2171          .SBTTL  CKMSG  - COMPARE WRITE CHAR. MESSAGE BUFFERS
2172          ;*
2173          ;
2174          ;ROUTINE TO COMPARE A WRITE CHARACTERISTICS EXPD AND RECV
2175          ;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
2176          ;ERROR PRINT ROUTINES.
2177          ;
2178          ;INPUT:
2179          ;
2180          ;      R0      RECV MESSAGE BUFFER HIGH ORDER ADDRESS
2181          ;      R1      RECV MESSAGE BUFFER LOW ORDER ADDRESS
2182          ;      R2      EXPD MESSAGE BUFFER ADDRESS
2183          ;OUTPUT:
2184          ;
2185          ;      CARRY   SET - MESSAGE BUFFERS MATCH
2186          ;             CLR -MESSAGE BUFFERS DON'T MATCH
2187          ;
2188          ;IMPLICIT OUTPUT:
2189          ;
2190          ;      EXPMSG   BUFFER IS SET TO EXPD DATA
2191          ;      RECVMSG  BUFFER IS SET TO RECV DATA
2192          ;      RCVHIADD  SET TO HIGH ORDER ADDRESS OF RECV
2193          ;      RCVLOADD  SET TO LOW ORDER ADDRESS OF RECV
2194          ;
2195          ;-
2196          CKMSG::
2197          SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
2198          MOV             RO,RCVHIADD    ;SAVE RECV HIGH ADDRESS
2199          MOV             R1,RCVLOAD    ;SAVE RECV LOW ADDRESS
2200          TST            KTENABLE      ;TESTING ABOVE 28K?
2201          BEQ            10$          ;BR IF NO
2202          JSR            PC,SETMAP     ;RETURN ADDRESS BIASED TO PAR6 IN RO
2203          MOV             R0,R1       ;GET RETURNED ADDRESS BIASED TO PAR6
2204          CLR             R4          ;WORD IN BUFFER
2205          CLR             R3          ;CLEAR ERROR SEEN FLAG
2206          MOV             R2,R5       ;GET EXPD BUFFER ADDRESS
2207          MOV             (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
2208          MOV             (R1),RECVMSG(R4) ;SAVE RECV FOR ERROR REPORT
2209          CMP             (R2)+,(R1)+  ;EXPD EQUAL RECV?
2210          BEQ            25$          ;BR IF YES
2211          INC             R3          ;SET ERROR SEEN FLAG
2212          ADD             #2,R4       ;POINT TO NEXT WORD ADDRESS
2213          CMP             R4,#14     ;DONE FIRST 7 WORDS?
2214          BLE            15$          ;BR IF NO
2215          BIT             #X2.EXTF,XST2(R5);IS EXTENDED FEATURES SET IN EXPD?
2216          BEQ            50$          ;BR IF NO
2217          CMP             R4,#16     ;DONE EXTENDED FEATURES WORD?
2218          BLE            15$          ;BR IF NO
2219          TST            R3          ;ANY ERRORS SEEN?
2220          BEQ            55$          ;BR IF NO
2221          CLC             ;SET FAILURE
2222          BR             60$          ;
2223          SEC             ;SET SUCCESS
2224          RTS            PC          ;RETURN
2225
2196 011032
2197 011032
2198 011036 010037 002250
2199 011042 010137 002252
2200 011046 005737 003102
2201 011052 001403
2202 011054 004737 020112
2203 011060 010001
2204 011062 005004 10$:
2205 011064 005003
2206 011066 010205
2207 011070 011264 002266 15$:
2208 011074 011164 002432
2209 011100 022221
2210 011102 001401
2211 011104 005203
2212 011106 062704 000002 25$:
2213 011112 020427 000014
2214 011116 003764
2215 011120 032765 000200 000012
2216 011126 001403
2217 011130 020427 000016
2218 011134 003755
2219 011136 005703 50$:
2220 011140 001402
2221 011142 000241
2222 011144 000401
2223 011146 000261 55$:
2224 011150 000207 60$:
2225
    
```

2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254 011152
2255 011152
2256 011156 020327 000144
2257 011162 003412
2258 011164 012703 000144
2259 011170
011170 012746 011304
011174 012746 000001
011200 010600
011202 104417
011204 062706 000004
2260 011210 010037 002250
2261 011214 010137 002252
2262 011220 005737 003102
2263 011224 001403
2264 011226 004737 020112
2265 011232 010001
2266 011234 005004
2267 011236 005005
2268 011240 111264 002266
2269 011244 111164 002432
2270 011250 122221
2271 011252 001401
2272 011254 005205
2273 011256 062704 000001
2274 011262 020403
2275 011264 002001
2276 011266 000764
2277 011270 005705
2278 011272 001402

```

.SBTTL CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS
;*
;ROUTINE TO COMPARE AN EXPECTED AND RECEIVED MESSAGE
;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
;ERROR PRINT ROUTINES.
;INPUT:
;      R0      RECV MESSAGE BUFFER HIGH ORDER ADDRESS
;      R1      RECV MESSAGE BUFFER LOW ORDER ADDRESS
;      R2      EXPD MESSAGE BUFFER ADDRESS
;      R3      NUMBER OF BYTES TO COMPARE
;OUTPUT:
;      CARRY   SET - MESSAGE BUFFERS MATCH
;             CLR - MESSAGE BUFFERS DON'T MATCH
;IMPLICIT OUTPUT:
;      EXPMSG  BUFFER IS SET TO EXPD DATA
;      RECMG   BUFFER IS SET TO RECV DATA
;      RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
;      RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
;
;--
CKMSG2::
  SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
  CMP             R3,#RECMG-EXPMSG;800 IS COUNT ABOVE MAX ALLOWED?
  BLE            5: ;800 BR IF NO
  MOV             #RECMG-EXPMSG,R3;800
  PRINTF         #DEBUGMSG ;800
  MOV             #DEBUGMSG,-(SP)
  MOV             #1,-(SP)
  MOV             SP,R0
  TRAP           C:PNTF
  ADD             #4,SP
5: MOV            R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
  MOV            R1,RCVLOAD ;SAVE RECV LOW ADDRESS
  TST            KTENABLE ;TESTING ABOVE 28K?
  BEQ            10: ;BR IF NO
  JSR            PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
  MOV            R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
10: CLR           R4 ;WORD IN BUFFER
  CLR           R5 ;CLEAR ERROR SEEN FLAG
15: MOVB         (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
  MOVB         (R1),RECMG(R4) ;SAVE RECV FOR ERROR REPORT
  CMPB         (R2),,(R1) ;EXPD EQUAL RECV?
  BEQ            25: ;BR IF YES
  INC           R5 ;SET ERROR SEEN FLAG
25: ADD           #1,R4 ;POINT TO NEXT BYTE
  CMP          R4,R3 ;DONE ALL BYTES?
  BGE          50: ;BR IF YES
  BR           15: ;DO NEXT BYTE
50: TST          R5 ;ANY ERRORS SEEN?
  BEQ          55: ;BR IF NO

```



```

2279 011274 000241          CLC          ;SET FAILURE
2280 011276 000401          BR          60$          ;
2281 011300 000261          55$: SEC          ;SET SUCCESS
2282 011302 000207          60$: RTS          PC          ;RETURN
2283
2284 011304 120 122 117 DEBUGMSG: .ASCIZ 'PROGRAM INTERNAL ERROR -CKMSG2 MESSAGE BUFFER EXCEEDED-' ;@@D
2285 011374 045 116 045 FERCM: .ASCII /NMA ***/
2286 011405 040 040 124 ERCM: .ASCIZ / TSSR ERROR CODE REC'D = /
2287 011440 056 056 056 SIMSG: .ASCIZ /.... AFTER DOING SOFT INIT/
2288 011473 124 105 123 TINERR: .ASCIZ /TEST: .../
2289 .EVEN

```

2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341

011506
011506
011506 004737 005264
011512 004737 017776
011516
011516 104423
011520
011520
011520 004737 005264
011524 012700 000004
011530 004737 006712
011534 013700 002716
011540 005001
011542 004737 013702
011546
011546
011546 104423

```

; *
; PRINT ROUTINE TO FATAL SOFT INIT ERRORS
; INPUT:
;       R1       CONTENTS OF TSSR AT ERROR
; SIDE EFFECTS:
;       EXECUTES DROP UNIT TO CEASE TESTING
; -
      BGNMSG SFMSG
SFMSG: JSR PC,PRITSSR ;PRINT CONTENTS OF TSSR REGISTER
        JSR PC,CKDROP ;DROP UNIT, IF ALLOWED
        ENDMSG
L10003: TRAP C#MSG

; *
; PRINT ROUTINE TO PRINT THE CONTENTS OF
; TSSR AND A COMMAND PACKET OTHER THAN GET STATUS COMMAND PACKET.
; INPUTS:
;       R1       TSSR CONTENTS
;       R4       ADDRESS OF COMMAND PACKET
; -
      BGNMSG PKTSSR
PKTSSR: JSR PC,PRITSSR ;PRINT THE CONTENTS OF TSSR REGISTER
        MOV #4,R0 ;NO. OF WORDS IN PACKET
        JSR PC,PRIPKT ;PRINT THE CONTENTS OF COMMAND PACKET
        MOV MESBFA,R0 ;ADDRESS OF MESSAGE BUFFER
        CLR R1 ;ASSUME NO HIGH MEMORY
        JSR PC,PRMESS ;PRINT THE MESSAGE BUFFER ALSO
        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
; -
```



```

2342
2343 011550          BGNMSG  PKTGETS
      011550          PKTGETS::
2344 011550 004737 005264      JSR    PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
2345 011554 012700 000002      MOV    #2,R0          ;NO. OF WORDS IN GET STATUS PACKET
2346 011560 004737 006712      JSR    PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
2347 011564          ENDMSG
      011564
      011564 104423
2348
2349
2350
2351          ;*
2352          ;PRINT TSSR ERRORS FOR INITIALIZATION TESTS
2353          ;
2354          ;INPUTS:
2355          ;
2356          ;       R1      TSSR CONTENTS
2357          ;       R4      ADDRESS OF COMMAND PACKET
2358          ;
2359 011566          BGNMSG  SFFMSG
      011566          SFFMSG::
2360 011566 004737 005264      JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
2361 011572          ENDMSG
      011572
      011572 104423
2362
2363          .SBTTL  PKTMES  - PRINT TSSR AND MESSAGE BUFFER
2364
2365          ;*
2366          ;PRINT ROUTINE TO PRINT THE CONTENTS OF TSSR AND MESSAGE
2367          ;BUFFER FOR ERROR REPORTS
2368          ;
2369          ;INPUTS:
2370          ;
2371          ;       R1      CONTENTS OF TSSR
2372          ;       R2      LOW ORDER MESSAGE BUFFER
2373          ;       R3      HIGH ORDER MESSAGE BUFFER ADDRESS
2374          ;       NOTE: R3 IS IGNORED IF KTENABLE FLAG IS CLEAR
2375          ;
2376          ;
2377 011574          BGNMSG  PKTMES
      011574          PKTMES::
2378 011574 004737 005264      JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR
2379 011600 010200          MOV    R2,R0          ;LOW ORDER ADDRESS
2380 011602 010301          MOV    R3,R1          ;HIGH ORDER ADDRESS
2381 011604 004737 013702      JSR    PC,PRMESS      ;PRINT THE MESSAGE BUFFER
2382 011610          ENDMSG
      011610
      011610 104423
2383

```

2385
 2386
 2387
 2388
 2389
 2390
 2391
 2392
 2393
 2394
 2395
 2396
 2397 011612
 011612
 2398 011612 004737 007646
 2399 011616 016501 000000
 2400 011622 004737 005264
 2401 011626
 011626
 011626 104423
 2402
 2403
 2404
 2405
 2406
 2407
 2408
 2409
 2410
 2411
 2412
 2413
 2414
 2415
 2416 011630
 011630
 2417 011630 012700 000007
 2418 011634 004737 015246
 2419 011640
 011640
 011640 104423
 2420
 2421

```

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

```

        .SBTTL  MSGEXP - PRINT WRITE CHAR. EXPD-RECV MESSAGE BUFFERS
        ;*
        ;PRINT ROUTINE TO PRINT WRITE CHARACTERISTIC MESSAGE BUFFER
        ;
        ;IMPLICIT INPUTS:
        ;
        ;       EXPMSG  - EXPECTED MESSAGE BUFFER
        ;       RECMSG  - RECEIVED MESSAGE BUFFER
        ;       RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
        ;       RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
        ;
        ;-
        BGNMSG  MSGEXP
    MSGEXP::
        MOV    #7,R0          ;ASSUME NO EXT FEATURES
        JSR    PC,PRMSGEXP    ;PRINT EXPD/RECV MESSAGE BUFFERS
        ENDMSG
    L10011:
        TRAP   C$MSG
    
```



```

2423          .SBTTL  FIFEXP  - PRINT FIFO EXP/RECV DATA
2424
2425          ;*
2426          ;PRINT ROUTINE TO PRINT FIFO EXP/RECV DATA
2427          ;
2428          ;       R1      - BYTE COUNT
2429          ;
2430          ;IMPLICIT INPUTS:
2431          ;
2432          ;       EXPMSG  - EXPECTED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
2433          ;       RECMSG  - RECEIVED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
2434          ;-
2435          BGNMSG  FIFEXP
2436          FIFEXP::
2437          PRINTX  #FIF1MSG,R1      ;PRINT BYTES TRANSFERRED
2438          MOV     R1,-(SP)
2439          MOV     #FIF1MSG,-(SP)
2440          MOV     #2,-(SP)
2441          MOV     SP,R0
2442          TRAP   C$PNTX
2443          ADD     #6,SP
2444          PRINTX  #FIF2MSG          ;PRINT HEADER MSG
2445          MOV     #FIF2MSG,-(SP)
2446          MOV     #1,-(SP)
2447          MOV     SP,R0
2448          TRAP   C$PNTX
2449          ADD     #4,SP
2450          MOV     R1,R0              ;GET BYTE COUNT
2451          JSR    PC,PRBYTEXP        ;PRINT FIFO BYTES IN ERROR
2452          ENDMSG
2453          L10012:
2454          TRAP   C$MSG
2455          .ASCIZ '  #N#A NUMBER OF BYTES TRANSFERRED = #D2'
2456          .ASCIZ '  #N#A FIFO DATA BYTES IN ERROR:'
2457          .EVEN
  
```

```

2446 .SBTTL MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS
2447 ;*
2448 ;
2449 ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RECV
2450 ;
2451 ;
2452 ;IMPLICIT INPUTS:
2453 ;
2454 ; EXPMSG - EXPECTED MESSAGE BUFFER
2455 ; RECMSG - RECEIVED MESSAGE BUFFER
2456 ; RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2457 ; RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2458 ;-
2459 012022 BGNMSG MSGSTAT
012022 MSGSTAT:
2460 012022 012701 012064 MOV #STATCOD,R1 ;ASCII ADDRESS TABLE
2461 012026 012100 10$: MOV (R1),R0 ;DONE ALL MSG LINES?
2462 012030 001410 BEQ 20$ ;BR IF YES
2463 012032 PRINTX R0 ;PRINT STATUS BIT NAMES
012032 010046 MOV R0,-(SP)
012034 012746 000001 MOV #1,-(SP)
012040 010600 MOV SP,R0
012042 104415 TRAP C$PNTX
012044 062706 000004 ADD #4,SP
2464 012050 000766 BR 10$ ;DO ANOTHER MSG LINE
2465 012052 012700 000012 20$: MOV #10,R0 ;NUMBER OF WORDS IN A READ STATUS BUFFER
2466 012056 004737 015246 JSR PC,PRMSGEXP ;PRINT EXPD/RECV MESSAGE BUFFERS
2467 012062 ENDMMSG
012062 104423 L10013: TRAP C$MSG
2468
2469 012064 012102 012144 012235 STATCOD: .WORD 1$,2$,3$,4$,5$,6$,0
2470 012102 045 116 045 1$: .ASCIZ 'N$A Tape Bus Signals in Word #8:'
2471 012144 045 116 045 2$: .ASCIZ 'N$A PARERR<15> IEOT <12> IFMK <9> IRDY<6> IRWD<2>'
2472 012235 045 116 045 3$: .ASCIZ 'N$A IRESV2<14> IIDENT<11> IHER <8> IONL<5> IFBY<1>'
2473 012326 045 116 045 4$: .ASCIZ 'N$A IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>'
2474 012417 045 116 045 5$: .ASCIZ 'N$A Tape Bus Signals in Word #9:'
2475 012461 045 116 045 6$: .ASCIZ 'N$A DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>'
2476 .EVEN
2477
2478
2479
2480 .SBTTL MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS
2481 ;*
2482 ;
2483 ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RECV
2484 ;
2485 ;IMPLICIT INPUTS:
2486 ;
2487 ; EXPMSG - EXPECTED MESSAGE BUFFER
2488 ; RECMSG - RECEIVED MESSAGE BUFFER
2489 ; RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2490 ; RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2491 ;-
2492 012536 BGNMSG MSGLOOP
012536 MSGLOOP:
2493 012536 012701 012600 MOV #LOOPCOD,R1 ;ASCII ADDRESS TABLE

```



```

2494 012542 012100          10$:  MOV    (R1)+,R0      ;DONE ALL MSG LINES?
2495 012544 001410          BEQ    20$              ;BR IF YES
2496 012546          PRINTX  R0              ;PRINT STATUS BIT NAMES
      012546 010046          MOV    R0,-(SP)
      012550 012746 000001          MOV    #1,-(SP)
      012554 010600          MOV    SP,R0
      012556 104415          TRAP  C$PNTX
      012560 062706 000004          ADD    #4,SP
2497 012564 000766          BR     10$              ;DO ANOTHER MSG LINE
2498 012566 012700 000012          20$:  MOV    #10,R0        ;NUMBER OF WORDS IN A READ STATUS BUFFER
2499 012572 004737 015246          JSR   PC,PRMSGEXP     ;PRINT EXPD/RECV MESSAGE BUFFERS
2500 012576          ENDMSG
      012576          L10014:
      012576 104423          TRAP  C$MSG
2501
2502 012600 012620 012673 012772 LOOPCOD: .WORD 1$,2$,3$,4$,5$,6$,7$,0
2503 012620          045 116 045 1$: .ASCIZ 'N$A Tape Bus Loopback Signals in Word #8:'
2504 012673          045 116 045 2$: .ASCIZ 'N$A PARERR<15> IRESV2<14> IRESV1<13>'
2505 012772          045 116 045 3$: .ASCIZ 'N$A IHISP=>IEOT<12> IWRT=>IIDENT<11> IREV =>ICER <10>'
2506 013071          045 116 045 4$: .ASCIZ 'N$A IWM =>IFMK<09> IEDIT=>IHER <08> IFAD =>ISPEED<07>'
2507 013170          045 116 045 5$: .ASCIZ 'N$A ITADO=>IRDY<06> ITAD1=>IONL <05> IERASE=>ILDPA <04>'
2508 013267          045 116 045 6$: .ASCIZ 'N$A IREW =>IDBY<03> IRWU =>IRWD <02> IFEN =>IFBY <01>'
2509 013366          045 116 045 7$: .ASCIZ 'N$A IGO =>IFPT<00>'
2510
2511          .EVEN

```

```

2513 .SBTTL MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER
2514 ;*
2515 ;
2516 ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
2517 ;
2518 ;
2519 ;IMPLICIT INPUTS:
2520 ;
2521 ; EXPMSG - EXPECTED MESSAGE BUFFER
2522 ; RECMMSG - RECEIVED MESSAGE BUFFER
2523 ; RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2524 ; RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2525 ;
2526 ;-
      BGNMSG MSGSUB
MSGSUB::
2527 013414 012700 000012   MOV    #10.,R0      ;SIZE OF WRITE SUBSYSTEM BUFFER
2528 013420 004737 015246   JSR    PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
2529 013424                                     ENDMSG
      L10015:
      TRAP  C$MSG

2530
2531
2532
2533
2534
2535 .SBTTL MEMADD - PRINT MEMORY ADDRESS DATA ERROR
2536 ;*
2537 ;
2538 ;PRINT ROUTINE TO PRINT MEMORY ADDRESS DATA COMPARE ERROR
2539 ;
2540 ;IMPLICIT INPUTS:
2541 ;
2542 ; ERRHI - MEMORY ERROR HIGH ORDER ADDRESS
2543 ; ERRLO - MEMORY ERROR LOW ORDER ADDRESS
2544 ; EXP - EXPECTED DATA
2545 ; RECV - RECEIVED DATA
2546 ;
2547 ;-
      BGNMSG MEMADD
MEMADD::
2548 013426 004737 007532   JSR    PC,PRIADD   ;PRINT MEMORY ADDRESS IN ERROR
2549 013432 013701 002176   MOV    EXPD,R1     ;GET EXPD DATA
2550 013436 013702 002200   MOV    RECV,R2    ;GET RECEIVED DATA
2551 013442 004737 007314   JSR    PC,PRIXOR   ;PRINT EXPD/RCV
2552 013446                                     ENDMSG
      L10016:
      TRAP  C$MSG
2553

```



```

2555 .SBTTL PRAMPKT - PRINT RAM AND PACKET DATA
2556 ;*
2557 ;
2558 ;PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
2559 ;WHEN THE RAM DATA DOES NOT MATCH.
2560 ;
2561 ;INPUTS:
2562 ;
2563 ; R4 POINTER TO COMMAND PACKET
2564 ;
2565 ;IMPLICIT INPUTS:
2566 ;
2567 ; RAMDATA DATA AS READ FROM THE RAM
2568 ; RAMSIZ NUMBER OF BYTES IN PACKET
2569 ; IF RAMSIZ=0 THEN DEFAULT TO 8.
2570 ;
2571 ;IMPLICIT OUTPUTS:
2572 ;
2573 ; RAMSIZ SET TO 0
2574 ;-
2575
2576 013450 PRAMPKT:
2577 013450 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
2578 013454 012701 002206 MOV #RAMDATA,R1 ;DATA FROM THE RAM
2579 013460 005002 CLR R2 ;INIT BYTE NUMBER
2580 013462 122124 5$: CMPB (R1)+,(R4)+ ;COMPARE EXPECTED, RECEIVED
2581 013464 001000 BNE 7$ ;BR IF NO MATCH
2582 013466 116105 177777 7$: MOVB -1(R1),R5 ;GET RECV RAM DATA
2583 013472 116403 177777 MOVB -1(R4),R3 ;GET EXPD PACKET DATA
2584 013476 XOR R5,R3 ;XOR EXPD/RECV
2585 013506 042703 177400 BIC #177400,R3 ;LOW BYTE ONLY
2586 013512 116137 177777 002200 MOVB -1(R1),RECV ;GET RECEIVED RAM DATA
2587 013520 116437 177777 002176 MOVB -1(R4),EXPD ;GET EXPECTED RAM DATA
2588 013526 PRINTB #RAMASC,R2,RECV,EXPD,R3
2589 013526 010346 MOV R3,-(SP)
2590 013564 005737 002246 10$: INC R2 ;UPDATE BYTE COUNT
2591 013570 001404 BEQ 15$ ;DEFAULT TO 8.?
2592 013572 020237 002246 CMP R2,RAMSIZ ;BR IF YES
2593 013576 003731 BLE 5$ ;DONE ALL BYTES?
2594 013600 000403 BR 25$ ;BR IF NO
2595 013602 020227 000010 15$: CMP R2,#8. ;DONE DEFAULT NUMBER OF BYTES?
2596 013606 002725 20$: BLT 5$ ;BR IF NO
2597 013610 005037 002246 25$: CLR RAMSIZ ;SET DEFAULT RAMSIZ
2598 013614 000207 RTS PC ;RETURN
2599
2600 013616 045 116 045 RAMASC: .ASCIZ 'N#A BYTE: #D2#A RAM: #03#A Packet: #03#A XOR:#03'
2601 .EVEN

```

```

2603          .SBTTL  PRMESS  - PRINT CONTENTS OF MESSAGE BUFFER
2604          ;*
2605          ;
2606          ; THIS ROUTINE PRINTS THE CONTENTS OF
2607          ; THE 7 WORD MESSAGE BUFFER RETURNED BY THE
2608          ; TK-25.
2609          ;
2610          ; INPUT:
2611          ;
2612          ;     R0     LOW ORDER ADDRESS OF MESSAGE BUFFER
2613          ;     R1     HIGH ORDER ADDRESS OF MESSAGE BUFFER
2614          ;     NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
2615          ;
2616          ; THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
2617          ;
2618          ; -
2619
2620 013702     PRMESS:
2621 013702     SAVREG          ;SAVE THE REGISTERS
2622 013706     010537 010642   MOV     R5,RAMR5H        ;SAVE DEVICE REGISTER POINTER
2623 013712     010005         MOV     R0,R5          ;SAVE LOW ORDER ADDRESS
2624 013714     005737 003102   TST     KTENABLE        ;ADDRESS ABOVE 28K?
2625 013720     001001         BNE     10$          ;BR IF YES
2626 013722     005001         CLR     R1          ;SET HIGH ORDER ADDRESS TO 0
2627 013724     010103 10$:   MOV     R1,R3          ;SAVE HIGH ORDER ADDRESS
2628 013726     006100         ROL     R0          ;SHIFT BIT15 TO C BIT
2629 013730     006101         ROL     R1          ;SHIFT TO HIGH ORDER FOR PRINTOUT
2630 013732     013732         PRINTX  @PROASC,R1,R5    ;PRINT MESSAGE BUFFER ADDRESS
2631 013732     010546         MOV     R5,-(SP)
2632 013734     010146         MOV     R1,-(SP)
2633 013736     012746 014540   MOV     @PROASC,-(SP)
2634 013742     012746 000003   MOV     @3,-(SP)
2635 013746     010600         MOV     SP,R0
2636 013750     104415         TRAP   C$PNTX
2637 013752     062706 000010   ADD     @10,SP
2638 013756     022715 177777   CMP     @177777,(R5)    ;MESSAGE BUFFER FULL OF ONES
2639 013762     001010         BNE     15$          ;BR IF BUFFER IS PROBABLY OKAY
2640 013764     013764         PRINTX  @MESBFN        ;"MESSAGE BUFFER PROBABLY NOT VALID"
2641 013764     012746 014460   MOV     @MESBFN,-(SP)
2642 013770     012746 000001   MOV     @1,-(SP)
2643 013774     010600         MOV     SP,R0
2644 013776     104415         TRAP   C$PNTX
2645 014000     062706 000004   ADD     @4,SP
2646 014004     014004 15$:   PRINTX  @PR1ASC        ;PRINT HEADER FOR CONTENTS
2647 014004     012746 014605   MOV     @PR1ASC,-(SP)
2648 014010     012746 000001   MOV     @1,-(SP)
2649 014014     010600         MOV     SP,R0
2650 014016     104415         TRAP   C$PNTX
2651 014020     062706 000004   ADD     @4,SP
2652 014024     005004         CLR     R4          ;NUMBER OF THE NEXT WORD
2653 014026     010501         MOV     R5,R1        ;COPY LOW ORDER ADDRESS
2654 014030     010300         MOV     R3,R0        ;COPY HIGH ORDER ADDRESS
2655 014032     001403         BEQ    20$          ;BR IF NOT ABOVE 28K
2656 014034     004737 020112   JSR    PC,SETMAP      ;SETUP PAR ADDRESS IN R0
2657 014040     010005         MOV     R0,R5        ;GET PAR FORMAT ADDRESS ABOVE 28K
2658 014042     014042 20$:   PRINTX  @MESHEA,(R5)+ ;PRINT "MESSAGE BUFFER HEADER ="
2659 014042

```


	014042	012546		MOV	(R5)+,-(SP)	
	014044	012746	014643	MOV	#MESH EA,-(SP)	
	014050	012746	000002	MOV	#2,-(SP)	
	014054	010600		MOV	SP,R0	
	014056	104415		TRAP	C#PNTX	
	014060	062706	000006	ADD	#6,SP	
2643	014064			PRINTX	#DATAFL,(R5)+	;PRINT "DATA FIELD LENGTH ="
	014064	012546		MOV	(R5)+,-(SP)	
	014066	012746	014710	MOV	#DATAFL,-(SP)	
	014072	012746	000002	MOV	#2,-(SP)	
	014076	010600		MOV	SP,R0	
	014100	104415		TRAP	C#PNTX	
	014102	062706	000006	ADD	#6,SP	
2644	014106			PRINTX	#RBP CRA,(R5)+	;PRINT "RESIDUAL BYTE COUNTER ="
	014106	012546		MOV	(R5)+,-(SP)	
	014110	012746	014755	MOV	#RBP CRA,-(SP)	
	014114	012746	000002	MOV	#2,-(SP)	
	014120	010600		MOV	SP,R0	
	014122	104415		TRAP	C#PNTX	
	014124	062706	000006	ADD	#6,SP	
2645	014130			PRINTX	#XSOC ON,(R5)+	;PRINT "XSTAT0 CONTENTS ="
	014130	012546		MOV	(R5)+,-(SP)	
	014132	012746	015022	MOV	#XSOC ON,-(SP)	
	014136	012746	000002	MOV	#2,-(SP)	
	014142	010600		MOV	SP,R0	
	014144	104415		TRAP	C#PNTX	
	014146	062706	000006	ADD	#6,SP	
2646	014152			PRINTX	#XS1CON,(R5)+	;PRINT "XSTAT1 CONTENTS ="
	014152	012546		MOV	(R5)+,-(SP)	
	014154	012746	015067	MOV	#XS1CON,-(SP)	
	014160	012746	000002	MOV	#2,-(SP)	
	014164	010600		MOV	SP,R0	
	014166	104415		TRAP	C#PNTX	
	014170	062706	000006	ADD	#6,SP	
2647	014174			PRINTX	#XS2CON,(R5)+	;PRINT "XSTAT2 CONTENTS ="
	014174	012546		MOV	(R5)+,-(SP)	
	014176	012746	015134	MOV	#XS2CON,-(SP)	
	014202	012746	000002	MOV	#2,-(SP)	
	014206	010600		MOV	SP,R0	
	014210	104415		TRAP	C#PNTX	
	014212	062706	000006	ADD	#6,SP	
2648	014216			PRINTX	#XS3CON,(R5)+	;PRINT "XSTAT3 CONTENTS ="
	014216	012546		MOV	(R5)+,-(SP)	
	014220	012746	015201	MOV	#XS3CON,-(SP)	
	014224	012746	000002	MOV	#2,-(SP)	
	014230	010600		MOV	SP,R0	
	014232	104415		TRAP	C#PNTX	
	014234	062706	000006	ADD	#6,SP	
2649	014240	022737	000001	CMP	#1,TRANSTST	;CHECK SOFTWARE P-TABLE
2650	014246	001402		BEQ	40\$;DO DUMP
2651	014250	000137	014360	JMP	50\$;DON'T DO THE DUMP
2652	014254			PRINTX	#RAMFHR	
	014254	012746	014362	MOV	#RAMFHR,-(SP)	
	014260	012746	000001	MOV	#1,-(SP)	
	014264	010600		MOV	SP,R0	
	014266	104415		TRAP	C#PNTX	
	014270	062706	000004	ADD	#4,SP	

