

TK25

TK25 FRT END FUNC #1
CZTKEB0

AH-T776B-MC

1 OF 1 OCT 1985

COPYRIGHT © 1984-85

digital
MADE IN USA

The image displays a grid of 144 small, illegible data plots or charts arranged in 12 rows and 12 columns. Each plot appears to be a technical or scientific visualization, possibly a spectrum or a data distribution, but the text and details within each plot are too small and faded to be discernible. The plots are organized in a regular grid pattern across the page.

d
E W

A ?;CZTKEBO TK25 FRT END FUNC #1
USER DOCUMENTATION

MACRO M1200 07-JUN-84 09:48 PAGE 2

SEQ 1

.REMA

IDENTIFICATION

PRODUCT ID: AC-T775B-MC
PRODUCT TITLE: CZTKEBO TK25 FRT END FUNC #1
PRODUCT DATE: 7 MAY 1985
DEPARTMENT: TAPE AND OPTICAL DIAGNOSTIC ENGINEERING
AUTHOR: RAYMOND CHANG

COPYRIGHT (C) 1984,1985 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 TK25 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

CZTKEAO (APRIL 1984)

NEW RELEASE

CZTKEBO (7 JUNE 1984)

REMOVE INTERRUPT PRIORITY HARDWARE QUESTION FROM P TABLE.

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 (CIQPMAD 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 "CIQPMO XXP" PROGRAMMER'S MANUAL NUMBER AC-S296A-AC.

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

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

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

RESTART ADDRESS: 152010 -----
THIS IS XXP+ 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 DATA 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; THEY 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 125252. 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.

```

746
747      .SBTTL PROGRAM HEADER
753      .MCALL SVC
754 000000 SVC ; INITIALIZE SUPERVISOR MACROS
755      .ENABLE LC
756      .NLIST BEX,CND
762 000000 .ENABL AMA,ABS
763      . = 2000
764 002000 BGNMOD TUV2A
       002000 TUV2A::
765
766      ;**
767      ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
768      ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
769      ;--
770
771
772 002000      POINTER BGNSH,BGNSFT,BGNAU,BGNDU,BGNRPT,BGNSETUP
773 002000      HEADER CZTKE,B,0,655.,0
       002000 L$NAME:: ;DIAGNOSTIC NAME
       002000      .ASCII /C/
       002001      .ASCII /Z/
       002002      .ASCII /T/
       002003      .ASCII /K/
       002004      .ASCII /E/
       002005      .BYTE 0
       002006      .BYTE 0
       002007      .BYTE 0
       002010 L$REV:: ;REVISION LEVEL
       002010      .ASCII /B/
       002011 L$DEPO:: ;0
       002011      .ASCII /0/
       002012 L$UNIT:: ;NUMBER OF UNITS
       002012 000001 .WORD T$PTHV
       002014 L$TIML:: ;LONGEST TEST TIME
       002014 001217 .WORD 655.
       002016 L$HPCP:: ;PTR. TO H.W. QUES.
       002016 046052 .WORD L$HARD
       002020 L$SPCP:: ;PTR. TO S.W. QUES.
       002020 046200 .WORD L$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$LADP:: ;DIAG. END ADDRESS
       002026 046424 .WORD L$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$DTP:: ;PTR. TO DISPATCH TABLE
       002040 046372 .WORD L$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	

```
775          .SBTTL  DEFAULT HARDWARE P-TABLE
776          ;++
777          ; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
778          ; THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
779          ; IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
780          ;--
781 002122    BGNHW  DFPTBL      ;DEFAULT HARD-P-TABLE
          002122 000003      .WORD  L10000-L$HW/2
          002124
          002124  L$HW::
782          DFPTBL::
783          .WORD  172522      ; 2ND (OF 2) REGISTERS.
784 002126 000224      .WORD  224      ; INTERRUPT VECTOR
785 002130 000240      .WORD  PRI05     ; INTERRUPT PRIORITY.
786 002132
          002132  L10000:
```

```
788 .SBTTL SOFTWARE P-TABLE
789
790 ;**
791 ; THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
792 ; PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
793 ;--
794 002132 BGNSW SFPTBL
      002132 000004 .WORD L10001-L$SW/2
      002134 L$SW::
      002134 SFPTBL::
795
796 002134 000000 TRANSTST:: .WORD 0 ;ENABLE RAM DUMP IF =1
797 002136 000000 NOITS:: .WORD 0 ; INHIBIT ITERATION OPTION.
798 ; ... 0 = ITERATE.
799 ; ...NZ = INHIBIT ITERATE.
800 002140 000031 LERRMAX:: .WORD 25. ; LOCAL (PER TEST) ERROR LIMIT
801 002142 000310 GERRMAX:: .WORD 200. ; GLOBAL (PER UNIT) ERROR LIMIT
802 002144 ENDSW
      002144 L10001:
803
```

805
812
817
823
824
825
826
827
828
829
830
831
832
836 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

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

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

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
000300
000240
000200
000140
000100
000040
000000

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

;
;OPERATOR FLAG BITS

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

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

837
838 002144

KT11 .. ;DEFINE MEMORY MANAGEMENT REGISTERS

.SBTTL MEMORY MANAGEMENT DEFINITIONS

;*KT11 VECTOR ADDRESS

000250

MMVEC= 250

;*KT11 STATUS REGISTER ADDRESSES

177572
177574
177576
172516

SR0= 177572
SR1= 177574
SR2= 177576
SR3= 172516

.IF NB

;*USER "I" PAGE DESCRIPTOR REGISTERS

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

.IF NB

;*USER "D" PAGE DESCRIPTOR 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
  .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
```

```

843                                     .SBTTL TK-25 REGISTER AND PACKET DEFINITIONS
844
845                                     ;
846                                     ; SOME GENERAL EQUATES.
847                                     ;
848
849         000004      ERRVEC==          4          ; POINTER TO ERROR VECTOR FOR BUS TIME OUT.
850         000060      TTIVEC==         60          ; INTERRUPT VECTOR FOR CONSOLE INPUT
851         177560      TTICSR==        177560       ; BUS ADDRESS OF CONSOLE INPUT
852         177562      TTIBFR==        177562       ; CONSOLE INPUT DATA BUFFER
853
854                                     ;+
855                                     ;BIT DEFINITIONS FOR TSSR REGISTER
856                                     ;-
857
858         100000      SC=          BIT15          ;SPECIAL CONDITION
859         040000      BIE=          BIT14          ;BUS INTERFACE ERROR
860         020000      SCE=          BIT13          ;SANITY CHECK ERROR
861         010000      RMR=          BIT12          ;MODIFICATION REFUSED
862         004000      NXM=          BIT11          ;NONEXISTANT MEMORY ERROR
863         002000      NBA=          BIT10          ;NEED BUFFER ADDRESS
864         001400      HIADDR= BIT9!BIT8          ;EXTENDED ADDRESS BITS
865         000200      SSR=          BIT7          ;SUB SYSTEM READY
866         000100      OFL=          BIT6          ;OFF LINE BIT
867         000060      FATERR= BIT4!BIT5          ;FATAL TERMINATION ERROR CODES
868         000016      TERCLS= BIT3!BIT2!BIT1     ;TERMINATION CODES
869
870
871                                     ;+
872                                     ;
873                                     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 0
874                                     ;(XST0)
875                                     ;
876                                     ;-
877
878         100000      XSOTMK= BIT15          ;TAPE MARK DETECTED
879         040000      XSORLS= BIT14          ;RECORD LENGTH SHORT
880         020000      XSOLET= BIT13          ;LOGICAL END OF TAPE
881         010000      XSORLL= BIT12          ;RECORD LENGTH LONG
882         004000      XSOWLE= BIT11          ;WRITE LOCK ERROR
883         002000      XSONEF= BIT10          ;NON EXECUTABLE FUNCTION
884         001000      XSQILC= BIT9          ;ILLEGAL COMMAND
885         000400      XSOILA= BIT8          ;ILLEGAL ADDRESS
886         000200      XSOMOT= BIT7          ;TAPE IN MOTION
887         000100      XSOONL= BIT6          ;TRANSPORT ON LINE
888         000040      XSOIE= BIT5          ;INTERRUPT ENABLE
889         000020      XSOVCK= BIT4          ;VOLUME CHECK BIT
890         000010      XSOPED= BIT3          ;PHASE ENCODED DRIVE
891         000004      XSOWLK= BIT2          ;WRITE LOCKED
892         000002      XSOBOT= BIT1          ;BEGINNING OF TAPE
893         000001      XSQEOI= BIT0          ;END OF TAPE
894
895
896                                     ;+
897                                     ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 1
898                                     ;(XST1)
899                                     ;-
    
```

```

900      100000      X1.DLT = BIT15      ;DATA LATE
901      040000      X1.SPARE= BIT14      ;NOT USED
902      020000      X1.COR = BIT13      ;CORRECTABLE DATA ERROR
903      017375      X1.MBZ = BIT12+BIT11+BIT10+BIT9+BIT7+BIT6+BIT5+BIT4+BIT3+BIT2+BIT0 ;ALWAYS 0
904      000400      X1.RBP = BIT8      ;READ BUS PARITY ERROR
905      000002      X1.UNC = BIT1      ;UNCORRECTABLE DATA OR HARD ERROR
906
907      ;+
908      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 2
909      ;(XST2)
910      ;-
911      100000      X2.OPM = BIT15      ;OPERATION IN PROGRESS (TAPE MOVING)
912      040000      X2.RCE = BIT14      ;RAM CHECKSUM ERROR
913      035400      X2.SPARE= BIT13+BIT12+BIT11+BIT9+BIT8 ;NOT USED BY TK-25 (ALWAYS=0)
914      002000      X2.WCF = BIT10      ;WRITE CLOCK FAILURE (FIFO NOT EMPTIED BY TRANSPORT)
915      000200      X2.EXTF = BIT7      ;IF WRITE CHAR CMD THEN = EXTENDED FEATURES ENABLED
916      000100      X2.BUFE = BIT6      ;IF WRITE CHAR CMD THEN = BUFFERING ENABLED
917      000077      X2.REV = 000077      ;IF WRITE CHAR CMD THEN = MICROCODE REVISION LEVEL
918      000007      X2.UNIT = BIT2+BIT1+BIT0 ;IF GET STATUS THEN = CURRENTLY SELECTED UNIT NO.
919
920      ;+
921      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 3
922      ;(XST3)
923      ;-
924      177400      X3.MDE = 177400      ;MICRO-DIAGNOSTIC ERROR CODE
925      000200      X3.SPARE= BIT7      ;NOT USED BY TK-25
926      000100      X3.OPI = BIT6      ;OPERATION INCOMPLETE
927      000040      X3.REV = BIT5      ;REVERSE
928      000020      X3.TRF = BIT4      ;TRANSPORT RESPONSE FAILURE
929      000010      X3.DCK = BIT3      ;DENSITY CHECK
930      000006      X3.MBZ =BIT2+BIT1      ;NOT USED ALWAYS 0
931      000001      X3.RIB = BIT0      ;REVERSE INTO BOT
932
933      ;+
934      ;BIT DEFINITIONS FOR EXTENDED STATUS REGISTER 4
935      ;(XST4)
936      ;-
937      100000      X4.HSP = BIT15      ;HIGH SPEED
938      040000      X4.RCE = BIT14      ;RETRY COUNT EXCEEDED
939      020000      X4.TSM = BIT13      ;TRANSPORT SPECIAL MODE
940      017400      X4.MBZ = BIT12+BIT11+BIT10+BIT9+BIT8 ;NOT USED ALWAYS 0
941      000377      X4.WRC = 000377      ;WRITE RETRY COUNT FIELD
942
943
944      ;+
945      ;
946      ;TSSR TERMINATION CODES (BIT 0-2)
947      ;
948      ;-
949
950      000006      TSREJ= 3*2      ;COMMAND REJECTED
951      000006      UNREC= 6      ;UNRECOVERABLE ERROR
952
953      ;+
954      ;
955      ;DEVICE REGISTER OFFSETS
956      ;
    
```

```

957          ;
958          ;
959          177776 TSBA== -2
960          177776 TSBAL== -2
961          177776 TSDB== -2          ;TSDB/TSBA REGISTER
962          177776 TSDBL== -2       ;TSDB/TSBA REGISTER
963          177777 TSBAH== -1
964          177777 TSDBH== -1       ;TSDB/TSBA REGISTER HIGH BYTE
965          000000 TSSR== 0         ;TSSR REGISTER
966          000001 TSSRH== 1        ;TSSR REGISTER HIGH BYTE
967
968          ;+
969          ; TSDB ADDRESS BIT DEFINITIONS
970          ;
971          000003 A1716 = BIT1+BIT0 ;ADDRESS BITS 17;16 ARE IN 1:0
972
973          ;+
974          ; COMMAND DEFINITIONS
975          ;
976          000017 P.GETSTAT      = 17 ;GET STATUS
977          000013 P.INIT         = 13 ;INITIALIZE
978          000012 P.CONTROL      = 12 ;CONTROL COMMANDS
979          000011 P.FORMAT       = 11 ;FORMAT
980          000010 P.POSITION     = 10 ;POSITION
981          000006 P.WRTSUB       = 6  ;SUBSYSTEM WRITE
982          000005 P.WRITE        = 5  ;WRITE
983          000004 P.WRTCHAR     = 4  ;WRITE CHARACTERISTICS
984          000001 P.READ         = 1  ;READ
985
986          ;+
987          ; COMMAND PACKET HEADER WORD BIT DEFINITIONS
988          ;
989          100000 P.ACK          = BIT15 ;BUFFER AVAIL FOR CONTROLLER
990          040000 P.CVC          = BIT14 ;CLEAR VOLUME CHECK
991          020000 P.OPP          = BIT13 ;REVERSE SEQUENCE OF DATA BITS
992          010000 P.SWB         = BIT12 ;SWAP BYTES IN MEMORY
993          007400 P.MODE        = BIT11!BIT10!BIT9!BIT8 ;EXTENDED COMMAND MODE FIELD
994          000200 P.IE          = BIT7  ;INTERRUPT ENABLE
995          000140 P.FMT= BIT6!BIT5 ;PACKET HEADER TYPE (ALWAYS=0)
996          000037 P.CMD         = 37  ;MAJOR COMMAND FIELD
997
998          ;+
999          ; CONTROL COMMAND MODE CODES
1000         ;
1001         000000 PC.RELEASE     = 0*256. ;RELEASE BUFFER
1002         000400 PC.REWIND     = 1*256. ;REWIND
1003         001000 PC.NOOP       = 2*256. ;NO-OP
1004         002000 PC.IEREW     = 4*256. ;REWIND IMMEDIATE INTERRUPT
1005         002400 PC.ERASE      = 5*256. ;SECURITY ERASE
1006
1007         ;+
1008         ; CONTROLLER RAM DEFINITIONS
1009         ;
1010         000167 RMCHBEG = 167 ;CHARACTERISTICS IO DATA BEGIN RAM ADDRESS
1011         000200 RMCHEND = 200 ;CHARACTERISTICS IO DATA END RAM ADDRESS
1012         000020 RMPKTBEG= 20  ;COMMAND PACKET BEGIN RAM ADDRESS
1013         000027 RMPKTEND= 27  ;COMMAND PACKET END RAM ADDRESS
1014         000104 RMMMSGBEG= 104 ;MESSAGE BUFFER BEGIN RAM ADDRESS
    
```

```

1014      000117      RMMSGEND= 117      ;MESSAGE BUFFER END RAM ADDRESS
1015      ;+
1016      ;
1017      ;REGISTER DEFINITIONS IN THE MESSAGE BUFFER
1018      ;
1019      ;-
1020
1021      000006      XST0== 6      ;EXTENDED STATUS REGISTER 0 (WORD 4)
1022      000010      XST1== 8.      ;EXTENDED STATUS REGISTER 1 (WORD 5)
1023      000012      XST2== 10.     ;EXTENDED STATUS REGISTER 2 (WORD 6)
1024      000014      XST3== 12.     ;EXTENDED STATUS REGISTER 3 (WORD 7)
1025      000016      XST4== 14.     ;EXTENDED STATUS REGISTER 4 (WORD 8)
1026
1027
1028      ;+
1029      ;
1030      ;OFFSETS TO WORD LOCATIONS IN PACKET DEFINITIONS
1031      ;
1032      ;-
1033
1034      000002      PKLOW = 2      ;LOW ORDER CHARACTERISTIC DATA POINTER
1035      000004      PKHI  = 4      ;HIGH ORDER CHARACTERISTIC DATA POINTER
1036      000006      PKBCNT = 6     ;NUMBER OF BYTES IN DATA PACKET
1037
1038      000010      EXBCNT=10      ;NUMBER OF BYTES IN EXTENDED DATA PACKET
1039
1040      ;+
1041      ;DATA PACKET OFFSETS FOR WRITE SUBSYSTEM COMMAND
1042      ;-
1043      000000      BSEL0 = 0      ;BYTE 0
1044      000001      BSEL1 = 1      ;BYTE 1
1045      000002      SEL2  = 2      ;WORD 2
1046      000004      SELDATA = 4    ;WORD 3
1047
1048      ;+
1049      ;BSEL0 SELECT CODES FOR WRITE SUBSYSTEM COMMAND
1050      ;-
1051      000000      PW.NOP      = 0      ;NO-OP
1052      000001      PW.RDRAM    = 1      ;READ RAM
1053      000002      PW.WTRAM    = 2      ;WRITE RAM
1054      000003      PW.RFIFO    = 3      ;READ FIFO
1055      000004      PW.WFIFO    = 4      ;WRITE FIFO
1056      000005      PW.RDSTAT   = 5      ;READ STATUS
1057      000006      PW.WCTL     = 6      ;WRITE TAPE CONTROL
1058      000007      PW.WFMT     = 7      ;WRITE TAPE FORMAT
1059      000010      PW.WMISC    = 10     ;WRITE MISCELLANEOUS
1060      000011      PW.WNPR     = 11     ;WRITE NPR CONTROL
1061      000020      PW.D22      = 20     ;DO MICROTEST 22
1062      000021      PW.D11      = 21     ;DO MICROTEST 11
1063      000022      PW.D13      = 22     ;DO MICROTEST 13
1064      000023      PW.NO1311   = 23     ;DISABLE MICROTEST 11 AND 13
1065      000024      PW.RDEXT     = 24     ;READ EXT. TAPE STATUS (NOT SUPPORTED BY ALL TRANSP
RTS
1066
1067      ;+
1068      ;BSEL1 CODES FOR WRITE TAPE CONTROL
1069      ;-
1070      000200      WC.IFAD      = BIT7   ;IFAD - FORMATTER ADDRESS
    
```

```

1071      000100      WC.IOTAD      = BIT6      ;ITADO - TRANSPORT ADDRESS BIT 0
1072      000040      WC.I1TAD      = BIT5      ;ITAD1 - TRANSPORT ADDRESS BIT 1
1073      000020      WC.ISRESV     = BIT4      ;IRESV5 - RESERVED #5
1074      000010      WC.IREW      = BIT3      ;IREW - REWIND
1075      000004      WC.IRWU      = BIT2      ;IRWU - REWIND AND UNLOAD
1076      000002      WC.IFEN      = BIT1      ;IFEN - FORMATTER ENABLE
1077      000001      WC.IGO       = BIT0      ;GO
1078
1079      ;+
1080      ;BSEL1 CODES FOR WRITE FORMAT
1081      ;-
1082      000200      WF.IHISP      = BIT7      ;IHISP - HIGH SPEED
1083      000100      WF.IWRT      = BIT6      ;IWRT - WRITE
1084      000040      WF.IREV      = BIT5      ;IREV - REVERSE
1085      000020      WF.IWFM      = BIT4      ;IWFM - WRITE FILE MARK
1086      000010      WF.IEDIT     = BIT3      ;IEDIT - EDIT
1087      000004      WF.IERASE    = BIT2      ;IERASE - ERASE
1088      000002      WF.I3RESV    = BIT1      ;IRESV3 - RESERVED #3
1089      000001      WF.I4RESV    = BIT0      ;IRESV4 - RESERVED #4
1090
1091      ;+
1092      ;BSEL1 CODES FOR WRITE MISCELLANEOUS SUBCOMMAND
1093      ;-
1094      000200      MS.EXT       = BIT7      ;INVERT SENSE OF EXTENDED FEATURES SWITCH
1095      000020      MS.RSFIFO     = BIT4      ;RESET FIFO AND INPUT PARITY ERRORR
1096      000010      MS.RSTAPE    = BIT3      ;RESET TAPE STATUS IN 2 FLIP-FLOPS
1097      000006      MS.ATTN      = BIT2!BIT1 ;ATTENTION TRIGGER FIELD
1098      000001      MS.RSD       = BIT0      ;RESET TIMER A,B THEN DELAY TIMES IN SEL2
1099
1100      ;+
1101      ; MS.ATTN SUBCODES
1102      ;-
1103      000000      MSA.NOP      = 0*2      ;NO-OP (NOTHING TRIGGERED)
1104      000002      MSA.VOL      = 1*2      ;SIMULATE ON-LINE/OFF-LINE TRANSISTION
1105      000004      MSA.NRAM     = 2*2      ;FORCE NON-FATAL RAM ERROR (FORCES ERRCODE 54)
1106      000006      MSA.FRAME    = 3*2      ;FORCE FATAL RAM ERROR (CAUSES SCE TO SET)
1107
1108      ;+
1109      ; WRITE SUBSYSTEM WRITE NPR BSEL1 BIT DEFINITIONS
1110      ;-
1110      000200      NP.IR        = BIT7      ;INTERRUPT REQUEST (0-1 TRANSITION)
1111      000100      NP.OUT       = BIT6      ;TAPE DATA DIRECTION OUT (0= IN)
1112      000040      NP.LOOP     = BIT5      ;ENABLE TRANSPORT LOOPBACK
1113      000020      NP.WRP      = BIT4      ;WRITE CORRECT PARITY (SET=0 TO WRITE WRONG)
1114
1115      ;+
1116      ; READ STATUS MESSAGE BUFFER BIT DEFINITIONS
1117      ;-
1118      000200      S2.DIM       = BIT7      ;WORD #9 BYTE 2 DATA MISs
1119      000100      S2.ILW      = BIT6      ;
1120      000040      S2.OURDY     = BIT5      ; ILW H
1121      000020      S2.INRDY    = BIT4      ; OUT RDY H
1122      000010      S2.ATIMR    = BIT3      ; IN RDY H
1123      000004      S2.BTIMR    = BIT2      ; TIMER A FLAG H
1124      000003      S2.UNDEF    = BIT1+BIT0 ; TIMER B FLAG H
1125      100000      S1.PARIN    = BIT15     ;(UNDEFINED)
1126      040000      S1.I2RESV   = BIT14     ;WORD #8 BYTE 1 PARIN H
1127      020000      S1.I1RESV   = BIT13     ; IRESV2
; IRESV1
    
```

```

1128      010000      S1.IEOT          = BIT12          ; IEOT L
1129      004000      S1.IIDENT       = BIT11          ; IIDENT H
1130      002000      S1.ICER         = BIT10          ; ICER H
1131      001000      S1.IFMK        = BIT9           ; IFMK H
1132      000400      S1.IHER         = BIT8           ; IHER H
1133      000200      SO.ISPEED      = BIT7           ;WORD #8 BYTE 0 ISPEED H
1134      000100      SO.IRDY        = BIT6           ; IRDY L
1135      000040      SO.IONL        = BIT5           ; IONL L
1136      000020      SO.ILDP        = BIT4           ; ILDP L
1137      000010      SO.IDBY        = BIT3           ; IDBY L
1138      000004      SO.IRWD        = BIT2           ; IRWD L
1139      000002      SO.IFBY        = BIT1           ; IFBY L
1140      000001      SO.IFPT        = BIT0           ; IFPT L
1141      ;
1142      ; SPECIAL KEYBOARD STUFF FOR MOVER PROGRAM
1143      177560      TKS             =177560          ;KEYBOARD STATUS REGISTER
1144      177562      TKB             =177562          ;KEYBOARD DATA REGISTER
1145      177564      TPS             =177564          ;CONSOLE PRINTER STATUS REGISTER
1146      177566      TPB             =177566          ;CONSOLE PRINTER DATA REGISTER
1147      007776      HIMEM          =007776          ;HIGH MEMORY MASK VALUE
1148      ;
1149      ;
1150      ;
1151      174400      CSR             =174400          ;STATUS AND CONTROL REGISTER
1152      174402      BAR             =174402          ;DL ADDRESS REGISTER
1153      174404      DAR             =174404          ;PLATTER ADDRESS
1154      174406      MPR             =174406          ;MULTIPURPOSE REGISTER
1155      ;
1156      ;
1157      ;
1158      ;
1159      ;
1160      ; CONTROLLER COMMANDS
1161      ;
1162      ;
1163      000004      DLGETS         =4              ;GET STATUS COMMAND
1164      000006      SEEK           =6              ;SEEK TRACK AND HEAD SELECT
1165      000010      DLRDHD         =10             ;READ SECTOR HEADER
1166      000014      READ           =14             ;READ COMMAND
1167      000016      DLRDNH         =16             ;READ SECTOR NO HEADER CHECK
1168      ;
1169      ;
1170      ;
1171      ;
1172      ;
1173      ;
1174      000001      READY          =1              ;DRIVE READY BIT IN STATUS REG.
1175      000013      DLSR           =13             ;STATUS AND RESET
1176      177730      DLERR          =177730          ;MASK FOR COVER OPEN
1177      000006      DLUN           =6              ;HEADS UNLOADED
1178      000177      DLCYL          =000177          ;MASK FOR CYLINDER ADDRESS
1179      100200      DLDNER         =100200          ;DONE SET OR ERROR SET BITS
1180      ;
1181      ;
1182      ;
1183      ;
1184      177560      TTICSR          = 177560          ;KEYBOARD INPUT STATUS
    
```