002134

40\$:

```

2653 014274 012737 000010 002246      MOV      #8.,RAMSIZ      ;RAM FIELD IS 8 BYTES LONG
2654 014302 012737 000020 010640      MOV      #20,RAMHLD     ;FIELD STARTS AT 20 OCTAL (10 HEX)
2655 014310 004737 010456                JSR      PC,RAMER       ;READ AND PRINT THEM
2656 014314 012737 000040 010640      MOV      #40,RAMHLD     ;FIELD STARTS AT 40 OCTAL (20 HEX)
2657 014322 004737 010456                JSR      PC,RAMER       ;READ AND PRINT THEM
2658 014326 012737 000060 010640      MOV      #60,RAMHLD     ;FIELD STARTS AT 60 OCTAL (30 HEX)
2659 014334 004737 010456                JSR      PC,RAMER       ;READ AND PRINT THEM
2660 014340 012737 000020 002246      MOV      #16.,RAMSIZ    ;RAM FIELD IS SIXTEEN BYTES LONG
2661 014346 012737 000100 010640      MOV      #100,RAMHLD    ;FIELD STARTS AT 100 OCTAL (40 HEX)
2662 014354 004737 010456                JSR      PC,RAMER       ;READ AND PRINT THEM
2663 014360 000207                50$:      RTS              ;RETURN
2664 014362      045      116      045  RAMFHR: .ASCIZ    '##A ***** SPECIAL CONTROLLER RAM MEMORY DUMP *****'
2665 014460      045      116      045  MESBFN: .ASCIZ    '##A MESSAGE BUFFER CONTENTS PROBABLY NOT VALID'
2666 014540      045      116      045  PROASC: .ASCIZ    '##A Message Buffer Address = #01#05'
2667 014605      045      116      045  PR1ASC: .ASCIZ    '##A Message Buffer Contents:'
2668
2669 014643      045      116      045  MESHEA: .ASCIZ    '##A Message Buffer Header      = #06'
2670 014710      045      116      045  DATAFL: .ASCIZ   '##A Data Field Length        = #06'
2671 014755      045      116      045  RBPCRA: .ASCIZ    '##A Residual Byte Counter    = #06'
2672 015022      045      116      045  XSOCON: .ASCIZ    '##A XSTAT0 Contents          = #06'
2673 015067      045      116      045  XS1CON: .ASCIZ    '##A XSTAT1 Contents          = #06'
2674 015134      045      116      045  XS2CON: .ASCIZ    '##A XSTAT2 Contents          = #06'
2675 015201      045      116      045  XS3CON: .ASCIZ    '##A XSTAT3 Contents          = #06'
2676      .EVEN
    
```



```

2678 .SBTTL PRMSGEXP - PRINT EXPD/RCV MESSAGE BUFFERS
2679 ;*(B
2680 ;ROUTINE TO PRINT EXPECTED AND RECEIVED MESSAGE BUFFERS
2681 ;
2682 ; RO - NUMBER OF WORDS IN BUFFER
2683 ;
2684 ;IMPLICIT INPUTS:
2685 ;
2686 ; EXPMSG - EXPECTED MESSAGE BUFFER
2687 ; RECMMSG - RECEIVED MESSAGE BUFFER
2688 ; RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2689 ; RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2690 ;
2691 ;-
2692 PRMSGEXP::
2693 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
2694 MOV RO,R5 ;SAVE NUMBER OF WORDS
2695 MOV RCVLOADD,RO ;GET RECV LOW ADDRESS
2696 MOV RO,R4 ;COPY LOW ADDRESS
2697 MOV RCVHIADD,R1 ;GET RECV HIGH ADDRESS
2698 ROL RO ;SHIFT BIT15 TO C BIT
2699 ROL R1 ;SHIFT TO HIGH ORDER FOR PRINTOUT
2700 PRINTX #PRMSGO,R1,R4 ;PRINT MESSAGE BUFFER ADDRESS
2701 MOV R4,-(SP)
2702 MOV R1,-(SP)
2703 MOV #PRMSGO,-(SP)
2704 MOV #3,-(SP)
2705 MOV SP,RO
2706 TRAP C#PNTX
2707 ADD #10,SP
2708 PRINTX #PRMSG1 ;PRINT HEADER FOR CONTENTS
2709 MOV #PRMSG1,-(SP)
2710 MOV #1,-(SP)
2711 MOV SP,RO
2712 TRAP C#PNTX
2713 ADD #4,SP
2714 CLR R4 ;NUMBER OF THE CURRENT WORD
2715 MOV #EXPMSG,R1 ;GET EXPD BUFFER ADDRESS
2716 MOV #RECMMSG,R2 ;GET RECV BUFFER ADDRESS
2717 MOV (R1),RO ;GET EXPD
2718 MOV (R2),R3 ;GET RECV
2719 XOR RO,R3 ;XOR EXPD/RCV
2720 PRINTX #PRMSG2,R4,(R1),,(R2),,R3
2721 MOV R3,-(SP)
2722 MOV (R2),,-(SP)
2723 MOV (R1),,-(SP)
2724 MOV R4,-(SP)
2725 MOV #PRMSG2,-(SP)
2726 MOV #5,-(SP)
2727 MOV SP,RO
2728 TRAP C#PNTX
2729 ADD #14,SP
2730 INC R4 ;NUMBER OF THE NEXT
2731 CMP R4,R5 ;DONE ALL YET?
2732 BGE 50$ ;BR IF YES
2733 BR 20$ ;DO ANOTHER
2734 RTS PC ;RETURN

```

2714
2715 015426 045 116 045 PRMSG0: .ASCIZ 'N#A Message Buffer Address - #01#05'
2716 015473 045 116 045 PRMSG1: .ASCIZ 'N#A Message Buffer Contents:'
2717 015531 045 116 045 PRMSG2: .ASCIZ 'N#A WORD #D2#A EXPD: #06#A RECV: #06#A XOR: #06#A'
2718 .EVEN
2719


```

2721                .SBTTL PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER
2722                ;*
2723                ;
2724                ;ROUTINE TO PRINT ERROR BYTES IN MESSAGE BUFFERS
2725                ; ONLY THE FIRST 8 ERRORS ENCOUNTERED ARE PRINTED DUE TO SCREEN SPACE
2726                ;
2727                ; RO - NUMBER OF BYTES IN BUFFER
2728                ;
2729                ;IMPLICIT INPUTS:
2730                ;
2731                ; EXPMSG - EXPECTED MESSAGE BUFFER
2732                ; RECMMSG - RECEIVED MESSAGE BUFFER
2733                ;
2734                PRBYTEXP::
2735                SAVREG                                ;SAVE R1-R5 UNTIL NEXT RETURN
2736                MOV R0,R5                            ;SAVE NUMBER OF BYTES
2737                CLR PRMNO                             ;INIT ERROR COUNT
2738                CLR R4                               ;NUMBER OF THE CURRENT BYTE
2739                MOV @EXPMSG,R1                       ;GET EXPD BUFFER ADDRESS
2740                MOV @RECMMSG,R2                     ;GET RECV BUFFER ADDRESS
2741                20$: MOVB (R1),R0                   ;GET EXPD BYTE
2742                BIC @+C<377>,R0                    ;CLEAR UPPER BYTE
2743                MOVB R0,PRBEXP                      ;SAVE FOR ERROR REPORT
2744                MOVB (R2),R3                       ;GET RECV BYTE
2745                BIC @+C<377>,R3                    ;CLEAR UPPER BYTE
2746                MOVB R3,PRBREC                    ;FOR ERROR REPORT
2747                XOR R0,R3                          ;XOR EXPD/RECV
2748                CMPB (R1)+,(R2)+                  ;EXPD = RECV?
2749                BEQ 30$                             ;BR IF YES
2750                INC PRMNO                          ;UPDATE ERROR COUNT
2751                CMP PRMNO,#8.                      ;PRINTED 8?
2752                BHI 30$                             ;BR IF YES
2753                27$: PRINTX @PRBMSG,R4,PRBEXP,PRBREC,R3
2754                MOV R3,-(SP)
2755                MOV PRBREC,-(SP)
2756                MOV PRBEXP,-(SP)
2757                MOV R4,-(SP)
2758                MOV @PRBMSG,-(SP)
2759                MOV #5,-(SP)
2760                MOV SP,R0
2761                TRAP C:PNTX
2762                ADD #14,SP
2763                FORCEEXIT 50$                        ;@@D
2764                BR 35$                               ;@D
2765                30$: FORCERROR 27$,NOTSSR          ;@D
2766                35$:                                ;@D
2767                INC R4                               ;NUMBER OF THE NEXT
2768                CMP R4,R5                          ;DONE ALL YET?
2769                BGE 50$                             ;BR IF YES
2770                BR 20$                              ;DO ANOTHER
2771                50$: PRINTX @PRBTOT,PRMNO          ;PRINT TOTAL ERROR COUNT
2772                MOV PRMNO,-(SP)
2773                MOV @PRBTOT,-(SP)
2774                MOV #2,-(SP)
2775                MOV SP,R0
2776                TRAP C:PNTX
    
```

E7

2764	016024	062706	000006		ADD	#6,SP	
	016030	000207			RTS	PC	;RETURN
2765							
2766	016032	045	116	045	PRBMSG: .ASCIZ	'#N#A BYTE #D2#A EXPD: #03#A RECV: #03#A XOR: #03'	
2767	016117	045	116	045	PRBTOT: .ASCIZ	'#N#A NUMBER OF BYTES IN ERROR = #D2'	
2768					.EVEN		
2769	016164	000000			PRBEXP: .WORD	0	;EXPD
2770	016166	000000			PRBREC: .WORD	0	;RECV
2771							


```

2773 .SBTTL EXPREC - PRINT EXPD/RECV WORD DATA
2774 ;*
2775 ;
2776 ;PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
2777 ;
2778 ;INPUTS:
2779 ;
2780 ; R1 RECEIVED DATA
2781 ; R2 EXPECTED DATA
2782 ;
2783 ;-
2784
2785 016170 BGNMSG EXPREC
016170 EXPREC::
2786 016170 004737 007314 JSR PC,PRIXOR ;PRINT THE DATA
2787 016174 ENDMSG
016174 L10017:
016174 104423 TRAP C$MSG
2788
2789

```

2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804 016176
016176
2805 016176 004737 007164
2806 016202
016202
016202 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
          ENDMSG
L10020: TRAP   C#MSG

```

2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831 016204
016204
2832 016204 004737 013450
2833 016210
016210
016210 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
          ENDMSG
L10021: TRAP   C#MSG

```

2834
2835
2836
2837
2838
2839
2840
2841

```

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

```



```

2842
2843          :          R4          POINTER TO COMMAND PACKET
2844          :
2845          :IMPLICIT INPUTS:
2846          :
2847          :          RAMDATA      DATA AS READ FROM THE RAM
2848          :          RAMSIZ      NUMBER OF BYTES IN PACKET
2849          :                      IF RAMSIZ=0 THEN DEFAULT TO 8.
2850          :          ERRHI      HIGH ORDER TEST ADDRESS
2851          :          ERRLO      LOW ORDER TEST ADDRESS
2852          :
2853          :IMPLICIT OUTPUTS:
2854          :
2855          :          RAMSIZ      SET TO 0
2856          :
2857          :
2858          :          BGNMSG      RAMTADD
2859          :          RAMTADD: :
2859          :          JSR          PC,PRITADD      ;PRINT TEST ADDRESS
2860          :          JSR          PC,PRAMPKT     ;PRINT RAM/PACKET DATA
2861          :          ENDMSG
2862          :
2863          :          L10022:
2864          :          TRAP          C$MSG
2865          :
2866          :          .SBTTL      RAMEXP - PRINT RAM EXPD/RECV DATA
2867          :
2868          :          ;*
2869          :          ;PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
2870          :
2871          :          ;INPUTS:
2872          :
2873          :          R1          RECEIVED DATA
2874          :          R2          EXPECTED DATA
2875          :          R4          CONTROLLER RAM ADDRESS
2876          :
2877          :          BGNMSG      RAMEXP
2878          :          RAMEXP: :
2878          :          BIC          @+C<377>,R1      ;SAVE EXPD RAM DATA BYTE
2879          :          BIC          @+C<377>,R2      ;SAVE EXPD RAM DATA BYTE
2880          :          JSR          PC,PRIRAM      ;PRINT THE RAM ADDRESS
2881          :          JSR          PC,PRIXOR     ;PRINT THE DATA
2882          :          ENDMSG
2883          :          L10023:
2884          :          TRAP          C$MSG
2885          :
2886          :          .SBTTL      TIMEXP - PRINT TIMER A,B AND EXP/REC
2887          :
2888          :          ;*
2889          :          ;PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
2890          :          ;AND TIMER A,B HEADER MESSAGE
2891          :
2892          :          ;INPUTS:
2893          :
2894          :          R1          RECEIVED DATA
2895          :          R2          EXPECTED DATA
    
```

```

2893
2894
2895 016246          BGNMSG  TIMEXP
      016246          TIMEXP::
2896 016246          PRINTX  #TIMSGO      ;PRINT HEADER
      016246 012746 016274      MOV      #TIMSGO, -(SP)
      016252 012746 000001      MOV      #1, -(SP)
      016256 010600      MOV      SP, RO
      016260 104415      TRAP     C$PNTX
      016262 062706 000004      ADD      #4, SP
2897 016266 004737 007314      JSR     PC, PR XOR      ;PRINT THE DATA
2898 016272          ENDMMSG
      016272          L10024:
      016272 104423      TRAP     C$MSG
2899
2900
2901 016274          045      116      045  TIMSGO: .ASCIZ  'N/A TIMER A STATUS IS IN BIT 3N/A TIMER B STATUS IS IN BIT 2'
2902          .EVEN

```



```

2904 .SBTTL BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS
2905
2906
2907
2908 ;*
2909 ;PRINT ROUTINE FOR TSSR ERRORS ON DATA TRANSFERS
2910 ;INPUTS:
2911 ;
2912 ; R1 CONTENTS OF TSSR
2913 ; R2 DATA WRITTEN (8 BITS)
2914 ;
2915 ;-
2916
2917 016374 BGNMSG BADSSR
016374 BADSSR:
2918 016374 010246 MOV R2, -(SP) ;SAVE DATA TRANSFERRED
2919 016376 042702 177400 BIC #177400,R2 ;GET JUST ONE BYTE
2920 016402 PRINTB #XFERASC,R2
016402 010246 MOV R2, -(SP)
016404 012746 016434 MOV #XFERASC, -(SP)
016410 012746 000002 MOV #2, -(SP)
016414 010600 MOV SP,R0
016416 104414 TRAP C:PNTB
016420 062706 000006 ADD #6,SP
2921 016424 012602 MOV (SP)+,R2 ;RESTORE R2
2922 016426 004737 005264 JSR PC,PRITSSR ;DECODE TSSR CONTENTS
2923 016432 ENDMSG
016432 L10025:
016432 104423 TRAP C:MSG
2924 016434 045 116 045 XFERASC: .ASCIZ '#N#A Data Transferred = #03'
2925

```

```

2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960 016470
2961 016470
2962 016474 012765 000000 000000
2963 016502 004737 016744
2964 016506 016500 000000
2965 016512 010004
2966 016514 042704 176277
2967 016520 052704 002200
2968 016524 020400
2969 016526 001402
2970 016530 000241
2971 016532 000401
2972 016534 000261
2973 016536 000207

```

```

.SBTTL GLOBAL SUBROUTINES SECTION
;+
; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
; THAT ARE USED IN MORE THAN ONE TEST.
;--
.SBTTL SOFINIT - SOFT INITIALIZE OF CONTROLLER
;+
;
;ROUTINE TO DO A SOFT INITIALIZE OF THE CONTROLLER
;BY WRITING INTO THE TSSR REGISTER. AFTER THE INIT,
;THE TSSR REGISTER IS TESTED FOR ERRORS. ANY ERRORS
;DETECTED SHOULD BE TREATED AS DEVICE FATAL ERRORS.
;
;INPUTS:
;
;      R5      ADDRESS OF FIRST REGISTER
;
;OUTPUTS:
;
;      R0      CONTENTS OF TSSR, IF ERROR
;      CARRY   SET IF INIT WAS OKAY
;              CLEAR IF FATAL ERROR
;
;CALLING SEQUENCE:
;
;      MOV     #ADDRESS,R5
;      JSR     PC,SOFINIT
;      BCS     CONTINUE
;      ERDF                    ;REPORT FATAL ERROR
;
;--
SOFINIT::
; SAVREG                    ; SAVE THE REGISTERS
MOV     #0,TSSR(R5)        ; DO THE INIT.
JSR     PC,WAITF           ; WAIT FOR SSR
MOV     TSSR(R5),R0        ;GET THE TSSR REGISTER
MOV     R0,R4              ;START SETUP OF EXPECTED TSSR
BIC     #C<HIADDR!OFL>,R4 ;CLEAR OUT UNUSED BITS
BIS     #SSR!NBA,R4        ;R4 HAS EXPECTED CONTENTS
CMP     R4,R0              ;ONLY EXPECTED BITS SET ?
BEQ     5$                 ;BRANCH IF OKAY
CLC                    ;CLEAR THE CARRY FOR ERROR
BR      10$                ;GO TO EXIT
5$: SEC                    ;SET THE CARRY BIT
10$: RTS PC                ;RETURN TO CALLER

```


2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019

```

.SBTTL  CHKAMB - CHECK TSSR FOR AMBIGUITY

;*
;
; THIS ROUTINE TESTS THE CONTENTS OF THE TSSR REGISTER
; FOR AMBIGUITY
;
; INPUT:
;
;      RO      CONTENTS OF TSSR
;
; OUTPUT:
;
;      RO      CONTENTS OF TSSR
;
;      CARRY   SET - NO AMBIGUITY
;              CLR - AMBIGUOUS CONTENTS
;
;-

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

```

010004
032700 100000
001004
032700 174077
001023
000424
032700 000200
001011
032700 000040
001414
042704 177761
020427 000016
001007
000410
032700 000040
001405
032700 000006
001002
000241
000401
000261
000207

```

3021          .SBTTL ENAIN,DSBINT - ENABLE/DISABLE INTERRUPTS
3022          ;
3023          ; DEFAULT DISPLAY INTERRUPT HANDLERS.
3024          ; IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
3025          ; OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
3026          ;
3027          ;
3028          ; BIT DEFINITIONS FOR "INTMASK" AND "INTFLAG" BYTES:
3029          ;
3030          000200          IOKCKIN=BIT7          ; DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
3031          000001          IOKSTP=BIT0          ; EXPECT "STOP" INTERRUPT.
3032          ;
3033          ; INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
3034          016640          000          INTMASK: .BYTE 0
3035          ; INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
3036          016641          000          INTFLAG: .BYTE 0
3037          ;
3038          ; SAVED INTERRUPT VECTOR:
3039          016642          000000          INTVEC: .WORD 0
3040          ; SAVE CPU PC
3041          016644          000000          INTCPC: .WORD 0
3042          ;
3043          ; SUBROUTINE TO ENABLE INTERRUPTS:
3044          016646          010046          ENAIN: MOV     RO,-(SP)          ;SAVE RO
3045          016650          013700          002156          MOV     IVEC,RO          ;GET POINTER TO VECTORS
3046          016654          012720          016712          MOV     @INTR,(RO)+          ;SET UP INTERRUPT VECTOR
3047          016660          012720          000340          MOV     @PRI07,(RO)+
3048          016664          012600          MOV     (SP)+,RO          ;RESTORE RO
3049          016666          011646          MOV     (SP),-(SP)
3050          016670          012766          000000          000002          MOV     @0,2(SP)          ;SET CPU TO LEVEL 0
3051          016676          000002          RTI
3052          ;
3053          ; SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 7)
3054          016700          011646          DSBINT: MOV     (SP),-(SP)
3055          016702          012766          000340          000002          MOV     @PRI07,2(SP)
3056          016710          000002          RTI
3057

```



```

3059          .SBTTL  INTR  - INTERRUPT HANDLERS
3060
3061 016712          BGNSRV  INTR          ;DEFINE INTERRUPT ENTRY
          016712          INTR::
3062 016712 012737 000001 002172          MOV      #1,INTRECV      ;SET FLAG TO SHOW INTERRUPT RECEIVED
3063 016720 105037 016641          CLR      INTFLAG        ;CLEAR FLAG TO SAY WE GOT INTERRUPT
3064 016724 132737 000001 016640          BIT      #IOKSTP,INTMASK ;EXPECTING STOP INTERRUPT?
3065 016732 001003          BNE      1$                ;BR IF YES
3066 016734 152737 00J001 016641          BIS      #IOKSTP,INTFLAG ;NO. SET THE ERROR FLAG.
3067
3068          ;SAVE REGISTERS, MSG BUFFER, ETC.
3069 016742          1$:
3070 016742          ENDSRV
          016742          L10026:
          016742 000002          RTI
3071
3072

```



```

3074 .SBTTL WAITF - WAIT FOR SUBSYSTEM READY
3075
3076 ; SUBROUTINE TO WAIT FOR THE SUBSYSTEM READY FLAG
3077 ;
3078 ; INPUTS:
3079 ;
3080 ; R5 ADDRESS OF FIRST DEVICE REGISTER
3081 ;
3082 ; OUTPUTS:
3083 ;
3084 ; R0 CONTENTS OF LAST TSSR READ
3085 ; CARRY SET - READY BIT SET
3086 ; CLR - TIMEOUT WAITING FOR READY
3087 ;
3088 016744 WAITF:: BREAK ; DO A SUPVSR BREAK FIRST.
016744 104422 TRAP C#BRK
3089 016746 012746 010000 MOV #10000,-(SP) ;BIG MSEC TIMER
3090 016752 DELAY 1 ;DELAY 100US
016752 012727 000001 MOV #1,(PC)
016756 000000 .WORD 0
016760 013727 002116 MOV L#DLY,(PC)
016764 000000 .WORD 0
016766 005367 177772 DEC -6(PC)
016772 001375 BNE .-4
016774 005367 177756 DEC -22(PC)
017000 001367 BNE .-20
3091 017002 016500 000000 2#: MOV TSSR(R5),R0 ;READ THE TSSR REGISTER
3092 017006 105700 TSTB R0 ;TEST FOR READY BIT SET
3093
3094 017010 100420 BMI 3# ; EXIT ON STOP FLAG.
3095 017012 DELAY 1 ; WAIT 100 USEC
017012 012727 000001 MOV #1,(PC)
017016 000000 .WORD 0
017020 013727 002116 MOV L#DLY,(PC)
017024 000000 .WORD 0
017026 005367 177772 DEC -6(PC)
017032 001375 BNE .-4
017034 005367 177756 DEC -22(PC)
017040 001367 BNE .-20
3096 017042 005316 DEC (SP) ;REDUCE DELAY COUNT
3097 017044 001356 BNE 2# ;RETRY UNTIL TIMER EXPIRES
3098 017046 000241 CLC ; C = 0, CONTROLLER STILL RUNNING...
3099 017050 000401 BR 4# ;...OR HUNG-UP AFTER 300 MSEC.
3100 017052 000261 3#: SEC ; C = 1, CONTROLLER IS STOPPED.
3101 017054 005326 4#: DEC (SP) ;RESTORE STACK WITHOUT CHANGING CARRY BIT
3102 017056 000207 RTS PC
    
```

3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135

017060
017060 004737 016744
017064 103014
017066 004737 016540
017072 103006
017074 032700 100000
017100 001405
017102 032700 074000
017106 001402
017110 000241
017112 000401
017114 000261
017116 000207

```

.SBTTL  CHK TSSR - 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 - OKAY
;         CLR - NOT READY AMBIGUOUS, OR SC SET
;
;
CHKTSSR:
    JSR    PC, WAITF      ;WAIT FOR READY
    BCC    20$           ;BRANCH IF TIME OUT
    JSR    PC, CHKAMB     ;TSSR AMBIGUOUS?
    BCC    10$           ;BR IF YES
    BIT    @SC, R0        ;SPECIAL CONDITION SET?
    BEQ    15$           ;BR IF NO
    BIT    @<SCE!BIE!RMR!NXM>, R0 ;ANY ERROR BITS SET?
    BEQ    15$           ;BR IF NO
10$:    CLC              ;SET FAILURE
        BR      20$      ;
15$:    SEC              ;SET SUCCESS
20$:    RTS              ;RETURN TO CALLER

```

```

3137          .SBTTL  XNXM  - CHECK FOR NONEXISTENT MEMORY
3138          ;*
3139          ; ROUTINE TO TEST FOR A NEXM IN THE RANGE (R1) THRU (R2).
3140          ; ON RETURN, IF "C" = 1, (R1) = NEXM ADDRESS.
3141          ;           "C" = 0, ALL ADDRESSES OK.
3142          ;
3143          ;CALL:  MOV  ADR1,R1
3144          ;         MOV  ADR2,R2
3145          ;         JSR  PC,NXM
3146          ;         RETURN
3147          ;           ;TEST "C" AND PROCEED.
3148 017120 012737 017152 000004 XNXM:  MOV  #2,#0#4          ; SET BUSERR VECTOR.
3149 017126 012737 000200 000006      MOV  #PRI04,#0#6
3150 017134 005003          CLR  R3          ;FLAG.
3151 017136 005711 1$:  TST  (R1)          ;TEST THE ADDRESS(ES).
3152          ;IF ANY TRAP, CONTINUE AT 2$.
3153 017140 020102          CMP  R1,R2          ;OTHERWISE, CONTINUE HERE.
3154 017142 001407          BEQ  3$          ;BR IF FINISHED (NO NEXM'S).
3155 017144 062701 000002          ADD  #2,R1          ;SET NEXT ADDRESS...
3156 017150 000772          BR   1$          ;...AND CONTINUE.
3157          ;
3158 017152 005103 2$:  COM  R3          ;GOT ONE, SET FLAG...
3159 017154 012716 017162          MOV  #3#,(SP)
3160 017160 000002          RTI          ;...AND DISMISS INTERRUPT...
3161 017162 3$:  CLRVEC #4          ;...AND GIVE BACK THE VECTOR.
3162 017162 012700 000004          MOV  #4,R0
3163 017166 104436          TRAP C#CVEC
3164 017170 005703          TST  R3          ;DID WE CATCH ONE ??
3165 017172 001401          BEQ  .+4          ;NO, "C" = 0, SKIP NEXT.
3166 017174 000261          SEC          ;YES, "C" = 1, (R1) = NEXM ADDR.
3167 017176 000207          RTS  PC
3168
3169
3170          .SBTTL  TSTLOOP - CHECK ITERATION COUNT
3171          ;*
3172          ; SUBROUTINE TO EXECUTE TEST ITERATIONS.
3173          ; EXIT WITH "C" SET IF LOOPS ALLOWED AND LOOP COUNT NON-ZERO.
3174          ; LOOP COUNTER IS SET BY "BEGIN.TEST" MACRO.
3175          ;
3176          ; CALL:  LOOPTO  ARG
3177          ;
3178 017200          TSTLOOP::
3179 017200 005737 002136          TST  NOITS          ; ITERATIONS INHIBITED?
3180 017204 001006          BNE  1$          ; YES.
3181 017206 005737 002152          TST  QVP          ; NO.
3182 017212 100403          BMI  1$          ;LOOPS DISALLOWED IN QUICK PASS.
3183 017214 005337 002164          DEC  LOOPCNT          ; BUMP LOOP COUNTER.
3184 017220 001002          BNE  2$
3185 017222 000241 1$:  CLC          ;LOOP DISALLOWED, OR DONE.
3186 017224 000401          BR   3$
3187 017226 000261 2$:  SEC          ;LOOP ENABLED.
3188 017230 000207 3$:  RTS  PC

```



```

3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218 017232
3219 017232 010046
3220 017234 005037 003106
3221 017240 005037 017500
3222 017244 005037 005232
3223 017250 105037 016640
3224 017254 013700 002150
3225 017260 006300
3226 017262 005737 003062
3227 017266 001430
3228 017270 100010
3229 017272 052760 160000 003130
3230 017300
    017300 104455
    017302 000001
    017304 003636
    017306 005176
3231 017310 000407
3232 017312 052760 160001 003130 3#:
3233 017320
    017320 104455
    017322 000002
    017324 004233
    017326 000000
3234 017330 012737 177777 003060 2#:
3235 017336
    017336 013700 002150
    017342 104451
3236 017344

```

```

      .SBTTL TSTSETUP - PRINT TEST NAME AND INIT ERROR COUNTS
      ;*
      ; PRINT THE NUMBER AND NAME OF EACH TEST AS WE GO ALONG.
      ; INCREMENT "TESTK" TO INDICATE THE NUMBER OF TESTS
      ; IN THE CURRENT RUN SEQUENCE.
      ; CLEAR THE ERROR COUNTER AND SIGNATURE EXTENSION FLAGS.
      ;
      ; INPUT:
      ;
      ;     R0     POINTER TO TEST ID ASCIZ STRING
      ;
      ; OUTPUT:
      ;
      ;     R5     ADDRESS OF FIRST DEVICE REGISTER
      ;
      ; IMPLICIT OUTPUTS:
      ;
      ;     TSTCNT UPDATED TO COUNT TESTS PERFORMED SINCE START OR RESTART
      ;
      ; SIDE EFFECTS:
      ;
      ;     INTERRUPT LEVEL IS RASIED TO LEVEL OF
      ;     THE DEVICE UNDER TEST
      ;
      ; -
TSTSETUP:;
      MOV     RO, -(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, RO          ; GET THE UNIT NUMBER,
      ASL     RO                 ; ... 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(RO) ; FLAG ERROR IN THE ERROR TABLE
      ERDF   1,NXR,NXRERR      ; NO DEVICE HERE -- PRINT IT
      TRAP   C$ERDF
      .WORD  1
      .WORD  NXR
      .WORD  NXRERR
      BR     2$
      BIS     @160001,ERTABL(RO) ; FLAG ERROR IN THE ERROR TABLE
      ERDF   2,NOINIT         ; DEVICE NOT IDLE
      TRAP   C$ERDF
      .WORD  2
      .WORD  NOINIT
      .WORD  0
      MOV     @-1,DUFLG        ; DROP THE UNIT
      DODU   UNITN
      MOV     UNITN, RO
      TRAP   C$DODU
      DOCLN                      ; ABORT THE PASS

```

3237	017344	104444			TRAP	C%DCLN		
3238	017346	000423			BR	5\$		
3239	017350			4\$:	RFLAGS	R0		; GET THE OPERATOR FLAGS.
	017350	104421			TRAP	C#RFLA		
3240	017352	032700	001000		BIT	#PNT,R0		; PRINT THE TEST NUMBERS?
3241	017356	001412			BEQ	1\$; BR IF NO
3242	017360	011600			MOV	(SP),R0		;GET THE ID MESSAGE
3243	017362				PRINTF	#TNAM,R0		;DISPLAY THE TEST ID
	017362	010046			MOV	R0,-(SP)		
	017364	012746	017426		MOV	#TNAM,-(SP)		
	017370	012746	000002		MOV	#2,-(SP)		
	017374	010600			MOV	SP,R0		
	017376	104417			TRAP	C#PNTF		
	017400	062706	000006		ADD	#6,SP		
3244	017404	005237	002162	1\$:	INC	TSTCNT		; BUMP TEST COUNTER.
3245	017410				SETPRI	IPRI		;PRIORITY THAT OF DEVICE
	017410	013700	002160		MOV	IPRI,R0		
	017414	104441			TRAP	C#SPRI		
3246	017416	005726		5\$:	TST	(SP)+		;FIX UP THE STACK
3247	017420	013705	002154		MOV	CSRADDR,R5		; ADDRESS OF TSV REGISTERS ON UNIBUS
3248	017424	000207			RTS	PC		
3249	017426	045	123	045	TNAM:	.ASCIZ	'#S#T#A Test'	
3250						.EVEN		

```

3252
3253
3254
3255
3256
3257 017442
      017442 104421
3258 017444 030027 020000
3259 017450 001412
3260 017452
      017452 013746 017500
      017456 012746 017502
      017462 012746 000002
      017466 010600
      017470 104417
      017472 062706 000006
3261 017476 000207
3262
3263 017500 000000
3264 017502 045 101 040
3265 017521 105 122 122
3266
3267
3268
3269
3270
3271
3272 017566 005237 017500
3273 017572 010046
3274 017574 013700 002150
3275 017600 006300
3276 017602 062700 003130
3277 017606 005210
3278 017610 032710 007777
3279 017614 001001
3280 017616 005310
3281 017620 012600
3282 017622 000207
3283
3284 017624 010046
3285 017626 013700 002150
3286 017632 006300
3287 017634 016000 003130
3288 017640 042700 170000
3289 017644 020037 002142
3290 017650 103004
3291 017652 023737 017500 002140
3292 017660 103417
3293 017662
      017662 104421
3294 017664 032700 000040
3295 017670 001013
3296 017672 012737 177777 003060
3297 017700
      017700 104455
      017702 000004
      017704 017521