1185	177562	TTIBFR =	177562	;KEYBOARD DATA REGISTER
1186	177564	TTOCSR =	177564	;CONSOLE PRINTER STATUS REGISTER
1187	177566	TTOBFR =	177566	;CONSOLE PRINTER DATA REGISTER
1188				

```
1190          .SBTTL SPECIAL MACROS AND OPDEFS.
1191
1192
1193          ;+
1194          ;SAVE GENERAL REGS 1 TO 5
1195          ;-
1196
1197          .MACRO SAVREG
1198          JSR    R5,REGSAV
1199          .ENDM
1200
1201          ;+
1202          ; MACRO TO FORCE AN ERROR
1203          ;-
1204          .MACRO FORCERROR TAG,NOTSSR
1205          .NLIST
1206          .IIF NDF LISTALL, .NLIST
1207          .LIST
1208          .IF B NOTSSR
1209             MOV    TSSR(R5),R1          ;READ TSSR
1210          .ENDC
1211             MOV    FORCER,FORCER        ;IS FORCER SET? (LEAVE C BIT ALONE)
1212             BNE    TAG                  ;BR IF YES
1213          .NLIST
1214          .IIF NDF LISTALL, .LIST
1215          .LIST
1216          .ENDM
1217
1218          ;+
1219          ; MACRO TO FORCE AN EXIT TO AVOID SECTION ITERATIONS
1220          ; WILL EXIT TO A LABEL IF FORCER IS NEGATIVE
1221          ; SO TO FORCE ERRORS AND EXIT ON 1 ERROR SET
1222          ; FORCER TO 177777
1223          ; TO FORCE ERRORS AND ITERATIONS SET FORCER TO 1.
1224          ;-
1225          .MACRO FORCEEXIT TAG
1226          .NLIST
1227          .IIF NDF LISTALL, .NLIST
1228          .LIST
1229             MOV    FORCER,FORCER        ;IS FORCER NEGATIVE?
1230             BMI    TAG                  ;BR IF YES
1231          .NLIST
1232          .IIF NDF LISTALL, .LIST
1233          .LIST
1234          .ENDM
1235          ;+
1236          ; MACRO TO INCREMENT ERROR COUNTS
1237          ;-
1238          .MACRO NEXT.ERRNO
1239          .NLIST
1240          ;;;.IIF NDF LISTALL, .NLIST
1241             ERRNO=ERRNO+1
1242          ;;;.IIF NDF LISTALL, .LIST
1243          .LIST
1244          .ENDM
1245          ;+
1246
```

```
1247           ;MACRO TO PERFORM XOR
1248           ;-
1249
1250           .MACRO XOR      A,B
1251           MOV      A,-(SP)
1252           BIC      B,(SP)
1253           BIC      A,B
1254           BIS      (SP)+,B
1255           .ENDM
1256
1257           000000           EN=0           ; INITIALIZE ERROR NUMBER
1258           .SBTTL FORCER - FORCE ERROR FLAG
1259
1260           ;
1261           ; THE FOLLOWING LOCATIONS MAY BE PATCHED BY THE USER
1262           ; TO OBTAIN THE RESULTS DESCRIBED FOR EACH.
1263           ;
1264
1265           002144 000000 FORCER::      0           ; FORCE TYPE ALL HARD ERRORS (THE ONES CALLED -
1266           ; - BY THE MACRO "IFERROR"). AN ERROR NEED NOT -
1267           ; - EXIST, JUST ASSUME AND TYPE THE MESSAGE.
1268
1269
1270
```

```

1272                                     .SBTTL GLOBAL DATA SECTION
1273
1274                                     ;**
1275                                     ;THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
1276                                     ;IN MORE THAN ONE TEST.
1277                                     ;--
1278
1279                                     ;
1280                                     ;THE FOLLOWING DATA ARE SET FOR EACH UNIT AT INIT TIME.
1281                                     ;SINGLE UNIT DEFAULTS (LISTED) ARE IN THE DEFAULT P-TABLE.
1282                                     ;
1283 002146 000000 EPRTSW::          .WORD 0           ;PRINT SWITCH
1284 002150 000000 UNITN::          .WORD 0           ;UNIT # UNDER TEST.
1285 002152 000000 QVP::            .WORD 0           ;QUICK VERIFY FLAG.
1286 002154 000000 CSRADDR::       .WORD 0           ;ADDRESS OF CSR FOR CURRENT DEVICE
1287 002156 000224 IVEC::            .WORD 224          ;INTERRUPT VECTOR
1288 002160 000200 IPRI::            .WORD PRI04        ;INTERRUPT PRIORITY.
1289 002162 000000 TSTCNT::         .WORD 0           ;NUMBER OF TESTS RUN IN THIS PASS
1290 002164 000000 LOOPCNT::        .WORD 0           ;REMAINING ITERATION COUNT FOR TEST
1291 002166 000000 DEVCNT::         .WORD 0           ;NUMBER OF DEVICE UNDER TEST
1292 002170 000000 FATFLG::         .WORD 0           ;SET IF FATAL ERROR IS DETECTED IN TEST
1293 002172 000000 INTRECV::        .WORD 0           ;SET IF TAPE INTERRUPT WAS RECEIVED
1294 002174 000000 BENBSW::         .WORD 0           ;BUFFER ENABLE SWITCH SW 0=OFF;1=ON
1295 002176 000000 EXPD::            .WORD 0           ;EXPECTED RAM DATA FOR PRAMPKT ROUTINE
1296 002200 000000 RECV::            .WORD 0           ;RECEIVED RAM DATA FOR PRAMPKT ROUTINE
1297 002202 000000 ERRHI::          .WORD 0           ;HIGH ADDRESS MEMORY ERROR
1298 002204 000000 ERRLO::          .WORD 0           ;LOW ADDRESS MEMORY ERROR
1299 002206 000000 RAMDATA::        .BLKW 16.         ;DATA READ FROM RAM PACKET OR MESSAGE BUF AREA
1300 002246 000000 RAMSIZ::         .WORD 0           ;RAM DATA SIZE FOR PRAMPKT ROUTINE
1301 002250 000000 RCVHIADD::       .WORD 0           ;RECEIVED BUFFER HIGH ADDRESS
1302 002252 000000 RCVLOADD::      .WORD 0           ;RECEIVED BUFFER LOW ADDRESS
1303 002254 000000 COUNT::          .WORD 0           ;TEST COUNT PATTERN
1304 002256 000000 DATA::          .WORD 0           ;TEST DATA
1305 002260 000000 TSTFLAG::        .WORD 0           ;TEST FLAG WORD
1306 002262 000000 TSTPTR::         .WORD 0           ;TSTBLK POINTER
1307 002264 000000 PRMNO::          .WORD 0           ;PRINT ROUTINE TEMP
1308 002266 000000 EXPMSG::         .BLKB 100.        ;EXPECTED MESSAGE BUFFER DATA
1309 002432 000000 RECMSG::         .BLKB 100.        ;RECEIVED MESSAGE BUFFER DATA
1310 002576 000000 TMPBFR::         .BLKB 80.         ;TEMPORARY STORAGE FOR PRINT
1311 002716 000000 MESBFA::         .WORD 0           ;STORES ADDRESS OF MESSAGE BUFFER FOR ERR PRT

```

1313
 1314
 1315
 1316
 1317
 1318
 1319
 1320
 1321
 1322
 1323
 1324
 1325
 1326
 1327
 1328
 1329 002720
 1330 002720 000000
 1331 002722 177777
 1332 002724 000001
 1333 002726 000002
 1334 002730 000004
 1335 002732 000010
 1336 002734 000020
 1337 002736 000040
 1338 002740 000100
 1339 002742 000200
 1340 002744 000400
 1341 002746 001000
 1342 002750 002000
 1343 002752 004000
 1344 002754 010000
 1345 002756 020000
 1346 002760 040000
 1347 002762 100000
 1348 002764 177776
 1349 002766 177775
 1350 002770 177773
 1351 002772 177767
 1352 002774 177757
 1353 002776 177737
 1354 003000 177677
 1355 003002 177577
 1356 003004 177377
 1357 003006 176777
 1358 003010 175777
 1359 003012 173777
 1360 003014 167777
 1361 003016 157777
 1362 003020 137777
 1363 003022 077777
 1364 003024 125252
 1365 003026 052525
 1366 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==.

```

1368                .SBTTL GLOBAL ENVIRONMENT STORAGE
1369
1370                ; STORAGE FOR DEVICE REGISTERS
1371                ;
1372 003030 000000 100000 000000 DUMMY: 0,100000,0,0           ;DUMMY DEVICE REGISTERS...
1373 003040 000000 000000 000000      0,0,0,0,0,0,0,0,0       ;...FOR MULTI-UNIT CHECKOUT.
1374
1375
1376
1377 003060 000000          DUFLG::      .WORD 0           ;"DROPPED UNIT" FLAG.
1378                                ;INHIBITS CODE IN "CLEAN-UP".
1379 003062 000000          NODEV::      .WORD 0           ;FLAG TO SAY NO DEVICE.
1380
1381 003064 000000          TEMP1::      .WORD 0           ;SOME TEMP LOCATIONS.
1382 003066 000000          TEMP2::      .WORD 0
1383 003070 000000          XXCOMM::     .WORD 0           ;XXDP+ COMM BLOCK POINTER.
1384 003072 000000          FREE::       .WORD 0           ;1ST FREE MEMORY ADDRESS...
1385 003074 000000          FRESIZ::    .WORD 0           ;...AND SIZE (IN WORDS).
1386 003076 000000          FREEHI:     .WORD 0           ;LAST WORD IN FREE SPACE
1387 003100 000000          KTFLG::      .WORD 0           ;KT11, MEM AVAIL FLAG -
1388                                ;- .WORD 0 = <24K OR NO KT -
1389                                ;- NZ = >24K AND KT.
1390 003102 000000          KTENABLE::    .WORD 0           ;SET BY TEST ROUTINES TO FLAG >28K UNDER TEST
1391 003104 002000          PST32W::     .WORD 2000        ;32W BLOCK ADDRESS FOR 32K START
1392 003106 000000          SIFLAG::     .WORD 0
1393 003110 000000          BADDAT::     .WORD 0           ;
1394 003112 000000          GDDAT::      .WORD 0           ;ACTUAL DATA
1395 003114 000000          LOOPFL::     .WORD 0           ;EXPECTED DATA
1396 003116
1397 003116 000000          CTAB::        .WORD 0           ;CONFIGURATION TABLES.
1398 003120 000000          CTABM::      .WORD 0           ;CONFIG WORK.
1399 003122 000000
1400 003124 000000
1401 003126 177777
1402 003130          CTABE::            .WORD -1          ;END OF MEM TABLE.
1403          CTABE::
1404          ;ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:
1405          ;
1406          ; 0 = UNIT NOT TESTED
1407          ; 100000 = UNIT ONLINE, NO ERRORS
1408          ; 10XXXX = UNIT ONLINE, ENCOUNTERED XXXX ERRORS
1409          ; 160000 = UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
1410          ; 160001 = UNIT DROPPED, NOT IDLE AT START
1411          ; 14XXXX = UNIT DROPPED, ENCOUNTERED XXXX ERRORS
1412 003130          ERTABL:              .BLKW 64.
1413 003330 000000          ERTABE:       .WORD 0
1414
1415 003332 000000          SKIPT:         .WORD 0           ;1=SKIP SUBTEST 0=NO SKIP OF SUBTEST

```

```

1417          .SBTTL GLOBAL TEXT MESSAGES
1418          ;++
1419          ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
1420          ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
1421          ; MORE THAN ONE TEST.
1422          ;--
1423
1424
1425
1426          ;+
1427          ;NAMES OF DEVICES SUPPORTED
1428          ;-
1429
1430          DEVTYP <TK-25>
003334 L$DVTYP::
003334          .ASCIZ /TK-25/
003334          .EVEN
124      113      055
1431
1432
1433          ;+
1434          ;TEST DESCRIPTION
1435          ;-
1436          DESCRIPT <CZTKEB TK-25 FRT END FUNC #1>
003342 L$DESC::
003342          .ASCIZ /CZTKEB TK-25 FRT END FUNC #1/
003342          .EVEN
103      132      124
1437
1438
1439          ;+
1440          ;BIT TO ASCII CONVERSION FOR TSSR REGISTER
1441          ;-
1442          TSSRBIT::          .WORD 1$,2$,3$,4$,5$,6$,7$,8$
1443          .WORD 9$,10$,11$,12$,13$,14$,15$,16$
003400 003440 003443 003447
003420 003501 003505 003511
1444          123      103      000 1$: .ASCIZ 'SC'
1445          102      111      105 2$: .ASCIZ 'BIE'
1446          123      103      105 3$: .ASCIZ 'SCE'
1447          122      115      122 4$: .ASCIZ 'RMR'
1448          116      130      115 5$: .ASCIZ 'NXM'
1449          116      102      101 6$: .ASCIZ 'NBA'
1450          102      111      124 7$: .ASCIZ 'BIT9'
1451          102      111      124 8$: .ASCIZ 'BIT8'
1452          123      123      122 9$: .ASCIZ 'SSR'
1453          117      106      114 10$: .ASCIZ 'OFL'
1454          102      111      124 11$: .ASCIZ 'BIT5'
1455          102      111      124 12$: .ASCIZ 'BIT4'
1456          102      111      124 13$: .ASCIZ 'BIT3'
1457          102      111      124 14$: .ASCIZ 'BIT2'
1458          102      111      124 15$: .ASCIZ 'BIT1'
1459          102      111      124 16$: .ASCIZ 'BIT0'
1460          .EVEN
1461          SFIERR: .ASCIZ 'TSSR ERROR AFTER SOFT INIT'
1462          SFHERR: .ASCIZ 'TSSR ERROR AFTER BUS RESET'
1463          NXR: .ASCIZ / NON-EXISTANT DEVICE REGISTER/
1464          NXR: .ASCIZ /%A ADDRESS: %06/
1465          TSSX: .ASCII /%A TSBA,TSSR EXP'D: %06%A,%06%N/
1466          .ASCIZ /%A TSBA,TSSR REC'D: %06%A,%06/
1467          FUSI: .ASCII /%N%A/
045      045      116      045
    
```

1468	004021	040	040	125	USI:	.ASCIZ	/ UNEXPECTED INTERRUPT/
1469	004050	040	040	111	NSI:	.ASCIZ	/ INTERRUPT EXPECTED, NOT RECEIVED/
1470	004113	045	116	045	FNOINTR:	.ASCII	/N/A/
1471	004117	040	040	116	NOINTR:	.ASCIZ	/ NO INTERRUPT WAS GENERATED/
1472	004154	040	040	111	IFault:	.ASCIZ	/ INTERRUPT FAULT/
1473	004176	045	101	040	INTX:	.ASCIZ	/A CPU PC: %06%A TSBA: %06/
1474	004233	040	040	042	NOINIT:	.ASCIZ	/ "BUS-INIT" DIDN'T INITIALIZE CONTROLLER/
1475	004305	040	040	042	NSINIT:	.ASCIZ	/ "SOFT-INIT" DIDN'T INITIALIZE THE DPU/
1476	004355	040	040	042	BRINIT:	.ASCIZ	/ "BUS-RESET" DIDN'T INITIALIZE THE DPU/
1477							
1478	004425	000			NUL:	.ASCIZ	//
1479	004426	045	116	000	NULCR:	.ASCIZ	/N/
1480	004431	045	101	040	EXPGOT:	.ASCIZ	/A EXP'D: %06%A, REC'D: %06/
1481	004465	045	116	045	EXPGT2:	.ASCIZ	/N%A EXP'D: %06%A, %06N%A REC'D: %0%A, %06/
1482	004541	045	101	040	DUAD12:	.ASCIZ	/A REG(W) WRITTEN TO: %06%A REG(R) READ; EXP'D: %06%A, REC'D: %06/
1483	004643	122	101	115	PKTRAM::	.ASCIZ	'RAM Contents Do Not Match Packet Sent'
1484	004711	040	040	103	SCME:	.ASCIZ	/ CONFIG DOESN'T MATCH MFG. MASTER/
1485	004754	127	122	111	WRMSG:	.ASCIZ	'WRITE CHARACTERISTICS Failed'
1486	005011	124	123	123	WRERR:	.ASCIZ	'TSSR Incorrect After WRITE Command, More Bits Set Than SSR'
1487	005104	124	123	123	RDERR:	.ASCIZ	'TSSR Incorrect After READ Command, More Bits Set Than SSR'
1488						.EVEN	
1489							
1490							
1491							

```

1493
1494
1495
1496
1497
1498
1499
1500
1501 005176
005176
1502 005176
005176 013746 003062
005202 012746 003675
005206 012746 000002
005212 010600
005214 104415
005216 062706 000006
1503 005222 004737 005230
1504 005226
005226
005226 104423
1505
1506
1507
1508
1509
1510
1511 005230 005727
1512 005232 000000
1513 005234 001402
1514 005236 004777 177770
1515 005242
005242 012746 004426
005246 012746 000001
005252 010600
005254 104415
005256 062706 000004
1516 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.
ENDMSG

L10002:
TRAP C$MSG

;
; THIS ROUTINE APPENDS A UNIQUE EXTENSION (IF REQUIRED)
; TO ANY OF THE ABOVE ERROR SIGNATURES.
;
EXTEND: TST (PC)+
EXTA: 0 ; 0 = NO EXTENSION.
BEQ 1$
JSR PC,@EXTA ; APPEND EXTENSION TEXT.
1$: PRINTX #NULCR ; PRINT A BLANK LINE
MOV #NULCR,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C$PNTX
ADD #4,SP
RTS PC

```

```

1519          .SBTTL  PRITSSR - PRINT TSSR CONTENTS
1520
1521          ;+
1522          ;
1523          ;ROUTINE TO DISPLAY THE CONTENTS, AND BIT DEFINITIONS, OF
1524          ;THE TSSR REGISTER. THIS ROUTINE IS NORMALLY CALLED ONLY
1525          ;BY A MESSAGE PRINTING ROUTINE
1526          ;
1527          ;INPUTS:
1528          ;
1529          ;      R1      CONTENTS OF TSSR
1530          ;
1531          ;SUBORDINATE ROUTINES:
1532          ;
1533          ;      CHKAMB  CHECK FOR AMBIGUOUS CONTENTS
1534          ;
1535          ;-
1536
1537 005264      PRITSSR:
1538 005264          SAVREG          ;SAVE GENERAL REGISTERS
1539 005270 010104      MOV          R1,R4          ;SAVE THE TSSR CONTENTS
1540 005272          PRINTB #TSSRFOR,R4          ;PRINT THE CONTENTS OF TSSR
1541          005272 010446      MOV          R4,-(SP)
1542          005274 012746 005736      MOV          #TSSRFOR,-(SP)
1543          0C5300 012746 000002      MOV          #2,-(SP)
1544          005304 010600          MOV          SP,R0
1545          005306 104414          TRAP         C$PNTB
1546          005310 062706 000006      ADD          #6,SP
1547          005314 010400          MOV          R4,R0          ;GET TSSR BACK FOR CHKAMB
1548          005316 004737 016540      JSR          PC,CHKAMB          ;ARE CONTENTS AMBIGUOUS ?
1549          005322 103410          BCS          5$          ;BRANCH IF NOT
1550          005324          PRINTX #AMBTSSR          ;SHOW CONTENTS ARE AMBIGUOUS
1551          005324 012746 006156      MOV          #AMBTSSR,-(SP)
1552          005330 012746 000001      MOV          #1,-(SP)
1553          005334 010600          MOV          SP,R0
1554          005336 104415          TRAP         C$PNTX
1555          005340 062706 000004      ADD          #4,SP
1556          005344 010403          5$: MOV          R4,R3          ;CONTENTS OF TSSR
1557          005346 042703 001476      BIC          #HIADDR!FATERR!TERCLS,R3          ;CLEAR ALL MULTIPLE BIT FIELDS
1558          005352 001434          BEQ          20$          ;NO BITS ARE SET
1559          005354 012702 002576      MOV          #TMPBFR,R2          ;TEMPORARY ASCII BUFFER
1560          005360 012701 003400      MOV          #TSSRBIT,R1          ;ASCII EQUIVALENT OF BITS
1561          005364 005703          10$: TST          R3          ;REMAINING BITS TO CONVERT
1562          005366 001413          BEQ          15$          ;BRANCH WHEN ALL ARE DONE
1563          005370 000241          CLC          ;CLEAR CARRY FOR SHIFT
1564          005372 006103          ROL          R3          ;SHIFT NEXT BIT TO CARRY
1565          005374 103006          BCC          13$          ;BRANCH IF BIT NOT SET
1566          005376 011100          MOV          (R1),R0          ;POINTER TO BIT DEFINITION
1567          005400 112022          11$: MOVB         (R0)+,(R2)+          ;MOVE ASCIZ TO BUFFER
1568          005402 001376          BNE          11$          ;MOVE ALL BITS
1569          005404 112762 000054 177777      MOVB         #' ,,-1(R2)          ;INSERT A COMMA TO TERMINATE
1570          005412 005721          13$: TST          (R1)+          ;POINT TO NEXT DESCRIPTION
1571          005414 000763          BR          10$          ;GET THE REMAINING BITS
1572          005416 105042          15$: CLRB         -(R2)          ;TERMINATE THE LINE
1573          005420          PRINTX #TSSDEF,#TMPBFR          ;PRINT THE BIT DEFINITIONS
1574          005420 012746 002576      MOV          #TMPBFR,-(SP)
1575          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			
1563								
1564	005444	010403		20\$:	MOV	R4,R3	;GET THE TSSR CONTENTS	
1565	005446	042703	177761		BIC	#+CTERCLS,R3	;CLEAR ALL BUT TERMINATION	
1566	005452	016303	006220		MOV	TCOCOD(R3),R3	;GET THE TERMINATION CODE MEANING	
1567	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		
1568	005500	010403			MOV	R4,R3	;TSSR CONTENTS AGAIN	
1569	005502	042703	177717		BIC	#+CFATERR,R3	;CLEAR ALL BUT FATAL TERMINATION	
1570	005506	001421			BEQ	25\$;DON'T PRINT IF ZERO	
1571	005510	006203			ASR	R3		
1572	005512	006203			ASR	R3		
1573	005514	006203			ASR	R3		
1574	005516	016303	006560		MOV	TSFCOD(R3),R3	;ALINE TERMINATION CODE FOR INDEX	
1575	005522				PRINTX	#TFCASC,R3	;GET THE FATAL TERMINATION CODE	
	005522	010346			MOV	R3,-(SP)	;PRINT THE FATAL TERMINATION CODE	
	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		
1576	005544	012737	000031	002170	MOV	#25.,FATFLG	;DROP UNIT AFTER THIS ERROR	
1577	005552	010403			25\$:	MOV	R4,R3	;GET TSSR CONTENTS
1578	005554	042703	176377		BIC	#+CHIADDR,R3	;CLEAR ALL BUT EXTENDED ADDRESS	
1579	005560	001411			BEQ	30\$;DON'T PRINT IF ZERO	
1580	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		
1581	005604	022704	100210		30\$:	CMP	#100210,R4	;CHECK FOR MEDIA ERROR
1582	005610	001003			BNE	31\$;BR, IF PROBABLY NOT TAPE ERROR	
1583	005612	012737	005672	002146	MOV	#EPRT3,EPRTSW	; "PROBABLY MEDIA RELETED ERROR - BAD TAPE"	
1584	005620	005737	002146		31\$:	TST	EPRTSW	;CHECK FOR THE SWITCH EMPTY
1585	005624	001003			BNE	310\$;BR, IF SWITCH IS NOT EMPTY	
1586	005626	012737	005672	002146	MOV	#EPRT1,EPRTSW	;SET SWITCH TO DEFAULT	
1587	005634	013737	002146	005644	310\$:	MOV	EPRTSW,32\$+2	;PUT REAL SWITCHABLE MESSAGE IN PLACE
1588	005642				32\$:	PRINTB	#EPRT1	;PRINT THE ERROR MESSAGE
	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		
1589	005662	012737	005672	002146	MOV	#EPRT1,EPRTSW	;RESET TO NORMAL ERROR POINTER	
1590	005670	000207			RTS	PC	;RETURN TO CALLER	
1591								
1592	005672				EPRT2:			

1593	005672				EPRT3:		
1594	005672	045	116	045	EPRT1:	.ASCIZ	'%N%A *****REPLACE CONTROLLER*****%S'
1595	005736	045	116	045	TSSRFOR:	.ASCIZ	'%N%A TSSR = %06'
1596	005756	045	116	045	TEXASC:	.ASCIZ	'%N%A Extended Address Bits = %06'
1597	006017	045	116	045	TCOASC:	.ASCIZ	'%N%A Termination Class Code = %T'
1598	006060	045	116	045	TFCASC:	.ASCIZ	'%N%A Fatal Termination Class Code = %T'
1599	006127	045	116	045	TSSDEF:	.ASCIZ	'%N%A TSSR Bits Set: %T'
1600	006156	045	116	045	AMBTSSR:	.ASCIZ	'%N%A TSSR Contents Are Ambiguous'
1601						.EVEN	
1602	006220	006240	006263	006311	TCOCOD:	.WORD	1\$,2\$,3\$,4\$,5\$,6\$,7\$,8\$
1603	006240	116	157	162	1\$:	.ASCIZ	'Normal Termination'
1604	006263	124	145	162	2\$:	.ASCIZ	'Termination Condition'
1605	006311	124	141	160	3\$:	.ASCIZ	'Tape Status Alert'
1606	006333	106	165	156	4\$:	.ASCIZ	'Function Reject'
1607	006353	122	145	143	5\$:	.ASCIZ	'Recoverable Error - Tape Position One Record Down'
1608	006435	122	145	143	6\$:	.ASCIZ	'Recoverable Error - Tape Was Not Moved'
1609	006504	125	156	162	7\$:	.ASCIZ	'Unrecoverable Error'
1610	006530	106	141	164	8\$:	.ASCIZ	'Fatal Controller Error'
1611						.EVEN	
1612							
1613	006560	006570	006624	006635	TSFCOD:	.WORD	1\$,2\$,3\$,4\$
1614	006570	111	156	164	1\$:	.ASCIZ	'Internal Diagnostic Failure'
1615	006624	122	145	163	2\$:	.ASCIZ	'Reserved'
1616	006635	102	165	163	3\$:	.ASCIZ	'Bus Interface or Sanity Check Error'
1617	006701	122	145	163	4\$:	.ASCIZ	'Reserved'
1618						.EVEN	

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

1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634