```

```

.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 C$RFLA
        BIT RO,#IER
        BEQ 1$ ; BR IF "IER" NOT SET.
        PRINTF #ESUM,ERRK ; PRINT ERROR COUNT.
        MOV ERRK,-(SP)
        MOV #ESUM,-(SP)
        MOV #2,-(SP)
        MOV SP,RO
        TRAP C$PNTF
        ADD #6,SP
1$: RTS PC

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

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

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

```


3298	017706	000000			.WORD	0	
	017710				DODU	UNITN	
	017710	013700	002150		MOV	UNITN,RO	
	017714	104451			TRAP	C#DODU	
3299	017716				DOCLN		
	017716	104444			TRAP	C#DCLN	
3300	017720	012600		2\$:	MOV	(SP)+,RO	; RESTORE RO
3301	017722	000207			RTS	PC	; RETURN TO CALLER
3302					.SBTTL	FATCHK - INC FATAL ERRORS AND CHECK FOR LIMIT	
3303							
3304							
3305							
3306							
3307							
3308							
3309	017724				FATCHK:		
3310	017724				SAVREG		;BETTER SAVE THE REGISTERS
3311	017730	013701	002150		MOV	UNITN,R1	;PICK UP THE UNIT NUMBER
3312	017734	006301			ASL	R1	;MAKE IT INTO A BYTE OFFSET
3313	017736	062761	000001	003130	ADD	#1,ERTABL(R1)	;ADD 1 TO THE PROPER UNIT'S ERROR COUNTER
3314	017744	005237	002170		INC	FATFLG	;BUMP FATAL ERROR COUNTER
3315	017750	023727	002170	000031	CMP	FATFLG,#25.	;CHECK AGAINST 25
3316	017756	002406			BLT	9\$;BR, IF LESS THAN 25 ERRORS
3317	017760				RFLAGS	RO	;READ THE FLAGS INTO RO
	017760	104421			TRAP	C#RFLA	
3318	017762	032700	040000		BIT	#BIT14,RO	;BR, IF LOOP ON ERROR IS SET
3319	017766	001002			BNE	9\$;OTHERWISE NEVER BE ABLE TO SCOPE ETC.
3320	017770	004737	017776		JSR	PC,CKDROP	;DROP UNIT IF ALLOWED
3321	017774	000207		9\$:	RTS	PC	;RETURN ETC.
3322							
3323							
3324							

```

3326          .SBTTL CKDROP - CHECK IF UNIT SHOULD BE DROPPED
3327          ;*
3328          ; CHECK IF UNIT SHOULD BE DROPPED
3329          ;-
3330 017776 010046          CKDROP: MOV      RO, -(SP)
3331 020000          FORCERROR      1$,NOTSSR
3332 020010          RFLAGS      RO
3333 020012 104421          TRAP      C$RFLA
3334 020016 032700 000040  BIT      @IDU,RO
3335 020020 001010          BNE      1$
3336 020022 011600          MOV      (SP),RO
3337 020022 012737 177777 003060  MOV      @-1,DUFLG
3338 020030          DODU      UNITN
3339 020030 013700 002150  MOV      UNITN,RO
3340 020034 104451          TRAP      C$DODU
3341          DOCLN          ;ABORT THE PASS
3342          TRAP      C$DCLN
3343          1$: MOV      (SP)+,RO
3344          RTS      PC
3345          .SBTTL CONFIG - DETERMINE CONFIGURATION OF SYSTEM
3346          ;
3347          ; SUBROUTINE - DETERMINE CONFIGURATION OF TK-25 SYSTEM.
3348          ;
3349          CONFIG:
3350 020044          JSR      PC,SOFINIT
3351 020050 004737 016470  RTS      PC
3352
3353
3354

```

```

3356 .SBTTL KTON,KTOFF - ENABLE/DISABLE MEMORY MANAGEMENT
3357
3358 ; SUBROUTINE - ENABLE MEM MGT.
3359 ;
3360 020052 005737 003100 KTON: TST KFLG ; GOT KT?
3361 020056 001403 BEQ 1$ ; NO.
3362 020060 012737 000001 177572 MOV #1,SRO ; YES. ENABLE KT11.
3363 020066 000207 1$: RTS PC
3364
3365
3366
3367 ; SUBROUTINE - DISABLE MEM MGT.
3368 ;
3369 ;
3370 020070 005737 003100 KTOFF: TST KFLG ; GOT KT11?
3371 020074 001405 BEQ 1$ ; NO.
3372 020076 000240 NOP
3373 020100 000240 NOP
3374 020102 012737 000000 177572 MOV #0,SRO ; DISABLE KT.
3375 020110 000207 1$: RTS PC
3376
3377

```



```

3379          .SBTTL  SETMAP  -  SETUP  PAR6  MAPPING
3380
3381          ;*
3382          ;
3383          ;THIS ROUTINE SETS UP KERNEL PAR6 TP HANDLE
3384          ;AN 18 BIT ADDRESS. THE OFFSET INTO THE PAGE
3385          ;IS RETURNED BIASED TO PAR6.
3386          ;
3387          ;INPUTS:
3388          ;
3389          ;      R0      HIGH ORDER ADDRESS BITS
3390          ;      R1      LOW ORDER ADDRESS BITS
3391          ;
3392          ;OUTPUTS:
3393          ;
3394          ;      R0      OFFSET INTO BLOCK WITH PAR6 BIAS (I.E. THE ADDRESS)
3395          ;      CARRY   SET IF SUCCESS
3396          ;              CLR IF ERROR
3397          ;-
3398          SETMAP:
3399          SAVREG          ;SAVE R1-R4 UNTIL NEXT RETURN
3400          TST            KTF LG          ;SYSTEM HAVE ABOVE 28K?
3401          BEQ            10$           ;BR IF NO
3402          MOV            R1,R2         ;SAVE LOW ORDER BITS
3403          .REPT          6
3404          ASR            R0            ;CONVERT WORD ADDRESS TO 32W BLOCKS
3405          ROR            R1            ;MAKE IT DOUBLE PRECISION
3406          .ENDR
3407          BIC            #177,R1       ;ALINE FOR LOWER 4K BOUNDARY
3408          CMP            R1,KTF LG     ;HIGHER THAN EXISTING MEMORY?
3409          BHIS          10$           ;BR IF YES
3410          MOV            R1,#KIPAR6   ;SETUP MAPPING REGISTER PAR6
3411          BIC            #160000,R2    ;SETUP DISPLACEMENT IN PAGE
3412          ADD            #140000,R2    ;ADD IN PAR6 BIAS
3413          MOV            R2,R0         ;RETURN IN R0
3414          SEC            ;SET SUCCESS
3415          BR            15$           ;
3416          CLC            ;SET FAILURE
3417          10$:          CLC            ;
3418          15$:          RTS            PC ;RETURN

```

```

3420 .SBTTL FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN
3421 ;*
3422 ; FILL MEMORY WITH A BACKGROUND PATTERN
3423 ;
3424 ; INPUTS:
3425 ;
3426 ; RO = BACKGROUND PATTERN
3427 ; FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
3428 ; KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
3429 ;
3430 ; OUTPUTS:
3431 ;
3432 ; NONE
3433 ;
3434 ;
3435 FILLMEM:
3436 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
3437 JSR PC,KTOFF ;DISABLE KT.
3438 MOV RO,R3 ;COPY TEST PATTERN
3439 MOV FREE,R1 ;GET FIRST FREE LOCATION
3440 MOV FRESIZ,R2 ;SIZE OF FREE SPACE BELOW 28K.
3441 10$: MOV R3,(R1)+ ;STORE A BACKGROUND WORD
3442 DEC R2 ;DONE ALL MEMORY IN FREE SPACE?
3443 BGT 10$ ;BR IF NO
3444 TST KTFLG ; GOT KT?
3445 BEQ 55$ ; NO. GET OUT.
3446 JSR PC,KTON ; YES. ENABLE KT.
3447 CLR RO ;HIGH ORDER ADDRESS START
3448 MOV PST32W,R1 ;GET >28K START ADDRESS (IN 32W BLOCKS)
3449 .REPT 6
3450 CLC ;CLEAR C BIT
3451 ROL R1 ;CONVERT BLOCKS TO WORDS
3452 ROL RO ;MAKE IT DOUBLE PRECISION
3453 .ENDR
3454 JSR PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
3455 30$: MOV R3,(R0)+ ;STORE TEST PATTERN IN >28K ADDRESS
3456 CMP RO,#160000 ;END OF PAR6 MAPPING AREA?
3457 BLO 30$ ;BR IF NO
3458 SUB #20000,RO ;BACKUP INTO PAR6 MAPPING BEGIN
3459 ADD #200,#KIPAR6 ;POINT TO NEXT 4K BLOCK >28K.
3460 CMP #KIPAR6,KTFLG ;END OF MEMORY?
3461 BEQ 50$ ;BR IF YES
3462 JMP 30$ ;KEEP GOING ON ETC.
3463 50$: JSR PC,KTOFF ; DISABLE KT.
3464 55$: RTS PC
3465
3466

```



```

3468 .SBTTL CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN
3469
3470 ;* COMPARE MEMORY WITH A BACKGROUND PATTERN
3471 ;
3472 ; INPUTS:
3473 ;
3474 ;     RO = BACKGROUND PATTERN
3475 ;     FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
3476 ;     KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
3477 ;
3478 ; OUTPUTS:
3479 ;
3480 ;     CARRY - SET IF NO ERROR
3481 ;     CARRY - CLR IF ERROR
3482 ;
3483 ; IMPLICIT OUTPUTS:
3484 ;
3485 ;     ERRHI - ERROR HIGH ADDRESS
3486 ;     ERRLO - ERROR LOW ADDRESS
3487 ;     EXPD  - EXPECTED DATA
3488 ;     RECV  - RECEIVED DATA
3489 ;
3490 ;-
3490 CMPMEM:
3491 SAVREG
3492 MOV R0,R3 ;SAVE R1-R5 UNTIL NEXT RETURN
3493 JSR PC,KTOFF ;COPY TEST PATTERN
3494 MOV FREE,R1 ;DISABLE KT.
3495 MOV FRESIZ,R2 ;GET FIRST FREE LOCATION
3496 10$: CMP R3,(R1) ;SIZE OF FREE SPACE BELOW 28K.
3497 BEQ 15$ ;FREE SPACE LOCATION EQUAL TO EXPD?
3498 MOV R1,ERRLO ;BR IF YES
3499 CLR ERRHI ;SAVE ADDRESS IN ERROR
3500 MOV R3,EXPD ;NO HIGH ADDRESS
3501 MOV (R1),RECV ;SAVE EXPD FOR ERROR REPORT
3502 BR 50$ ;SAVE RECV FOR ERROR REPORT
3503 15$: TST (R1)+ ;
3504 DEC R2 ;POINT TO NEXT ADDRESS
3505 BGT 10$ ;DONE ALL MEMORY IN FREE SPACE?
3506 TST KTFLG ;BR IF NO
3507 BEQ 55$ ; GOT KT?
3508 JSR PC,KTON ; NO. GET OUT.
3509 CLR R0 ; YES. ENABLE KT.
3510 MOV PST32W,R1 ;HIGH ORDER ADDRESS START
3511 .REPT 6 ;GET >28K START ADDRESS (IN 32W BLOCKS)
3512 ROL R1 ;CONVERT BLOCKS TO WORDS
3513 ROL R0 ;MAKE IT DOUBLE PRECISION
3514 .ENDR
3515 BIC #177,R1 ;ALINE 4K BOUNDARY
3516 MOV R0,-(SP) ;SAVE HIGH ORDER
3517 MOV R1,-(SP) ;SAVE LOW ORDER
3518 JSR PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
3519 MOV R0,R4 ;COPY ADDRESS BIASED TO PAR6
3520 MOV (SP)+,R1 ;RESTORE LOW ORDER IN NON PAR6 FORMAT
3521 MOV (SP)+,R0 ;RESTORE HIGH ORDER IN NON PAR6 FORMAT
3522 30$: CMP R3,(R4) ;ABOVE 28K LOCATION EQUAL EXPD?
3523 BEQ 32$ ;BR IF YES
3524 MOV R0,ERRHI ;SAVE HIGH ORDER IN ERROR
    
```


3525	020562	010137	002204		MOV	R1,ERRLO	;SAVE LOW ORDER IN ERROR
3526	020566	010337	002176		MOV	R3,EXPD	;SAVE EXPD FOR ERROR REPORT
3527	020572	011437	002200		MOV	(R4),RECV	;SAVE RECV FOR ERROR REPORT
3528	020576	000421			BR	50\$;
3529	020600	062701	000002	32\$:	ADD	#2,R1	;UPDATE NON PAR6 ADDRESS
3530	020604	005500			ADC	R0	;MAKE IT DOUBLE PRECISION ADD
3531	020606	062704	000002		ADD	#2,R4	;UPDATE PAR FORMAT ADDRESS
3532	020612	020427	160000		CMP	R4,#160000	;END OF PAR6 MAPPING AREA?
3533	020616	103755			BLO	30\$;BR IF NO
3534	020620	162704	020000		SUB	#20000,R4	;BACKUP INTO PAR6 MAPPING BEGIN
3535	020624	062737	000200	172354	ADD	#200,#KIPAR6	;POINT TO NEXT 4K BLOCK >28K.
3536	020632	023737	172354	003100	CMP	#KIPAR6,KTFLG	;END OF MEMORY?
3537	020640	101744			BLOS	30\$;BR IF NO
3538	020642	004737	020070	50\$:	JSR	PC,KTOFF	;TURN OFF MEMORY MAPPING
3539	020646	000241			CLC		;SET FAILURE
3540	020650	000403			BR	60\$;
3541	020652	004737	020070	55\$:	JSR	PC,KTOFF	;TURN OFF MEMORY MAPPING
3542	020656	000261			SEC		;SET SUCCESS
3543	020660	000207		60\$:	RTS	PC	
3544							

```

3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566 020662
3567 020662
      020662 104422
3568 020664 010446
3569 020666 010346
3570 020670 010246
3571 020672 010146
3572 020674 010546
3573 020676 016605 000012
3574 020702 004736
3575 020704 012601
3576 020706 012602
3577 020710 012603
3578 020712 012604
3579 020714 012605
3580 020716
      020716 104422
3581 020720 000207
3582

```

```

      .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:
      BREAK
      TRAP      C:BRK
      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
      BREAK
      TRAP      C:BRK
      RTS      PC
;LOOK FOR CNTL C
;LOOK FOR CNTL C

```

```

3584 .SBTTL GETPAT - GET 8 BIT PATTERN FROM OPERATOR
3585 ;*
3586 ;ROUTINE TO REQUEST AN 8 BIT DATA PATTERN FROM THE OPERATOR
3587 ;
3588 ;INPUTS:
3589 ;
3590 ; NONE.
3591 ;
3592 ;OUTPUTS:
3593 ;
3594 ; RO OCTAL NUMBER FROM THE OPERATOR
3595 ;
3596 ;CALLING SEQUENCE:
3597 ;
3598 ; JSR PC,GETPAT
3599 ;
3600 ;
3601 ;-
3602
3603 GETPAT::
3604 1$: SAVREG ;SAVE THE GENERAL REGISTERS
3605 1$: 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$HILIM
      10000$: BNCOMPLETE 1$ ;RETRY IF ERROR
      BCC 1$
      MOV PATDAT,RO ;DATA PATTERN FROM OPERATOR
      RTS PC ;RETURN TO CALLER
3610 ;*
3611 ;LOCAL DATA AREA
3612 ;-
3613
3614 PATDAT: .WORD 0 ;TEMPORARY STORAGE FOR DATA
3615 DATASC: .ASCIZ 'ENTER DATA PATTERN'
3616 .EVEN

```

```

020722
020722
020726 104443
020730 000406
020732 020756
020734 000022
020736 020760
020740 000377
020742 000000
020744 000377
020746
3606 020746 103367 020756
3607 020750 013700
3608 020754 000207
020756
000000
020760 105 116 124

```



```

3618 .SBTTL GETSEL - ISSUE MENU AND GET OPERATOR RESPONSE
3619 ;*
3620 ;ROUTINE TO ISSUE A MENU AND GET
3621 ;THE OPERATOR'S RESPONSE.
3622 ;
3623 ;INPUTS:
3624 ;
3625 ; R0 ADDRESS OF ASCIZ STRING OF MENU
3626 ; R1 MAXIMUM ALLOWABLE OPERATOR RESPONSE
3627 ;
3628 ;OUTPUTS:
3629 ;
3630 ; R0 NUMBER OF THE OPERATOR'S SELECTION
3631 ;-
3632 GETSEL::
3633 SAVREG ;SAVE GENERAL REGISTERS
3634 MOV R0,R2 ;SAVE THE MENU ADDRESS
3635 MOV R2,R3 ;START OF MENU STRING
3636 TST (R3) ;END OF ASCII ?
3637 BEQ 3$ ;BRANCH IF ALL LINES DISPLAYED
3638 PRINTF @SELASC,(R3), ;DISPLAY THE MENU
MOV (R3),-(SP)
MOV @SELASC,-(SP)
MOV @2,-(SP)
MOV SP,R0
TRAP C:PNTF
ADD @6,SP
BR 2$
3639 021004 010002
3640 021010 012746 021170
021012 012746 000002
021014 010600
021016 104417
021020 062706 000006
021042 000764
3641 021044 104443
021046 000406
021050 021224
021052 000042
021054 021175
021056 177777
021060 000000
021062 177777
021064
3642 021064 103352
3643 021066 013700 021224
3644 021072 020001
3645 021074 101411
021076 012746 021122
021102 012746 000001
021106 010600
021110 104417
021112 062706 000004
3646 021116 000735
3647 021120 000207
3648 021122 045 116 045 MENERR: .ASCIZ 'MNA *** Menu Selection Too Large ***'
3649 021170 045 116 045 SELASC: .ASCIZ 'MNT'
3650 021175 105 156 164 MENASC: .ASCIZ 'Enter Menu Selection: '
3651 .EVEN
3652 021224 000000 MENRES: .WORD 0

```

```

3654 .SBTTL CHKMAN - CHECK MANUAL INTERVENTION LEGALITY
3655 ;*
3656 ;ROUTINE TO TEST FOR MANUAL INTERVENTION LEGALITY.
3657 ;INPUT:
3658 ;
3659 ; NONE.
3660 ;
3661 ;OUTPUT:
3662 ;
3663 ; CARRY 0 MANUAL INTERVENTION NOT ALLOWED
3664 ; CARRY 1 MANUAL INTERVENTION IS OK
3665 ;
3666 ;SIDE EFFECTS:
3667 ;
3668 ; A MESSAGE IS DISPLAYED WARNING THAT TEST IS
3669 ; NOT EXECUTED IF MANUAL INTERVENTION IS NOT
3670 ; ALLOWED.
3671 ;
3672 ;-
3673 ;
3674 ;-
3675 ;
3676 021226 CHKMAN:: SAVREG ;SAVE THE REGISTERS
3677 021226 MANUAL ;SEE IF MANUAL INTERVENTION OK
3678 021232 TRAP C$MANI
3679 021234 BCOMPLETE 1$ ;BRANCH IF ALLOWED
3680 021234 103411 BCS 1$
3681 021236 012746 021262 PRINTF @NOMAN ;PRINT THE WARNING MESSAGE
3682 021242 012746 000001 MOV @NOMAN,-(SP)
3683 021246 010600 MOV @1,-(SP)
3684 021250 104417 TRAP SP,R0
3685 021252 062706 000004 TRAP C$PNTF
3686 021256 000241 ADD @4,SP
3687 021260 000207 CLC ;CLEAR CARRY FOR ERROR
3688 1$: RTS PC ;RETURN
3689 021262 045 116 045 NOMAN: .ASCIZ 'N$A *** Manual Intervention not Allowed - Test Aborted ***'
3690 .even
    
```



```

3707          .SBTTL  KTINIT  - SETUP KT11 MEMORY MANAGEMENT REGISTERS
3708          ;*
3709          ;
3710          ;ROUTINE TO INIT KT-11
3711          ;
3712          ;-
3713
3714          KTINIT:
3715 021444 005037 003100          CLR      KTFLG          ; INIT >28K MEMORY FLAG
3716 021450 005037 003102          CLR      KTENABLE       ; INIT TEST >28K FLAG
3717 021454 023727 002120 001577  CMP      L$HIME,#01577   ; GOT ENOUGH MEMORY (>28K)?
3718 021462 101444                BLOS    9$              ; NO.
3719 021464 013700 000004          MOV      @ERRVEC,RO     ; SAVE OLD ERR VEC PTR.
3720 021470 012737 021562 000004  MOV      #2$,@ERRVEC   ; SET ERR VEC PTR.
3721 021476 005737 177572          TST     @SRO           ; GOT KT11?
3722 021502 000240                NOP                    ; (TRAP IF NO).
3723 021504 013737 002120 003100  MOV      L$HIME,KTFLG  ; YES. SET KT FLAG.
3724 021512 042737 000177 003100  BIC     #177,KTFLG     ;
3725 021520 010037 000004          MOV      RO,@ERRVEC   ; RESTORE OLD ERR VEC PTR.
3726 021524 005000                CLR      RO            ; RO = AR DATA.
3727 021526 012701 172340          MOV      @KIPAR0,R1   ; R1 = KI REGS PTR.
3728 021532 012761 077406 177740 1$:  MOV      #77406,-40(R1) ; SET DESCRIPTOR REG.
3729 021540 010021                MOV      RO,(R1)+     ; SET KIPAR REG.
3730 021542 062700 000200          ADD     #200,RO        ; BUMP AR DATA BY "4K".
3731 021546 020027 002000          CMP     RO,#2000      ; AT "I/O"?
3732 021552 001367                BNE     1$            ; NO.
3733 021554 012741 177600          MOV     #177600,-(R1) ; YES. SET KTPAR7 FOR I/O.
3734 021560 000405                BR      9$            ;
3735
3736 021562 012716 021570          2$:  MOV     #6$, (SP)    ; SET UP RETURN
3737 021566 000002                RTI                    ; RTI TO NEXT LOCATION
3738
3739 021570 010037 000004          6$:  MOV     RO,@ERRVEC ; RESTORE OLD ERR VEC PTR.
3740
3741 021574 000207          9$:  RTS     PC
3742          ;          .IIF DF ONEFILE, .PAGE
3751
3752
3758

```

H9

CZTKEA TK25 FRT END FUNC #1
PROTECTION TABLE

MACRO M1200 20-APR-84 08:12 PAGE 83

SEQ 111

```
3760 .SBTTL PROTECTION TABLE
3761 021576 BGNPROT
    021576
3762 021576 177777 177777 177777 L$PROT::
3763 021606 .WORD -1. -1. -1. -1 ;NO DEVICE PROTECTION REQUIRED.
3764 ENDPROT
```

```

3766          .SBTTL  INITIALIZE SECTION
3767
3768          ;**
3769          ;THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
3770          ;AT THE BEGINNING OF EACH PASS.
3771          ;
3772          ;IF "START" OR "RESTART", SET QUICK-PASS FLAG AND BUS-INIT.
3773          ;IF "CONTINUE", NOTHING IS REQUIRED.
3774          ;
3775          ;--
3776          ;*
3777          ;INSERT TEMPORARY JUMP TO ODT
3778          ;-
3779          021606          BGNINIT
3780          021606          L$INIT::
3781          021606          012737 005672 002146          40$:  MOV      #EPR1,EPR1SW ;SET UP PRIMARY MESSAGE FOR REPLACEMENT
3782          021614          005037 003106          CLR      SIFLAG      ;CLEAR "SOFT INIT" FLAG
3783          021620          005037 003102          CLR      KTENABLE   ;CLEAR TEST ABOVE 28K FLAG
3784          021624          005037 002246          CLR      RAMSIZ     ;CLEAR RAM SIZE FOR RAMERR ROUTINE
3785          021630          021630 012700 000036          READEF  #EF.CONTINUE
3786          021636          021636 103023          MOV      #EF.CONTINUE,R0
3787          021640          023737 002150 002012          TRAP    C$REFG
3788          021646          103064          BNCOMPLETE 1$
3789          021650          005737 003060          BCC     1$
3790          021654          100466          CMP     UNITN,L$UNIT ;UNIT IN RANGE?
3791          021656          013701 002150          BHIS   4$           ;BR IF NO.
3792          021662          006301          TST    DUFLG       ;DROPPED UNIT?
3793          021664          005761 003130          BMI   NXTU        ;BR IF YES
3794          021670          001512          MOV    UNITN,R1
3795          021672          032761 040000 003130          ASL    R1
3796          021700          001054          TST    ERTABL(R1)
3797          021702          021702 104432          BEQ    SETU
3798          021704          000412          BIT    #BIT14,ERTABL(R1) ;DROPPED?
3799          021706          021706 012700 000035          BNE   NXTU
3800          021712          104447          EXIT   INIT        ;DO NOTHING IF "CONTINUE".
3801          021714          021714 103046          TRAP   C$EXIT
3802          021716          012716 000040          .WORD  L10030-.
3803          021722          104447          1$:  READEF  #EF.NEW
3804          021724          103404          MOV    #EF.NEW,R0
3805          021726          012726 000037          TRAP   C$REFG
3806          021732          104447          BNCOMPLETE NXTU    ;TAKE NEXT UNIT IF NOT NEW PASS.
3807          021734          103025          BCC   NXTU
3808          021736          005037 002162          READEF  #EF.START
3809          021738          005037 002162          MOV    #EF.START,R0
3810          021740          005037 002162          TRAP   C$REFG
3811          021742          005037 002162          BCOMPLETE 2$
3812          021744          005037 002162          BCS   2$
3813          021746          005037 002162          READEF  #EF.RESTART
3814          021748          005037 002162          MOV    #EF.RESTART,R0
3815          021750          005037 002162          TRAP   C$REFG
3816          021752          005037 002162          BNCOMPLETE 31$
3817          021754          005037 002162          BCC   31$
3818          021756          005037 002162          2$:  BRESET
3819          021758          005037 002162          TRAP   C$RESET
3820          021760          005037 002162          CLR    TSTCNT
3821          021762          005037 002162          ;1ST PASS, BUS-INIT...
3822          021764          005037 002162          ;BUS RESET.
3823          021766          005037 002162          ;NUMBER OF TESTS RUN IN PASS
3824          021768          005037 002162

```



```

3857 022164 010221          MOV      R2,(R1)+      ;...AND PRIORITY.
3858
3859 022166                1$:
3860                        ;      TST      QVP          ;1ST PASS ??
3861                        ;      BEQ      5$          ;NO, SKIP THE PASS 1 STUFF.
3862
3863                        ;
3864                        ;1ST PASS, CHECK THAT DEVICE ADDRESSES ARE VALID, AND
3865                        ;THAT THE DISPLAY STATUS IS PROPERLY INITIALIZED.
3866                        ;
3867 022166 013701 002150    MOV      UNITN,R1
3868 022172 006301          ASL      R1
3869 022174 052761 100000 003130  BIS      #BIT15,ERTABL(R1) ;SAY DEVICE RUNNING
3870 022202 005037 005232    CLR      EXTA          ;CLEAR ERROR EXTENSION FLAG.
3871 022206 023727 002012 000001  CMP      L$UNIT,#1     ;ARE WE TESTING MULTIPLE UNITS?
3872 022214 101416          BLOS     10$          ;BR IF NO.
3873 022216                RFLAGS    RO          ;YES -- GET OPERATOR FLAGS.
3874 022220 104421          TRAP     C$RFLA
3875 022224 032700 001000    BIT      #PNT,RO      ;SHOULD WE PRINT UNIT #?
3876 022226 001412          BEQ      10$          ;BR IF NOT.
3877 022226                PRINTF   #PUNIT,UNITN ;PRINT THE UNIT #
3878 022226 013746 002150    MOV      UNITN,-(SP)
3879 022232 012746 022320    MOV      #PUNIT,-(SP)
3880 022236 012746 000002    MOV      #2,-(SP)
3881 022242 010600          MOV      SP,RO
3882 022244 104417          TRAP     C$PNTF
3883 022246 062706 000006    ADD      #6,SP
3884 022252                10$:
3885 022252 005037 003062    CLR      NODEV
3886 022256 013701 002154    MOV      CSRADDR,R1  ;ADDRESS OF FIRST REGISTER
3887 022262 010102          MOV      R1,R2        ;START OF REGISTERS
3888 022264 062702 000000    ADD      #TSSR,R2    ;ADDRESS OF TSSR REGISTER
3889 022270 004737 017120    JSR      PC,XNXM     ;TEST BOTH CONTROLLER REGISTERS...
3890 022274 103005          BCC      2$          ;...AND BR IF ALL OK.
3891 022276 010137 003062    MOV      R1,NODEV    ;FLAG DEVICE AS NON-EXISTENT
3892 022302 012737 177777 003060  MOV      #-1,DUFLG   ;DROP THIS UNIT.
3893 022310                2$:
3894 022310                ;
3895 022310                ;FINALLY, SET CPU PRIORITY AND WE'RE DONE.
3896 022310                ;
3897 022310                5$:
3898 022310 012700 000000    SETPRI   #PRI00      ;ENABLE INTERRUPTS.
3899 022314 104441          MOV      #PRI00,RO
3900 022316                TRAP     C$SPRI
3901 022316                ENDINIT
3902 022316 104411          L10030: TRAP     C$INIT
3903 022320 045 116 045 PUNIT: .ASCIZ  /#N#N#A***** TESTING UNIT #D2#A *****/
3904                                .EVEN

```

```

3896                                     .SBTTL  ADD AND DROP UNITS SECTIONS
3897
3898
3899                                     ;**
3900                                     ; THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
3901                                     ; TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
3902                                     ; OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
3903                                     ;--
3903 022366                                     BGNAU
3904 022366 010001                               L$AU::
3905 022370 006301                               MOV     R0,R1           ; GET UNIT TO BE ADDED (R0)
3906 022372 052761 100000 003130                ASL     R1             ; MAKE IT A WORD INDEX
3907 022400 042761 040000 003130                BIS     #100000,ERTABL(R1) ; SET THE "ACTIVE" BIT
3908 022406                                     BIC     #40000,ERTABL(R1) ; CLEAR THE "DROPPED" BIT
3909 022406 010046                               PRINTF  #1$,R0
3910 022410 012746 022434                               MOV     R0,-(SP)
3911 022414 012746 000002                               MOV     #1$,-(SP)
3912 022420 010600                               MOV     #2,-(SP)
3913 022422 104417                               MOV     SP,R0
3914 022424 062706 000006                               TRAP   C$PNTF
3915 022430                                     ADD     #6,SP
3916 022430 000167                               EXIT   AU
3917 022432 000026                               .WORD  J$JMP
3918 022434 045 116 045 1$:                       .WORD  L10031-2-.
3919                                     .ASCIZ /#N#A UNIT #D#A ADDED/
3920                                     .EVEN
3921
3922                                     ENDAU           ; UNUSED.
3923 022462 104452                               L10031:
3924 022462                                     TRAP   C$AU
3925
3926                                     ;**
3927                                     ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
3928                                     ; TO BE REMOVED FROM THE TEST LIST.
3929
3930                                     ;
3931                                     ; SUPVSR DOES THE "DROPPING". THIS IS JUST TO TELL THE MAN,
3932                                     ; "DROPPED" UNITS ARE RE-SELECTED ON OPERATOR "STA" OR "ADD"
3933                                     ; COMMAND, OTHERWISE REMAIN INACTIVE. THE "DISPLAY" COMMAND
3934                                     ; WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
3935                                     ; WHICH ARE STILL ACTIVE.
3936                                     ; UPON ENTRY, R0 CONTAINS THE UNIT TO BE DROPPED.
3937
3938                                     ;--
3939 022464                                     BGNDU
3940 022464                                     L$DU::
3941 022472 012737 177777 003060                MOV     #-1,DUFLG
3942 022474 010001                               MOV     R0,R1
3943 022476 006301                               ASL     R1
3944 022478 052761 140000 003130                BIS     #140000,ERTABL(R1) ; SAY DROPPED
3945 022504 000240 000240 000240                240,240,240           ; ??????????
3946 022512                                     PRINTF  #1$,R0
3947 022514 010046                               MOV     R0,-(SP)
3948 022516 012746 022540                               MOV     #1$,-(SP)
3949 022518 012746 000002                               MOV     #2,-(SP)
3950 022520 010600                               MOV     SP,R0
3951 022522 104417                               TRAP   C$PNTF
3952 022524 062706 000006                               ADD     #6,SP
3953 022530                                     EXIT   DU
3954 022532 000167                               .WORD  J$JMP
3955 022534 000030                               .WORD  L10032-2-.

```


3933	022540	045	116	045	1\$:	.ASCIZ /#N#A UNIT #D#A DROPPED/ .EVEN ENDDU	
3934							
3935	022570				L10032:	TRAP C#DU	
	022570						
	022570	104453					
3936							
3937							
3938							
3939	022572					BGNAUTO	
	022572				L\$AUTO::		
3940	022572	012703	000550			MOV #360.,R3	;ENOUGH TIME FOR 2400' REEL TO REWIND
3941	022576	004737	016744		10\$:	JSR PC,WAITF	;WAIT FOR SSR TO SET
3942	022602	103420				BCS 20\$;LEAVE WHEN SSR IS SET
3943	022604					DELAY 250.	;WAIT FOR .25 SECONDS
	022604	012727	000372			MOV #250.,(PC)+	
	022610	000000				.WORD 0	
	022612	013727	002116			MOV L\$DLY,(PC)+	
	022616	000000				.WORD 0	
	022620	005367	177772			DEC -6(PC)	
	022624	001375				BNE .-4	
	022626	005367	177756			DEC -22(PC)	
	022632	001367				BNE .-20	
3944	022634	005303				DEC R3	;BUMP COUNTER DOWN
3945	022636	001357				BNE 10\$;KEEP GOING
3946	022640	004737	017776			JSR PC,CKDROP	;TRY AND DROP UNIT
3947	022644				20\$:		
3948	022644					ENDAUTO	; UNUSED.
	022644				L10033:		
	022644	104461				TRAP C\$AUTO	

```

3950          .SBTTL  CLEAN-UP AND REPORT CODING SECTIONS
3951
3952          ;**
3953          ; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS
3954          ; EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).
3955          ; USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.
3956          ;--
3957 022646          BGNCLN
          L$CLEAN::
3958 022646 005737 003060          TST      DUFLG          ;"DROPPED" FLAG IS SET ON...
3959 022652 100407          BMI      1$          ;...AND GROSS CONTROLLER FAULT...
3960          ;...DON'T TRY TO XCT CLEANUP CODE.
3961
3962 022654 013705 002154          MOV      CSRADDR,R5          ; ADDRESS OF TSV REGISTERS ON UNIBUS
3963 022660 012765 000000 000000          MOV      #0,TSSR(R5)          ;DO SOFT INIT
3964 022666 004737 016744          JSR      PC,WAITF
3965 022672          1$:
3966 022672          2$:          ENDCLN
          L10034:          TRAP      C$CLEAN
          022672 104412
3967          ;**
3968          ; THE REPORT CODING SECTION CONTAINS THE
3969          ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
3970          ;--
3971 022674          BGNRPT
          L$RPT::
3972 022674          PRINTS  #DEVSUM
          022674 012746 023136          MOV      #DEVSUM,-(SP)
          022700 012746 000001          MOV      #1,-(SP)
          022704 010600          MOV      SP,R0
          022706 104416          TRAP      C$PNTS
          022710 062706 000004          ADD      #4,SP
3973 022714 010246          MOV      R2,-(SP)
3974 022716 010346          MOV      R3,-(SP)
3975 022720 010446          MOV      R4,-(SP)
3976 022722 012704 003130          MOV      #ERTABL,R4          ; GET START OF ERROR TABLE.
3977 022726 005003          CLR      R3          ; CLEAR UNIT NUMBER
3978 022730 011402          1$: MOV      (R4),R2          ; GET ERROR TABLE ENTRY & TEST IT.
3979 022732 001467          BEQ      4$          ; ZERO IF UNIT NOT RUN
3980 022734 100066          BPL      4$
3981 022736 032702 040000          BIT      #BIT14,R2          ; WAS UNIT DROPPED?
3982 022742 001015          BNE      2$          ; BR IF YES
3983 022744 042702 170000          BIC      #C7777,R2          ; GET ERROR COUNT FIELD
3984 022750          PRINTS  #DEVONL,R3,R2          ; PRINT
          022750 010246          MOV      R2,-(SP)
          022752 010346          MOV      R3,-(SP)
          022754 012746 023173          MOV      #DEVONL,-(SP)
          022760 012746 000003          MOV      #3,-(SP)
          022764 010600          MOV      SP,R0
          022766 104416          TRAP      C$PNTS
          022770 062706 000010          ADD      #10,SP
3985 022774 000446          BR      4$
3986 022776 020227 160000          2$: CMP      R2,#160000          ; WAS UNIT NON-EXISTENT?
3987 023002 001012          BNE      3$          ; BR IF NO
3988 023004          PRINTS  #DEVNXR,R3
          023004 010346          MOV      R3,-(SP)
          023006 012746 023255          MOV      #DEVNXR,-(SP)
    
```



```

023012 012746 000002      MOV      #2,-(SP)
023016 010600             MOV      SP,R0
023020 104416             TRAP    C:PNTS
023022 062706 000006      ADD      #6,SP
3989 023026 000431             BR      4$
3990 023030 020227 160001    3$:    CMP      R2,#160001      ; WAS UNIT NOT READY AT STARTUP?
3991 023034 001012             BNE     30$              ; BR IF NO.
3992 023036             PRINTS  #DEVNRD,R3
023036 010346             MOV      R3,-(SP)
023040 012746 023337             MOV      #DEVNRD,-(SP)
023044 012746 000002             MOV      #2,-(SP)
023050 010600             MOV      SP,R0
023052 104416             TRAP    C:PNTS
023054 062706 000006      ADD      #6,SP
3993 023060 000414             BR      4$
3994 023062 042702 170000    30$:   BIC      #C7777,R2
3995 023066             PRINTS  #DEVDR0,R3,R2
023066 010246             MOV      R2,-(SP)
023070 010346             MOV      R3,-(SP)
023072 012746 023420             MOV      #DEVDR0,-(SP)
023076 012746 000003             MOV      #3,-(SP)
023102 010600             MOV      SP,R0
023104 104416             TRAP    C:PNTS
023106 062706 000010      ADD      #10,SP
3996 023112 062704 000002    4$:    ADD      #2,R4
3997 023116 005203             INC      R3
3998 023120 020427 003330    CMP      R4,#ERTABE
3999 023124 103701             BLO     1$
4000 023126 012604             MOV      (SP),R4
4001 023130 012603             MOV      (SP),R3
4002 023132 012602             MOV      (SP),R2
4003 023134             ENDRPT                    ; UNUSED.
023134             L10035:
023134 104425             TRAP    C:RPT
4004
4005
4006 023136      045      116      045  DEVSUM: .ASCIZ  /#N#ADEVICE STATUS SUMMARY:#N/
4007 023173      045      101      040  DEVONL: .ASCIZ  /#A UNIT #D3#A CONTROLLER READY, ERRORS = #D#N/
4008 023255      045      101      040  DEVNXR: .ASCIZ  /#A UNIT #D3#A DROPPED, NON-EXISTENT REGISTER#N/
4009 023337      045      101      040  DEVNRD: .ASCIZ  /#A UNIT #D3#A DROPPED, NOT READY AT STARTUP#N/
4010 023420      045      101      040  DEVDR0: .ASCIZ  /#A UNIT #D3#A DROPPED, ERRORS = #D#N/
4011
4014
4021
4027
4035

```


4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076 023470
023470
4077 023470 005037 002170
4078 023474 012737 005672 002146
4079 023502 005037 003100
4084 023506 012700 023704
4085 023512 004737 017232
4086 023516 012737 000005 002164
4087 023524
4088 023524 005003
4089
4090 023526
023526
023526 104402
4091
4092 023530
023530 104433
4093 023532 004737 016744

.SBTTL TEST 1: BUS RESET TEST

```

THIS TEST VERIFIES THAT THE 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 CONTROLLER. IF THE CONTROLLER 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.

BGNTST

```

CLR FATFLG ;CLEAR FATAL ERROR FLAG
MOV #EPRT1,EPRTSW ;SET UP ERROR MESSAGE SWITCH
CLR KTF LG ;HOLD OFF KT11
MOV #TST1ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
MOV #5,LOOPCNT ;PERFORM 5 ITERATIONS

T1LOOP: CLR R3 ;USE R3 AS FATAL ERROR FLAG

BGNSUB ;////////// BEGIN SUBTEST ////////////
T1.1: TRAP C#BSUB

BRESET ;ISSUE A BUS RESET TRAP C#RESET

JSR PC,WAITF ;WAIT FOR READY
    
```



```

4109 023574 005703          TST      R3          ;DID WE HAVE FATAL ERROR ?
4110 023576 001402          BEQ      20$         ;BRANCH IF NOT
4111 023600 004737 017776   JSR      PC,CKDROP  ;GO DROP THIS UNIT, IF ALLOWED
4112 023604 005003          CLR      R3          ;RESET FATAL ERROR FLAG
4113
4114
4115 023606          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      023606          T1.2:          TRAP      C$BSUB
      023606 104402
4116
4117 023610 005065 000000   CLR      TSSR(R5)   ;WRITE TO ISSUE A SOFT RESET
4118 023614 004737 016744   JSR      PC,WAITF   ;WAIT FOR READY TO SET
4119 023620 016501 000000   MOV      TSSR(R5),R1 ;GET REGISTER TSSR DATA
4120 023624 010102          MOV      R1,R2      ;START SETUP OF EXPECTED TSSR
4121 023626 042702 176277   BIC      @+C<HIADDR!OFL>,R2 ;CLEAR OUT UNUSED BITS
4122 023632 052702 002200   BIS      @SSR!NBA,R2 ;R4 HAS EXPECTED CONTENTS
4123 023636 020102          CMP      R1,R2      ;COMPARE EXPECTED TO RECEIVED
4124 023640 001405          BEQ      10$         ;BRANCH IF COMPARE
4128 023642          ERDF          ERRNO,SFIERR,SFFMSG ;REPORT A FATAL ERROR
      023642 104455          TRAP      C$ERDF
      023644 000146          .WORD    102
      023646 003550          .WORD    SFIERR
      023650 011566          .WORD    SFFMSG
4129 023652 005203          INC      R3          ;SET THE ERROR FLAG
4130 023654          10$:
4131 023654          ENDSUB          ;//////////////// END SUBTEST //////////////////
      023654 104403          L10040:        TRAP      C$ESUB
4132
4133
4134 023656 005703          TST      R3          ;FATAL ERROR DETECTED ?
4135 023660 001402          BEQ      20$         ;BRANCH IF NOT
4136 023662 004737 017776   JSR      PC,CKDROP  ;SEE IF TIME TO DROP UNIT
4137 023666 004737 017200   JSR      PC,TSTLOOP ;SHOULD WE DO ITERATIONS ?
4138 023672 103002          BCC      40$         ;BRANCH IF NOT
4139 023674 000137 023524   JMP      T1LOOP     ;LOOP UNTIL COUNT EXPIRED
4140 023700          40$:          EXIT      TST       ;ALL DONE THIS TEST
      023700 104432          TRAP      C$EXIT
      023702 000022          .WORD    L10036-.
4141
4142          ;+
4143          ;LOCAL TEXT MESSAGES FOR TEST
4144          ;-
4145
4146 023704          111      156      151 TST1ID: .ASCIZ 'Initialization'
4147          .EVEN
4148 023724          ENDTST
      023724          L10036:        TRAP      C$ETST
      023724 104401
4149
4150

```


4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199

.SBTTL TEST 2: RAM TEST

THIS TEST VERIFIES THAT ALL LOCATIONS OF THE RAM ON THE CONTROLLER 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 2, SUBTEST 1: -

THIS SUBTEST VERIFIES EACH RAM LOCATION BY FIRST PLACING THE CONTROLLER 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.

4200 023726
023726
4201 023726 005037 002170
4202 023732 012737 005672 002146
4203 023740 005037 003100
4204
4205 023744
023744
023744 104402
4206

BGNTST

CLR FATFLG ;CLEAR FATAL ERROR FLAG
MOV #EPRT1,EPRTSW ;SET UP ERROR MESSAGE SWITCH
CLR KTFLG ;HOLD OFF KT11

BGNSUB

T2::
;CLEAR FATAL ERROR FLAG
;SET UP ERROR MESSAGE SWITCH
;HOLD OFF KT11
////////// BEGIN SUBTEST //////////
T2.1: TRAP C#BSUB

```

4211 023746 012700 024672          MOV    #TST2ID,R0          ;ASCII MESSAGE TO IDENTIFY TEST
4212 023752 004737 017232          JSR    PC,TSTSETUP        ;DO INITIAL TEST SETUP
4213 023756 012737 000002 002164  MOV    #2,LOOPCNT        ;PERFORM 2 ITERATIONS
4214 023764                                T2LOOP:
4215 023764 004737 016470          JSR    PC,SOFINIT        ;DO INITIALIZE ON CONTROLLER
4216 023770 103405                    BCS    20$                ;BR IF INIT WAS OK
4220 023772 010001                    MOV    R0,R1              ;CONTENTS OF TSSR REGISTER
4221 023774                    ERRDF  ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
                                TRAP    C$ERDF
                                .WORD   201
                                .WORD   SFIERR
                                .WORD   SFIMSG
                                023774 104455
                                023776 000311
                                024000 003550
                                024002 011506
4222 024004 012704 000002 20$:  MOV    #2,R4                ;SET RAM ADDRESS AT TWO
4223 024010 25$:
4224
4225 024010 110402                    MOVB   R4,R2              ;EXPECTED DATA FROM WRAP-AROUND
4226 024012 110465 177777          MOVB   R4,TSDBH(R5)       ;LOAD ADDRESS INTO TSDB
4227 024016 110265 177776          MOVB   R2,TSDBL(R5)       ;LOADS DATA INTO RAM LOCATION
4228 024022 116501 177776          MOVB   TSBAL(R5),R1       ;READS WRAP DATA
4229 024026 120102                    CMPB   R1,R2              ;DOES WRITTEN(WRAP) = READ
4230 024030 001404                    BEQ    30$                ;BR IF OK, THEY ARE EQUAL
4234 024032                    ERRHRD ERRNO,TSBAM2,EXPREC ;DATA NOT WRAPPED CORRECTLY
                                TRAP    C$ERHRD
                                .WORD   202
                                .WORD   TSBAM2
                                .WORD   EXPREC
                                024032 104456
                                024034 000312
                                024036 024530
                                024040 016170
4235 024042 30$:
4236
4237 024042 005204                    INC    R4                  ;NEXT ADDRESS
4238 024044 020427 000400          CMP    R4,#400           ;END OF RAM MEMORY CHECK
4239 024050 001357                    BNE    25$                ;LOOP TILL ALL RAM WRITTEN
4240 024052 005002                    CLR    R2                  ;CLEAR OUT R2 HIGH BITS
4241 024054 005304                    DEC    R4                  ;SET BACK TO 377
4242 024056 110402 40$:  MOVB   R4,R2              ;GET DATA PATTERN BACK IN SHAPE
4243 024060 110465 177777          MOVB   R4,TSDBH(R5)       ;LOAD UP RAM ADDRESS POINTER
4244 024064 116501 177776          MOVB   TSBAL(R5),R1       ;READ RAM CONTENTS BACK
4245 024070 120102                    CMPB   R1,R2              ;CHECK WITH DATA WRITTEN
4246 024072 001404                    BEQ    45$                ;BR IF OK, DATA IN = DATA OUT
4250 024074                    ERRHRD ERRNO,TSBAM2,EXPREC ;WRITTEN DATA NOT = TO READ
                                TRAP    C$ERHRD
                                .WORD   203
                                .WORD   TSBAM2
                                .WORD   EXPREC
                                024074 104456
                                024076 000313
                                024100 024530
                                024102 016170
4251 024104 45$:  CKLOOP                    ;SCOPE LOOP
                                TRAP    C$CLP1
4252 024106 005304                    DEC    R4                  ;DROP DATA COUNTER (PATTERN)
4253 024110 022704 000002          CMP    #2,R4              ;AT LOC TWO YET
4254 024114 001360                    BNE    40$                ;BR, IF NOT AT TWO YET
4255
4256 024116                    ENDSUB                    ;////////// END SUBTEST ////////////
                                L10042:
                                TRAP    C$ESUB
                                024116 104403
4257

```

```

4259
4260 024120          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      024120          ;              T2.2:          TRAP      C$BSUB
      024120 104402          ;
4261          ;          TEST 2, SUBTEST 2
4262          ;
4263          ;
4264          ;          THIS SUBTEST WRITES RAM WITH ALL ZEROS
4265          ;          THEN WALKS AN ALL ONES WORD DOWN THROUGH MEMORY
4266          ;
4267 024122 004737 016470          JSR      PC,SOFINIT          ;DO INITIALIZE ON CONTROLLER
4268 024126 103405          BCS      20$          ;BR IF INIT WAS OK
4272 024130 010001          MOV      R0,R1          ;CONTENTS OF TSSR REGISTER
4273 024132          ERRDF      ERRNO,SFIERR,SFIMSG          ;FATAL ERROR TSSR WAS NOT OK
      024132 104455          ;              TRAP      C$ERDF
      024134 000314          ;              .WORD    204
      024136 003550          ;              .WORD    SFIERR
      024140 011506          ;              .WORD    SFIMSG
4274 024142 005002          20$:    CLR      R2          ;TEST DATA = 0
4275 024144 012704 000002          MOV      #2,R4          ;STARTING RAM ADDRESS = 2
4276 024150          25$:
4277
4278 024150 110465 177777          MOVB     R4,TSDBH(R5)          ;LOAD ADDRESS INTO TSDB
4279 024154 110265 177776          MOVB     R2,TSDBL(R5)          ;LOADS DATA INTO RAM LOCATION
4280 024160 116501 177776          MOVB     TSBAL(R5),R1          ;READS WRAP DATA
4281 024164 120102          CMPB     R1,R2          ;DOES WRITTEN(WRAP) = READ ?
4282 024166 001404          BEQ      30$          ;BR IF OK, THEY ARE EQUAL
4286 024170          ERRHRD     ERRNO,TSBAM2,EXPREC          ;DATA NOT WRAPPED CORRECTLY
      024170 104456          ;              TRAP      C$ERHRD
      024172 000315          ;              .WORD    205
      024174 024530          ;              .WORD    TSBAM2
      024176 016170          ;              .WORD    EXPREC
4287 024200          30$:
4288
4289 024200 005204          INC      R4          ;NEXT ADDRESS
4290 024202 020427 000400          CMP      R4,#400          ;END OF RAM MEMORY CHECK
4291 024206 001360          BNE      25$          ;BR, MORE RAM TO GO
4292
4293 024210 005304          35$:    DEC      R4          ;SET BACK TO 377
4294 024212 005002          CLR      R2          ;SET TO ALL ZEROS
4295 024214          40$:
4296 024214 110465 177777          MOVB     R4,TSDBH(R5)          ;LOAD UP THE ADDRESS FOR RAM
4297 024220 116501 177776          MOVB     TSBAL(R5),R1          ;READ THE RAM CONTENTS BACK
4298 024224 005002          CLR      R2          ;LOOKING FOR 000000 (EXPECTED)
4299 024226 120102          CMPB     R1,R2          ;BOTH SHOULD BE 00000000 BINARY
4300 024230 001404          BEQ      43$          ;BR, IF DATA IS GOOD
4304 024232          ERRHRD     ERRNO,TSBAM3,EXPREC          ;CHARACTERISTICS DATA NOT CORRECT
      024232 104456          ;              TRAP      C$ERHRD
      024234 000316          ;              .WORD    206
      024236 024612          ;              .WORD    TSBAM3
      024240 016170          ;              .WORD    EXPREC
4305 024242 012702 000377          43$:    MOV      #000377,R2          ;SET ALL ONES WORD
4306 024246 110465 177777          MOVB     R4,TSDBH(R5)          ;LOAD UP RAM ADDRESS POINTER
4307 024252 110265 177776          MOVB     R2,TSDBL(R5)          ;WRITE DATA INTO RAM
4308 024256 116501 177776          MOVB     TSBAL(R5),R1          ;READ RAM CONTENTS BACK
4309 024262 120102          CMPB     R1,R2          ;CHECK WITH DATA WRITTEN
4310 024264 001404          BEQ      45$          ;BR IF OK, DATA IN = DATA OUT

```



```

4314 024266          ERRHRD  ERRNO,TSBAM2,EXPREC      ;WRITTEN DATA NOT = TO READ
      024266 104456          TRAP                      C$ERHRD
      024270 000317          .WORD                    207
      024272 024530          .WORD                    TSBAM2
      024274 016170          .WORD                    EXPREC
4315 024276          45$:  CKLOOP                      ;SCOPE LOOP
      024276 104406          TRAP                      C$CLP1
4316 024300          DEC      R4                        ;DROP RAM ADDRESS POINTER
4317 024302 005304      CMP      #2,R4                 ;AT LOC 2 YET
4318 024306 001342      BNE      40$                   ;BR, IF NOT AT TWO YET
4319
4320 024310          ENDSUB                             ;////////////////// END SUBTEST ////////////////////
      024310          L10043:                          TRAP      C$ESUB
      024310 104403
4321

```


.SBTTL TEST 3: COMMAND REJECT

4401
4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457

```

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

```

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

```


4508	025072	011520		22\$:	MOV	#SC!NBA!SSR!TSREJ,R2		.WORD	PKTSSR
4509	025100	004737	102206		JSR	PC,CHKTSSR			
4510	025104	016501	000000		MOV	TSSR(R5),R1			
4511	025110	032701	000100		BIT	#OFL,R1			
4512	025114	001402			BEQ	25\$			
4513	025116	052702	000100		BIS	#OFL,R2			
4514	025122	020201		25\$:	CMP	R2,R1			
4515	025124	001404			BEQ	30\$			
4519	025126				ERRHRD	ERRNO,T3NBA,PKTSSR			
	025126	104456						TRAP	C\$ERHRD
	025130	000460						.WORD	304
	025132	025630						.WORD	T3NBA
	025134	011520						.WORD	PKTSSR
4520	025136			30\$:	CKLOOP				
	025136	104406							
4521	025140	004737	017060		JSR	PC,CHKTSSR		TRAP	C\$CLP1
4522	025144	016501	177776		MOV	TSBA(R5),R1			
4523	025150	010402			MOV	R4,R2			
4524	025152	020102			CMP	R1,R2			
4525	025154	001404			BEQ	35\$			
4529	025156				ERRHRD	ERRNO,T3TSBA,EXPREC			
	025156	104456						TRAP	C\$ERHRD
	025160	000461						.WORD	305
	025162	026071						.WORD	T3TSBA
	025164	016170						.WORD	EXPREC
4530									
4531									
4532	025166	004737	010354	35\$:	JSR	PC,CKRAM			
4533	025172	103404			BCS	40\$			
4537	025174				ERRHRD	ERRNO,PKTRAM,RAMERR			
	025174	104456						TRAP	C\$ERHRD
	025176	000462						.WORD	306
	025200	004643						.WORD	PKTRAM
	025202	016204						.WORD	RAMERR
4538	025204			40\$:	ENDSEG				
	025204								
	025204	104405							
4539	025206	011300			MOV	(R3),R0			
4540	025210	042700	177740		BIC	#177740,R0			
4541	025214	020027	000004		CMP	R0,#4			
4542	025220	001002			BNE	45\$			
4543	025222	062703	000002		ADD	#2,R3			
4544	025226	020327	003030	45\$:	CMP	R3,#TBLEND			
4545	025232	103002			BHIS	50\$			
4546	025234	000137	024754		JMP	5\$			
4547									
4548	025240			50\$:	ENDSUB				
	025240								
	025240	104403							
4549								TRAP	C\$ESUB
4550	025242	005737	002170		TST	FATFLG			
4551	025246	001402			BEQ	60\$			
4552	025250	004737	017776		JSR	PC,CKDROP			

```

;EXPECTED CONTENTS OF TSSR
;WAIT FOR READY, NON-AMBIGUOUS
;GET THE CONTENTS OF TSSR
;IS OFF-LINE BIT SET ?
;BRANCH IF NOT OFF-LINE
;SET OFF-LINE IN EXPECTED DATA
;DOES EXPECTED MATCH RECEIVED ?
;DOES EXPECTED MATCH RECEIVED?
;NBA NOT SET TO REJECT

TRAP C$ERHRD
.WORD 304
.WORD T3NBA
.WORD PKTSSR

;LOOP ON ERROR ?

TRAP C$CLP1
;WAIT FOR READY, NON-AMBIGUOUS
;GET TSBA REGISTER CONTENTS
;START OF THE PACKET
;COMPARE EXPECTED TO RECEIVED
;ERROR IF NOT EQUAL
;PRINT THE ERROR & EXPD/RECV

TRAP C$ERHRD
.WORD 305
.WORD T3TSBA
.WORD EXPREC

;SEE IF DATA IN RAM IS CORRECT
;BRANCH IF PACKET IN RAM IS CORRECT
;REPORT THE RAM ERROR(S)

TRAP C$ERHRD
.WORD 306
.WORD PKTRAM
.WORD RAMERR

;<<<<<<<<<<<<<<<< END SEGMENT 10000$
TRAP C$ESEG

;PACKET COMMAND WORD
;GET BITS 0-4
;DON'T TEST WRITE CHARACTERISTICS
;BRANCH IF OK
;GET NEXT WORD FROM DATA TABLE
;REACHED END OF TABLE ?
;BRANCH IF END OF TABLE
;CONTINUE TEST WITH NEW DATA

;////////// END SUBTEST L10046:
TRAP C$ESUB

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


4649 025626 052525 .WORD 052525

4650

4651

4652

4653

4654

4655

;;
;LOCAL TEXT MESSAGES FOR TEST
;*

4656	025630	103	157	155	T3NBA:	.ASCIZ	'Command Not Rejected'
4657	025655	103	157	156	T3SSR:	.ASCIZ	'Contents of TSSR Incorrect After Write Packet'
4658	025733	125	156	145	T3INT:	.ASCIZ	'Unexpected Interrupt Received On Write Packet'
4659	026011	105	170	160	T3NINT:	.ASCIZ	'Expected Interrupt Not Received On Write Packet'
4660	026071	111	156	143	T3TSBA:	.ASCIZ	'Incorrect TSBA Address After Packet Write'
4661	026143	103	157	155	TST3ID:	.ASCIZ	'Command Reject'

4662 .EVEN
4663 026162 .ENDTST

L10045: TRAP C\$ETST

026162 104401


```

4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872 026774          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      026774          ;              T4.3:          TRAP      C$BSUB
      026774 104402
4873
4874 026776          SETPRI  #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
      026776 012700 000000          ;              MOV      #PRI00,R0
      027002 104441          ;              TRAP      C$SPRI
4875 027004 012703 027314          ;START OF TEST DATA FOR SUBTEST
4876 027010 012704 027250          ;GET THE ADDRESS OF COMMAND PACKET
4877 027014 004737 030146          ;RESTORE PACKET TO STARTING VALUES
4878
4879
4880 027020 004737 016470          JSR      PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
4881 027024 103405          BCS     10$          ;BR IF SOFT INIT = OK
4885 027026 010001          MOV     R0,R1          ;SAVE CONTENTS OF TSSR
4886 027030          ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      027030 104455          ;              TRAP      C$ERDF
      027032 000634          ;              .WORD    412
      027034 003550          ;              .WORD    SFIERR
      027036 011506          ;              .WORD    SFIMSG
4887 027040 005037 002172          ;CLEAR INTERRUPT RECEIVED FLAG
4888 027044 052737 000001 027260 10$: CLR  INTRECV          ;MAKE ADDRESS ODD
4889 027052 010465 177776          BIS    #1,T4DATA          ;SET THE PACKET ADDRESS
4890 027056 004737 016744          MOV    R4,TSDB(R5)          ;SET THE PACKET ADDRESS
4891 027062 103405          JSR    PC,WAITF          ;WAIT FOR SSR TO SET
4892 027064 010001          BCS   15$          ;BR IF CARRY SET (GOOD RETURN)
4896 027066          MOV    R0,R1          ;SAVE CONTENTS OF TSSR
      027066 104455          ERRDF  ERRNO,T4SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      027070 000635          ;              TRAP      C$ERDF
      027072 027656          ;              .WORD    413
      027074 011520          ;              .WORD    T4SSR
4897 027076          ;              .WORD    PKTSSR
      027076 104406          15$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
4898 027100          ESCAPE  SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      027100 104410          ;              TRAP      C$CLP1
      027102 000116          ;              .WORD    C$ESCAPE
4899 027104 005737 002172          ;              .WORD    L10053-.
4900 027110 001404          TST   INTRECV          ;DID AN INTERRUPT OCCUR ?
4904 027112          BEQ   22$          ;BRANCH IF NOT
      027112 104456          ERRHRD ERRNO,T4INT,PKTSSR
      027114 000636          ;              TRAP      C$ERHRD
      027116 027745          ;              .WORD    414
      027120 011520          ;              .WORD    T4INT
4905 027122 016501 000000          ;              .WORD    PKTSSR
      027122 016501 000000          22$: MOV  TSSR(R5),R1          ;GET THE CONTENTS OF TSSR
4906 027126 012702 102206          MOV    #SC!SSR!TSREJ!NBA,R2 ;EXPECTED CONTENTS OF TSSR
4907 027132 032701 000100          BIT   #OFL,R1          ;IS OFF-LINE BIT SET ?
4908 027136 001402          BEQ   25$          ;BRANCH IF NOT OFF-LINE

```



```

4909 027140 052702 000100
4910 027144 020201
4911 027146 001414
4912 027150 010100
4913 027152
4914 027162 020027 002000
4915 027166 001404
4919 027170
      027170 104456
      027172 000637
      027174 027560
      027176 011520
4920 027200
      027200 104406
4921 027202 032701 002000
4922 027206 001004
4926 027210
      027210 104456
      027212 000640
      027214 027330
      027216 011520
4927
4928 027220
      027220
      027220 104403
4929
4930 027222 005737 002170
4931 027226 001402
4932 027230 004737 017776
4933 027234
4934 027234
      027234 104432
      027236 000756
4935
4936
4937
4938
4939
4941 027240
4943 027250
4944 027250 100004
4945 027252 027260
4946 027254 000000
4947 027256 000010
4948
4949 027260
4950 027260 027274
4951 027262 000000
4952 027264 000016
4953 027266 000000
4954 027270
4955
4956 027270 000000 000000
4957 027274
4958
4959
4960

      25$: BIS #OFL,R2 ;SET OFF-LINE IN EXPECTED DATA
          CMP R2,R1 ;DOES EXPECTED MATCH RECEIVED ?
          BEQ 30$ ;OKAY IF MATCH
          MOV R1,R0 ;DATA FROM TSSR
          XOR R2,R0 ;FIND BITS IN ERROR
          CMP R0,#NBA ;IS NBA ONLY BIT IN ERROR ?
          BEQ 30$ ;DON'T PRINT ERROR IF NBA ONLY BAD BIT
          ERRHRD ERRNO,T44REJ,PKTSSR ;COMMAND NOT REJECTED
                                     TRAP C$ERHRD
                                     .WORD 415
                                     .WORD T44REJ
                                     .WORD PKTSSR

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

          BIT #NBA,R1 ;IS NBA BIT SET ?
          BNE 35$ ;OKAY IF NBA SET
          ERRHRD ERRNO,T42NBA,PKTSSR ;NBA NOT SET
                                     TRAP C$ERHRD
                                     .WORD 416
                                     .WORD T42NBA
                                     .WORD PKTSSR

      35$: ENDSUB ;////////////////// END SUBTEST ////////////////////
                                     L10053: TRAP C$ESUB

          TST FATFLG ;ANY FATAL ERRORS ?
          BEQ 60$ ;BRANCH IF NOT
          JSR PC,CKDROP ;TRY TO DROP THE UNIT

      60$: EXIT TST ;ALL DONE THIS TEST
                                     TRAP C$EXIT
                                     .WORD L10050-.