```

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

1635 006712
1636 006712
1637 006716 010005
1638 006720 005737 003102
1639 006724 001001
1640 006726 005003
1641 006730 010301
1642 006732 010400
1643 006734 006100
1644 006736 006101
1645 006740
006740 010446
006742 010146
006744 012746 007116
006750 012746 000003
006754 010600
006756 104414
006760 062706 000010
1646 006764 010300
1647 006766 001404
1648 006770 010401
1649 006772 004737 020112
1650 006776 010004
1651 007000 005001
1652 007002 012402
1653 007004
007004 010246
007006 010146
007010 012746 007060
007014 012746 000003
007020 010600
007022 104414
007024 062706 000010
1654 007030 005201
1655 007032 020105
1656 007034 002762
1657 007036
007036 012746 007153
007042 012746 000001
007046 010600
007050 104414
007052 062706 000004

```

PRIPKT::
  SAVREG          ;SAVE THE REGISTERS
  MOV R0,R5       ;SAVE NO. OF WORDS IN PACKET
  TST KTENABLE    ;ABOVE 28K UNDER TEST?
  BNE 10$         ;BR IF YES
  CLR R3          ;SET HIGH ORDER ADDRESS TO 0
10$: MOV R3,R1     ;COPY HIGH ORDER ADDRESS
  MOV R4,R0       ;GET LOWER ADDRESS
  ROL R0          ;SHIFT BIT 15 INTO C BIT
  ROL R1          ;AND INTO HIGH ORDER.
  PRINTB #PKTADD,R1,R4 ;PRINT PACKET ADDRESS
  MOV R4,-(SP)
  MOV R1,-(SP)
  MOV #PKTADD,-(SP)
  MOV #3,-(SP)
  MOV SP,R0
  TRAP C#PNTB
  ADD #10,SP
15$: MOV R3,R0     ;GET HIGH ORDER ADDRESS
  BEQ 20$        ;BR IF NOT ABOVE 28K.
  MOV R4,R1       ;GET LOW ORDER ADDRESS
  JSR PC,SETMAP  ;SETUP PAR6 MAPPING FOR 18 BIT ADDRESS
  MOV R0,R4       ;GET RETURNED PAR6 ADDRESS BIAS
20$: CLR R1       ;SAVE WORD NUMBER
25$: MOV (R4)+,R2 ;GET PACKET CONTENTS
  PRINTB #PKTFRM,R1,R2 ;PRINT THE DATA
  MOV R2,-(SP)
  MOV R1,-(SP)
  MOV #PKTFRM,-(SP)
  MOV #3,-(SP)
  MOV SP,R0
  TRAP C#PNTB
  ADD #10,SP
  INC R1          ;NEXT WORD NUMBER
  CMP R1,R5       ;DONE ALL PACKET WORDS?
  BLT 25$        ;LOOP TILL ALL DONE
  PRINTB #PKTNEW  ;JUST A COUPLE NEW LINES
  MOV #PKTNEW,-(SP)
  MOV #1,-(SP)
  MOV SP,R0
  TRAP C#PNTB
  ADD #4,SP
  
```

			RTS	PC		;RETURN
1658	007056	000207				
1659						
1660	007060	045	116	045	PKTFRM: .ASCIZ	'%N%A Packet Word #D1%A = %06'
1661	007116	045	116	045	PKTADD: .ASCIZ	'%N%A Packet Address = %01%05'
1662						
1663	007153	045	116	045	PKTNEW: .ASCIZ	'%N%A ' .EVEN
1664						
1665						

.SBTTL PRIBXOR - PRINT EXPD, RECV AND XOR BYTE

```

1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685 007164
1686 007164
1687 007170 010203
1688 007172
1689 007202 012700 177400
1690 007206 040001
1691 007210 040002
1692 007212 040003
1693 007214
    007214 010346
    007216 010146
    007220 010246
    007222 012746 007246
    007226 012746 000004
    007232 010600
    007234 104414
    007236 062706 000012
1694 007242 010300
1695 007244 000207
1696
1697 007246 045 116 045 XORBFOR: .ASCIZ '#N#A EXPD: #03#A RECV: #03#A XOR: #03#
1698
1699
  
```

```

;+
;
;PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE DATA BYTE
;THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
;
;INPUTS:
;
;      R1      RECEIVED DATA
;      R2      EXPECTED DATA
;
;OUTPUT:
;
;      R0      XOR OF EXPECTED/RECEIVED DATA
;-
PRIBXOR::
    SAVREG                ;SAVE THE REGISTERS
    MOV      R2,R3        ;EXPECTED DATA
    XOR      R1,R3        ;FORM THE EXCLUSIVE OR
    MOV      #+C<377>,R0  ;BYTE MASK
    BIC      R0,R1        ;SAVE LOW BYTE RECV
    BIC      R0,R2        ;SAVE LOW BYTE EXPD
    BIC      R0,R3        ;SAVE LOW BYTE XOR
    PRINTB   #XORBFOR,R2,R1,R3 ;PRINT THE MESSAGE
    MOV      R3,-(SP)
    MOV      R1,-(SP)
    MOV      R2,-(SP)
    MOV      #XORBFOR,-(SP)
    MOV      #4,-(SP)
    MOV      SP,R0
    TRAP    C#PNTB
    ADD     #12,SP
    MOV     R3,R0        ;R0 HAS XOR ON RETURN
    RTS     PC           ;RETURN TO CALLER
  
```

```

1701          .SBTTL  PRI XOR - PRINT EXPD, RECV AND XOR
1702
1703          ;*
1704          ;
1705          ;PRINT EXPECTED DATA, RECEIVED DATA, AND XOR OF THE TWO
1706          ;THIS ROUTINE IS NORMALLY CALLED ONLY FOR PRINT ROUTINES.
1707          ;
1708          ;INPUTS:
1709          ;
1710          ;      R1      RECEIVED DATA
1711          ;      R2      EXPECTED DATA
1712          ;
1713          ;OUTPUT:
1714          ;
1715          ;      R0      XOR OF EXPECTED/RECEIVED DATA
1716          ;
1717          ;-
1718
1719 007314      PRTYDP..
    
```

```

1720 007314      SAVREG          ;SAVE THE REGISTERS
1721 007320 010203      MOV      R2,R3          ;EXPECTED DATA
1722 007322          XOR      R1,R3          ;FORM THE EXCLUSIVE OR
1723 007332          PRINTB   #XORFOR,R2,R1,R3 ;PRINT THE MESSAGE
          007332 010346      MOV      R3,-(SP)
          0C7334 010146      MOV      R1,-(SP)
          0C7336 010246      MOV      R2,-(SP)
          007340 012746 007364      MOV      #XORFOR,-(SP)
          007344 012746 000004      MOV      #4,-(SP)
          007350 010600          MOV      SP,R0
          007352 104414          TRAP   C$PNTB
          007354 062706 000012      ADD      #12,SP
1724 007360 010300      MOV      R3,R0          ;R0 HAS XOR ON RETURN
1725 007362 000207      RTS      PC          ;RETURN TO CALLER
1726
1727 007364 045 116 045 XORFOR: .ASCIZ 'N#A EXPD: #06#A RECV: #06#A XOR: #06#
1728          .EVEN
    
```

```

1730 .SBTTL PRIEQU - PRINT BIT NUMBERS AS ASCII EQUIVALENT
1731
1732 ;*
1733 ;
1734 ;ROUTINE TO CONVERT BIT VALUES TO ASCII AND PRINT THE STRING
1735 ;THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
1736 ;
1737 ;INPUTS:
1738 ;
1739 ; R0 OCTAL VALUE TO CONVERT
1740 ; R1 TABLE OF POINTERS TO ASCII EQUIVALENT
1741 ;
1742 ;-
1743
1744 007432 PRIEQU:
1745 007432 SAVREG ;SAVE THE REGISTERS
1746 007436 000207 RTS PC ;RETURN TO CALLER
1747
1748
1749
1750
1751 .SBTTL PRIRAM - PRINT RAM ADDRESS
1752 ;*
1753 ;
1754 ;PRINT CONTROLLER RAM ADDRESS.
1755 ;THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
1756 ;
1757 ;INPUTS:
1758 ;
1759 ; R4 RAM ADDRESS
1760 ;
1761 ;-
1762 007440 PRIRAM:
1763 007440 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
1764 007444 PRINTB #RAMFOR,R4 ;PRINT RAM ADDRESS IN ERROR
1765 007446 010446 MOV R4,-(SP)
1766 007446 012746 007470 MOV #RAMFOR,-(SP)
1767 007452 012746 000002 MOV #2,-(SP)
1768 007456 010600 MOV SP,R0
1769 007460 104414 TRAP C#PNTB
1770 007462 062706 000006 ADD #6,SP
1771 007466 000207 RTS PC ;RETURN
1772
1773 1767 007470 045 116 045 RAMFOR: .ASCIZ '#N#A CONTROLLER RAM ADDRESS = #06'
1774 .EVEN
1775
1776
1777 .SBTTL PRIADD - PRINT MEMORY ERROR ADDRESS
1778 ;*
1779 ;
1780 ;PRINT MEMORY ADDRESS
1781 ;THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
1782 ;
1783 ; IMPLICIT INPUTS
1784 ;
1785 ; ERRHI - HIGH ORDER ADDRESS
1786 ; ERRLO - LOW ORDER ADDRESS