;+
;LOCAL STORAGE FOR THIS TEST
;-

      T4PACKET: .BLKB 10-<.-TUV2A&7>
               .WORD 100004 ;COMMAND PACKET FOR TEST
               .WORD T4DATA ;WRITE CHARACTERISTICS COMMAND, WITH ACK
               .WORD 0 ;ADDRESS OF CHARACTERISTICS BLOCK
               .WORD 8. ;STARTING VALUE OF BLOCK SIZE

      T4DATA: .WORD T4BFR ;CHARACTERISTICS DATA BLOCK
              .WORD 0 ;ADDRESS OF MESSAGE BUFFER
              .WORD 14. ;LENGTH OF MESSAGE BUFFER
              .WORD 0

      T4SP:

      T4BFR: .WORD 0,0 ;SPACE
            .BLKW 8. ;MESSAGE BUFFER

;+
;

```

```

4961 ;TEST DATA FOR SUBTEST TWO
4962 ;
4963 ;DATA HAS FORMAT:
4964 ;
4965 ;      1ST WORD      OFFSET TO TEST WORD IN PACKET
4966 ;      2ND WORD      BITS TO SET FOR TEST
4967 ;
4968 ;-
4969
4970 T42DATA:
4971 .WORD 0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
4972 .WORD 2,BIT0
4973 .WORD 4,BIT6!BIT15
4974 T42DONE=.
4975
4976
4977 ;*
4978 ;LOCAL TEXT MESSAGES FOR TEST
4979 ;-
4980
4981 101 T42NBA: .ASCIZ 'NBA Not Set On Rejected WRITE CHARACTERISTICS'
4982 111 T4NBA: .ASCIZ 'WRITE CHARACTERISTICS Command Not Accepted'
4983 111 T42REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
4984 111 T44REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
4985 156 T4SSR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
4986 145 T4INT: .ASCIZ 'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
4987 143 T4TSBA: .ASCIZ 'Incorrect TSBA Address After WRITE CHARACTERISTICS'
4988 151 TST4ID: .ASCIZ 'Write Characteristics'
4989 .EVEN
4990

```

4970	027314			
4971	027314	000000	037140	
4972	027320	000002	000001	
4973	027324	000004	100100	
4974		027330		
4981	027330	116	102	101
4982	027406	127	122	111
4983	027461	127	122	111
4984	027560	127	122	111
4985	027656	103	157	156
4986	027745	125	156	145
4987	030034	111	156	143
4988	030117	127	162	151

4992
4993
4994
4995
4996
4997
4998
4999 030146
5000 030146
5001 030152 012701 027250
5002 030156 012721 100004
5003 030162 012721 027260
5004 030166 005021
5005 030170 012721 000010
5006 030174 012721 027274
5007 030200 005021
5008 030202 012721 000020
5009 030206 005021
5010 030210 005011
5011 030212 000207
5012 030214
030214 104401
5013

```

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;
;-
T4REST:
    SAVREG
    MOV    #T4PACKET,R1    ;SAVE THE REGISTERS
    MOV    #100004,(R1)+   ;START OF THE PACKET
    MOV    #T4DATA,(R1)+  ;WRITE CHARACTERISTICS WITH ACK
    CLR    (R1)+           ;ADDRESS OF CHAR DATA BLOCK
    MOV    #8,(R1)+       ;EXTENDED ADDRESS
    MOV    #T4BFR,(R1)+   ;SIZE OF DATA BLOCK IN BYTES
    CLR    (R1)+           ;ADDRESS OF MESSAGE BUFFER
    MOV    #16,(R1)+      ;LENGTH OF MESSAGE BUFFER
    CLR    (R1)+
    CLR    (R1)
    RTS    PC
    ENDTST
;RETURN

```

L10050: TRAP C#ETST

5074	030320	004737	017060			JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET		
5075	030324	103405				BCS	15#		;BR IF CARRY SET (GOOD RETURN)		
5076	030326	010001				MOV	R0,R1		;SAVE CONTENTS OF TSSR		
5080	030330					ERRDF	ERRNO,T5SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET		
	030330	104455							TRAP	C#ERDF	
	030332	000766							.WORD	502	
	030334	031221							.WORD	T5SSR	
	030336	011520							.WORD	PKTSSR	
5081	030340				15#:	CKLOOP			;LOOP ON ERROR, IF FLAG SET		
	030340	104406							TRAP	C#CLP1	
5082	030342					ESCAPE	TST		;EXIT IF FATAL ERROR		
	030342	104410							TRAP	C#ESCAPE	
	030344	001060							.WORD	L10054-.	
5083	030346	016203	000006			MOV	XSTO(R2),R3		;STORE STATUS FOR A WHILE		
5084	030352	020327	052525			CMP	R3,#052525		;CHECK FOR XSTATO OVER WRITTEN (GOOD!)		
5085	030356	001006				BNE	20#		;BR, IF XSTATO HAS BEEN UPDATED		
5086	030360	016501	000000			MOV	TSSR(R5),R1		;PICK UP TSSR FOR ERROR PRINTOUT		
5090	030364					ERRHRD	ERRNO,T5NMSG,PKTSSR		; "NO MESSAGE PACKET RETURNED"		
	030364	104456							TRAP	C#ERHRD	
	030366	000767							.WORD	503	
	030370	031310							.WORD	T5NMSG	
	030372	011520							.WORD	PKTSSR	
5091	030374	032762	000020	000006	20#:	BIT	#XSOVCK,XSTO(R2)		;IS VOLUME CHECK CLEAR IN XSTO ?		
5092	030402	001006				BNE	23#		;OKAY IF VOLUME CHECK IS CLEARED		
5096	030404	016501	000000			MOV	TSSR(R5),R1		;CONTENTS OF TSSR FOR ERROR REPORT		
5097	030410					ERRHRD	ERRNO,T5VCK2,PKTMES		;VOLUME CHECK NOT SET		
	030410	104456							TRAP	C#ERHRD	
	030412	000770							.WORD	504	
	030414	031055							.WORD	T5VCK2	
	030416	011574							.WORD	PKTMES	
5098	030420				23#:	CKLOOP			;LOOP ON ERROR ?		
	030420	104406							TRAP	C#CLP1	
5099	030422	010465	177776			MOV	R4,T5DB(R5)		;SET THE PACKET ADDRESS FOR WRITE CHAR		
5100	030426	004737	017060			JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET		
5101	030432	103405				BCS	25#		;BR IF CARRY SET (GOOD RETURN)		
5102	030434	010001				MOV	R0,R1		;SAVE CONTENTS OF TSSR		
5106	030436					ERRDF	ERRNO,T5SSR,PKTSSR		;DEVICE FATAL SSR FAILED TO SET		
	030436	104455							TRAP	C#ERDF	
	030440	000771							.WORD	505	
	030442	031221							.WORD	T5SSR	
	030444	011520							.WORD	PKTSSR	
5107	030446				25#:	CKLOOP			;LOOP ON ERROR, IF FLAG SET		
	030446	104406							TRAP	C#CLP1	
5108	030450					ESCAPE	TST		;EXIT IF FATAL ERROR		
	030450	104410							TRAP	C#ESCAPE	
	030452	000752							.WORD	L10054-.	
5109	030454	026203	000006			CMP	XSTO(R2),R3		;THE XSTO SHOULD NOT HAVE CHANGED		
5110	030460	001406				BEQ	27#		;OKAY IF VOLUME CHECK IS SET		
5114	030462	016501	000000			MOV	TSSR(R5),R1		;CONTENTS OF TSSR FOR ERROR REPORT		
5115	030466					ERRHRD	ERRNO,T5NVCK,PKTMES		;VOLUME CHECK NOT SET		
	030466	104456							TRAP	C#ERHRD	
	030470	000772							.WORD	506	
	030472	031131							.WORD	T5NVCK	
	030474	011574							.WORD	PKTMES	
5116	030476				27#:	CKLOOP			;LOOP ON ERROR ?		
	030476	104406							TRAP	C#CLP1	
5117	030500	032762	000020	000006	30#:	BIT	#XSOVCK,XSTO(R2)		;IS VOLUME CHECK SET IN XSTO ?		


```

5203 .SBTTL TEST 6: COMPLETION INTERRUPT
5204
5205
5206 ;
5207 ; THIS TEST VERIFIES THAT AN INTERRUPT IS GENERATED AT THE
5208 ; COMPLETION OF THE WRITE CHARACTERISTICS COMMAND IF THE INTERRUPT
5209 ; ENABLE (IE) BIT IN THE COMMAND HEADER WORD IS SET. THIS TEST
5210 ; CHECKS THE FUNCTIONING OF THE INTERRUPT LOGIC AND BASIC
5211 ; PROCESSING OF THE IE BIT.
5212 ;
5213 ; THE SEQUENCES OF TEST 7 ARE REPEATED, EXCEPT THAT THE INTERRUPT
5214 ; SERVICE ROUTINE IS SET UP TO EXPECT INTERRUPTS AND EACH WRITE
5215 ; CHARACTERISTICS COMMAND IS ISSUED WITH THE IE BIT SET (1). IT
5216 ; IS VERIFIED, WHERE APPROPRIATE, THAT THE IE STATUS BIT IN XSTO
5217 ; OF ANY MESSAGE PACKET IS SET AND THAT A COMPLETION INTERRUPT IS
5218 ; GENERATED. FINALLY, A SEQUENCE OF TWO COMMANDS ARE ISSUED, THE
5219 ; FIRST WITH IE=1 AND THE SECOND WITH IE=0. IT IS VERIFIED THAT
5220 ; NO INTERRUPT IS GENERATED AFTER THE SECOND COMMAND AND THAT THE
5221 ; IE BIT IN XSTO IS 0.
5222 ;
5222 031426 BGNTST
5223 031426
5223 031426 005037 002170 CLR FATFLG ;CLEAR FATAL ERROR FLAG
5224 031432 012737 005672 002146 MOV #EPRT1,EPRTSW ;SET UP ERROR MESSAGE SWITCH
5225 031440 005037 003100 CLR KTFLG ;HOLD OFF KT11
5230 031444 012700 033401 MOV #TST6ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
5231 031450 004737 017232 JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
5232 031454 012737 000002 002164 MOV #2.,LOOPCNT ;PERFORM 2 ITERATIONS
5233 031462 T6LOOP:
5234 031462 BGNSUB ;////////// BEGIN SUBTEST //////////
5235 031462 104402 T6.1: TRAP C#BSUB
5236 031464 004737 033426 JSR PC,T6REST ;SET PACKET TO INITIAL VALUES
5237 031470 012700 000000 SETPRI #PRI00 ;LOWER PRIORITY TO ALLOW INTERRUPTS
5238 031476 012703 002732 MOV #TSTBLK+10.,R3 ;START OF TEST DATA
5239 031502 012704 032330 MOV #T6PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
5240 031506 012764 000010 000006 MOV #8.,PKBCNT(R4) ;START WITH MINIMUM ALLOWABLE VALUE
5241 031514 5$:
5242 031514 104404 BGNSEG ;>>>>>>>>>> BEGIN SEGMENT >>>>>>>>>>
5243 031516 004737 016470 JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
5244 031522 103405 BCS 10$ ;BR IF SOFT INIT = OK
5249 031524 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
5250 031526 104455 ERDFF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
5251 031536 005037 002170 10$: CLR FATFLG ;CLEAR FATAL ERROR FLAG
5252 031542 005037 002172 CLR INTRECV ;CLEAR INTERRUPT RECEIVED FLAG
5253 031546 010465 177776 MOV R4,TSD(B(R5)) ;SET THE PACKET ADDRESS
5254 031552 004737 017060 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
5255 031556 103407 BCS 15$ ;BR IF CARRY SET (GOOD RETURN)
5256 031560 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR

```



```

5362          ;*
5363          ;
5364          ;TEST 6, SUBTEST 3
5365          ;
5366          ;SUBTEST TO VERIFY THAT A WRITE CHARACTERISTICS COMMAND IS
5367          ;REJECTED IF AN ILLEGAL DATA BLOCK ADDRESS IS ISSUED.
5368          ;
5369          ;
5370          ;-
5371 032126          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
          032126          T6.3:
          032126 104402          TRAP          C$BSUB
5372
5373 032130          SETPRI  #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
          032130 012700 000000          MOV          #PRI00,R0
          032134 104441          TRAP          C$SPRI
5374 032136 012703 032372          ;START OF TEST DATA FOR SUBTEST
5375 032142 012704 032330          5$: MOV          #T6DATA,R3          ;GET THE ADDRESS OF COMMAND PACKET
5376 032146 004737 033426          JSR          PC,T6REST          ;RESTORE PACKET TO STARTING VALUES
5377
5378
5379 032152 004737 016470          JSR          PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
5380 032156 103405          BCS          10$          ;BR IF SOFT INIT = OK
5384 032160 010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
5385 032162          ERRDF          ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL ERROR DURING INIT
          032162 104455          TRAP          C$ERDF
          032164 001141          .WORD          609
          032166 003550          .WORD          SFIERR
          032170 011506          .WORD          SFIMSG
5386 032172 005037 002172          10$: CLR          INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
5387 032176 052737 000001 032340 BIS          #1,T6DATA          ;MAKE ADDRESS ODD
5388 032204 010465 177776          MOV          R4,TSDB(R5)          ;SET THE PACKET ADDRESS
5389 032210 004737 016744          JSR          PC,WAITF          ;WAIT FOR SSR TO SET
5390 032214 103405          BCS          15$          ;BR IF CARRY SET (GOOD RETURN)
5391 032216 010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
5395 032220          ERRDF          ERRNO,T6SSR,PKTSSR          ;DEVICE FATAL SSR FAILED TO SET
          032220 104455          TRAP          C$ERDF
          032222 001142          .WORD          610
          032224 033047          .WORD          T6SSR
          032226 011520          .WORD          PKTSSR
5396 032230          15$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
          032230 104406          TRAP          C$CLP1
5397 032232          ESCAPE          SUB          ;BY-PASS SUBTEST IF FATAL ERROR
          032232 104410          TRAP          C$ESCAPE
          032234 000056          .WORD          L10060-.
5398 032236 005737 002172          TST          INTRECV          ;DID AN INTERRUPT OCCUR ?
5399 032242 001004          BNE          22$          ;BRANCH IF YES
5403 032244          ERRHRD          ERRNO,T6NINT,PKTSSR
          032244 104456          TRAP          C$ERHRD
          032246 001143          .WORD          611
          032250 033136          .WORD          T6NINT
          032252 011520          .WORD          PKTSSR
5404 032254 016501 000000          22$: MOV          TSSR(R5),R1          ;GET THE CONTENTS OF TSSR
5405 032260 012702 102206          MOV          #SC!SSR!TSREJ!NBA,R2          ;EXPECTED CONTENTS OF TSSR
5406 032264 032701 000100          BIT          #OFL,R1          ;IS OFF-LINE BIT SET ?
5407 032270 001402          BEQ          25$          ;BRANCH IF NOT OFF-LINE
5408 032272 052702 000100          BIS          #OFL,R2          ;SET OFF LINE IN EXPECTED DATA

```



```

5409 032276 020201          25$:  CMP      R2,R1          ;DOES EXPECTED MATCH RECEIVED ?
5410 032300 001404          BEQ      30$          ;OKAY IF MATCH
5414 032302          ERRHRD  ERRNO,T64REJ,PKTSSR ;COMMAND NOT REJECTED
      032302 104456          TRAP     C$ERHRD
      032304 001144          .WORD   612
      032306 032653          .WORD   T64REJ
      032310 011520          .WORD   PKTSSR
5415 032312          30$:
5416
5417 032312          ENDSUB          ;////////////////// END SUBTEST ////////////////////
      032312          L10060:
      032312 104403          TRAP     C$ESUB
5418

```

```

5420
5421 032314          EXIT   TST          ;ALL DONE THIS TEST
      032314 104432
      032316 001162          TRAP      C$EXIT
                                   .WORD   L10055-.
5422
5423
5424      ;*
5425      ;LOCAL STORAGE FOR THIS TEST
5426      ;
5428 032320          .BLKB   10-<.-TUV2A&7>
5430 032330          T6PACKET:          ;COMMAND PACKET FOR TEST
      032330 100204          .WORD   100204          ;WRITE CHAR COMMAND, WITH IE, ACK
      032332 032340          .WORD   T6DATA          ;ADDRESS OF CHARACTERISTICS BLOCK
      032334 000000          .WORD   0
      032336 000010          .WORD   8.              ;STARTING VALUE OF BLOCK SIZE
5435
5436 032340          T6DATA:          ;CHARACTERISTICS DATA BLOCK
      032340 032352          .WORD   T6BFR          ;ADDRESS OF MESSAGE BUFFER
      032342 000000          .WORD   0
      032344 000016          .WORD   14.             ;LENGTH OF MESSAGE BUFFER
      032346 000000 000000          .WORD   0,0
5441
5442 032352          T6BFR:  .BLKW   8.              ;MESSAGE BUFFER
5443
5444      ;*
5445      ;
5446      ;TEST DATA FOR SUBTEST TWO
5447      ;
5448      ;DATA HAS FORMAT:
5449      ;
5450      ;          1ST WORD      OFFSET TO TEST WORD IN PACKET
5451      ;          2ND WORD      BITS TO SET FOR TEST
5452      ;
5453      ;-
5454
5455 032372          T62DATA:
5456 032372 000000 036140          .WORD   0,BIT5!BIT6!BIT6!BIT10!BIT11!BIT12!BIT13
5457 032376 000002 000001          .WORD   2,BIT0
5458 032402 000004 100100          .WORD   4,BIT6!BIT15
5459          032406          T62DONE=.
5460
5461
5462      ;*
5463      ;LOCAL TEXT MESSAGES FOR TEST
5464      ;-
5465
5466 032406          127      122      111  T6NBA:  .ASCIZ  'WRITE CHARACTERISTICS Command Not Accepted'
5467 032461          127      122      111  T62REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
5468 032560          127      122      111  T63REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
5469 032653          127      122      111  T64REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
5470 032751          127      122      111  T65REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
5471 033047          103      157      156  T6SSR:  .ASCIZ  'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
5472 033136          105      170      160  T6NINT: .ASCIZ  'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
5473 033227          125      156      145  T6INT:  .ASCIZ  'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
5474 033316          111      156      143  T6TSBA: .ASCIZ  'Incorrect TSBA Address After WRITE CHARACTERISTICS'
5475 033401          103      157      155  TST6ID: .ASCIZ  'Completion Interrupt'
5476          .EVEN
  
```

```

5478
5479
5480
5481
5482
5483
5484
5485 033426
5486 033426
5487 033432 012701 032330
5488 033436 012721 100204
5489 033442 012721 032340
5490 033446 005021
5491 033450 012721 000010
5492 033454 012721 032352
5493 033460 005021
5494 033462 012721 000016
5495 033466 005021
5496 033470 005011
5497 033472 005037 032352
5498 033476 000207
5499 033500
      033500
      033500 104401

```

```

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;
;-
T6REST:
  SAVREG          ;SAVE THE REGISTERS
  MOV             #T6PACKET,R1 ;START OF THE PACKET
  MOV             #100204,(R1)+ ;WRITE CHARACTERISTICS WITH ACK, IE
  MOV             #T6DATA,(R1)+ ;ADDRESS OF CHAR DATA BLOCK
  CLR             (R1)+         ;EXTENDED ADDRESS
  MOV             #8.,(R1)+     ;SIZE OF DATA BLOCK IN BYTES
  MOV             #T6BFR,(R1)+ ;ADDRESS OF MESSAGE BUFFER
  CLR             (R1)+
  MOV             #14.,(R1)+    ;LENGTH OF MESSAGE BUFFER
  CLR             (R1)+
  CLR             (R1)
  CLR             T6BFR        ;CLEAR 1ST LOC IN MESSAGE BUFFER
  RTS             PC          ;RETURN
  ENDTST

```

```

L10055: TRAP C$ETST

```



```

5600 034032 032701 000100
5601 034036 001402
5602 034040 052702 000100
5603 034044 020201
5604 034046 001404
5608 034050
      034050 104456
      034052 001303
      034054 036270
      034056 011520
5609 034060
5610 034060 013701 035662
5611 034064 012702 025252
5612 034070 020102
5613 034072 001404
5617 034074
      034074 104456
      034076 001304
      034100 035744
      034102 016170
5618 034104
5619 034104 005737 002170
5620 034110 001403
5621 034112 004737 017776
5622 034116
      034116
      034116 104405
5623 034120
5624 034120
      034120
      034120 104403
5625

```

```

55$: BIT #0FL,R1
      BEQ 55$
      BIS #0FL,R2
      CMP R2,R1
      BEQ 60$
      ERRHRD ERRNO,T7SSR,PKTSSR

60$: MOV T7BFR,R1
      MOV #025252,R2
      CMP R1,R2
      BEQ 70$
      ERRHRD ERRNO,T7MBF,EXPREC

70$: TST FATFLG
      BEQ 80$
      JSR PC,CKDROP
      ENDSEG

80$: ENDSUB

```

```

;IS OFF-LINE BIT SET ?
;BRANCH IF NOT OFF-LINE
;SET OFF-LINE IN EXPECTED DATA
;DOES EXPECTED MATCH RECEIVED ?
;OKAY IF MATCH
;"TSSR INCORRECT AFTER WRT CHARA."
      TRAP C$ERHRD
      .WORD 707
      .WORD T7SSR
      .WORD PKTSSR

;PICK UP THE 1ST WORD OF MESSAGE BUFFER
;SET UP EXPECTED DATA
;WAS ANY MESSAGE REC'D
;BR, IF OK (EQUAL)
;MESSAGE BUFFER WAS MODIFIED
      TRAP C$ERHRD
      .WORD 708
      .WORD T7MBF
      .WORD EXPREC

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

;////////// END SUBTEST ////////////
      L10062: TRAP C$ESUB

```



```

5781 034664 001402           BEQ       25$              ;BRANCH IF NOT OFF-LINE
5782 034666 052702 000100    BIS       #OFL,R2         ;SET OFF-LINE IN EXPECTED DATA
5783 034672 020201           CMP       R2,R1           ;DOES EXPECTED MATCH RECEIVED ?
5784 034674 001404           BEQ       30$              ;OKAY IF MATCH
5788 034676           ERRHRD   ERRNO,T7SSR,PKTSSR ;NBA NOT ZERO
                                 TRAP       C$ERHRD
                                 .WORD     720
5789 034706           ENDSEG
5790 034706           BGNSEG
5791 034710           JSR      PC,T7RST        ;RESET PACKETS AND COMMANDS
5792 034712 004737 036566     CLR      INTRECV         ;CLEAR INTERRUPT RECEIVED FLAG
5793 034716 005037 002172     MOV      #025252,T7BFR   ;WIPE OUT MESSAGE BUFFER AREA
5794 034722 012737 025252 035662 MOV      #100212,(R4)     ;SET COMMAND PACKET TO MESS BUF REL
5795 034730 012714 100212     MOV      R4,TSDB(R5)    ;SET THE PACKET ADDRESS
5796 034734 010465 177776     JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
5797 034740 004737 017060     BCS     45$              ;BR IF CARRY SET (GOOD RETURN)
5798 034744 103407           MOV      R0,R1           ;SAVE CONTENTS OF TSSR
5799 034746 010001           ERRDF   ERRNO,T7SSRM,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
5800 034750           JSR      PC,FATCHK       ;INC AND CHECK FOR MORE THAN 25 ERRORS
5801 034752 001321           CKLOOP
5802 034754 036200           TST     INTRECV         ;DID AN INTERRUPT OCCUR ?
5803 034756 011520           BEQ     52$              ;BRANCH IF NO
5804 034760 004737 017724     ERRHRD   ERRNO,T7INT,PKTSSR ;NBA NOT SET
5805 034764           TRAP    C$ERHRD
5806 034766 005737 002172     .WORD   721
5807 034772 001404           .WORD   T7SSRM
5808 034774           .WORD   PKTSSR
5809 034776 001322           JSR      PC,INTRECV
5810 035000 036450           TST     INTRECV
5811 035002 011520           BEQ     55$
5812 035004 016501 000000     ERRHRD   ERRNO,T7INT,PKTSSR ;NBA NOT SET
5813 035010 012702 000200     TRAP    C$ERHRD
5814 035014 032701 000100     .WORD   722
5815 035020 001402           .WORD   T7INT
5816 035022 052702 000100     .WORD   PKTSSR
5817 035026 020201           MOV      TSSR(R5),R1     ;GET THE CONTENTS OF TSSR
5818 035030 001404           MOV      #SSR,R2        ;EXPECTED CONTENTS OF TSSR
5822 035032           BIT     #OFL,R1         ;IS OFF-LINE BIT SET ?
5823 035042           BEQ     55$              ;BRANCH IF NOT OFF-LINE
5824 035046           BIS     #OFL,R2         ;SET OFF-LINE IN EXPECTED DATA
5825 035052 020102           CMP     R2,R1           ;DOES EXPECTED MATCH RECEIVED ?
5826 035054 001404           BEQ     60$              ;OKAY IF MATCH
5827 035056           ERRHRD   ERRNO,T7SSR,PKTSSR ;NBA NOT SET
5831 035056           TRAP    C$ERHRD
                                 .WORD   723
                                 .WORD   T7SSR
                                 .WORD   PKTSSR
5832 035056           MOV     T7BFR,R1        ;PICK UP THE 1ST WORD OF MESSAGE BUFFER
5833 035056           MOV     #025252,R2     ;SET UP EXPECTED DATA
5834 035056           CMP     R1,R2          ;WAS ANY MESSAGE REC'D
5835 035056           BEQ     70$            ;BR, IF OK (EQUAL)
5836 035056           ERRHRD   ERRNO,T7MBF,EXPREC ;MESSAGE BUFFER WAS MODIFIED
5837 035056           TRAP    C$ERHRD

```


035060	001324						.WORD	724
035062	035744						.WORD	T7MBF
035064	016170						.WORD	EXPREC
5832								
5833	035066	104406	70\$:	CK'LOOP				
	035066	104406					TRAP	C\$CLP1
5834								
5835	035070	005037		CLR	INTRECV			
5836	035074	004737		JSR	PC,T7RST			
5837	035100	042714		BIC	#100000,(R4)			
5838	035104	010465		MOV	R4,TSDB(R5)			
5839	035110	004737		JSR	PC,CHKTSSR			
5840	035114	103407		BCS	75\$			
5841	035116	010001		MOV	R0,R1			
5845	035120			ERRDF	ERRNO,T7SSR,PKTSSR			
	035120	104455					TRAP	C\$ERDF
	035122	001325					.WORD	725
	035124	036270					.WORD	T7SSR
	035126	011520					.WORD	PKTSSR
5846	035130	004737	017724	JSR	PC,FATCHK			
5847	035134		75\$:	CKLOOP				
	035134	104406					TRAP	C\$CLP1
5848	035136			ESCAPE	SEG			
	035136	104410					TRAP	C\$ESCAPE
	035140	000062					.WORD	10001\$-.
5849	035142	005737	002172	TST	INTRECV			
5850	035146	001006		BNE	82\$			
5854	035150	016500	000000	MOV	TSSR(R5),R0			
5855	035154			ERRHRD	ERRNO,T7NINT,PKTSSR			
	035154	104456					TRAP	C\$ERHRD
	035156	001326					.WORD	726
	035160	036357					.WORD	T7NINT
	035162	011520					.WORD	PKTSSR
5856	035164	016501	000000	82\$:	MOV	TSSR(R5),R1		
5857	035170	012702	000200		MOV	#SSR,R2		
5858	035174	032701	000100		BIT	#OFL,R1		
5859	035200	001402			BEQ	85\$		
5860	035202	052702	000100		BIS	#OFL,R2		
5861	035206	020201		85\$:	CMP	R2,R1		
5862	035210	001404			BEQ	90\$		
5866	035212				ERRHRD	ERRNO,T7SSR,PKTSSR		
	035212	104456					TRAP	C\$ERHRD
	035214	001327					.WORD	727
	035216	036270					.WORD	T7SSR
	035220	011520					.WORD	PKTSSR
5867	035222		90\$:					
5868	035222			ENDSEG				
	035222							
	035222	104405					TRAP	C\$ESEG
5869	035224	005737	002170	TST	FATFLG			
5870	035230	001403		BEQ	95\$			
5871	035232	004737	017776	JSR	PC,CKDROP			
5872								
5873	035236			BGNSEG				
	035236	104404					TRAP	C\$BSEG
5874	035240	005037	002172	95\$:	CLR	INTRECV		
5875	035244	004737	036566		JSR	PC,T7RST		


```

5979
5980      ;+
5981      ;LOCAL STORAGE FOR THIS TEST
5982      ;-
5984 035636      .BLKB    10-<.-TUV2A&7>
5986 035640      T7PACKET:
5987 035640 100204      .WORD    100204      ;COMMAND PACKET FOR TEST
5988 035642 035650      .WORD    T7DATA      ;WRITE CHAR COMMAND, WITH IE, ACK
5989 035644 000000      .WORD    0            ;ADDRESS OF CHARACTERISTICS BLOCK
5990 035646 000010      .WORD    8.          ;STARTING VALUE OF BLOCK SIZE
5991
5992 035650      T7DATA:
5993 035650 035662      .WORD    T7BFR      ;CHARACTERISTICS DATA BLOCK
5994 035652 000000      .WORD    0            ;ADDRESS OF MESSAGE BUFFER
5995 035654 000016      .WORD    14.         ;LENGTH OF MESSAGE BUFFER
5996 035656 000000 000000      .WORD    0,0
5997
5998 035662      T7BFR: .BLKW    8.          ;MESSAGE BUFFER
5999
6000      ;+
6001      ;
6002      ;TEST DATA FOR SUBTEST FOUR
6003      ;
6004 035702      T7PKT:
6005 035702 100204      .WORD    100204      ;COMMAND PACKET FOR TEST
6006 035704 035712      .WORD    T7DTA      ;WRITE CHAR COMMAND, WITH IE, ACK
6007 035706 000000      .WORD    0            ;ADDRESS OF CHARACTERISTICS BLOCK
6008 035710 000010      .WORD    8.          ;STARTING VALUE OF BLOCK SIZE
6009
6010 035712      T7DTA:
6011 035712 035724      .WORD    T7BUFR     ;CHARACTERISTICS DATA BLOCK
6012 035714 000000      .WORD    0            ;ADDRESS OF MESSAGE BUFFER
6013 035716 000016      .WORD    14.         ;LENGTH OF MESSAGE BUFFER
6014 035720 000000 000000      .WORD    0,0
6015
6016 035724      T7BUFR: .BLKW    8.          ;MESSAGE BUFFER
6017
6018      ;+
6019      ;LOCAL TEXT MESSAGES FOR TEST
6020      ;-
6021
6022
6023 035744      115      145      163  T7MBF: .ASCIZ  'Message Buffer Modified after MESSAGE BUFFER RELEASE Command'
6024 036041      116      102      101  T7NBA: .ASCIZ  'NBA Not Clear After WRITE CHARACTERISTICS Command'
6025 036123      116      102      101  T7NNBA: .ASCIZ 'NBA Set After MESSAGE BUFFER RELEASE Command'
6026
6027 036200      103      157      156  T7SSRM: .ASCIZ  'Contents Of TSSR Incorrect After Message Buffer Release'
6028 036270      103      157      156  T7SSR: .ASCIZ  'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
6029 036357      105      170      160  T7NINT: .ASCIZ  'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
6030 036450      125      156      145  T7INT: .ASCIZ  'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
6031 036537      102      141      163  TST7ID: .ASCIZ  'Basic Packet Protocol'
6032      .EVEN
6033

```

```

6035
6036
6037
6038
6039
6040
6041
6042 036566
6043 036566
6044 036572 012701 035640
6045 036576 012721 100204
6046 036602 012721 035650
6047 036606 005021
6048 036610 012721 000010
6049 036614 012721 035662
6050 036620 005021
6051 036622 012721 000016
6052 036626 005021
6053 036630 005011
6054 036632 005037 035662
6055 036636 000207
6056
6057
6058
6059
6060
6061
6062 036640
6063 036640
6064 036644 012701 035702
6065 036650 012721 100204
6066 036654 012721 035712
6067 036660 005021
6068 036662 012721 000010
6069 036666 012721 035724
6070 036672 005021
6071 036674 012721 000016
6072 036700 005021
6073 036702 005011
6074 036704 005037 035724
6075 036710 000207
6076 036712
      036712
      036712 104401

```

```

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;
;-
T7RST:
      SAVREG                ;SAVE THE REGISTERS
      MOV      #T7PACKET,R1 ;START OF THE PACKET
      MOV      #100204,(R1)+ ;WRITE CHARACTERISTICS WITH ACK, IE
      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      #14,(R1)+    ;LENGTH OF MESSAGE BUFFER
      CLR      (R1)+
      CLR      (R1)
      CLR      T7BFR        ;CLEAR 1ST LOC IN MESSAGE BUFFER
      RTS      PC           ;RETURN
;+
;
;ROUTINE TO RESTORE COMMAND PACKET #2 TO START-UP (DEFAULT) VALUES
;
;-
T7RT2:
      SAVREG                ;SAVE THE REGISTERS
      MOV      #T7PKT,R1    ;START OF THE PACKET
      MOV      #100204,(R1)+ ;WRITE CHARACTERISTICS WITH ACK, IE
      MOV      #T7DTA,(R1)+ ;ADDRESS OF CHAR DATA BLOCK
      CLR      (R1)+         ;EXTENDED ADDRESS
      MOV      #8,(R1)+     ;SIZE OF DATA BLOCK IN BYTES
      MOV      #T7BUFR,(R1)+ ;ADDRESS OF MESSAGE BUFFER
      CLR      (R1)+
      MOV      #14,(R1)+    ;LENGTH OF MESSAGE BUFFER
      CLR      (R1)+
      CLR      (R1)
      CLR      T7BUFR        ;CLEAR 1ST LOC IN MESSAGE BUFFER
      RTS      PC           ;RETURN
      ENDTST

```

```

L10061: TRAP C$ETST

```



```

6248 037436          EXIT   TST          ;ALL DONE THIS TEST
      037436 104432
      037440 000770          TRAP      C#EXIT
                                   .WORD  L10066-.
6249
6250          ;*
6251          ;LOCAL STORAGE FOR THIS TEST
6252          ;-
6253
6255 037442          .BLKB   10-<.-TUV2A&7>
6257 037450          T8PACKET:
6258 037450 100204          .WORD   100204          ;COMMAND PACKET FOR TEST
6259 037452 037460          .WORD   T8DATA          ;WRITE CHAR COMMAND, WITH IE, ACK
6260 037454 000000          .WORD   0              ;ADDRESS OF CHARACTERISTICS BLOCK
6261 037456 000010          .WORD   8.              ;STARTING VALUE OF BLOCK SIZE
6262
6263 037460          T8DATA:          ;CHARACTERISTICS DATA BLOCK
6264 037460 037472          .WORD   T8BFR          ;ADDRESS OF MESSAGE BUFFER
6265 037462 000000          .WORD   0
6266 037464 000016          .WORD   14.            ;LENGTH OF MESSAGE BUFFER
6267 037466 000000 000000 .WORD   0,0
6268
6269 037472          T8BFR:   .BLKW   8.          ;MESSAGE BUFFER
6270
6271
6273 037512          .BLKB   10-<.-TUV2A&7>
6275 037520          T8PK2:
6276 037520 100204          .WORD   100204          ;COMMAND PACKET FOR TEST
6277 037522 037530          .WORD   T8DTA          ;WRITE CHAR COMMAND, WITH IE, ACK
6278 037524 000000          .WORD   0              ;ADDRESS OF CHARACTERISTICS BLOCK
6279 037526 000010          .WORD   8.              ;STARTING VALUE OF BLOCK SIZE
6280
6281 037530          T8DTA:          ;CHARACTERISTICS DATA BLOCK
6282 037530 037542          .WORD   T8BF2          ;ADDRESS OF MESSAGE BUFFER
6283 037532 000000          .WORD   0
6284 037534 000016          .WORD   14.            ;LENGTH OF MESSAGE BUFFER
6285 037536 000000 000000 .WORD   0,0
6286
6287 037542          T8BF2:   .BLKW   8.          ;MESSAGE BUFFER
6288
6289
6290
6291          ;*
6292          ;LOCAL TEXT MESSAGES FOR TEST
6293          ;-
6294
6295 037562          111      116      111  T8NBA:  .ASCIZ  'INITIALIZE Command Not Accepted'
6296 037622          111      116      111  T82REJ: .ASCIZ  'INITIALIZE Not Rejected With Non-Zero Mode Field'
6297 037703          107      105      124  T83REJ: .ASCIZ  'GET STATUS Not Accepted'
6298 037733          107      105      124  T84REJ: .ASCIZ  'GET STATUS Not Rejected With Non-Zero Mode Field'
6299 040014          103      157      156  T8SSR:  .ASCIZ  'Contents of TSSR Incorrect After INITIALIZE'
6300 040070          103      157      156  T8SR2:  .ASCIZ  'Contents of TSSR Incorrect After GET STATUS'
6301 040144          105      170      160  T8NINT: .ASCIZ  'Expected Interrupt Not Received On INITIALIZE'
6302 040222          111      156      143  T8TSBA: .ASCIZ  'Incorrect TSBA Address After INITIALIZE'
6303 040272          116      157      156  T8T8ID: .ASCIZ  'Non-Tape Motion Commands'
6304          .EVEN
6305

```

```

6307
6308
6309
6310
6311
6312
6313
6314
    ;+
    ;
    ;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
    ;INITIALIZE COMMAND
    ;
    ;-
    
```

```

6315 040324
6316 040324
6317 040330 012701 037450
6318 040334 012721 100213
6319 040340 005021
6320 040342 005021
6321 040344 005021
6322 040346 005021
6323 040350 005021
6324 040352 005021
6325 040354 005021
6326 040356 005011
6327 040360 005037 037472
6328 040364 000207
6329
6330
6331
6332
6333
6334
6335
    T8REST:
        SAVREG
        MOV #T8PACKET,R1 ;SAVE THE REGISTERS
        MOV #100213,(R1)+ ;START OF THE PACKET
        CLR (R1)+ ;INITIALIZE WITH ACK, IE
        CLR (R1)+ ;ADDRESS OF CHAR DATA BLOCK
        CLR (R1)+ ;EXTENDED ADDRESS
        CLR (R1)+ ;SIZE OF DATA BLOCK IN BYTES
        CLR (R1)+ ;ADDRESS OF MESSAGE BUFFER
        CLR (R1)+ ;LENGTH OF MESSAGE BUFFER
        CLR (R1)+
        CLR (R1)+
        CLR T8BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
        RTS PC ;RETURN
    
```

```

6329
6330
6331
6332
6333
6334
6335
    ;+
    ;
    ;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
    ;GET STATUS COMMAND
    ;
    ;-
    
```

```

6336 040366
6337 040366
6338 040372 012701 037450
6339 040376 012721 100217
6340 040402 005021
6341 040404 005021
6342 040406 005021
6343 040410 005021
6344 040412 005021
6345 040414 005021
6346 040416 005021
6347 040420 005011
6348 040422 005037 037472
6349 040426 000207
6350 040430
        T8RT2:
            SAVREG
            MOV #T8PACKET,R1 ;SAVE THE REGISTERS
            MOV #100217,(R1)+ ;START OF THE PACKET
            CLR (R1)+ ;GET STATUS WITH ACK, IE
            CLR (R1)+ ;ADDRESS OF CHAR DATA BLOCK
            CLR (R1)+ ;EXTENDED ADDRESS
            CLR (R1)+ ;SIZE OF DATA BLOCK IN BYTES
            CLR (R1)+ ;ADDRESS OF MESSAGE BUFFER
            CLR (R1)+ ;LENGTH OF MESSAGE BUFFER
            CLR (R1)+
            CLR (R1)+
            CLR T8BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
            RTS PC ;RETURN
            ENDTST
    
```

```

040430
040430 104401
    L10066: TRAP C$ETST
    
```

6353
6354
6355
6356
6357
6358
6359
6360
6361
6362
6363
6364
6365
6366
6367
6368
6369
6370
6371
6372
6373
6374
6375
6376
6377
6378

.SBTTL TEST 9: DMA MEMORY ADDRESSING

```

: **
: TEST 1
: TEST DESCRIPTION
:
:   This test verifies that the controller can properly address and
:   access all available CPU memory (other than that occupied by the
:   diagnostic and diagnostic supervisor code) for both reading (DATI)
:   and writing (DATO). Verified are the LSI-11 Bus drivers for all
:   available address lines. Up to this point only 16 bits have been
:   used for DMA transfers.
:
: TEST STEPS
:
: REPEAT FROM 1 TO LOOPCNT
: BEGIN
:   Do Subtest 1 - Verify GET STATUS selected locations
:   Do Subtest 2 - Verify message packets selected locations
:   Do Subtest 3 - Verify Characteristic data selected locations
:   Do Subtest 4 - Verify NXM to selected invalid addresses
: END
:
: --

```

6379 040432
040432
6380 040432 005037 002170
6381 040436 012737 005672 002146
6382 040444 005037 003100
6387 040450 012700 042100
6388 040454 004737 017232
6389 040460 012737 000002 002164
6390 040466
6391

```

BGNTST
CLR FATFLG ;CLEAR FATAL ERROR FLAG
MOV #EPRT1,EPRTSW ;SET UP ERROR MESSAGE SWITCH
CLR KTFLG ;HOLD OFF KT11
MOV #TST9ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
MOV #2.,LOOPCNT ;PERFORM 2 ITERATIONS
T9LOOP: ;LOOP ON TEST LABEL

```



```

6393 .SBTTL TEST 9: SUBTEST 1: GET STATUS SELECTED LOCATIONS
6394 ;**
6395 ; TEST 9: SUBTEST 1:
6396 ;
6397 ; SUBTEST DESCRIPTION:
6398 ;
6399 ; This subtest verifies the controller can fetch a get status
6400 ; command from all available memory locations.
6401 ; Two word blocks are tested one at a time by first setting
6402 ; all available memory to a background pattern of 125252.
6403 ; A Get Status command is then executed to various addresses in
6404 ; each available memory 4k word block. The various addresses
6405 ; are determined by floating a 1 then a 0 through the address bits.
6406 ;
6407 ; TEST STEPS:
6408 ;
6409 ; BEGIN
6410 ; Fill Memory with background pattern of 125252
6411 ; Write to TSSR to soft initialize
6412 ; Do a WRITE CHARACTERISTICS to setup a message buffer
6413 ;
6414 ; REPEAT FOR SELECTED VALID ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
6415 ; BEGIN
6416 ; Get a valid modulo-4 test address
6417 ; Do a GET STATUS command from the test address
6418 ;
6419 ; END
6420 ;
6421 ;--
6422 040466 BGNSUB ;////////// BEGIN SUBTEST //////////
6423 040466 ; T9.1: TRAP C$BSUB
6424 040 56 104402
6425 ;Fill Memory with background pattern of 125252
6426 040470 012700 125252 MOV #125252,R0 ;BACKGROUND DATA
6427 040474 004737 020216 JSR PC,FILLMEM ;FILL MEMORY WITH BACKGROUND DATA
6428 ;Write to TSSR to soft initialize
6429 040500 004737 016470 JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
6430 040504 103405 BCS 15$ ;BR IF SOFT INIT = OK
6431 040506 NEXT.ERRNO
6432 040506 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
6433 040510 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
6434 040510 104455 TRAP C$ERDF
6435 040512 001605 .WORD 901
6436 040514 003550 .WORD SFIERR
6437 040516 011506 .WORD SFIMSG
6438 ;Do a WRITE CHARACTERISTICS to setup a message buffer
6439 15$:
6440 MOV #T9PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
6441 JSR PC,T9SWRT ;RESTORE PACKET TO STARTING VALUES
6442 CLR KTENABLE ;TURN OFF KT-11
6443 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS
6444 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
6445 FORCERROR 17$
6446 BCS 20$ ;BR IF SSR SET IN CHKTSSR

```

```

6444 040562 010001          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
6445 040564          NEXT.ERRNO
6446 040564          17$:  ERRDF  ERRNO,T9WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
        040564 104455          TRAP      C$ERDF
        040566 001606          .WORD    902
        040570 042202          .WORD    T9WRTSSR
        040572 011520          .WORD    PKTSSR

6447
6448          ;Verify a Get Status can be fetched from each address
6449          ;Get a valid modulo-4 test address
6450          ;Do a GET STATUS command from the test address
6451 040574 005037 002170      20$:  CLR      FATFLG          ;CLEAR FATAL ERROR FLAG
6452 040600 005037 041740      CLR      T9KT            ;TEST ABOVE 28K SWITCH
6453 040604 012702 041744      MOV      @T9BLK,R2       ;POINT TO TEST PATTERN TABLE
6454 040610          T91LOOP:
6455 040610 005037 003102      CLR      KTENABLE        ;TURN OFF ABOVE 28K TEST FLAG
6456 040614 012201          MOV      (R2)+,R1        ;GET TEST PATTERN ADDRESS
6457 040616 005000          CLR      R0              ;ASSUME NO TEST ABOVE 28K
6458 040620 005737 041740      TST      T9KT            ;TEST ABOVE 28K THIS TIME?
6459 040624 001407          BEQ      25$             ;BR IF NO
6460 040626 016200 177776      MOV      -2(R2),R0       ;GET TEST PATTERN AGAIN
6461 040632 042700 177774      BIC      @+C<A1716>,R0   ;SAVE 18 BIT ADDRESS ONLY
6462 040636 012737 000001 003102      MOV      @1,KTENABLE     ;TURN ON ABOVE 28K TEST FLAG
6463 040644 004737 042746      25$:  JSR      PC,T9CONVERT ;CONVERT TEST PATTERN TO TEST ADDRESS
6464 040650 103034          BCC      65$             ;BR IF INVALID PACKET ADDRESS
6465 040652 013704 041734      MOV      T9LOADD,R4      ;COPY CURRENT PACKET LOW ADDRESS
6466 040656 013703 041732      MOV      T9HIADD,R3      ;COPY CURRENT PACKET HIGH ADDRESS
6467 040662 004737 043504      JSR      PC,T9SETGET     ;SETUP CURRENT PACKET TO GET STATUS
6468 040666 042703 177774      BIC      @+C<A1716>,R3   ;SAVE ADDRESS BITS 17+16
6469 040672 050304          BIS      R3,R4           ;SETUP 18 BIT PACKET ADDRESS
6470 040674 004737 020070      JSR      PC,KTOFF        ;TURN OFF KT-11
6471 040700 010465 177776      MOV      R4,TSDB(R5)     ;SET THE PACKET ADDRESS TO EXECUTE
6472 040704 004737 017060      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
6473 040710          FORCERROR 32$
6474 040724 103405          BCS      40$             ;BR IF SSR SET IN CHKTSSR
6475 040726 010001          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
6476 040730          NEXT.ERRNO
6477 040730          32$:  ERRDF  ERRNO,T9GETSSR,PKTGETS ;DEVICE FATAL SSR FAILED TO SET
        040730 104455          TRAP      C$ERDF
        040732 001607          .WORD    903
        040734 042126          .WORD    T9GETSSR
        040736 011550          .WORD    PKTGETS
6478 040740          40$:  CKLOOP              ;LOOP ON ERROR, IF FLAG SET
        040740 104406          TRAP      C$CLP1
6479 040742          65$:
6480 040742          FORCEEXIT 80$
6481 040752 020227 042076      CMP      R2,@T9TBE       ;DONE ALL TSTBLK TEST PATTERNS?
6482 040756 103002          BHIS    70$             ;BR IF YES
6483 040760 000137 040610      JMP      T91LOOP         ;DO ANOTHER MODULO- 4 ADDRESS
6484 040764 005737 041740      70$:  TST      T9KT            ;DONE ABOVE 28K TESTING TOO?
6485 040770 003012          BGT      80$             ;BR IF YES
6486 040772 005737 003100      TST      KTFLG          ;ANY MEMORY ABOVE 28K ON SYSTEM?
6487 040776 001407          BEQ      80$             ;BR IF NO
6488 041000 012737 000001 041740      MOV      @1,T9KT        ;SET SWITCH
6489 041006 012702 041744      MOV      @T9BLK,R2       ;RESET TEST PATTERN TABLE
6490 041012 000137 040610      JMP      T91LOOP         ;DO ABOVE 28K TESTING
6491 041016 004737 020070      80$:  JSR      PC,KTOFF        ;TURN OFF KT11
    
```

J14

6492	041022			ENDSUB		
	041022					
	041022	104403				
6493	041024	005737	002170	TST	FATFLG	
6494	041030	001402		BEQ	100\$	
6495	041032	004737	017776	JSR	PC,CKDROP	
6496	041036					
6497						

100\$:

```
;/;;;;;;;;;;;;;;;; END SUBTEST ;;;;;;;;;;;;;;;;;  
L10072: TRAP C$ESUB  
;ANY FATAL ERRORS ?  
;BRANCH IF NOT  
;TRY TO DROP THE UNIT
```



```

6499          .SBTTL TEST 9: SUBTEST 2: MESSAGE PACKETS TO SELECTED LOCATIONS
6500          ;**
6501          ; TEST 9: SUBTEST 2:
6502          ;
6503          ; SUBTEST DESCRIPTION:
6504          ;
6505          ; This subtest verifies the controller can deposit message packets
6506          ; to all available memory locations.
6507          ; First all available memory is set to a background pattern
6508          ; of 125252.
6509          ; Write Characteristics commands are then executed with message
6510          ; buffer addresses set to various addresses in each available
6511          ; memory location.
6512          ; The various addresses are determined by floating a 1 then a 0
6513          ; through the address bits.
6514          ;
6515          ; TEST STEPS:
6516          ;
6517          ; BEGIN
6518          ; Fill Memory with background pattern of 125252
6519          ; Write to TSSR to soft initialize
6520          ; Do a WRITE CHARACTERISTICS to setup a message buffer to compare
6521          ;
6522          ; REPEAT FOR SELECTED ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
6523          ; BEGIN
6524          ; Get a valid modulo-4 test address
6525          ; Set the packet message buffer to the TEST ADDRESS
6526          ; Do a WRITE CHARACTERISTICS
6527          ; Restore the test message buffer to background pattern
6528          ;
6529          ; END
6530          ; END
6531          ;--
6532          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
6533          041036          T9.2:          TRAP          C$BSUB
6534          041036          104402
6535          ;Fill Memory with background pattern of 125252
6536          041040          012700          125252          MOV          #125252,R0          ;BACKGROUND DATA
6537          041044          004737          020216          JSR          PC,FILLMEM          ;FILL MEMORY WITH BACKGROUND DATA
6538          ;Write to TSSR to soft initialize
6539          041050          004737          016470          JSR          PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
6540          041054          103405          BCS          15$          ;BR IF SOFT INIT = OK
6541          041056          NEXT.ERRNO
6542          041056          010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
6543          041060          ERRDF          ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL ERROR DURING INIT
6544          041060          104455          TRAP          C$ERDF
6545          041062          001610          .WORD          904
6546          041064          003550          .WORD          SFIERR
6547          041066          011506          .WORD          SFIMSG
6548          ;Do a WRITE CHARACTERISTICS to setup a message buffer to compare
6549          15$:          MOV          #T9PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
6550          JSR          PC,T9SWRT          ;SET PACKET TO WRITE CHARACTERISTICS
6551          JSR          PC,KTOFF          ;TURN OFF KT-11

```

```

6550 041104 010465 177776      MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS
6551 041110 004737 017060      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
6552 041114                      FORCERROR      17$
6553 041130 103405                      BCS      20$            ;BR IF SSR SET IN CHKTSSR
6554 041132 010001                      MOV      R0,R1          ;SAVE CONTENTS OF TSSR
6555 041134                      NEXT.ERRNO
6556 041134 17$:      ERRDF  ERRNO,T9WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP      C$ERDF
                                .WORD    905
                                .WORD    T9WRTSSR
                                .WORD    PKTSSR
041134 104455
041136 001611
041140 042202
041142 011520
6557
6558      ;Get a valid modulo-4 test address
6559      ;Set the packet message buffer to the test address
6560      ;Do a WRITE CHARACTERISTICS
20$:      CLR      FATFLG      ;CLEAR FATAL ERROR FLAG
                                MOV      #T9BLK,R3      ;POINT TO TEST PATTERN TABLE
T92LOOP:
6561 041144 005037 002170      MOV      (R3)+,R1      ;GET TEST PATTERN ADDRESS
6562 041150 012703 041744      MOV      R1,R0          ;GET ADDRESS ALL "18 BITS"
6563 041154                      BIC      #177774,R0     ;LEAVE ONLY A17 AND A16
6564 041154 012301                      BIC      #1,R1          ;ALWAYS ON A WORD BOUNDRY
6565 041156 010100                      JSR      PC,T9CT2      ;CONVERT TEST PATTERN TO TEST ADDRESS
6566 041160 042700 177774                      BCS      25$            ;BR IF VALID MESSAGE BUFFER ADDRESS
6567 041164 042701 000001                      JMP      150$          ;GET ANOTHER TEST PATTERN TO TRY
6568 041170 004737 043140      MOV      #T9PACKET,R4  ;SET THE COMMAND PACKET ADDRESS
6569 041174 103402                      JSR      PC,T9SWRT     ;SETUP T9PACKET TO WRITE CHAR.
6570 041176 000137 041274      MOV      T9LOADD,T9DATA ;SETUP LOW ORDER MESSAGE BUFFER ADD.
6571 041202 012704 041670      MOV      T9HIADD,T9DATA+2 ;SETUP HIGH ORDER MESSAGE BUFFER ADD.
6572 041206 004737 043436      JSR      PC,KTOFF      ;TURN OFF KT-11
6573 041212 013737 041734 041700      MOV      R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
6574 041220 013737 041732 041702      JSR      PC,CHKTSSR    ;WAIT FOR SSR TO SET
6575 041226 004737 020070                      FORCERROR      32$
6576 041232 010465 177776                      BCS      50$            ;BR IF SSR SET IN CHKTSSR
6577 041236 004737 017060      MOV      R0,R1          ;SAVE CONTENTS OF TSSR
6578 041242                      NEXT.ERRNO
6579 041256 103405                      ERRDF  ERRNO,T9WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
6580 041260 010001
6581 041262
6582 041262 32$:      TRAP      C$ERDF
                                .WORD    906
                                .WORD    T9WRTSSR
                                .WORD    PKTSSR
                                TRAP      C$CLP1
041262 104455
041264 001612
041266 042202
041270 011520
6583 041272 50$:      CKLOOP      ;LOOP ON ERROR, IF FLAG SET
041272 104406      TRAP      C$CLP1
6584 041274 150$:
6585 041274      FORCEXIT      160$
6586 041304 020327 042076      CMP      R3,#T9TBE     ;DONE ALL TST9BLK TEST PATTERNS?
6587 041310 103002                      BHS      160$          ;BR IF YES
6588 041312 000137 041154      JMP      T92LOOP      ;DO ANOTHER MODULO- 4 ADDRESS
6589 041316 004737 020070      JSR      PC,KTOFF      ;TURN OFF KT11
6590 041322 160$:      ENDSUB
                                ;////////////////// END SUBTEST ////////////////////
                                L10073:
                                TRAP      C$ESUB
041322 104403
6591 041324 005737 002170      TST      FATFLG      ;ANY FATAL ERRORS ?
6592 041330 001402                      BEQ      180$          ;BRANCH IF NOT
6593 041332 004737 017776      JSR      PC,CKDROP     ;TRY TO DROP THE UNIT
6594 041336 180$:

```



```

6596          .SBTTL TEST 9: SUBTEST 3: CHARACTERISTIC DATA SELECTED LOCATIONS
6597          ;**
6598          ; TEST 9: SUBTEST 3:
6599          ;
6600          ; SUBTEST DESCRIPTION:
6601          ;
6602          ;     This subtest verifies the controller can fetch a
6603          ;     Write Characteristics data block from all available
6604          ;     memory locations.
6605          ;     First all available memory is set to a background
6606          ;     pattern of 125252.
6607          ;     Then Write Characteristics commands are executed with
6608          ;     characteristic data blocks at various memory addresses.
6609          ;     The various memory addresses are determined by floating
6610          ;     a 1 then a 0 through the address bits.
6611          ;
6612          ; TEST STEPS:
6613          ;
6614          ;     BEGIN
6615          ;     Fill Memory with background pattern of 125252
6616          ;     Write to TSSR to soft initialize
6617          ;
6618          ;     REPEAT FOR SELECTED VALID ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
6619          ;     BEGIN
6620          ;     Get a valid test address
6621          ;     Set the test packet characteristics data pointer to the
6622          ;     test address.
6623          ;     Store expected characteristic data in test address block
6624          ;     Do a WRITE CHARACTERISTIC command
6625          ;     END
6626          ;     END
6627          ;--
6628
6629          BGNSUB          ;////////// BEGIN SUBTEST ////////////
6630          041336          T9.3:          TRAP      C$BSUB
6631          041336          104402
6632          ;Fill Memory with background pattern of 125252
6633          041340 012700 125252          MOV      #125252,R0          ;BACKGROUND DATA
6634          041344 004737 020216          JSR      PC,FILLMEM          ;FILL MEMORY WITH BACKGROUND DATA
6635
6636          ;Write to TSSR to soft initialize
6637          041350 004737 016470          JSR      PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
6638          041354 103405          BCS     20$          ;BR IF SOFT INIT = OK
6639          041356 010001          NEXT.ERRNO
6640          041360          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
6641          041360 104455          ERRDF   ERRNO,SFERR,SFMSG          ;DEVICE FATAL ERROR DURING INIT
6642          041362 001613          TRAP    C$ERDF
6643          041364 003550          .WORD  907
6644          041366 011506          .WORD  SFIERR
6645          .WORD  SFMSG
6646
6647          ;Get a valid test address
6648          20$:          CLR      FATFLG          ;CLEAR FATAL ERROR FLAG
6649          041370 005037 002170          CLR      T9KT          ;TEST ABOVE 28K SWITCH
6650          041374 005037 041740          MOV      #T9BLK,R3          ;POINT TO TEST PATTERN TABLE
6651          041400 012703 041744
6652          041404
6653          T93LOOP:
    