```

```

1781
1782
1783 007532
1784 007532
1785 007536 013700 002202
1786 007542 013701 002204
1787 007546 010102
1788 007550 006101
1789 007552 006100
1790 007554
    007554 010246
    007556 010046
    007560 012746 007602
    007564 012746 000003
    007570 010600
    007572 104414
    007574 062706 000010
1791 007600 000207
1792
1793 007602 045 116 045 PRIA0: .ASCIZ '%N%A MEMORY ERROR ADDRESS = %01%05'
1794 .EVEN
1795
1796
1797 .SBTTL PRITADD - PRINT MEMORY TEST ADDRESS
1798
1799 ;+
1800 ;PRINT MEMORY ADDRESS
1801 ;THIS ROUTINE IS NORMALLY CALLED ONLY FROM PRINT ROUTINES.
1802 ;
1803 ; IMPLICIT INPUTS
1804 ;
1805 ; ERRHI - HIGH ORDER ADDRESS
1806 ; ERRLO - LOW ORDER ADDRESS
1807 ;
1808 ;-
1809 PRITADD:
1810 007646
1811 007652 013700 002202
1812 007656 013701 002204
1813 007662 010102
1814 007664 006101
1815 007666 006100
1816 007670
    007670 010246
    007672 010046
    007674 012746 007716
    007700 012746 000003
    007704 010600
    007706 104414
    007710 062706 000010
1817 007714 000207
1818
1819 007716 045 116 045 PRIT0: .ASCIZ '%N%A MEMORY TEST ADDRESS = %01%05'
1820 .EVEN
1821
1822
1823
    
```

```

1825                                     .SBTTL SPACE - SPACE RECORDS (FORWARD AND REVERSE) COMMAND
1826
1827
1828                                     ;*
1829                                     ;
1830                                     ;ROUTINE TO ISSUE A SPACE RECORDS
1831                                     ;COMMAND (FORWARD OR REVERSE)
1832                                     ;
1833                                     ;INPUT:
1834                                     ;
1835                                     ;       R3       NUMBER OF RECORDS TO BE SPACED OVER
1836                                     ;               BIT15 CONTROLS DIRECTION
1837                                     ;               BIT15 = 0 IS FORWARD
1838                                     ;               BIT15 = 1 IS REVERSE
1839                                     ;       R5       FIRST DEVICE UNIBUS ADDRESS
1840                                     ;
1841                                     ;       REQUIRES A WRITE CHARACTERISTICS DONE PREVIOUSLY
1842                                     ;
1843                                     ;OUTPUT:
1844                                     ;
1845                                     ;       CARRY   SET - SPACE RECORDS COMMAND OK
1846                                     ;               CLR - SPACE RECORDS FAILED
1847                                     ;
1848                                     ;
1849                                     ;       R0       THE CONTENTS OF R4 IS MOVED TO R0
1850                                     ;
1851                                     ;IMPLICIT OUTPUT:
1852                                     ;
1853                                     ;       TAPE HAS BEEN MOVED
1854                                     ;
1855                                     ;SIDE EFFECTS:
1856                                     ;
1857                                     ;
1858                                     ;-
1859
1860 007760                               SPACE::
1861 007760                               SAVREG
1862 007764 012737 000764 010150           MOV     #500.,SDELAY      ;SAVE THE GENERAL REGISTERS
1863 007772 012737 140010 010140         MOV     #140010,80$     ;SET UP DELAY
1864 010000 005703                       TST     R3              ;SET UP COMMAND, SPACE FORWARD
1865 010002 100403                       BMI     5$              ;CHECK FOR DIRECTION
1866 010004 010337 010142                 MOV     R3,90$         ;BR, IF REVERSE INDICATED
1867 010010 000407                       BR      10$            ;LOAD UP NUMBER OF RECORDS TO SPACE
1868 010012 042703 100000                 5$:    BIC     #BIT15,R3 ;GO DO COMMAND
1869 010016 010337 010142                 MOV     R3,90$         ;CLEAR DIRECTION BIT
1870 010022 052737 000400 010140         BIS     #BIT8,80$     ;LOAD UP NUMBER OF RECORDS TO SPACE
1871 010030 012704 010140                 10$:   MOV     #80$,R4    ;SET REVERSE BIT IN COMMAND PACKET
1872 010034 010465 177776                 MOV     R4,TSDB(R5)   ;SET UP R4 WITH PACKET ADDRESS
1873 010040 004737 016744                 15$:   JSR     PC,WAITF ;SEND OUT COMMAND
1874 010044 103420                       BCS    20$            ;WAIT FOR SSR
1875 010046                               DELAY  250            ;BR, IF SSR IS SET AND OK
1876 010046 012727 000250                 MOV     #250,(PC)+    ;DELAY ABOUT .25 SECONDS
1877 010052 000000                       .WORD  0
1878 010054 013727 002116                 MOV     L$DLY,(PC)+
1879 010060 000000                       .WORD  0
1880 010062 005367 177772                 DEC     -6(PC)
1881 010066 001375                               BNE    .-4
    
```

010070	005367	177756		DEC	-22(PC)	
010074	001367			BNE	.-20	
1876	010076	005337	010150	DEC	SDELAY	;BUMP DELAY COUNTER DOWN
1877	010102	001356		BNE	15\$;BR, IF MORE DELAY
1878	010104	000411		BR	60\$;BR IF TROUBLE CARRY = CLEAR
1879	010106	016501	000000	20\$: MOV	TSSR(R5),R1	;READ TSSR
1880	010112	012702	000200	MOV	#SSR,R2	;SET UP EXPECTED
1881	010116	020201		25\$: CMP	R2,R1	;ARE THEY OK
1882	010120	001401		BEQ	40\$;BR, IF EQUAL = OK
1883	010122	000402		BR	60\$;TROUBLE EXIT
1884	010124	000261		40\$: SEC		;SET CARRY NO TROUBLE
1885	010126	000401		BR	70\$;EXIT
1886	010130	000241		60\$: CLC		;CARRY CLEAR = ERROR
1887	010132			70\$:		
1888	010132	010400		MOV	R4,R0	;PASS PACKET ADDRESS
1889	010134	000207		RTS	PC	;RETURN

```
1891      ;
1892      ;
1893      ;
1894      ;PACKET FOR SPACE COMMAND
1895      ;
1897 010136      .BLKB  10-<.-TUV2A&7>
1899      ;
1900      ;COMMAND WORD
1901 010140 000000 80$: .WORD
1902      ;NUMBER OF RECORDS TO BE SPACED OVER WORD
1903 010142 000000 90$: .WORD
1904 010144 000000      .WORD
1905 010146 000000      .WORD
1906 010150 000000 SDELAY: .WORD 0          ;DELAY COUNTER
1907      .EVEN
```

```

1909          .SBTTL  WRTCHR  - WRITE CHARACTERISTICS COMMAND
1910
1911          ;+
1912          ;
1913          ;ROUTINE TO ISSUE A WRITE CHARACTERISTICS
1914          ;COMMAND SO THAT OTHER COMMANDS WILL BE ACCEPTED
1915          ;
1916          ;INPUT:
1917          ;
1918          ;      R4      ADDRESS OF PACKET FROM TEST
1919          ;      R5      FIRST DEVICE UNIBUS ADDRESS
1920          ;      REQUIRES A CALL TO SOFINIT BE DONE PREVIOUSLY
1921          ;
1922          ;OUTPUT:
1923          ;
1924          ;      R0      TSSR CONTENTS
1925          ;      CARRY   SET - WRITE CHARACTERISTICS COMMAND OK
1926          ;              CLR - WRITE CHARACTERISTICS FAILED
1927          ;
1928          ;IMPLICIT OUTPUT:
1929          ;
1930          ;      MESSAGE BUFFER AND OTHER BUFFERS ALL SET UP
1931          ;      SOFTWARE SWITCHES SET AS FOLLOWS:
1932          ;              BENBSW = BUFFER ENABLE SWITCH ON OR OFF
1933          ;
1934          ;
1935          ;SIDE EFFECTS:
1936          ;
1937          ;
1938          ;-
1939

```

```

1940 010152          WRTCHR::
1941 010152          SAVREG          ;SAVE THE GENERAL REGISTERS
1942 010156 005037 002174          CLR      BENBSW          ;CLEAR BUFFER ENABLE SWITCH
1943 010162 010465 177776          10$:  MOV      R4,TSDB(R5)      ;SEND OUT COMMAND
1944 010166 004737 017060          JSR      PC,CHKTSSR      ;WAIT FOR SSR
1945 010172 103401          BCS      20$          ;BR, IF SSR IS SET AND OK
1946 010174 000423          BR       60$          ;BR IF TROUBLE CARRY = CLEAR
1947 010176 016501 000000          20$:  MOV      TSSR(R5),R1      ;READ TSSR
1948 010202 012702 000200          MOV      #SSR,R2          ;SET UP EXPECTED
1949 010206 032701 000100          BIT      #OFL,R1          ;WAS OFF LINE SET IN TSSR
1950 010212 001402          BEQ      25$          ;BR, IF NO OFL SET
1951 010214 052702 000100          BIS      #OFL,R2          ;MAKE THEM LOOK ALIKE
1952 010220 020201          25$:  CMP      R2,R1          ;ARE THEY OK
1953 010222 001401          BEQ      40$          ;BR, IF EQUAL = OK
1954 010224 000407          BR       60$          ;TROUBLE EXIT
1955 010226 062704 000010          40$:  ADD      #8.,R4          ;POINT TO WRT CHARA DATA PACKET
1956 010232 011403          MOV      (R4),R3          ;GET ADDRESS OF MESSAGE BUFFER
1957 010234 010337 002716          MOV      R3,MESBFA        ;STORE FOR PRINT ROUTINES
1958 010240 000261          SEC          ;SET CARRY NO TROUBLE
1959 010242 000401          BR       70$          ;EXIT
1960 010244 000241          60$:  CLC          ;CARRY CLEAR = ERROR
1961 010246 016500 000000          70$:  MOV      TSSR(R5),R0      ;RETURN TSSR CONTENTS
1962 010252 000207          RTS      PC          ;RETURN
1963
1964

```

1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993

.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
;
;
;-

```

1994 010254
 1995 010254
 1996 010260 012704 010350
 1997 010264 010465 177776
 1998 010270 012703 000550
 1999 010274 004737 016744
 2000 010300 103417
 2001 010302
 010302 012727 000372
 010306 000000
 010310 013727 002116
 010314 000000
 010316 005367 177772
 010322 001375
 010324 005367 177756
 010330 001367
 2002 010332 005303
 2003 010334 001357
 2004 010336 000241
 2005 010340 010400
 2006 010342 000207
 2007
 2008
 2010 010344
 2012 010350
 2013 010350 102010
 2014 010352 000000

```

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

```

```

2016          .SBTTL CKRAM - COMPARE RAM TO I/O PACKET
2017
2018          ;+
2019          ;
2020          ;ROUTINE TO READ THE FIRST 8 BYTES FROM RAM
2021          ;MEMORY AND COMPARE THIS DATA TO A COMMAND PACKET.
2022          ;
2023          ;INPUT:
2024          ;
2025          ;       R4      ADDRESS OF THE COMMAND PACKET
2026          ;       R5      FIRST DEVICE UNIBUS ADDRESS
2027          ;
2028          ;OUTPUT:
2029          ;
2030          ;       CARRY   SET - RAM MATCHES PACKET
2031          ;               CLR - RAM DOES NOT MATCH PACKET
2032          ;
2033          ;IMPLICIT OUTPUT:
2034          ;
2035          ;       THE TABLE RAMDATA IS FILLED WITH THE
2036          ;       DATA HELD IN RAM.
2037          ;       RAMSIZ IS SET TO 8. FOR PRAMPKT ROUTINE
2038          ;
2039          ;SIDE EFFECTS:
2040          ;
2041          ;
2042          ;-
2043
2044 010354      CKRAM:
2045 010354      SAVREG          ;SAVE THE GENERAL REGISTERS
2046 010360 012701 002206      MOV      #RAMDATA,R1      ;ADDRESS TO SAVE THE RAM DATA
2047 010364 012702 000020      MOV      #RMPKTBEG,R2     ;BYTE ADDRESS OF FIRST RAM DATA
2048 010370 005003              CLR      R3              ;CLEAR THE ERROR FLAG
2049 010372 004737 017060      JSR      PC,CHKTSSR      ;WAIT FOR SSR
2050 010376 004737 017060 10$: JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2051 010402 110265 177777      MOVVB   R2,TSDBH(R5)    ;SELECT NEXT RAM ADDRESS
2052 010406 004737 017060      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2053 010412 116511 177776      MOVVB   TSBAL(R5),(R1) ;READ THE RAM DATA
2054 010416 122124              CMPB    (R1)+,(R4)+    ;COMPARE TO EXPECTED
2055 010420 001401              BEQ     20$          ;BRANCH IF OK
2056 010422 005203              INC     R3              ;SET ERROR FLAG
2057 010424 005202 000027 20$: INC     R2              ;ADDRESS OF NEXT RAM LOCATION
2058 010426 020227 000027      CMP     R2,#RMPKTEND  ;REACHED END YET ?
2059 010432 003761              BLE     10$          ;BRANCH TILL ALL READ
2060 010434 005703              TST    R3              ;WAS AN ERROR FOUND ?
2061 010436 001402              BEQ     30$          ;BRANCH IF NOT
2062 010440 000241              CLC     ;CLEAR CARRY TO SHOW ERROR
2063 010442 000401              BR     50$          ;AND EXIT
2064 010444 000261 30$:      SEC     ;SHOW GOOD COMPARE
2065 010446 012737 000010 50$: MOV     #8.,RAMSIZ    ;SETUP RAMSIZ FOR PRAMPKT ROUTINE
2066 010454 000207              RTS     PC           ;RETURN
2067

```

```

2069          .SBTTL RAMER - READ AND DISPLAY SELECTED RAM
2070          ;+
2071          ;
2072          ;ROUTINE TO READ THE SELECTED RAM LOCATIONS
2073          ;
2074          ;INPUT:
2075          ;
2076          ;       R5       FIRST DEVICE UNIBUS ADDRESS
2077          ;       CONSOLE WILL ALSO BE PRINTED TO
2078          ;
2079          ;IMPLICIT OUTPUT:
2080          ;
2081          ;       THE TABLE RAMDATA IS FILLED WITH THE
2082          ;       DATA HELD IN RAM.
2083          ;
2084          ;SIDE EFFECTS:
2085          ;
2086          ;
2087          ;-
2088
2089 010456      RAMER::
2090 010456      SAVREG          ;SAVE THE GENERAL REGISTERS
2091 010462 013705 010642      MOV      RAMR5H,R5          ;RESET R5 TO FIRST DEVICE REGISTER
2092 010466 012701 002206      MOV      #RAMDATA,R1        ;ADDRESS TO SAVE THE RAM DATA
2093 010472 013702 010640      MOV      RAMHLD,R2          ;BYTE ADDRESS OF THE FIRST RAM DATA
2094 010476 013703 002246      MOV      RAMSIZ,R3          ;SET THE SIZE OF THE READ UP
2095 010502 004737 017060      10$:   JSR      PC,CHKTSSR        ;WAIT FOR THE SSR TO SET
2096 010506 110265 177777      MOVVB  R2,TSDBH(R5)      ;SELECT NEXT RAM ADDRESS
2097 010512 004737 017060      JSR      PC,CHKTSSR        ;WAIT FOR SSR TO SET
2098 010516 116521 177776      MOVVB  TSBAL(R5),(R1)+ ;READ THE RAM DATA
2099 010522 062702 000001      20$:   ADD      #1,R2          ;ADDRESS OF THE NEXT RAM LOCATION
2100 010526 077313              SOB      R3,10$          ;NUMBER OF LOCATIONS COUNTER
2101 010530 013704 002246      MOV      RAMSIZ,R4          ;GET THE RAM SIZE
2102 010534 013702 010640      MOV      RAMHLD,R2          ;GET THE STARTING RAM ADDRESS
2103 010540 060204              ADD      R2,R4            ;CALCULATE THE END ADDRESS
2104 010542 162704 000001      SUB      #1,R4            ;CORRECT VALUE OF PRINTOUT
2105 010546              PRINTX #RAMIOP,R2,R4          ;RAM ADDRESS = 10 - 17, ETC.
2106 010546              MOV      R4,-(SP)
2107 010550 010246              MOV      R2,-(SP)
2108 010552 012746 010644      MOV      #RAMIOP,-(SP)
2109 010556 012746 000003      MOV      #3,-(SP)
2110 010562 010600              MOV      SP,R0
2111 010564 104415              TRAP    C:PNTX
2112 010566 062706 000010      ADD      #10,SP
2113 010572 012701 002206      MOV      #RAMDATA,R1        ;ADDRESS OF WHERE RAM DATA IS
2114 010576 013703 002246      MOV      RAMSIZ,R3          ;THE SIZE OF THE RAM FIELD READ
2115 010602 005004              30$:   CLR      R4                ;NO EXTRA DATA LEFT OVER
2116 010604 112104              MOVVB  (R1)+,R4            ;PICK UP BYTE OF RAM DATA
2117 010606 042704 177400      BIC      #177400,R4          ;GET RID OF SIGN EXTEND
2118 010612              PRINTX #RAMPD,R4          ;"010 211 111 222 377 000 123 134 ETC."
2119 010612 010446              MOV      R4,-(SP)
2120 010614 012746 010715      MOV      #RAMPD,-(SP)
2121 010620 012746 000002      MOV      #2,-(SP)
2122 010624 010600              MOV      SP,R0
2123 010626 104415              TRAP    C:PNTX
2124 010630 062706 000006      ADD      #6,SP
2125 010634 077316              SOB      R3,30$          ;LOOP UNTIL ALL PRINTED
  
```

```
2113 010636 000207          50$:  RTS   PC           ;RETURN
2114
2115 010640 000000          RAMHLD: .WORD 0           ;RAM ADDR HOLDER 1ST ADDRESS
2116 010642 000000          RAMR5H: .WORD 0           ;HOLDS R5 FOR LATER
2117 010644   045    116   045  RAMIOP: .ASCIZ 'N%A Ram Address (Octal) = %03%A - %03%N'
2118 010715   045    101   040  RAMPD: .ASCIZ 'A %03%A '
2119
2120          .EVEN
```

```

2122          .SBTTL  CKRAM2 - COMPARE RAM TO I/O CHARACTERISTICS DATA
2123          ;+
2124          ;
2125          ;ROUTINE TO READ THE FIRST 8 OR 10 BYTES FROM RAM
2126          ;MEMORY AND COMPARE THIS DATA TO A CHARACTERISTICS DATA BLOCK.
2127          ;
2128          ;INPUT:
2129          ;
2130          ;      R4      ADDRESS OF THE CHARACTERISTICS DATA
2131          ;      R5      FIRST DEVICE UNIBUS ADDRESS
2132          ;
2133          ;OUTPUT:
2134          ;
2135          ;      CARRY   SET - RAM MATCHES PACKET
2136          ;             CLR - RAM DOES NOT MATCH PACKET
2137          ;
2138          ;IMPLICIT OUTPUT:
2139          ;
2140          ;      THE TABLE RAMDATA IS FILLED WITH THE
2141          ;      DATA HELD IN RAM.
2142          ;      RAMSIZ IS SET TO 8. OR 10. FOR PRAMPKT ROUTINE
2143          ;
2144          ;SIDE EFFECTS:
2145          ;
2146          ;
2147          ;-
2148
2149 010730      CKRAM2::
2150 010730          SAVREG          ;SAVE THE GENERAL REGISTERS
2151 010734      012701 002206      MOV      #RAMDATA,R1      ;ADDRESS TO SAVE THE RAM DATA
2152 010740      012702 000167      MOV      #RMCHBEG,R2      ;BYTE ADDRESS OF FIRST RAM DATA
2153 010744      005003              CLR      R3          ;CLEAR THE ERROR FLAG
2154 010746      004737 017060      JSR      PC,CHKTSSR      ;WAIT FOR SSR
2155 010752      004737 017060      10$:   JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2156 010756      110265 177777      MOV      R2,TSDBH(R5)      ;SELECT NEXT RAM ADDRESS
2157 010762      004737 017060      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
2158 010766      116511 177776      MOV      TSBAL(R5),(R1)    ;READ THE RAM DATA
2159 010772      122124              CMP      (R1)+,(R4)+      ;COMPARE TO EXPECTED
2160 010774      001401              BEQ      20$          ;BRANCH IF OK
2161 010776      005203              INC      R3          ;SET ERROR FLAG
2162 011000      005202              INC      R2          ;ADDRESS OF NEXT RAM LOCATION
2163 011002      012737 000010 002246  MOV      #8.,RAMSIZ      ;ASSUME NORMAL NOT SET
2164 011010      020227 000176      CMP      R2,#RMCHEND-2    ;REACHED END YET ?
2165 011014      003756              BLE      10$          ;BRANCH TILL ALL READ
2166 011016      005703              27$:   TST      R3          ;WAS AN ERROR FOUND ?
2167 011020      001402              BEQ      30$          ;BRANCH IF NOT
2168 011022      000241              CLC          ;CLEAR CARRY TO SHOW ERROR
2169 011024      000401              BR      50$          ;AND EXIT
2170 011026      000261              30$:   SEC          ;SHOW GOOD COMPARE
2171 011030      000207              50$:   RTS      PC      ;RETURN
2172

```

```

2174 .SBTTL CKMSG - COMPARE WRITE CHAR. MESSAGE BUFFERS
2175 ;+
2176 ;
2177 ;ROUTINE TO COMPARE A WRITE CHARACTERISTICS EXPD AND RECV
2178 ;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
2179 ;ERROR PRINT ROUTINES.
2180 ;
2181 ;INPUT:
2182 ;
2183 ; R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS
2184 ; R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS
2185 ; R2 EXPD MESSAGE BUFFER ADDRESS
2186 ;OUTPUT:
2187 ;
2188 ; CARRY SET - MESSAGE BUFFERS MATCH
2189 ; CLR -MESSAGE BUFFERS DON'T MATCH
2190 ;
2191 ;IMPLICIT OUTPUT:
2192 ;
2193 ; EXPMSG BUFFER IS SET TO EXPD DATA
2194 ; RECMMSG BUFFER IS SET TO RECV DATA
2195 ; RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
2196 ; RCVLOADD SET TO LOW ORDER ADDRESS OF RECV
2197 ;
2198 ;-
2199 011032 CKMSG::
2200 011032 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
2201 011036 010037 002250 MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
2202 011042 010137 002252 MOV R1,RCVLOAD ;SAVE RECV LOW ADDRESS
2203 011046 005737 003102 TST KTENABLE ;TESTING ABOVE 28K?
2204 011052 001403 BEQ 10$ ;BR IF NO
2205 011054 004737 020112 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
2206 011060 010001 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
2207 011062 005004 10$: CLR R4 ;WORD IN BUFFER
2208 011064 005003 CLR R3 ;CLEAR ERROR SEEN FLAG
2209 011066 010205 MOV R2,R5 ;GET EXPD BUFFER ADDRESS
2210 011070 011264 002266 15$: MOV (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
2211 011074 011164 002432 MOV (R1),RCMMSG(R4) ;SAVE RECV FOR ERROR REPORT
2212 011100 022221 CMP (R2)+,(R1)+ ;EXPD EQUAL RECV?
2213 011102 001401 BEQ 25$ ;BR IF YES
2214 011104 005203 INC R3 ;SET ERROR SEEN FLAG
2215 011106 062704 000002 25$: ADD #2,R4 ;POINT TO NEXT WORD ADDRESS
2216 011112 020427 000014 CMP R4,#14 ;DONE FIRST 7 WORDS?
2217 011116 003764 BLE 15$ ;BR IF NO
2218 011120 032765 000200 000012 BIT #X2.EXTF,XST2(R5); IS EXTENDED FEATURES SET IN EXPD?
2219 011126 001403 BEQ 50$ ;BR IF NO
2220 011130 020427 000016 CMP R4,#16 ;DONE EXTENDED FEATURES WORD?
2221 011134 003755 BLE 15$ ;BR IF NO
2222 011136 005703 50$: TST R3 ;ANY ERRORS SEEN?
2223 011140 001402 BEQ 55$ ;BR IF NO
2224 011142 000241 CLC ;SET FAILURE
2225 011144 000401 BR 60$ ;
2226 011146 000261 55$: SEC ;SET SUCCESS
2227 011150 000207 60$: RTS PC ;RETURN
2228

```

```

2230 .SBTTL CKMSG2 - COMPARE EXPD RECV MESSAGE BUFFERS
2231 ;*
2232 ;
2233 ;ROUTINE TO COMPARE AN EXPECTED AND RECEIVED MESSAGE
2234 ;BUFFER. THE EXPECTED AND RECEIVED BUFFERS ARE STORED FOR
2235 ;ERROR PRINT ROUTINES.
2236 ;
2237 ;INPUT:
2238 ;
2239 ; R0 RECV MESSAGE BUFFER HIGH ORDER ADDRESS
2240 ; R1 RECV MESSAGE BUFFER LOW ORDER ADDRESS
2241 ; R2 EXPD MESSAGE BUFFER ADDRESS
2242 ; R3 NUMBER OF BYTES TO COMPARE
2243 ;
2244 ;OUTPUT:
2245 ;
2246 ; CARRY SET - MESSAGE BUFFERS MATCH
2247 ; CLR - MESSAGE BUFFERS DON'T MATCH
2248 ;
2249 ;IMPLICIT OUTPUT
2250 ;
2251 ; EXPMSG BUFFER IS SET TO EXPD DATA
2252 ; RECMG2 BUFFER IS SET TO RECV DATA
2253 ; RCVHIADD SET TO HIGH ORDER ADDRESS OF RECV
2254 ; RCVLOAD SET TO LOW ORDER ADDRESS OF RECV
2255 ;
2256 ;-
2257 CKMSG2::
2258 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
2259 CMP R3,#RECMG2-EXPMSG;@@D IS COUNT ABOVE MAX ALLOWED?
2260 BLE 5$ ;@@D BR IF NO
2261 MOV #RECMG2-EXPMSG,R3;@@D
2262 PRINTF #DEBUGMSG ;@@D
011170 MOV #DEBUGMSG,-(SP)
011174 MOV #1,-(SP)
011200 MOV SP,R0
011202 TRAP C:PNTF
011204 ADD #4,SP
2263 5$: MOV R0,RCVHIADD ;SAVE RECV HIGH ADDRESS
2264 MOV R1,RCVLOAD ;SAVE RECV LOW ADDRESS
2265 TST KTENABLE ;TESTING ABOVE 28K?
2266 BEQ 10$ ;BR IF NO
2267 JSR PC,SETMAP ;RETURN ADDRESS BIASED TO PAR6 IN R0
2268 MOV R0,R1 ;GET RETURNED ADDRESS BIASED TO PAR6
2269 10$: CLR R4 ;WORD IN BUFFER
2270 CLR R5 ;CLEAR ERROR SEEN FLAG
2271 15$: MOVB (R2),EXPMSG(R4) ;SAVE EXPD FOR ERROR REPORT
2272 MOVB (R1),RECMG2(R4) ;SAVE RECV FOR ERROR REPORT
2273 CMPB (R2)+,(R1)+ ;EXPD EQUAL RECV?
2274 BEQ 25$ ;BR IF YES
2275 INC R5 ;SET ERROR SEEN FLAG
2276 25$: ADD #1,R4 ;POINT TO NEXT BYTE
2277 CMP R4,R3 ;DONE ALL BYTES?
2278 BGE 50$ ;BR IF YES
2279 BR 15$ ;DO NEXT BYTE
2280 50$: TST R5 ;ANY ERRORS SEEN?
2281 BEQ 55$ ;BR IF NO
  
```

```
2282 011274 000241          CLC          ;SET FAILURE
2283 011276 000401          BR          60$          ;
2284 011300 000261          55$: SEC          ;SET SUCCESS
2285 011302 000207          60$: RTS          PC          ;RETURN
2286
2287 011304 120 122 117 DEBUGMSG: .ASCIZ 'PROGRAM INTERNAL ERROR -CKMSG2 MESSAGE BUFFER EXCEEDED-' ;@@D
2288 011374 045 116 045 FERCM: .ASCII /%N%A ***/
2289 011405 040 040 124 ERCM: .ASCIZ / TSSR ERROR CODE REC'D = /
2290 011440 056 056 056 SIMSG: .ASCIZ /.... AFTER DOING SOFT INIT/
2291 011473 124 105 123 TINERR: .ASCIZ /TEST: .../
2292 .EVEN
```

```

2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310 011506          BGNMSG  SFIMSG
      011506          SFIMSG::
2311 011506 004737 005264      JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
2312 011512 004737 017776      JSR    PC,CKDROP      ;DROP UNIT, IF ALLOWED
2313 011516          ENDMSG
      011516          L10003:
      011516 104423      TRAP   C$MSG

2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326 011520          BGNMSG  PKTSSR
      011520          PKTSSR::
2327 011520 004737 005264      JSR    PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
2328 011524 012700 000004      MOV    #4,RO          ;NO. OF WORDS IN PACKET
2329 011530 004737 006712      JSR    PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
2330 011534 013700 002716      MOV    MESBFA,RO      ;ADDRESS OF MESSAGE BUFFER
2331 011540 005001          CLR    R1              ;ASSUME NO HIGH MEMORY
2332 011542 004737 013702      JSR    PC,PRMESS      ;PRINT THE MESSAGE BUFFER ALSO
2333 011546          ENDMSG
      011546          L10004:
      011546 104423      TRAP   C$MSG

2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
  
```

```

2345
2346 011550          BGNMSG  PKTGETS
      011550          PKTGETS::
2347 011550 004737 005264      JSR    PC,PRITSSR      ;PRINT THE CONTENTS OF TSSR REGISTER
2348 011554 012700 000002      MOV    #2,R0           ;NO. OF WORDS IN GET STATUS PACKET
2349 011560 004737 006712      JSR    PC,PRIPKT      ;PRINT THE CONTENTS OF COMMAND PACKET
2350 011564          ENDMSG
      011564          L10005:
      011564 104423      TRAP   C$MSG

2351
2352
2353
2354          ;+
2354          ;PRINT TSSR ERRORS FOR INITIALIZATION TESTS
2355          ;
2356          ;INPUTS:
2357          ;
2358          ;       R1       TSSR CONTENTS
2359          ;       R4       ADDRESS OF COMMAND PACKET
2360          ;-
2361
2362 011566          BGNMSG  SFFMSG
      011566          SFFMSG::
2363 011566 004737 005264      JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR REGISTER
2364 011572          ENDMSG
      011572          L10006:
      011572 104423      TRAP   C$MSG

2365
2366
2367          .SBTTL  PKTMES  - PRINT TSSR AND MESSAGE BUFFER
2368          ;+
2369          ;
2370          ;PRINT ROUTINE TO PRINT THE CONTENTS OF TSSR AND MESSAGE
2371          ;BUFFER FOR ERROR REPORTS
2372          ;
2373          ;INPUTS:
2374          ;
2375          ;       R1       CONTENTS OF TSSR
2376          ;       R2       LOW ORDER MESSAGE BUFFER
2377          ;       R3       HIGH ORDER MESSAGE BUFFER ADDRESS
2378          ;       NOTE: R3 IS IGNORED IF KTENABLE FLAG IS CLEAR
2379          ;-
2380 011574          BGNMSG  PKTMES
      011574          PKTMES::
2381 011574 004737 005264      JSR    PC,PRITSSR      ;PRINT CONTENTS OF TSSR
2382 011600 010200          MOV    R2,R0           ;LOW ORDER ADDRESS
2383 011602 010301          MOV    R3,R1           ;HIGH ORDER ADDRESS
2384 011604 004737 013702      JSR    PC,PRMESS      ;PRINT THE MESSAGE BUFFER
2385 011610          ENDMSG
      011610          L10007:
      011610 104423      TRAP   C$MSG
2386

```

2388
 2389
 2390
 2391
 2392
 2393
 2394
 2395
 2396
 2397
 2398
 2399
 2400 011612
 011612
 2401 011612 004737 007646
 2402 011616 016501 000000
 2403 011622 004737 005264
 2404 011626
 011626
 011626 104423
 2405
 2406
 2407
 2408
 2409
 2410
 2411
 2412
 2413
 2414
 2415
 2416
 2417
 2418
 2419 011630
 011630
 2420 011630 012700 000007
 2421 011634 004737 015246
 2422 011640
 011640
 011640 104423
 2423
 2424

```

.SBTTL ADDSSR - PRINT TEST ADDRESS AND TSSR
;+
;PRINT ROUTINE TO PRINT THE CONTENTS OF
;TSSR AND A MEMORY TEST ADDRESS
;
;INPUTS:
;
;      R5      FIRST DEVICE UNIBUS ADDRESS
;      ERRHI   HIGH ORDER MEMORY TEST ADDRESS
;      ERRLO   LOW ORDER MEMORY TEST ADDRESS
;-

      BGNMSG  ADDSSR
ADDSSR::
      JSR     PC,PRITADD      ;PRINT MEMORY TEST ADDRESS
      MOV     TSSR(R5),R1    ;GET CURRENT TSSR
      JSR     PC,PRITSSR     ;PRINT THE CONTENTS OF TSSR REGISTER
      ENDMSG
L10010:
      TRAP    C$MSG

.SBTTL MSGEXP - PRINT WRITE CHAR. EXPD-RCV MESSAGE BUFFERS
;+
;
;PRINT ROUTINE TO PRINT WRITE CHARACTERISTIC MESSAGE BUFFER
;
;IMPLICIT INPUTS:
;
;      EXPMSG  - EXPECTED MESSAGE BUFFER
;      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/RCV MESSAGE BUFFERS
      ENDMSG
L10011:
      TRAP    C$MSG
  
```