```


6647	041404	005037	003102	CLR	KTENABLE		;TURN OFF ABOVE 28K TEST FLAG
6648	041410	012301		MOV	(R3),R1		;GET TEST PATTERN ADDRESS
6649	041412	010100		MOV	R1,R0		;GET ADDRESS ALL "18 BITS"
6650	041414	042700	177774	BIC	#177774,R0		;LEAVE ONLY A17 AND A16
6651	041420	042701	000003	BIC	#3,R1		;GET RID OF A17 AND A16
6652	041424	005737	041740	TST	T9KT		;TEST ABOVE 28K THIS TIME?
6653	041430	001407		BEQ	25\$;BR IF NO
6654	041432	016300	177776	MOV	-2(R3),R0		;GET TEST PATTERN AGAIN
6655	041436	042700	177774	BIC	#1C<A1716>,R0		;SAVE 18 BIT ADDRESS ONLY
6656	041442	012737	000001	MOV	#1,KTENABLE	003102	;TURN ON ABOVE 28K TEST FLAG
6657	041450	004737	042746	JSR	PC,T9CONVERT	25\$:	;CONVERT TEST PATTERN TO TEST ADDRESS
6658	041454	103402		BCS	30\$;BR IF VALID TEST ADDRESS
6659	041456	000137	041560	JMP	60\$;GET NEXT TEST PATTERN
6660				;Set the test packet characteristics data pointer to the test address			
6661	041462	012704	041670	30\$:	MOV	#T9PACKET,R4	;GET THE ADDRESS OF COMMAND PACKET
6662	041466	004737	043436	JSR	PC,T9SWRT		;RESTORE PACKET TO STARTING VALUES
6663	041472	013764	041734	000002	MOV	T9LOADD,PKLOW(R4)	;STORE CHAR. DATA PTR LOW ADDRESS
6664	041500	013764	041732	000004	MOV	T9HIADD,PKHI(R4)	;STORE CHAR. DATA PTR HIGH ADDRESS
6665	041506	004737	043546	JSR	PC,T9CHAR		;STORE EXPECTED DATA IN DATA BLOCK
6666				;Do a WRITE CHARACTERISTIC command			
6667	041512	004737	020070	JSR	PC,KTOFF		;TURN OFF KT-11
6668	041516	010465	177776	MOV	R4,TSDB(R5)		;SET THE PACKET ADDRESS TO EXECUTE
6669	041522	004737	017060	JSR	PC,CHKTSSR		;WAIT FOR SSR TO SET
6670	041526			FORCERROR	32\$		
6671	041542	103405		BCS	40\$;BR IF SSR SET IN CHKTSSR
6672	041544	010001		MOV	R0,R1		;SAVE CONTENTS OF TSSR
6673	041546			NEXT.ERRNO			
6674	041546			32\$:	ERRDF	ERRNO,T9WRTSSR,PKTSSR	;DEVICE FATAL SSR FAILED TO SET
	041546	104455					TRAP C\$ERDF
	041550	001614					.WORD 908
	041552	042202					.WORD T9WRTSSR
	041554	011520					.WORD PKTSSR
6675	041556			40\$:	CKLOOP		;LOOP ON ERROR, IF FLAG SET
	041556	104406					TRAP C\$CLP1
6676	041560			60\$:			
6677	041560	020327	042076	CMP	R3,#T9TBE		;DONE ALL TSTBLK TEST PATTERNS?
6678	041564	103002		BHIS	65\$;BR IF YES
6679	041566	000137	041404	JMP	T93LOOP		;DO ANOTHER MODULO- 4 ADDRESS
6680	041572	005737	041740	65\$:	TST	T9KT	;DONE ABOVE 28K TESTING TOO?
6681	041576	003012		BGT	70\$;BR IF YES
6682	041600	005737	003100	TST	KTFLG		;ANY MEMORY ABOVE 28K ON SYSTEM?
6683	041604	001407		BEQ	70\$;BR IF NO
6684	041606	012737	000001	041740	MOV	#1,T9KT	;SET SWITCH
6685	041614	012703	041744	MOV	#T9BLK,R3		;RESET TEST PATTERN TABLE
6686	041620	000137	041404	JMP	T93LOOP		;DO ABOVE 28K TESTING
6687	041624	004737	020070	70\$:	JSR	PC,KTOFF	;TURN OFF KT11
6688	041630			ENDSUB			;////////// END SUBTEST //////////
	041630						L10074:
	041630	104403					TRAP C\$ESUB
6689	041632	005737	002170	TST	FATFLG		;ANY FATAL ERRORS ?
6690	041636	001402		BEQ	75\$;BRANCH IF NOT
6691	041640	004737	017776	JSR	PC,CKDROP		;TRY TO DROP THE UNIT
6692	041644			75\$:			
6693	041644	004737	017200	100\$:	JSR	PC,TSTLOOP	;SHOULD WE DO ITERATIONS?
6694	041650	103002		BCC	105\$;BR IF NO
6695	041652	000137	040466	JMP	T9LOOP		;LOOP UNTIL ITERATION COUNT DONE
6696	041656			105\$:			

6697 041656 004737 020070
6698 041662
041662 104432
041664 001724

JSR PC,KTOFF
EXIT TST

;TURN OFF MEMORY MANAGEMENT
;ALL DONE THIS TEST

TRAP C\$EXIT
.WORD L10071-

6699
6700
6701
6702
6703
6704

;*
;LOCAL STORAGE FOR THIS TEST
;-

6706 041666
6708 041670
6709 041670 100004
6710 041672 041700
6711 041674 000000
6712 041676 000010

T9PACKET: .BLKB 10-<.-TUV2A&7>
.WORD 100004
.WORD T9DATA
.WORD 0
.WORD 8.

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

;STARTING VALUE OF BLOCK SIZE

6714 041700
6715 041700 041712
6716 041702 000000
6717 041704 000016
6718 041706 000000 000000

T9DATA: .WORD T9BFR
.WORD 0
.WORD 14.
.WORD 0,0

;CHARACTERISTICS DATA BLOCK
;LOW ADDRESS OF MESSAGE BUFFER
;HIGH ORDER OF MESSAGE BUFFER
;LENGTH OF MESSAGE BUFFER

6719
6720 041712
6721
6722 041732 000000
6723 041734 000000
6724 041736 000000
6725 041740 000000
6726 041742 000000

T9BFR: .BLKW 8.
T9HIADD: .WORD 0
T9LOADD: .WORD 0
T9PAR6: .WORD 0
T9KT: .WORD 0
T9TST: .WORD 0

;MESSAGE BUFFER
;HIGH ADDRESS
;LOW ADDRESS
;ADDRESS IN PAR FORMAT
;TEST ABOVE 28K SWITCH
;ADDRESS TEST BIT

6727
6728
6729
6730

;*
;TABLE OF ADDRESSES
;-

6731
6732 041744 000001
6733 041746 000002
6734 041750 000003
6735 041752 000005
6736 041754 000006
6737 041756 000007
6738 041760 000011
6739 041762 000012
6740 041764 000013
6741 041766 000021
6742 041770 000022
6743 041772 000023
6744 041774 000041
6745 041776 000042
6746 042000 000043
6747 042002 000101
6748 042004 000102
6749 042006 000103
6750 042010 000201
6751 042012 000202
6752 042014 000203
6753 042016 000401

T9BLK: .WORD 000001
.WORD 000002
.WORD 000003
.WORD 000005
.WORD 000006
.WORD 000007
.WORD 000011
.WORD 000012
.WORD 000013
.WORD 000021
.WORD 000022
.WORD 000023
.WORD 000041
.WORD 000042
.WORD 000043
.WORD 000101
.WORD 000102
.WORD 000103
.WORD 000201
.WORD 000202
.WORD 000203
.WORD 000401

6754	042020	000402				.WORD	000402	
6755	042022	000403				.WORD	000403	
6756	042024	001001				.WORD	001001	
6757	042026	001002				.WORD	001002	
6758	042030	001003				.WORD	001003	
6759	042032	002001				.WORD	002001	
6760	042034	002002				.WORD	002002	
6761	042036	002003				.WORD	002003	
6762	042040	004001				.WORD	004001	
6763	042042	004002				.WORD	004002	
6764	042044	004003				.WORD	004003	
6765	042046	010001				.WORD	010001	
6766	042050	010002				.WORD	010002	
6767	042052	010003				.WORD	010003	
6768	042054	020001				.WORD	020001	
6769	042056	020002				.WORD	020002	
6770	042060	020003				.WORD	020003	
6771	042062	040001				.WORD	040001	
6772	042064	040002				.WORD	040002	
6773	042066	040003				.WORD	040003	
6774	042070	100001				.WORD	100001	
6775	042072	100002				.WORD	100002	
6776	042074	100003				.WORD	100003	
6777	042076	177777				.WORD	177777	
6778						;		
6779						;	LOCAL TEXT MESSAGES FOR TEST	
6780						;		
6781								
6782	042100	104	115	101	TST9ID:	.ASCIZ	'DMA Memory Addressing'	
6783	042126	103	157	156	T9GETSSR:	.ASCIZ	'Contents of TSSR Incorrect After GET STATUS'	
6784	042202	103	157	156	T9WRTSSR:	.ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'	
6785	042271	115	145	163	T9MSGBUF:	.ASCIZ	'Message Buffer Contents Incorrect After WRITE CHARACTERISTICS'	
6786	042367	102	141	143	T9BKGNB:	.ASCIZ	'Background Pattern Disturbed By WRITE CHARACTERISTICS'	
6787	042455	105	170	160	T9NINT:	.ASCIZ	'Expected Interrupt Not Received On WRIT. CHARACTERISTICS'	
6788	042546	127	162	151	T9DPR:	.ASCIZ	'Write Characteristic data in rnr does not match expected'	
6789	042637	124	123	123	T9NXM:	.ASCIZ	'TSSR NXM bit failed to set when non-existent memory address specifi	
ed								
6790						.EVEN		
6791								


```

6793
6794
6795
6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808
6809
6810
6811
6812
6813 042746
6814 042746
6815 042752 005037 041734
6816 042756 005037 041732
6817 042762 005037 041736
6818 042766 042701 170000
6819 042772 010005
6820 042774 004737 020070
6821 043000 013702 003072
6822 043004 062702 000020
6823 043010 060102
6824 043012 042702 000003
6825 043016 013703 003076
6826 043022 162703 000020
6827 043026 010237 041734
6828 043032 010237 041736
6829 043036 020203
6830 043040 101007
6831 043042 020237 003072
6832 043046 103007
6833 043050 005737 003102
6834 043054 001004
6835 043056 000424
6836 043060 162702 000020
6837 043064 000754
6838 043066
6839 043066 005737 003102
6840 043072 001420
6841 043074 005737 003100
6842 043100 001413
6843 043102 004737 020052
6844 043106 010500
6845 043110 010037 041732
6846 043114 010201
6847 043116 004737 020112
6848 043122 010037 041736
6849 043126 103403

```

```

; *
; ROUTINE TO CONVERT A TEST PATTERN TO A VALID ADDRESS IN DIAGNOSTIC FREE SPACE
; DIAGNOSTIC FREE SPACE IS BETWEEN THE END OF THE DIAGNOSTIC AND THE
; BEGINNING OF THE SUPERVISOR. THIS IS ALWAYS BELOW 24K.
; IF MEMORY ABOVE 28K SPECIFIED (VIA R1) THEN PAR 6 IS SET
; TO THE RELOCATION BASE.
;
; INPUTS:
;
; R0      HIGH ORDER ADDRESS BITS
; R1      LOW ORDER ADDRESS BITS
;
; OUPUTS:
; T9PAR6 = ADDRESS BIASED TO PAR6 IF >28K UNDER TEST
; T9HIADD = HIGH ORDER ADDRESS IN NON PAR6 FORMAT
; T9LOADD = LOW ORDER ADDRESS IN NON PAR6 FORMAT
; C BIT = 1 IF GOOD ADDRESS RETURNED
; C BIT = 0 IF TEST PATTERN DID NOT YIELD A VALID ADDRESS
;
; T9CONVERT:
; SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
; CLR T9LOADD    ;CLEAR LOW ADDRESS
; CLR T9HIADD    ;CLEAR HIGH ADDRESS
; CLR T9PAR6     ;CLEAR PAR6 BIASED ADDRESS
; BIC @C<7777>,R1 ;FORCE TO LOWER 12 BITS OF ADDRESS
; MOV R0,R5      ;SAVE HIGH ORDER ADDRESS BITS
; JSR PC,KTOFF  ;SHUTOFF MEMORY MANAGEMENT
; MOV FREE,R2   ;GET FIRST FREE ADDRESS
; ADD @16.,R2   ;IN CASE TEST PATTERN=0
; ADD R1,R2     ;ADD IN TEST PATTERN
; BIC @3,R2     ;MAKE IT MODULO-4
25$: ; MOV FREEHI,R3 ;GET LAST FREE ADDRESS
; SUB @16.,R3  ;SAVE AT LEAST 8 WORDS (IN CASE MESSAGE BUFFER)
; MOV R2,T9LOADD ;SAVE POSSIBLE LOW ADDRESS
; MOV R2,T9PAR6 ;SAVE IT IN PAR6 BIASED TOO
; CMP R2,R3    ;IS THIS ADDRESS ABOVE FREE SPACE?
; BHI 35$     ;BR IF YES
; CMP R2,FREE ;IS IT IN FREE SPACE?
; BHIS 50$    ;BR IF YES- ITS GOOD
; TST KTENABLE ;TESTING ABOVE 28K?
; BNE 50$     ;BR IF YES
; BR 90$     ;BR IF NOT IN FREE SPACE
35$: ; SUB @16.,R2 ;FORCE FIT THE TEST PATTERN
; BR 25$    ;TRY THIS TEST PATTERN ADDRESS
50$: ; TST KTENABLE ;TESTING ABOVE 28K?
; BEQ 100$   ;BR IF NO
; TST KTFLG ;ANY MEMORY ABOVE 28K?
; BEQ 90$   ;BR IF NO
; JSR PC,KTON ;TURN ON MEMORY MANAGEMENT
; MOV R5,R0 ;GET HIGH ORDER ADDRESS
; MOV R0,T9HIADD ;SAVE POSSIBLE HIGH ADDRESS
; MOV R2,R1 ;GET COMPUTED LOW ORDER ADDRESS
; JSR PC,SETMAP ;RETURN PAR6 BIASED ADDRESS IN R0
; MOV R0,T9PAR6 ;COPY PAR6 BIASED ADDRESS
; BCS 105$   ;BR IF VALID ADDRESS

```

E15

6850 043130 000241
6851 043132 000401
6852 043134 000261
6853 043136 000207
6854
6855

90#: CLC
 BR 105#
100#: SEC
105#: RTS PC

;CLR C BIT FOR FAILURE
;
;SET SUCCESS
;RETURN

6857
 6858
 6859
 6860
 6861
 6862
 6863
 6864
 6865
 6866
 6867
 6868
 6869
 6870
 6871
 6872
 6873
 6874
 6875
 6876
 6877
 6878 043140
 6879 043140
 6880 043144 005037 041734
 6881 043150 005037 041732
 6882 043154 005037 041736
 6883 043160 042701 170000
 6884 043164 010005
 6885 043166 004737 020070
 6886 043172 013702 003072
 6887 043176 062702 000020
 6888 043202 060102
 6889 043204 013703 003076
 6890 043210 162703 000020
 6891 043214 010237 041734
 6892 043220 010237 041736
 6893 043224 020203
 6894 043226 101007
 6895 043230 020237 003072
 6896 043234 103007
 6897 043236 005737 003102
 6898 043242 001004
 6899 043244 000424
 6900 043246 162702 000020
 6901 043252 000754
 6902 043254
 6903 043254 005737 003102
 6904 043260 001420
 6905 043262 005737 003100
 6906 043266 001413
 6907 043270 004737 020052
 6908 043274 010500
 6909 043276 010037 041732
 6910 043302 010201
 6911 043304 004737 020112
 6912 043310 010037 041736
 6913 043314 103403

```

; *
; ONLY FOR MESSAGE BUFFER ADDRESSES
; ROUTINE TO CONVERT A TEST PATTERN TO A VALID ADDRESS IN DIAGNOSTIC FREE SPACE
; DIAGNOSTIC FREE SPACE IS BETWEEN THE END OF THE DIAGNOSTIC AND THE
; BEGINNING OF THE SUPERVISOR. THIS IS ALWAYS BELOW 24K.
; IF MEMORY ABOVE 28K SPECIFIED (VIA R1) THEN PAR 6 IS SET
; TO THE RELOCATION BASE.
;
; INPUTS:
;
; R0      HIGH ORDER ADDRESS BITS
; R1      LOW ORDER ADDRESS BITS
;
; OUPUTS:
; T9PAR6  = ADDRESS BIASED TO PAR6 IF >28K UNDER TEST
; T9HIADD = HIGH ORDER ADDRESS IN NON PAR6 FORMAT
; T9LOADD = LOW ORDER ADDRESS IN NON PAR6 FORMAT
; C BIT   = 1 IF GOOD ADDRESS RETURNED
;         = 0 IF TEST PATTERN DID NOT YIELD A VALID ADDRESS
;
; -
T9CT2:
    SAVREG                ;SAVE R1-R5 UNTIL NEXT RETURN
    CLR T9LOADD           ;CLEAR LOW ADDRESS
    CLR T9HIADD          ;CLEAR HIGH ADDRESS
    CLR T9PAR6           ;CLEAR PAR6 BIASED ADDRESS
    BIC #C<7777>,R1     ;FORCE TO LOWER 12 BITS OF ADDRESS
    MOV R0,R5            ;SAVE HIGH ORDER ADDRESS BITS
    JSR PC,KTOFF        ;SHUTOFF MEMORY MANAGEMENT
    MOV FREE,R2          ;GET FIRST FREE ADDRESS
    ADD #16.,R2         ;IN CASE TEST PATTERN=0
    ADD R1,R2            ;ADD IN TEST PATTERN
25$:  MOV FREEHI,R3      ;GET LAST FREE ADDRESS
    SUB #16.,R3         ;SAVE AT LEAST 8 WORDS (IN CASE MESSAGE BUFFER)
    MOV R2,T9LOADD      ;SAVE POSSIBLE LOW ADDRESS
    MOV R2,T9PAR6       ;SAVE IT IN PAR6 BIASED TOO
    CMP R2,R3           ;IS THIS ADDRESS ABOVE FREE SPACE?
    BHI 35$             ;BR IF YES
    CMP R2,FREE         ;IS IT IN FREE SPACE?
    BHIS 50$           ;BR IF YES- ITS GOOD
    TST KTENABLE        ;TESTING ABOVE 28K?
    BNE 50$            ;BR IF YES
    BR 90$              ;BR IF NOT IN FREE SPACE
35$:  SUB #16.,R2       ;FORCE FIT THE TEST PATTERN
    BR 25$              ;TRY THIS TEST PATTERN ADDRESS
50$:  TST KTENABLE      ;TESTING ABOVE 28K?
    BEQ 100$           ;BR IF NO
    TST KTFLG          ;ANY MEMORY ABOVE 28K?
    BEQ 90$            ;BR IF NO
    JSR PC,KTON        ;TURN ON MEMORY MANAGEMENT
    MOV R5,R0          ;GET HIGH ORDER ADDRESS
    MOV R0,T9HIADD     ;SAVE POSSIBLE HIGH ADDRESS
    MOV R2,R1          ;GET COMPUTED LOW ORDER ADDRESS
    JSR PC,SETMAP      ;RETURN PAR6 BIASED ADDRESS IN R0
    MOV R0,T9PAR6     ;COPY PAR6 BIASED ADDRESS
    BCS 105$          ;BR IF VALID ADDRESS
    
```


6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931
6932
6933
6934
6935
6936
6937
6938
6939
6940
6941
6942
6943
6944
6945
6946 043326
6947 043326
6948 043332 012701 002206
6949 043336 012702 000020
6950 043342 005003
6951 043344 004737 017060
6952 043350 112765 000000 177776
6953 043356 004737 017060 10\$:
6954 043362 010265 177776
6955 043366 004737 017060
6956 043372 116511 177776
6957 043376 122124
6958 043400 001401
6959 043402 005203
6960 043404 005202
6961 043406 020227 000022
6962 043412 002761
6963 043414 005703
6964 043416 001402
6965 043420 000241
6966 043422 000401
6967 043424 000261
6968 043426 012737 000002 002246
6969 043434 000207
6970
6971
6972
6973
6974
6975 043436
6976 043436

```

;+
; ROUTINE TO READ THE FIRST 2 BYTES FROM RAM
; MEMORY AND COMPARE THIS DATA TO A COMMAND PACKET.
; INPUT:
; R4 ADDRESS OF THE COMMAND PACKET
; R5 FIRST DEVICE UNIBUS ADDRESS
; OUTPUT:
; CARRY SET - RAM MATCHES PACKET
; CLR - RAM DOES NOT MATCH PACKET
; IMPLICIT OUTPUT:
; THE TABLE RAMDATA IS FILLED WITH THE
; DATA HELD IN RAM.
; RAMSIZ SET TO 2 FOR PRAMPKT ROUTINE
; SIDE EFFECTS:
; THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
;-
T9CKRAM:
  SAVREG ;SAVE THE GENERAL REGISTERS
  MOV #RAMDATA,R1 ;ADDRESS TO SAVE THE RAM DATA
  MOV #RMPKTBEG,R2 ;BYTE ADDRESS OF FIRST RAM DATA
  CLR R3 ;CLEAR THE ERROR FLAG
  JSR PC,CHKTSSR ;WAIT FOR SSR
  MOVB #0,TSDB(R5) ;SET MAINTENANCE MODE
  JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
  MOV R2,TSDB(R5) ;SELECT NEXT RAM ADDRESS
  JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
  MOVB TSBA(R5),(R1) ;READ THE RAM DATA
  CMPB (R1)+,(R4)+ ;COMPARE TO EXPECTED
  BEQ 20$ ;BRANCH IF OK
  INC R3 ;SET ERROR FLAG
  INC R2 ;ADDRESS OF NEXT RAM LOCATION
  CMP R2,#RMPKTBEG+2 ;DONE 2 BYTES?
  BLT 10$ ;BR IF NO
  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 #2,RAMSIZ ;SETUP RAMSIZ
  RTS PC ;RETURN
;+
; ROUTINE TO SETUP PACKET TO WRITE CHARACTERISTICS
;-
T9SWRT:
  SAVREG ;SAVE THE REGISTERS

```



```

6977 043442 012701 041670      MOV      #T9PACKET,R1      ;START OF THE PACKET
6978 043446 012721 100004      MOV      #100004,(R1)+    ;WRITE CHARACTERISTICS WITH ACK
6979 043452 012721 041700      MOV      #T9DATA,(R1)+   ;ADDRESS OF CHAR DATA BLOCK
6980 043456 005021              CLR      (R1)+           ;EXTENDED ADDRESS
6981 043460 012721 000010      MOV      #8.,(R1)+       ;SIZE OF DATA BLOCK IN BYTES
6982 043464 012721 041712      MOV      #T9BFR,(R1)+   ;ADDRESS OF MESSAGE BUFFER
6983 043470 005021              CLR      (R1)+           ;
6984 043472 012721 000016      MOV      #14.,(R1)+     ;LENGTH OF MESSAGE BUFFER
6985 043476 005021              CLR      (R1)+           ;
6986 043500 005011              CLR      (R1)            ;
6987 043502 000207      RTS      PC              ;RETURN
6988
6989
6990      ;ROUTINE TO SETUP A GET STATUS COMMAND PACKET AT CURRENT PACKET ADDRESS
6991      ;
6992      ;      R3      HIGH ORDER PACKET ADDRESS
6993      ;      R4      LOW ORDER PACKET ADDRESS
6994      ;      NOTE: R3 IS IGNORED IF KTENABLE FLAG CLEAR
6995      ;
6996      ;-
6997
6998 043504      T9SETGET:
6999 043504      SAVREG              ;SAVE THE REGISTERS
7000 043510 010401      MOV      R4,R1          ;GET LOW ORDER ADDRESS
7001 043512 005737 003102      TST      KTENABLE      ;TESTING ABOVE 28K?
7002 043516 001404      BEQ      10$           ;BR IF NO
7003 043520 010300      MOV      R3,R0         ;GET HIGH ORDER ADDRESS
7004 043522 004737 020112      JSR      PC,SETMAP     ;RETURN ADDRESS BIASED TO PAR6 IN R0
7005 043526 010001      MOV      R0,R1         ;GET ADDRESS
7006 043530 012700 000017      10$:  MOV      #P.GETSTATUS,R0 ;GET STATUS COMMAND CODE NO IE
7007 043534 052700 100000      BIS      #P.ACK,R0     ;SET ACK
7008 043540 010021      MOV      R0,(R1)+     ;STORE GET STATUS IN PACKET
7009 043542 005021      CLR      (R1)+       ;CLEAR UNUSED WORD
7010 043544 000207      RTS      PC           ;RETURN
7011
7012
7013      ;ROUTINE TO SETUP A CHARACTERISTIC DATA BLOCK AT A TEST ADDRESS
7014      ;
7015      ;-
7016 043546      T9CHAR:
7017 043546      SAVREG              ;SAVE R1-R5 UNTIL NEXT RETURN
7018 043552 012700 041700      MOV      #T9DATA,R0    ;GET T9PACKET DATA POINTER
7019 043556 013701 041734      MOV      T9LOADD,R1    ;ASSUME NOT ABOVE 28K
7020 043562 005737 003102      TST      KTENABLE      ;TESTING ABOVE 28K?
7021 043566 001404      BEQ      10$           ;BR IF NO
7022 043570 013701 041736      MOV      T9PAR6,R1     ;SET TEST ADDRESS ABOVE 28K
7023 043574 012021      10$:  MOV      (R0)+,(R1)+   ;STORE DATA WORD 1
7024 043576 012021      MOV      (R0)+,(R1)+   ;STORE DATA WORD 2
7025 043600 012021      MOV      (R0)+,(R1)+   ;STORE DATA WORD 3
7026 043602 012021      MOV      (R0)+,(R1)+   ;STORE DATA WORD 4
7027 043604 012021      MOV      (R0)+,(R1)+   ;STORE DATA WORD 5
7028 043606 000207      RTS      PC           ;RETURN
7029 043610      ENDTST
      043610
      043610 104401

```

L10071: TRAP C\$ETST


```

7031 .SBTTL TEST 10: INITIALIZE AFTER WRITE CHARACTERISTICS
7032 ;*
7033 ; TEST DESCRIPTION:
7034 ;
7035 ; This test verifies that a Hardware Initialize command
7036 ; invoked after a Write Characteristics command sets up
7037 ; the Command, Message and Characteristic image blocks
7038 ; in the controller ram correctly.
7039 ;
7040 ; TEST STEPS:
7041 ;
7042 ; REPEAT FOR LOOPCNT
7043 ; BEGIN
7044 ; Do WRITE CHARACTERISTICS command.
7045 ; If the NBA bit in the TSSR register is NOT=0 then Print Error.
7046 ; Write to TSSR register to soft initialize the controller
7047 ; If controller RAM 310-377 NOT=0 then Print Error
7048 ; END
7049 ;--
7050
7051
7052 043612 BGNTST
7053 043612
7053 043612 005037 002170 CLR FATFLG ;CLEAR FATAL ERROR FLAG
7054 043616 012737 005672 002146 MOV #EPRT1,EPRTSW ;SET UP ERROR MESSAGE SWITCH
7055 043624 005037 003100 CLR KTFLG ;HOLD OFF KT11
7061 043630 012700 044262 MOV #TST10ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
7062 043634 004737 017232 JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
7063 043640 012737 000002 002164 MOV #2.,LOOPCNT ;PERFORM 2 ITERATIONS
7064 043646
7065 043646 004737 044536 T10LOOP: JSR PC,T10REST ;SET PACKET TO START-UP VALUES
7066
7067 043652 012703 002732 MOV #TSTBLK+10.,R3 ;START OF TEST DATA
7068 043656 012704 044220 MOV #T10PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
7069 043662 012764 000010 000006 MOV #8.,PKBCNT(R4) ;START WITH MINIMUM ALLOWABLE VALUE
7070 043670
7071 043670 004737 016470 S#: JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
7072 043674 103405 BCS 10# ;BR IF SOFT INIT OKAY
7073 043676 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
7074 043700 ERRDF ER#NO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
7075
7076
7077 043710 005037 002170 ;Do WRITE CHARACTERISTICS command.
7078 043714 010465 177776 10#: CLR FATFLG ;CLEAR FATAL ERROR FLAG
7079 043720 004737 017060 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
7080 043724 FORCERROR 12# JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
7081 043740 103407 BCS 15# ;GOODFORCE ERROR IF FORCER=1
7082 043742 010001 MOV R0,R1 ;BR IF CARRY SET (GOOD RETURN)
7083 043744 NEXT.ERRNO ;SAVE CONTENTS OF TSSR
7084 043744 104455 12#: ERRDF ERRNO,T10SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
7085 043746 001751 TRAP C$ERDF
7086 043750 044447 .WORD 1000
7087 .WORD SFIERR
7088 .WORD SFIMSC

```

```

043752 011520
7085 043754 004737 017724      JSR      PC,FATCHK      ;INC AND CHECK FOR MORE THAN 25 ERRORS
7086 043760      15$: CKLOOP      ;LOOP ON ERROR, IF FLAG SET
043760 104406      TRAP      C$CLP1
7087 043762 016501 000000      MOV      TSSR(R5),R1    ;GET THE CONTENTS OF TSSR
7088 043766 012702 000200      MOV      #SSR,R2       ;EXPECTED CONTENTS OF TSSR
7089 043772 032701 000100      BIT      #0FL,R1       ;IS OFF-LINE BIT SET ?
7090 043776 001402      BEQ      25$           ;BRANCH IF NOT OFF-LINE
7091 044000 052702 000100      BIS      #0FL,R2       ;SET OFF-LINE IN EXPECTED DATA
7092
7093      ;If the NBA bit in the TSSR register is NOT=0 then Print Error.
7094 044004 25$:      FORCERROR      27$           ;@@D
7095 044004      CMP      R2,R1         ;DOES EXPECTED MATCH RECEIVED ?
7096 044020 020201      BEQ      30$           ;OKAY IF MATCH
7097 044022 001404      NEXT.ERRNO
7098 044024      27$: ERRHRD  ERRNO,T1ONBA,PKTSSR ;NBA NOT ZERO
7099 044024      TRAP      C$ERHRD
044024 104456      .WORD      1002
044026 001752      .WORD      T1ONBA
044030 044374      .WORD      PKTSSR
044032 011520
7100 044034 30$:      CKLOOP      ;LOOP ON ERROR ?
044034 104406      TRAP      C$CLP1
7101
7102      ;Write to TSSR register to soft initialize the controller
7103 044036 40$:      JSR      PC,SOFINIT    ;WRITE TO TSSR TO SOFT INITIALIZE
7104 044036 004737 016470      FORCERROR      42$           ;@@D
7105 044042      BCS      50$           ;BR IF SOFT INIT OKAY
7106 044056 103405      MOV      R0,R1         ;SAVE CONTENTS OF TSSR
7107 044060 010001      NEXT.ERRNO
7108 044062      42$: ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
7109 044062      TRAP      C$ERDF
044062 104455      .WORD      1003
044064 001753      .WORD      SFIERR
044066 003550      .WORD      SFIMSG
044070 011506
7110
7111      ;If controller RAM 310-377 NOT=0 then Print Error
7112 044072 012704 000310      50$: MOV      #310,R4     ;START WITH LOC 310
7113 044076 005002      CLR      R2            ;MEMORY EXPECTED SHOULD BE 000000
7114 044100 004737 017060      JSR      PC,CHKTSSR    ;WAIT FOR SSR READY
7115 044104 110465 177777      60$: MOVB     R4,TSDBH(R5) ;SELECT RAM ADDRESS
7116 044110 116501 177776      MOVB     TSBAL(R5),R1  ;READ LOC CONTENTS
7117 044114      FORCERROR      62$,NOTSSR ;@@D
7118 044124 120102      CMPB     R1,R2         ;CHECK MEMORY FOR 000000
7119 044126 001406      BEQ      70$           ;BRANCH IF DATA OKAY
7120 044130      NEXT.ERRNO
7121 044130      62$: ERRDF  ERRNO,T1OMEM,RAMEXP ;MEMORY NOT ZERO AFTER INIT.
044130 104455      TRAP      C$ERDF
044132 001754      .WORD      1004
044134 044335      .WORD      T1OMEM
044136 016224      .WORD      RAMEXP
7122 044140 004737 017724      JSR      PC,FATCHK    ;INC AND CHECK FOR MORE THAN 25 ERRORS
7123 044144      70$: CKLOOP
044144 104406      TRAP      C$CLP1
7124 044146      ESCAPE  TST          ;EXIT ON FATAL ERROR
044146 104410      TRAP      C$ESCAPE

```



```

044150 000434 .WORD L10075-.
7125
7126 044152 005204 82$: INC R4 ;LOOK AT NEXT RAM LOC.
7127 044154 020427 000377 CMP R4,#377 ;AT TOP OF RAM ADDRESS SPACE
7128 044160 001351 BNE 60$ ;BRANCH TILL ALL MEMORY TESTED
7129
7130
7131 044162 005737 002170 TST FATFLG ;ANY FATAL ERRORS ?
7132 044166 001402 BEQ 160$ ;BRANCH IF NOT
7133 044170 004737 017776 JSR PC,CKDROP ;TRY TO DROP THE UNIT
7134 044174 004737 017200 160$: JSR PC,TSTLOOP ;DONE ALL ITERATIONS?
7135 044200 103002 BCC 165$ ;BR IF YES
7136 044202 000137 043646 JMP T10LOOP ;LOOP UNTIL ITERATION COUNT DONE
7137 044206
7138 044206 EXIT TST
044206 104432 TRAP C$EXIT
044210 000374 .WORD L10075-.
7139
7140

```



```

7142
7143
7144
7145
7147 044212
7149 044220
7150 044220 100004
7151 044222 044230
7152 044224 000000
7153 044226 000010
7154
7155 044230
7156 044230 044242
7157 044232 000000
7158 044234 000016
7159 044236 000000 000000
7160
7161 044242
7162
7163
7164
7165 044262 111 156 151 TST10ID: .ASCIZ 'Initialization After WRITE CHARACTERISTICS'
7166 044335 111 156 143 T10MEM: .ASCIZ 'Incorrect RAM Data After Init'
7167 .EVEN
7168 044374 127 122 111 T10NBA: .ASCIZ 'WRITE CHARACTERISTICS Command Not Accepted'
7169 044447 103 157 156 T10SSR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
7170

```

```

;+
;LOCAL STORAGE FOR THIS TEST
;-

      .BLKB   10-<.-TUV2A&7>
T10PACKET:
      .WORD   100004
      .WORD   T10DATA
      .WORD   0
      .WORD   8.
;COMMAND PACKET FOR TEST
;WRITE CHARACTERISTICS COMMAND, WITH ACK
;ADDRESS OF CHARACTERISTICS BLOCK
;STARTING VALUE OF BLOCK SIZE

T10DATA:
      .WORD   T10BFR
      .WORD   0
      .WORD   14.
      .WORD   0.0
;CHARACTERISTICS DATA BLOCK
;ADDRESS OF MESSAGE BUFFER
;LENGTH OF MESSAGE BUFFER