```

2426 .SBTTL FIFEXP - PRINT FIFO EXP/RECV DATA
2427 ;+
2428 ;
2429 ;PRINT ROUTINE TO PRINT FIFO EXP/RECV DATA
2430 ;
2431 ; R1 - BYTE COUNT
2432 ;
2433 ;IMPLICIT INPUTS:
2434 ;
2435 ; EXPMSG - EXPECTED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
2436 ; RECVMSG - RECEIVED MESSAGE BUFFER (CONTAINS FIFO DATA ONLY)
2437 ;-
2438 011642 BGNMSG FIFEXP
011642
2439 011642 FIFEXP::
011642 010146 PRINTX #FIF1MSG,R1 ;PRINT BYTES TRANSFERRED
011644 012746 011714 MOV R1,-(SP)
011650 012746 000002 MOV #FIF1MSG,-(SP)
011654 010600 MOV #2,-(SP)
011656 104415 MOV SP,R0
011660 062706 000006 TRAP C$PNTX
2440 011664 ADD #6,SP
011664 012746 011763 PRINTX #FIF2MSG ;PRINT HEADER MSG
011670 012746 000001 MOV #FIF2MSG,-(SP)
011674 010600 MOV #1,-(SP)
011676 104415 MOV SP,R0
011700 062706 000004 TRAP C$PNTX
2441 011704 010100 ADD #4,SP
2442 011706 004737 015616 MOV R1,R0 ;GET BYTE COUNT
2443 011712 JSR PC,PRBYTEXP ;PRINT FIFO BYTES IN ERROR
011712 ENDMMSG
011712 104423 L10012:
2444 011714 045 116 045 FIF1MSG: TRAP C$MSG
2445 011763 045 116 045 FIF2MSG: .ASCIZ '%N%A NUMBER OF BYTES TRANSFERRED = %D2'
2446 .ASCIZ '%N%A FIFO DATA BYTES IN ERROR:'
2447 .EVEN
  
```

```

2449          .SBTTL MSGSTAT - PRINT STATUS HEADER AND MESSAGE BUFFERS
2450          ;+
2451          ;
2452          ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
2453          ;
2454          ;
2455          ;IMPLICIT INPUTS:
2456          ;
2457          ;   EXPMSG - EXPECTED MESSAGE BUFFER
2458          ;   RECMSG - RECEIVED MESSAGE BUFFER
2459          ;   RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2460          ;   RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2461          ;-
2462 012022      BGNMSG MSGSTAT
          012022      MSGSTAT::
2463 012022 012701 012064      MOV #STATCOD,R1 ;ASCII ADDRESS TABLE
2464 012026 012100      10$: MOV (R1)+,RO ;DONE ALL MSG LINES?
2465 012030 001410      BEQ 20$ ;BR IF YES
2466 012032      PRINTX RO ;PRINT STATUS BIT NAMES
          012032 010046      MOV RO,-(SP)
          012034 012746 000001      MOV #1,-(SP)
          012040 010600      MOV SP,RO
          012042 104415      TRAP C$PNTX
          012044 062706 000004      ADD #4,SP
2467 012050 000766      BR 10$ ;DO ANOTHER MSG LINE
2468 012052 012700 000012      20$: MOV #10.,RO ;NUMBER OF WORDS IN A READ STATUS BUFFER
2469 012056 004737 015246      JSR PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
2470 012062      ENDMSG
          012062      L10013:
          012062 104423      TRAP C$MSG
2471
2472 012064 012102 012144 012235      STATCOD: .WORD 1$,2$,3$,4$,5$,6$,0
2473 012102 045 116 045 1$: .ASCIZ 'N#A Tape Bus Signals in Word #8:'
2474 012144 045 116 045 2$: .ASCIZ 'N#A PARERR<15> IEOT <12> IFMK <9> IRDY<6> IRWD<2>'
2475 012235 045 116 045 3$: .ASCIZ 'N#A IRESV2<14> IIDENT<11> IHER <8> IONL<5> IFBY<1>'
2476 012326 045 116 045 4$: .ASCIZ 'N#A IRESV1<13> ICER <10> ISPEED<7> ILDP<4> IFPT<0>'
2477 012417 045 116 045 5$: .ASCIZ 'N#A Tape Bus Signals in Word #9:'
2478 012461 045 116 045 6$: .ASCIZ 'N#A DATMIS<7> ILW<6> OUTRDY<5> INRDY<4>'
2479          .EVEN
2480
2481
2482
2483          .SBTTL MSGLOOP - PRINT LOOPBACK HEADER AND MESSAGE BUFFERS
2484          ;+
2485          ;
2486          ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
2487          ;
2488          ;IMPLICIT INPUTS:
2489          ;
2490          ;   EXPMSG - EXPECTED MESSAGE BUFFER
2491          ;   RECMSG - RECEIVED MESSAGE BUFFER
2492          ;   RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2493          ;   RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2494          ;-
2495 012536      BGNMSG MSGLOOP
          012536      MSGLOOP::
2496 012536 012701 012600      MOV #LOOPCOD,R1 ;ASCII ADDRESS TABLE
  
```

```
2497 012542 012100          10$:  MOV      (R1)+,R0      ;DONE ALL MSG LINES?
2498 012544 001410          BEQ      20$              ;BR IF YES
2499 012546                PRINTX   R0              ;PRINT STATUS BIT NAMES
      012546 010046          MOV      RO,-(SP)
      012550 012746 000001    MOV      #1,-(SP)
      012554 010600          MOV      SP,R0
      012556 104415          TRAP     C$PNTX
      012560 062706 000004    ADD      #4,SP
2500 012564 000766          BR       10$              ;DO ANOTHER MSG LINE
2501 012566 012700 000012    20$:  MOV      #10.,R0      ;NUMBER OF WORDS IN A READ STATUS BUFFER
2502 012572 004737 015246    JSR      PC,PRMSGEXP     ;PRINT EXPD/RECV MESSAGE BUFFERS
2503 012576                ENDMSG
      012576                L10014:
      012576 104423          TRAP     C$MSG
2504
2505 012600 012620 012673 012772 LOOPCOD:  .WORD  1$,2$,3$,4$,5$,6$,7$,0
2506 012620          045  116  045  1$: .ASCIZ  '%N%A Tape Bus Loopback Signals in Word #8:'
2507 012673          045  116  045  2$: .ASCIZ  '%N%A PARERR<15> IRESV2<14> IRESV1<13>'
2508 012772          045  116  045  3$: .ASCIZ  '%N%A IHISP=>IEOT<12> IWRT=>IIDENT<11> IREV =>ICER <10>'
2509 013071          045  116  045  4$: .ASCIZ  '%N%A IWFM =>IFMK<09> IEDIT=>IHER <08> IFAD =>ISPEED<07>'
2510 013170          045  116  045  5$: .ASCIZ  '%N%A ITADO=>IRDY<06> ITAD1=>IONL <05> IERASE=>ILDLP <04>'
2511 013267          045  116  045  6$: .ASCIZ  '%N%A IREW =>IDBY<03> IRWU =>IRWD <02> IFEN =>IFBY <01>'
2512 013366          045  116  045  7$: .ASCIZ  '%N%A IGO =>IFPT<00>'
2513                .EVEN
2514
```

```

2516 .SBTTL MSGSUB - PRINT WRITE SUBSYSTEM MESSAGE BUFFER
2517 ;+
2518 ;
2519 ;PRINT ROUTINE TO PRINT MESSAGE BUFFER EXPD/RCV
2520 ;
2521 ;
2522 ;IMPLICIT INPUTS:
2523 ;
2524 ; EXPMSG - EXPECTED MESSAGE BUFFER
2525 ; RECMG - RECEIVED MESSAGE BUFFER
2526 ; RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2527 ; RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2528 ;-
2529 013414 BGNMSG MSGSUB
      013414 MSGSUB::
2530 013414 012700 000012 MOV #10.,R0 ;SIZE OF WRITE SUBSYSTEM BUFFER
2531 013420 004737 015246 JSR PC,PRMSGEXP ;PRINT EXPD/RCV MESSAGE BUFFERS
2532 013424 ENDMSG
      013424 L10015:
      013424 104423 TRAP C$MSG
2533
2534
2535
2536
2537
2538 .SBTTL MEMADD - PRINT MEMORY ADDRESS DATA ERROR
2539 ;+
2540 ;
2541 ;PRINT ROUTINE TO PRINT MEMORY ADDRESS DATA COMPARE ERROR
2542 ;
2543 ;IMPLICIT INPUTS:
2544 ;
2545 ; ERRHI - MEMORY ERROR HIGH ORDER ADDRESS
2546 ; ERRLO - MEMORY ERROR LOW ORDER ADDRESS
2547 ; EXP - EXPECTED DATA
2548 ; RECV - RECEIVED DATA
2549 ;-
2550 013426 BGNMSG MEMADD
      013426 MEMADD::
2551 013426 004737 007532 JSR PC,PRIADD ;PRINT MEMORY ADDRESS IN ERROR
2552 013432 013701 002176 MOV EXPD,R1 ;GET EXPD DATA
2553 013436 013702 002200 MOV RECV,R2 ;GET RECEIVED DATA
2554 013442 004737 007314 JSR PC,PRIXOR ;PRINT EXPD/RCV
2555 013446 ENDMSG
      013446 L10016:
      013446 104423 TRAP C$MSG
2556

```

2558
 2559
 2560
 2561
 2562
 2563
 2564
 2565
 2566
 2567
 2568
 2569
 2570
 2571
 2572
 2573
 2574
 2575
 2576
 2577
 2578

```
.SBTTL PRAMPKT - PRINT RAM AND PACKET DATA
;+
;
;PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
;WHEN THE RAM DATA DOES NOT MATCH.
;
;INPUTS:
;
; R4 POINTER TO COMMAND PACKET
;
;IMPLICIT INPUTS:
;
; RAMDATA DATA AS READ FROM THE RAM
; RAMSIZ NUMBER OF BYTES IN PACKET
; IF RAMSIZ=0 THEN DEFAULT TO 8.
;
;IMPLICIT OUTPUTS:
;
; RAMSIZ SET TO 0
;-
```

2579 013450
 2580 013450
 2581 013454 012701 002206
 2582 013460 005002
 2583 013462 122124
 2584 013464 001000
 2585 013466 116105 177777
 2586 013472 116403 177777
 2587 013476
 2588 013506 042703 177400
 2589 013512 116137 177777 002200
 2590 013520 116437 177777 002176
 2591 013526
 013526 010346
 013530 013746 002176
 013534 013746 002200
 013540 010246
 013542 012746 013616
 013546 012746 000005
 013552 010600
 013554 104414
 013556 062706 000014
 2592 013562 005202
 2593 013564 005737 002246
 2594 013570 001404
 2595 013572 020237 002246
 2596 013576 003731
 2597 013600 000403
 2598 013602 020227 000010
 2599 013606 002725
 2600 013610 005037 002246
 2601 013614 000207
 2602
 2603 013616 045 116 045
 2604

```
PRAMPKT:
    SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
    MOV #RAMDATA,R1 ;DATA FROM THE RAM
    CLR R2 ;INIT BYTE NUMBER
5$: CMPB (R1)+,(R4)+ ;COMPARE EXPECTED, RECEIVED
    BNE 7$ ;BR IF NO MATCH
7$: MOVB -1(R1),R5 ;GET RECV RAM DATA
    MOVB -1(R4),R3 ;GET EXPD PACKET DATA
    XOR R5,R3 ;XOR EXPD/RECV
    BIC #177400,R3 ;LOW BYTE ONLY
    MOVB -1(R1),RECV ;GET RECEIVED RAM DATA
    MOVB -1(R4),EXPD ;GET EXPECTED RAM DATA
    PRINTB #RAMASC,R2,RECV,EXPD,R3
    MOV R3,-(SP)
    MOV EXPD,-(SP)
    MOV RECV,-(SP)
    MOV R2,-(SP)
    MOV #RAMASC,-(SP)
    MOV #5,-(SP)
    MOV SP,R0
    TRAP C#PNTB
    ADD #14,SP
10$: INC R2 ;UPDATE BYTE COUNT
    TST RAMSIZ ;DEFAULT TO 8.?
    BEQ 15$ ;BR IF YES
    CMP R2,RAMSIZ ;DONE ALL BYTES?
    BLE 5$ ;BR IF NO
    BR 25$ ;
15$: CMP R2,#8. ;DONE DEFAULT NUMBER OF BYTES?
20$: BLT 5$ ;BR IF NO
25$: CLR RAMSIZ ;SET DEFAULT RAMSIZ
    RTS PC ;RETURN
RAMASC: .ASCIZ 'N%A BYTE: #D2%A RAM: #03%A Packet: #03%A XOR:#03%A'
        .EVEN
```

```

2606                                     .SBTTL PRMESS - PRINT CONTENTS OF MESSAGE BUFFER
2607                                     ;+
2608                                     ;
2609                                     ;THIS ROUTINE PRINTS THE CONTENTS OF
2610                                     ;THE 7 WORD MESSAGE BUFFER RETURNED BY THE
2611                                     ;TK-25.
2612                                     ;
2613                                     ;INPUT:
2614                                     ;
2615                                     ;     RO     LOW ORDER ADDRESS OF MESSAGE BUFFER
2616                                     ;     R1     HIGH ORDER ADDRESS OF MESSAGE BUFFER
2617                                     ;     NOTE: R1 IS IGNORED IF KTENABLE FLAG IS CLEAR
2618                                     ;
2619                                     ;THIS ROUTINE IS NORMALLY CALLED FROM A PRINT ROUTINE
2620                                     ;
2621                                     ;-
2622
2623 013702 PRMESS:
2624 013702 SAVREG
2625 013706 010537 010642 MOV R5,RAMRSH ;SAVE THE REGISTERS
2626 013712 010005 MOV RO,R5 ;SAVE DEVICE REGISTER POINTER
2627 013714 005737 003102 TST KTENABLE ;SAVE LOW ORDER ADDRESS
2628 013720 001001 BNE 10$ ;ADDRESS ABOVE 28K?
2629 013722 005001 CLR R1 ;BR IF YES
2630 013724 010103 10$: MOV R1,R3 ;SET HIGH ORDER ADDRESS TO 0
2631 013726 006100 ROL RO ;SAVE HIGH ORDER ADDRESS
2632 013730 006101 ROL R1 ;SHIFT BIT15 TO C BIT
2633 013732 PRINTX #PROASC,R1,R5 ;SHIFT TO HIGH ORDER FOR PRINTOUT
2634 013732 010546 MOV R5,-(SP) ;PRINT MESSAGE BUFFER ADDRESS
2635 013734 010146 MOV R1,-(SP)
2636 013736 012746 014540 MOV #PROASC,-(SP)
2637 013742 012746 000003 MOV #3,-(SP)
2638 013746 010600 MOV SP,RO
2639 013750 104415 TRAP C$PNTX
2640 013752 062706 000010 ADD #10,SP
2641 013756 022715 177777 CMP #177777,(R5) ;MESSAGE BUFFER FULL OF ONES
2642 013762 001010 BNE 15$ ;BR IF BUFFER IS PROBABLY OKAY
2643 013764 PRINTX #MESBFN ;"MESSAGE BUFFER PROBABLY NOT VALID"
2644 013764 012746 014460 MOV #MESBFN,-(SP)
2645 013770 012746 000001 MOV #1,-(SP)
2646 013774 010600 MOV SP,RO
2647 013776 104415 TRAP C$PNTX
2648 014000 062706 000004 ADD #4,SP
2649 014004 012746 014605 15$: PRINTX #PRIASC ;PRINT HEADER FOR CONTENTS
2650 014004 012746 000001 MOV #PRIASC,-(SP)
2651 014010 012746 000001 MOV #1,-(SP)
2652 014014 010600 MOV SP,RO
2653 014016 104415 TRAP C$PNTX
2654 014020 062706 000004 ADD #4,SP
2655 014024 005004 CLR R4 ;NUMBER OF THE NEXT WORD
2656 014026 010501 MOV R5,R1 ;COPY LOW ORDER ADDRESS
2657 014030 010300 MOV R3,RO ;COPY HIGH ORDER ADDRESS
2658 014032 001403 BEQ 20$ ;BR IF NOT ABOVE 28K
2659 014034 004737 020112 JSR PC,SETMAP ;SETUP PAR ADDRESS IN RO
2660 014040 010005 MOV RO,R5 ;GET PAR FORMAT ADDRESS ABOVE 28K
2661 014042 20$: PRINTX #MESHEA,(R5)+ ;PRINT "MESSAGE BUFFER HEADER ="
2662 014042
    