T10BFR: .BLKW  8.
;MESSAGE BUFFER

;LOCAL TEXT MESSAGES FOR TEST
;-

```

```

7172
7173
7174
7175
7176
7177
7178
7179
7180
7181 044536
7182 044536
7183 044542 012701 044220
7184 044546 012721 100004
7185 044552 012721 044230
7186 044556 005021
7187 044560 012721 000010
7188 044564 012721 044242
7189 044570 005021
7190 044572 012721 000016
7191 044576 005021
7192 044600 005011
7193 044602 000207
7194 044604
    044604
    044604 104401

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

T10REST:
    SAVREG                ;SAVE THE REGISTERS
    MOV #T10PACKET,R1     ;START OF THE PACKET
    MOV #100004,(R1)+     ;WRITE CHARACTERISTICS WITH ACK
    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)
    RTS PC                ;RETURN
    ENDTST

L10075: TRAP C$ETST
    
```



```

7302 045504 000000          .WORD 0
7303 045506 000010          .WORD 8.          ;STARTING VALUE OF BLOCK SIZE
7304
7305
7306 045510          T11DTA:          ;SELECT DATA BLOCK
7307 045510 045076          .WORD T11BFR      ;ADDRESS OF MESSAGE BUFFER
7308 045512 000000          .WORD 0
7309 045514 000400          .WORD 256.        ;LENGTH OF MESSAGE BUFFER
7310 045516 000000 000000  .WORD 0.0
7311
7312
7313          ;*
7314          ;LOCAL TEXT MESSAGES FOR TEST
7315          ;-
7316
7317 045522          127      122      111  T11NBA: .ASCIZ 'WRITE SUBSYSTEM MEMORY Command Not Accepted'
7318 045576          105      170      160  T11NINI: .ASCIZ 'Expected Interrupt Not Received On WRITE SUBSYSTEM MEMORY'
7319 045670          102      141      163  TST11ID: .ASCIZ 'Basic WRITE SUBSYSTEM MEMORY Command'
7320          .EVEN
7321

```



```

7323
7324
7325
7326
7327
7328
7329
7330
7331 045736
7332 045736
7333 045742 012701 045060
7334 045746 012721 100206
7335 045752 012721 045070
7336 045756 005021
7337 045760 012721 000006
7338 045764 005021
7339 045766 005021
7340 045770 005011
7341 045772 000207
7342
7343
7344 045774
7345 045774
7346 046000 012701 045500
7347 046004 012721 100204
7348 046010 012721 045510
7349 046014 005021
7350 046016 012721 000010
7351 046022 012721 045076
7352 046026 005021
7353 046030 012721 000400
7354 046034 005021
7355 046036 005011
7356 046040 005037 045076
7357 046044 000207
7358 046046
    046046
    046046 104401

```

```

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;WRITE SUBSYSTEM MEMORY COMMAND
;
;-
T11REST:
    SAVREG
    MOV #T11PACKET,R1
    MOV #100206,(R1)+
    MOV #T11DATA,(R1)+
    CLR (R1)+
    MOV #6.,(R1)+
    CLR (R1)+
    CLR (R1)+
    CLR (R1)
    RTS PC
;SAVE THE REGISTERS
;START OF THE PACKET
;WRITE SUBSYSTEM MEM. WITH ACK, IE
;ADDRESS OF DATA BLOCK
;EXTENDED ADDRESS
;SIZE OF DATA BLOCK IN BYTES
;CLEAR BSELO AND BSEL1
;CLEAR SEL2
;CLEAR DATA AREA
;RETURN

```

```

T11RST:
    SAVREG
    MOV #T11PK2,R1
    MOV #100204,(R1)+
    MOV #T11DTA,(R1)+
    CLR (R1)+
    MOV #8.,(R1)+
    MOV #T11BFR,(R1)+
    CLR (R1)+
    MOV #256.,(R1)+
    CLR (R1)+
    CLR (R1)
    CLR T11BFR
    RTS PC
    ENDTST
;SAVE THE REGISTERS
;START OF THE PACKET
;WRITE CHARA. WITH ACK, IE
;ADDRESS OF CHARAISTICS DATA BLOCK
;EXTENDED ADDRESS
;SIZE OF DATA BLOCK IN BYTES
;MESSAGE BUFFER ADDRESS
;LENGTH OF MESSAGE BUFFER
;CLEAR 1ST LOC IN MESSAGE BUFFER
;RETURN

```

```

L10076: TRAP C$ETST

```

```

7360          .SBTTL  HARDWARE PARAMETER CODING SECTION
7361
7362          ;**
7363          ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
7364          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
7365          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
7366          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
7367          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
7368          ; WITH THE OPERATOR.
7369          ;--
7370 046050      BGNHRD
7370 046050      000015
7370 046052
7371
7372 046052      GPRMA  HPM1,0,0,160000,177776,YES      ;GET TSBA/TSDB REGISTER ADDRESS.
7372 046052      000031
7372 046054      046104
7372 046056      160000
7372 046060      177776
7373 046062      GPRMA  HPM2,2,0,0,776,YES              ;GET VECTOR ADDRESS.
7373 046062      001031
7373 046064      046133
7373 046066      000000
7373 046070      000776
7374 046072      GPRMD  HPM3,4,0,340,0,7,YES           ;GET INTERRUPT PRIORITY.
7374 046072      002032
7374 046074      046157
7374 046076      000340
7374 046100      000000
7374 046102      000007
7375 046104      ENDHRD
7375 046104
7376 046104      104      105      126  HPM1:  .ASCIZ  'DEVICE ADDRESS (TSSR) '
7377 046133      111      116      124  HPM2:  .ASCIZ  'INTERRUPT VECTOR '
7378 046157      111      116      124  HPM3:  .ASCIZ  'INTERRUPT PRIORITY '
7379
7380

```



```

7416
7417
7418
7419
7420
7421 046432
7422
7423
7424
7425 046432

      046432 046450
      046434 000005
      046436

7426
7427
7428
7429
7430 046436
7431 046436
      046436 000000
      046440 000003
      046442
7432 046442 172522
7433 046444 000224
7434 046446 000240
7435 046450
      046450
7436 046450
7437
7438      000001

```

```

;
; AND AN ADJUSTMENT TO ACCOUNT FOR THE "LASTAD BIT7" HACK
; DESCRIBED IN "SUPPRG.MEM" (FOR REV C).
;

```

```

PATCH::
;      .IF      NZ,..E377
;      .=.!377+1
;      .ENDC
LASTAD      ;SET LAST USED ADDRESS.

```

```

.EVEN
.WORD T$FREE
.WORD T$SIZE

```

```

L$LAST::
.SBTTL  HARD CODED P-TABLE
;+
;      DIAGNOSTIC IS PRE-PARAMETERIZED PER THIS TABLE
;--

```

```

BGNSETUP      1
BGNPTAB

```

```

.WORD 0
.WORD L10104-./2-1

```

L10102:

```

.WORD      172522
.WORD      224
.WORD      PRI05
ENDPTAB

```

L10104:

```

ENDSETUP
.END

```


ADDSSR 011612 G	C\$AU = 000052	DEBUGM 011304	FATERR= 000060	HIADDR= 001400
ADR = 000020 G	C\$AUTO= 000061	DEVCNT 002166 G	FATFLG 002170 G	HIMEM = 007776
AMBTSS 006156	C\$BRK = 000022	DEVDR0 023420	FERCM 011374	HOE = 100000 G
ASSEMB= 000010	C\$BSEG= 000004	DEVNRD 023337	FIFEXP 011642 G	HPM1 046104
A1716 = 000003	C\$BSUB= 000002	DEVNXR 023255	FIF1MS 011714	HPM2 046133
BADDAI 003110 G	C\$CEFG= 000045	DEVONL 023173	FIF2MS 011763	HPM3 046157
BADSSR 016374 G	C\$CLCK= 000062	DEVSUM 023136	FILLME 020216	IBE = 010000 G
BAR = 174402	C\$CLEA= 000012	DFPTBL 002124 G	FNOINT 004113	IDU = 000040 G
BENBSW 002174 G	C\$CLOS= 000035	DIAGMC= 000000	FORCER 002144 G	IER = 020000 G
BIE = 040000	C\$CLP1= 000006	DICEA = 000001	FREE 003072 G	IFAU 004154
BIT0 = 000001 G	C\$CVEC= 000036	DLCYL = 000177	FREEHI 003076	INCRK 017566
BIT00 = 000001 G	C\$DCLN= 000044	DLNER= 100200	FRESIZ 003074 G	INTCPC 016644
BIT01 = 000002 G	C\$DODU= 000051	DLERR = 177730	FUSI 004015	INTFLA 016641
BIT02 = 000004 G	C\$DRPT= 000024	DLGETS= 000004	F\$AU = 000015	INTMAS 016640
BIT03 = 000010 G	C\$DU = 000053	DLRDHD= 000010	F\$AUTO= 000020	INTR 016712 G
BIT04 = 000020 G	C\$EDIT= 000003	DLRDNH= 000016	F\$BGN = 000040	INTREC 002172 G
BIT05 = 000040 G	C\$ERDF= 000055	DLSR = 000013	F\$CLEA= 000007	INTVEC 016642
BIT06 = 000100 G	C\$ERHR= 000056	DLUN = 000006	F\$DU = 000016	INTX 004176
BIT07 = 000200 G	C\$ERRO= 000060	DSBINT 016700	F\$END = 000041	IOKCKI= 000200
BIT08 = 000400 G	C\$ERSF= 000054	DUAD12 004541	F\$HARD= 000004	IOKSTP= 000001
BIT09 = 001000 G	C\$ERSO= 000057	DUFLG 003060 G	F\$HW = 000013	IPRI 002160 G
BIT1 = 000002 G	C\$ESCA= 000010	DUMMY 003030	F\$INIT= 000006	ISR = 000100 G
BIT10 = 002000 G	C\$ESEG= 000005	EF.CON= 000036 G	F\$JMP = 000050	IVEC 002156 G
BIT11 = 004000 G	C\$ESUB= 000003	EF.NEW= 000035 G	F\$MOD = 000000	IXE = 004000 G
BIT12 = 010000 G	C\$ETST= 000001	EF.PWR= 000034 G	F\$MSG = 000011	I\$AU = 000041
BIT13 = 020000 G	C\$EXIT= 000032	EF.RES= 000037 G	F\$PROT= 000021	I\$AUTO= 000041
BIT14 = 040000 G	C\$GETB= 000026	EF.STA= 000040 G	F\$PWR = 000017	I\$CLN = 000041
BIT15 = 100000 G	C\$GETW= 000027	EMAXDU 017521	F\$RPT = 000012	I\$DU = 000041
BIT2 = 000004 G	C\$GMAN= 000043	EN = 000000	F\$SEG = 000003	I\$HRD = 000041
BIT3 = 000010 G	C\$GPHR= 000042	ENAIN 016646	F\$SOFT= 000005	I\$INIT= 000041
BIT4 = 000020 G	C\$GPLO= 000030	ENVIRN 021356	F\$SRV = 000010	I\$MOD = 000040
BIT5 = 000040 G	C\$GPRI= 000040	EPRTSW 002146 G	F\$SUB = 000002	I\$MSG = 000041
BIT6 = 000100 G	C\$INIT= 000011	EPRT1 005672	F\$SW = 000014	I\$PROT= 000040
BIT7 = 000200 G	C\$INLP= 000020	EPRT2 005672	F\$TEST= 000001	I\$PTAB= 000041
BIT8 = 000400 G	C\$MANI= 000050	EPRT3 005672	GDDAT 003112 G	I\$PWR = 000041
BIT9 = 001000 G	C\$MEM = 000031	ERCM 011405	GERRMA 002142 G	I\$RPT = 000041
BOE = 000400 G	C\$MSG = 000023	ERRHI 002202 G	GETPAT 020722 G	I\$SEG = 000041
BRINIT 004355	C\$OPEN= 000034	ERRK 017500	GETSEL 021004 G	I\$SETU= 000041
BSELO = 000000	C\$PNTB= 000014	ERRLO 002204 G	G\$CNT0= 000200	I\$SF? = 000041
BSEL1 = 000001	C\$PNTF= 000017	ERRNO = 002120	G\$DELM= 000372	I\$SRV = 000041
CHKAMB 016540	C\$PNTS= 000016	ERRVEC= 000004 G	G\$DISP= 000003	I\$SUB = 000041
CHKMAN 021226 G	C\$PNTX= 000015	ERTABE 003330	G\$EXCP= 000400	I\$TST = 000041
CHKTSS 017060	C\$QIO = 000377	ERTABL 003130	G\$HILI= 000002	J\$JMP = 000167
CKDROP 017776	C\$RDBU= 000007	ESUM 017502	G\$LOLI= 000001	KIPAR0= 172340
CKEMAX 017624	C\$REFG= 000047	EVL = 000004 G	G\$NO = 000000	KIPAR1= 172342
CKMSG 011032 G	C\$RESE= 000033	EXBCNT= 000010	G\$OFFS= 000400	KIPAR2= 172344
CKMSG2 011152 G	C\$REVI= 000003	EXPBRE 016176 G	G\$OFSI= 000376	KIPAR3= 172346
CKRAM 010354 G	C\$RFLA= 000021	EXPD 002176 G	G\$PRMA= 000001	KIPAR4= 172350
CKRAM2 010730 G	C\$RPT = 000025	EXPGOT 004431	G\$PRMD= 000002	KIPAR5= 172352
CMPMEM 020402	C\$SEFG= 000046	EXPGT2 004465	G\$PRML= 000000	KIPAR6= 172354
CONFIG 020044	C\$SPRI= 000041	EXPMMSG 002266 G	G\$RADA= 000140	KIPAR7= 172356
COUNT 002254 G	C\$SVEC= 000037	EXPREC 016170 G	G\$RADB= 000000	KIPDR0= 172300
CSR = 174400	C\$TPRI= 000013	EXTA 005232	G\$RADD= 000040	KIPDR1= 172302
CSRADD 002154 G	DAR = 174404	EXTEND 005230	G\$RADL= 000120	KIPDR2= 172304
CTAB 003116 G	DATA = 002256 G	E\$END = 002100	G\$RADO= 000020	KIPDR3= 172306
CTABE 003130 G	DATAFL 014710	E\$LOAD= 000035	G\$XFER= 000004	KIPDR4= 172310
CTABM 003116 G	DATASC 020760	FATCHK 017724	G\$YES = 000010	KIPDR5= 172312

KIPDR6= 172314	L\$REV 002010 G	L10057 032124	NULCR 004426	PRI04 = 000200 G
KIPDR7= 172316	L\$RPT 022674 G	L10060 032312	NXM = 004000	PRI05 = 000240 G
KTENAB 003102 G	L\$SOFT 046212 G	L10061 036712	NXR 003636	PRI06 = 000300 G
KTFLG 003100 G	L\$SPC 002056 G	L10062 034120	NXRERR 005176 G	PRI07 = 000340 G
KTINIT 021444	L\$SPCP 002020 G	L10063 034512	NXRX 003675	PRMESS 013702
KTOFF 020070	L\$SPTP 002024 G	L10064 035402	NXTU 022032	PRMNO 002264 G
KTON 020052	L\$STA 002030 G	L10065 035630	OFL = 000100	PRMSG0 015246 G
LERRMA 002140 G	L\$SW 002134 G	L10066 040430	ONEFIL = 000000	PRMSG0 015426
LERRNO= 000000	L\$TEST 002114 G	L10067 037176	O\$APTS = 000000	PRMSG1 015473
LISTAL = 000001	L\$TIML 002014 G	L10070 037434	O\$AU = 000001	PRMSG2 015531
LOE = 040000 G	L\$UNIT 002012 G	L10071 043610	O\$BGNR = 000001	PROASC 014540
LOOPCN 002164 G	L10000 002132	L10072 041022	O\$BGNS = 000001	PRIASC 014605
LOOPCO 012600	L10001 002144	L10073 041322	O\$DU = 000001	PST32W 003104 G
LOOPFL 003114 G	L10002 005226	L10074 041630	O\$ERRT = 000000	PUNIT 022320
LOT = 000010 G	L10003 011516	L10075 044604	O\$GNSW = 000001	PW.D11 = 000021
L\$ACP 002110 G	L10004 011546	L10076 046046	O\$POIN = 000001	PW.D13 = 000022
L\$APT 002036 G	L10005 011564	L10077 045032	O\$SETU = 000001	PW.D22 = 000020
L\$AU 022366 G	L10006 011572	L10100 046104	PASRPT 022064	PW.NOP = 000000
L\$AUT 002070 G	L10007 011610	L10101 046226	PATCH 046432 G	PW.NO1 = 000023
L\$AUTO 022572 G	L10010 011626	L10102 046442	PATDAT 020756	PW.RDE = 000024
L\$CCP 002106 G	L10011 011640	L10104 046450	PC.ERA = 002400	PW.RDR = 000001
L\$CLEA 022646 G	L10012 011712	MEMADD 013426 G	PC.IER = 002000	PW.RDS = 000005
L\$CO 002032 G	L10013 012062	MENASC 021175	PC.NOO = 001000	PW.RFI = 000003
L\$DEPO 002011 G	L10014 012576	MENERR 021122	PC.REL = 000000	PW.WCT = 000006
L\$DESC 003342 G	L10015 013424	MENRES 021224	PC.REW = 000400	PW.WFI = 000004
L\$DESP 002076 G	L10016 013446	MESBFA 002716 G	PKBCNT = 000006	PW.WFM = 000007
L\$DEVP 002060 G	L10017 016174	MESBFN 014460	PKHI = 000004	PW.WMI = 000010
L\$DISP 046404 G	L10020 016202	MESHEA 014643	PKLOW = 000002	PW.WNP = 000011
L\$DLY 002116 G	L10021 016210	MMVEC = 000250	PKTADD 007116	PW.WTR = 000002
L\$DTP 002040 G	L10022 016222	MPR = 174406	PKTFRM 007060	P.ACK = 100000
L\$DTYP 002034 G	L10023 016244	MSA.FR = 000006	PKTGET 011550 G	P.CMD = 000037
L\$DU 022464 G	L10024 016272	MSA.NO = 000000	PKTMES 011574 G	P.CONT = 000012
L\$DUT 002072 G	L10025 016432	MSA.NR = 000004	PKTNEW 007153	P.CVC = 040000
L\$DVTY 003334 G	L10026 016742	MSA.VO = 000002	PKTRAM 004643 G	P.FMT = 000140
L\$EF 002052 G	L10030 022316	MSGEXP 011630 G	PKTSSR 011520 G	P.FORM = 000011
L\$ENVI 002044 G	L10031 022462	MSGLOO 012536 G	PNT = 001000 G	P.GETS = 000017
L\$ETP 002102 G	L10032 022570	MSGSTA 012022 G	PRAMPK 013450	P.IE = 000200
L\$EXP1 002046 G	L10033 022644	MSGSUB 013414 G	PRBEXP 016164	P.INIT = 000013
L\$EXP4 002064 G	L10034 022672	MS.ATT = 000006	PRBMSG 016032	P.MODE = 007400
L\$EXP5 002066 G	L10035 023134	MS.EXT = 000200	PRBTOT 016166	P.OPP = 020000
L\$HARD 046052 G	L10036 023724	MS.RSD = 000001	PRBYTE 015616 G	P.POSI = 000010
L\$HIME 002120 G	L10037 023572	MS.RSF = 000020	PRI = 002000 G	P.READ = 000001
L\$HPCP 002016 G	L10040 023654	MS.RST = 000010	PRIADD 007532	P.SWB = 010000
L\$HPTP 002022 G	L10041 024676	NBA = 002000	PRIAO 007602	P.WRIT = 000005
L\$HW 002124 G	L10042 024116	NEWPAS 022020	PRIAXO 007164 G	P.WRTC = 000004
L\$ICP 002104 G	L10043 024310	NODEV 003062 G	PRIBXO 007432	P.WRTS = 000006
L\$INIT 021606 G	L10044 024510	NOINIT 004233	PRIPKT 006712 G	QVP 002152 G
L\$LADP 002026 G	L10045 026162	NOINTR 004117	PRIRAM 007440	RAMASC 013616
L\$LAST 046436 G	L10046 025240	NOITS 002136 G	PRITAD 007646	RAMDAT 002206 G
L\$LOAD 002100 G	L10047 025560	NOMAN 021262	PF.ITSS 005264	RAMER 010456 G
L\$LUN 002074 G	L10050 030214	NP.IR = 000200	PRITO 007716	RAMERR 016204 G
L\$MREV 002050 G	L10051 026506	NP.LOO = 000040	PRIXOR 007314 G	RAMEXP 016224 G
L\$NAME 002000 G	L10052 026772	NP.OUT = 000100	PRI00 = 000000 G	RAMFHR 014362
L\$PRIO 002042 G	L10053 027220	NP.WRP = 000020	PRI01 = 000040 G	RAMFOR 007470
L\$PROT 021576 G	L10054 031424	NSI 004050	PRI02 = 000100 G	RAMHLD 010640
L\$PRT 002112 G	L10055 033500	NSINIT 004305	PRI03 = 000140 G	RAMIOP 010644
L\$REPP 002062 G	L10056 031700	NUL 004425		RAMPD 010715

SYMBOL TABLE

RAMR5H	010642		SO.IDB=	000010		TSSX	003716		T\$\$AU =	010031		T3PACK	025620
RAMSIZ	002246	G	SO.IFB=	000002		TSTBLK	002720	G	T\$\$AUT=	010033		T3SSR	025655
RAMTAD	016212	G	SO.IFP=	000001		TSTCNT	002162	G	T\$\$CLE=	010034		T3TSBA	026071
RBPCRA	014755		SO.ILD=	000020		TSTEND	017442		T\$\$DAT=	010104		T3.1	024734
RCVHIA	002250	G	SO.ION=	000040		TSTFLA	002260	G	T\$\$DU =	010032		T3.2	025254
RCVLOA	002252	G	SO.IRD=	000100		TSTLOO	017200	G	T\$\$HAR=	010100		T4	026164
RDERR	005104		SO.IRW=	000004		TSTPTR	002262	G	T\$\$HW =	010000		T4BFR	027274
READ =	000014		SO.ISP=	000200		TSTSET	017232	G	T\$\$INI=	010030		T4DATA	027260
READY =	000001		S1.ICE=	002000		TST1ID	023704		T\$\$MSG=	010025		T4INT	027745
RECMSG	002432	G	S1.IEO=	010000		TST1OI	044262		T\$\$PC =	000001		T4LOOP	026220
RECV	002200	G	S1.IFM=	001000		TST1II	045670		T\$\$PRO=	010027		T4NBA	027406
REGSAV	020662		S1.IHE=	000400		TST2ID	024672		T\$\$PTA=	010103		T4PACK	027250
REWIND	010254	G	S1.IID=	004000		TST3ID	026143		T\$\$RPT=	010035		T4REST	030146
RMCHBE=	000167		S1.IIR=	020000		TST4ID	030117		T\$\$SEG=	010000		T4SP	027270
RMCHEN=	000200		S1.I2R=	040000		TST5ID	031407		T\$\$SOF=	010101		T4SSR	027656
RMMSGB=	000104		S1.PAR=	100000		TST6ID	033401		T\$\$SRV=	010026		T4TSBA	030034
RMMSGE=	000117		S2.ATI=	000010		TST7ID	036537		T\$\$SUB=	010077		T4.1	026220
RMPKTB=	000020		S2.BTI=	000004		TST8ID	040272		T\$\$SW =	010001		T4.2	026522
RMPKTE=	000027		S2.DIM=	000200		TST9ID	042100		T\$\$TES=	010076		T4.3	026774
RMR =	010000		S2.ILW=	000100		TTIBFR=	177562	G	T1	023470	G	T42DAT	027314
RWPACK	010350		S2.INR=	000020		TTICSR=	177560	G	T1LOOP	023524		T42DON=	027330
SC =	100000		S2.OUT=	000040		TTIVEC=	000060	G	T1.1	023526		T42NBA	027330
SCE =	020000		S2.UND=	000003		TTOBFR=	177566		T1.2	023606		T42REJ	027461
SCME	004711		TBLEND=	003030	G	TTOCSR=	177564		T10	043612	G	T44REJ	027560
SDELAY	010150		TCOASC	006017		TUV2A	002000	G	T10BFR	044242		T5	030216
SEEK =	000006		TCOCOD	006220		T\$ARGC=	000003		T10DAT	044230		T5BFR	030742
SELASC	021170		TEMP1	003064	G	T\$CODE=	001130		T10LOO	043646		T5DATA	030730
SELDAT=	000004		TEMP2	003066	G	T\$ERRN=	002120		T10MEM	044335		T5LOOP	030252
SEL2 =	000002		TERCLS=	000016		T\$EXCP=	000000		T10NBA	044374		T5NMSG	031310
SETMAP	020112		TESTNO=	000013		T\$FLAG=	000040		T10PAC	044220		T5NVCK	031131
SETU	022116		TEXASC	005756		T\$FREE=	046450		T10RES	044536		T5PACK	030720
SFFMSG	011566	G	TFCASC	006060		T\$GMAN=	000000		T10SSR	044447		T5SSR	031221
SFHERR	003603		TIMEXP	016246	G	T\$HILI=	000007		T11	044606	G	T5VCK	030762
SFIERR	003550		TIMSGO	016274		T\$LAST=	000001		T11BFR	045076		T5VCK2	031055
SFIMSG	011506	G	TINERR	011473		T\$LOLI=	000000		T11BS0	045070		T6	031426
SFPTBL	002134	G	TKB =	177562		T\$LSYM=	010000		T11BS1	045071		T6BFR	032352
SIFLAG	003106	G	TKS =	177560		T\$LTNO=	000013		T11BS2	045072		T6DATA	032340
SIMSG	011440		TMPBFR	002576	G	T\$NEST=	000000		T11DAT	045070		T6INT	033227
SKIPT	003332		TNAM	017426		T\$NS0 =	000000		T11DTA	045510		T6LOOP	031462
SOFINI	016470	G	TPB =	177566		T\$NS1 =	000005		T11LOO	044642		T6NBA	032406
SPACE	007760	G	TPS =	177564		T\$NS2 =	000002		T11NBA	045522		T6NINT	033136
SPM1	046226		TRANST	002134	G	T\$NS3 =	000003		T11NIN	045576		T6PACK	032330
SPM4	046272		TSBA =	177776	G	T\$PCNT=	000000		T11PAC	045060		T6REST	033426
SPM6	046322		TSBAH =	177777	G	T\$PTAB=	010103		T11PK2	045500		T6SSR	033047
SPM7	046352		TSBAL =	177776	G	T\$PTHV=	000001		T11RES	045736		T6TSBA	033316
SRO =	177572		TSBAM2	024530		T\$PTNU=	000001		T11RST	045774		T6.1	031462
SR1 =	177574		TSBAM3	024612		T\$SAVL=	177777		T11.1	044642		T6.2	031714
SR2 =	177576		TSDB =	177776	G	T\$SEGL=	177777		T2	023726	G	T6.3	032126
SR3 =	172516		TSDBH =	177777	G	T\$SEKO=	010000		T2LOOP	023764		T62DAT	032372
SSR =	000200		TSDBL =	177776	G	T\$SIZE=	000005		T2.1	023744		T62DON=	032406
STATCO	012064		TSFCOD	006560		T\$SUBN=	000001		T2.2	024120		T62REJ	032461
SVCGBL=	000000		TSREJ =	000006		T\$TAGL=	177777		T2.3	024312		T63REJ	032560
SVCINS=	000001		TSSDEF	006127		T\$TAGN=	010105		T3	024700	G	T64REJ	032653
SVCSUB=	000001		TSSR =	000000	G	T\$TEMP=	000014		T3INT	025733		T65REJ	032751
SVCTAG=	000001		TSSRBI	003400	G	T\$TEST=	000013		T3LOOP	024734		T7	033502
SVCTST=	000001		TSSRFO	005736		T\$TSTM=	177777		T3NBA	025630		T7BFR	035662
S\$LSYM=	010000		TSSRH =	000001	G	T\$TSTS=	000001		T3NINT	026011		T7BUFR	035724

T7DATA	035650	T8SSR	040014	T9.1	040466	XFERAS	016434	X\$FALS=	000040
T7DTA	035712	T8TSBA	040222	T9.2	041036	XNXM	017120	X\$OFFS=	000400
T7INT	036450	T8.1	036750	T9.3	041336	XORBFO	007246	X\$TRUE=	000020
T7LOOP	033536	T8.2	037212	T91L00	040610	XORFOR	007364	X1.COR=	020000
T7MBF	035744	T82REJ	037622	T92L00	041154	XST0 =	000006 G	X1.DLT=	100000
T7NBA	036041	T83REJ	037703	T93L00	041404	XST1 =	000010 G	X1.MBZ=	017375
T7NINT	036357	T84REJ	037733	T94TST	041742	XST2 =	000012 G	X1.RBP=	000400
T7NNBA	036123	T9	040432 G	UAM =	000200 G	XST3 =	000014 G	X1.SPA=	040000
T7PACK	035640	T9BFR	041712	UNITN	002150 G	XST4 =	000016 G	X1.UNC=	000002
T7PKT	035702	T9BKGN	042367	UNREC =	000006	XSOBOT=	000002	X2.BUF=	000100
T7RST	036566	T9BLK	041744	USI	004021	XSOCON	015022	X2.EXT=	000200
T7RT2	036640	T9CHAR	043546	WAITF	016744 G	XSOEOT=	000001	X2.OPM=	100000
T7SSR	036270	T9CKRA	043326 G	WC.IFA=	000200	XSOIE =	000040	X2.RCE=	040000
T7SSRM	036200	T9CONV	042746	WC.IFE=	000002	XSOILA=	000400	X2.REV=	000077
T7.1	033536	T9CT2	043140	WC.IGO=	000001	XSOILC=	001000	X2.SPA=	035400
T7.2	034122	T9DATA	041700	WC.IRE=	000010	XSOLET=	020000	X2.UNI=	000007
T7.3	034514	T9DPR	042546	WC.IRW=	000004	XSOMOT=	000200	X2.WCF=	002000
T7.4	035404	T9GETS	042126	WC.IOT=	000100	XSONEF=	002000	X3.DCK=	000010
T8	036714 G	T9HIAD	041732	WC.IIT=	000040	XSOONL=	000100	X3.MBZ=	000006
T8BFR	037472	T9KT	041740	WC.ISR=	000020	XSOPED=	000010	X3.MDE=	177400
T8BF2	037542	T9LOAD	041734	WF.IED=	000010	XSORLL=	010000	X3.OPI=	000100
T8DATA	037460	T9LOOP	040466	WF.IER=	000004	XSORLS=	040000	X3.REV=	000040
T8DTA	037530	T9MSGB	042271	WF.IHI=	000200	XSOTMK=	100000	X3.RIB=	000001
T8LOOP	036750	T9NINT	042455	WF.IRE=	000040	XSOVCK=	000020	X3.SPA=	000200
T8NBA	037562	T9NXM	042637	WF.IWF=	000020	XSOWLE=	004000	X3.TRF=	000020
T8NINT	040144	T9PACK	041670	WF.IWR=	000100	XSOWLK=	000004	X4.HSP=	100000
T8PACK	037450	T9PAR6	041736	WF.I3R=	000002	XS1CON	015067	X4.MBZ=	017400
T8PK2	037520	T9SETG	043504	WF.I4R=	000001	XS2CON	015134	X4.RCE=	040000
T8REST	040324	T9SWRT	043436	WRCHR	010152 G	XS3CON	015201	X4.TSM=	020000
T8RT2	040366	T9TBE	042076	WRERR	005011	XXCOMM	003070 G	X4.WRC=	000377
T8SR2	040070	T9WRTS	042202	WRTMSG	004754	X\$ALWA=	000000		

. ABS. 046450 000
 000000 001
 ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 31240 WORDS (123 PAGES)
 DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
 ELAPSED TIME: 00:30:00
 CZTKEA.BIC,CZTKEA/-SP=SVC/ML,CZTKEA