```

	014042	012546		MOV	(R5)+, -(SP)	
	014044	012746	014643	MOV	#MESHEA, -(SP)	
	014050	012746	000002	MOV	#2, -(SP)	
	014054	010600		MOV	SP, R0	
	014056	104415		TRAP	C#PNTX	
	014060	062706	000006	ADD	#6, SP	
2646	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	
2647	014106			PRINTX	#RBPCRA, (R5)+ ;PRINT "RESIDUAL BYTE COUNTER ="	
	014106	012546		MOV	(R5)+, -(SP)	
	014110	012746	014755	MOV	#RBPCRA, -(SP)	
	014114	012746	000002	MOV	#2, -(SP)	
	014120	010600		MOV	SP, R0	
	014122	104415		TRAP	C#PNTX	
	014124	062706	000006	ADD	#6, SP	
2648	014130			PRINTX	#XSOCN, (R5)+ ;PRINT "XSTAT0 CONTENTS ="	
	014130	012546		MOV	(R5)+, -(SP)	
	014132	012746	015022	MOV	#XSOCN, -(SP)	
	014136	012746	000002	MOV	#2, -(SP)	
	014142	010600		MOV	SP, R0	
	014144	104415		TRAP	C#PNTX	
	014146	062706	000006	ADD	#6, SP	
2649	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	
2650	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	
2651	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	
2652	014240	022737	000001	002134	CMP	#1, TRANSTST ;CHECK SOFTWARE P-TABLE
2653	014246	001402			BEQ	40\$;DO DUMP
2654	014250	000137	014360		JMP	50\$;DON'T DO THE DUMP
2655	014254			40\$:	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

```

2656 014274 012737 000010 002246      MOV      #8.,RAMSIZ      ;RAM FIELD IS 8 BYTES LONG
2657 014302 012737 000020 010640      MOV      #20,RAMHLD     ;FIELD STARTS AT 20 OCTAL (10 HEX)
2658 014310 004737 010456              JSR      PC,RAMER       ;READ AND PRINT THEM
2659 014314 012737 000040 010640      MOV      #40,RAMHLD     ;FIELD STARTS AT 40 OCTAL (20 HEX)
2660 014322 004737 010456              JSR      PC,RAMER       ;READ AND PRINT THEM
2661 014326 012737 000060 010640      MOV      #60,RAMHLD     ;FIELD STARTS AT 60 OCTAL (30 HEX)
2662 014334 004737 010456              JSR      PC,RAMER       ;READ AND PRINT THEM
2663 014340 012737 000020 002246      MOV      #16.,RAMSIZ    ;RAM FIELD IS SIXTEEN BYTES LONG
2664 014346 012737 000100 010640      MOV      #100,RAMHLD    ;FIELD STARTS AT 100 OCTAL (40 HEX)
2665 014354 004737 010456              JSR      PC,RAMER       ;READ AND PRINT THEM
2666 014360 000207              50$:   RTS      PC      ;RETURN
2667 014362      045      116      045  RAMFHR: .ASCIZ  'N%A ***** SPECIAL CONTROLLER RAM MEMORY DUMP *****'
2668 014460      045      116      045  MESBFN: .ASCIZ  'N%A MESSAGE BUFFER CONTENTS PROBABLY NOT VALID'
2669 014540      045      116      045  PROASC: .ASCIZ  'N%A Message Buffer Address = #01#05'
2670 014605      045      116      045  PR1ASC: .ASCIZ  'N%A Message Buffer Contents:'
2671
2672 014643      045      116      045  MESHEA: .ASCIZ  'N%A Message Buffer Header      = #06'
2673 014710      045      116      045  DATAFL: .ASCIZ  'N%A Data Field Length      = #06'
2674 014755      045      116      045  RBPCRA: .ASCIZ  'N%A Residual Byte Counter    = #06'
2675 015022      045      116      045  XSOCON: .ASCIZ  'N%A XSTAT0 Contents      = #06'
2676 015067      045      116      045  XS1CON: .ASCIZ  'N%A XSTAT1 Contents      = #06'
2677 015134      045      116      045  XS2CON: .ASCIZ  'N%A XSTAT2 Contents      = #06'
2678 015201      045      116      045  XS3CON: .ASCIZ  'N%A XSTAT3 Contents      = #06'
2679              .EVEN
  
```

```

2681          .SBTTL PRMSGEXP - PRINT EXPD/RCV MESSAGE BUFFERS
2682          ;+[B
2683          ;
2684          ;ROUTINE TO PRINT EXPECTED AND RECEIVED MESSAGE BUFFERS
2685          ;
2686          ;      RO      - NUMBER OF WORDS IN BUFFER
2687          ;
2688          ;IMPLICIT INPUTS:
2689          ;
2690          ;      EXPMSG  - EXPECTED MESSAGE BUFFER
2691          ;      RECMSG  - RECEIVED MESSAGE BUFFER
2692          ;      RCVHIADD- RECEIVED MESSAGE BUFFER HIGH ORDER ADDRESS
2693          ;      RCVLOADD- RECEIVED MESSAGE BUFFER LOW ORDER ADDRESS
2694          ;
2695          PRMSGEXP::
2696          SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
2697          MOV      RO,R5   ;SAVE NUMBER OF WORDS
2698          MOV      RCVLOADD,RO ;GET RECV LOW ADDRESS
2699          MOV      RO,R4   ;COPY LOW ADDRESS
2700          MOV      RCVHIADD,R1 ;GET RECV HIGH ADDRESS
2701          ROL      RO      ;SHIFT BIT15 TO C BIT
2702          ROL      R1      ;SHIFT TO HIGH ORDER FOR PRINTOUT
2703          PRINTX   #PRMSG0,R1,R4 ;PRINT MESSAGE BUFFER ADDRESS
2704          MOV      R4,-(SP)
2705          MOV      R1,-(SP)
2706          MOV      #PRMSG0,-(SP)
2707          MOV      #3,-(SP)
2708          MOV      SP,RO
2709          TRAP     C#PNTX
2710          ADD     #10,SP
2711          PRINTX   #PRMSG1          ;PRINT HEADER FOR CONTENTS
2712          MOV      #PRMSG1,-(SP)
2713          MOV      #1,-(SP)
2714          MOV      SP,RO
2715          TRAP     C#PNTX
2716          ADD     #4,SP
2717          CLR      R4          ;NUMBER OF THE CURRENT WORD
2718          MOV      #EXPMSG,R1   ;GET EXPD BUFFER ADDRESS
2719          MOV      #RECMSG,R2   ;GET RECV BUFFER ADDRESS
2720          MOV      (R1),RO      ;GET EXPD
2721          MOV      (R2),R3      ;GET RECV
2722          XOR      RO,R3        ;XOR EXPD/RCV
2723          PRINTX   #PRMSG2,R4,(R1)+,(R2)+,R3
2724          MOV      R3,-(SP)
2725          MOV      (R2)+,-(SP)
2726          MOV      (R1)+,-(SP)
2727          MOV      R4,-(SP)
2728          MOV      #PRMSG2,-(SP)
2729          MOV      #5,-(SP)
2730          MOV      SP,RO
2731          TRAP     C#PNTX
2732          ADD     #14,SP
2733          INC      R4          ;NUMBER OF THE NEXT
2734          CMP      R4,R5      ;DONE ALL YET?
2735          BGE     50$         ;BR IF YES
2736          BR      20$         ;DO ANOTHER
2737          BR      50$         ;RETURN
2738          RTS      PC
    
```

2717
2718 015426 045 116 045 PRMSG0: .ASCIZ 'N#A Message Buffer Address = #01#05'
2719 015473 045 116 045 PRMSG1: .ASCIZ 'N#A Message Buffer Contents:'
2720 015531 045 116 045 PRMSG2: .ASCIZ 'N#A WORD #D2#A EXPD: #06#A RECV: #06#A XOR: #06'
2721 .EVEN
2722

```

2724          .SBTTL PRBYTEXP - PRINT ERROR BYTES IN EXP/REC MESSAGE BUFFER
2725          ;+
2726          ;
2727          ;ROUTINE TO PRINT ERROR BYTES IN MESSAGE BUFFERS
2728          ; ONLY THE FIRST 8 ERRORS ENCOUNTERED ARE PRINTED DUE TO SCREEN SPACE
2729          ;
2730          ; RO - NUMBER OF BYTES IN BUFFER
2731          ;
2732          ;IMPLICIT INPUTS:
2733          ;
2734          ; EXPMSG - EXPECTED MESSAGE BUFFER
2735          ; RECMMSG - RECEIVED MESSAGE BUFFER
2736          ;-
2737 015616 PRBYTEXP::
2738 015616 SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
2739 015622 010005 MOV RO,R5          ;SAVE NUMBER OF BYTES
2740 015624 005037 002264 CLR PRMNO          ;INIT ERROR COUNT
2741 015630 005004 CLR R4          ;NUMBER OF THE CURRENT BYTE
2742 015632 012701 002266 MOV #EXPMSG,R1      ;GET EXPD BUFFER ADDRESS
2743 015636 012702 002432 MOV #RECMMSG,R2    ;GET RECV BUFFER ADDRESS
2744 015642 111100 20$: MOVB (R1),R0      ;GET EXPD BYTE
2745 015644 042700 177400 BIC #C<377>,R0    ;CLEAR UPPER BYTE
2746 015650 110037 016164 MOVB RO,PRBEXP     ;SAVE FOR ERROR REPORT
2747 015654 111203 MOVB (R2),R3      ;GET RECV BYTE
2748 015656 042703 177400 BIC #C<377>,R3    ;CLEAR UPPER BYTE
2749 015662 110337 016166 MOVB R3,PRBREC     ;FOR ERROR REPORT
2750 015666 XOR R0,R3          ;XOR EXPD/RECV
2751 015676 122122 CMPB (R1)+,(R2)+   ;EXPD = RECV?
2752 015700 001431 BEQ 30$          ;BR IF YES
2753 015702 005237 002264 INC PRMNO          ;UPDATE ERROR COUNT
2754 015706 023727 002264 000010 CMP PRMNO,#8      ;PRINTED 8?
2755 015714 101023 BHI 30$          ;BR IF YES
2756 015716 27$: PRINTX #PRBMSG,R4,PRBEXP,PRBREC,R3
          015716 010346 MOV R3,-(SP)
          015720 013746 016166 MOV PRBREC,-(SP)
          015724 013746 016164 MOV PRBEXP,-(SP)
          015730 010446 MOV R4,-(SP)
          015732 012746 016032 MOV #PRBMSG,-(SP)
          015736 012746 000005 MOV #5,-(SP)
          015742 010600 MOV SP,R0
          015744 104415 TRAP C$PNTX
          015746 062706 000014 ADD #14,SP
2757 015752 FORCEEXIT 50$ ;@@D
2758 015762 000404 BR 35$ ;@@D
2759 015764 30$: FORCERROR 27$,NOTSSR ;@@D
2760 015764 35$: ;@@D
2761 015774 INC R4          ;NUMBER OF THE NEXT
2762 015774 005204 CMP R4,R5          ;DONE ALL YET?
2763 015776 020405 BGE 50$          ;BR IF YES
2764 016000 002001 BR 20$          ;DO ANOTHER
2765 016002 000717 50$: PRINTX #PRBTOT,PRMNO ;PRINT TOTAL ERROR COUNT
          016004 013746 002264 MOV PRMNO,-(SP)
          016010 012746 016117 MOV #PRBTOT,-(SP)
          016014 012746 000002 MOV #2,-(SP)
          016020 010600 MOV SP,R0
          016022 104415 TRAP C$PNTX

```



```
2776 .SBTTL EXPREC - PRINT EXPD/RECV WORD DATA
2777 ;+
2778 ;
2779 ;PRINT ROUTINE TO DISPLAY EXPD/RECV DATA
2780 ;
2781 ;INPUTS:
2782 ;
2783 ; R1 RECEIVED DATA
2784 ; R2 EXPECTED DATA
2785 ;
2786 ;-
2787
2788 016170 BGNMSG EXPREC
016170 EXPREC::
2789 016170 004737 007314 JSR PC,PRIXOR ;PRINT THE DATA
2790 016174 ENDMSG
016174 L10017:
016174 104423 TRAP C$MSG
2791
2792
```

```
2794 .SBTTL EXPBREC - PRINT EXPD/RECV BYTE DATA
2795 ;+
2796 ;
2797 ;PRINT ROUTINE TO DISPLAY BYTE EXPD/RECV DATA
2798 ;
2799 ;
2800 ;INPUTS:
2801 ;
2802 ; R1 RECEIVED DATA BYTE
2803 ; R2 EXPECTED DATA BYTE
2804 ;
2805 ;-
2806
2807 016176 BGNMSG EXPBREC
016176 EXPBREC::
2808 016176 004737 007164 JSR PC,PRIBXOR ;PRINT THE DATA
2809 016202 ENDMSG
016202 L10020:
016202 104423 TRAP C$MSG
```

```
2810 .SBTTL RAMERR - PRINT RAM AND PACKET DATA
2811 ;+
2812 ;
2813 ;PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
2814 ;
2815 ;
2816 ;INPUTS:
2817 ;
2818 ; R4 POINTER TO COMMAND PACKET
2819 ;
2820 ;
2821 ;IMPLICIT INPUTS:
2822 ;
2823 ; RAMDATA DATA AS READ FROM THE RAM
2824 ; RAMSIZ NUMBER OF BYTES IN PACKET
2825 ; IF RAMSIZ=0 THEN DEFAULT TO 8.
2826 ;
2827 ;IMPLICIT OUTPUTS:
2828 ;
2829 ; RAMSIZ SET TO 0
2830 ;-
2831
2832 BGNMSG RAMERR
2833 RAMERR::
2834 016204 JSR PC,PRAMPKT ;PRINT RAM/PACKET DATA
016204 ENDMSG
2835 016204 004737 013450 L10021:
2836 016210 TRAP C$MSG
016210 104423
```

```
2837 .SBTTL RAMTADD - PRINT TEST ADDRESS, RAM AND PACKET DATA
2838 ;+
2839 ;
2840 ;PRINT ROUTINE TO DISPLAY RAM/PACKET DATA
2841 ;
2842 ;
2843 ;INPUTS:
2844 ;
```

```

2845      ;
2846      ;       R4       POINTER TO COMMAND PACKET
2847      ;
2848      ;IMPLICIT INPUTS:
2849      ;
2850      ;       RAMDATA   DATA AS READ FROM THE RAM
2851      ;       RAMSIZ   NUMBER OF BYTES IN PACKET
2852      ;                   IF RAMSIZ=0 THEN DEFAULT TO 8.
2853      ;       ERRHI    HIGH ORDER TEST ADDRESS
2854      ;       ERRLO    LOW ORDER TEST ADDRESS
2855      ;
2856      ;IMPLICIT OUTPUTS:
2857      ;
2858      ;       RAMSIZ   SET TO 0
2859      ;
2860      ;
2861 016212      BGNMSG   RAMTADD
2862 016212      004737   007646   RAMTADD::
2863 016216      004737   013450   JSR     PC,PRITADD   ;PRINT TEST ADDRESS
2864 016222      016222   104423   JSR     PC,PRAMPKT  ;PRINT RAM/PACKET DATA
2865      ;
2866      ;
2867      ;
2868      ;
2869      ;
2870      ;
2871      ;
2872      ;
2873      ;
2874      ;
2875      ;
2876      ;
2877      ;
2878      ;
2879 016224      016224   104423   BGNMSG   RAMEXP
2880 016224      042701   177400   RAMEXP::
2881 016230      042702   177400   BIC     #+C<377>,R1   ;SAVE EXPD RAM DATA BYTE
2882 016234      004737   007440   BIC     #+C<377>,R2   ;SAVE EXPD RAM DATA BYTE
2883 016240      004737   007314   JSR     PC,PRIRAM    ;PRINT THE RAM ADDRESS
2884 016244      016244   104423   JSR     PC,PRIXOR    ;PRINT THE DATA
2885      ;
2886      ;
2887      ;
2888      ;
2889      ;
2890      ;
2891      ;
2892      ;
2893      ;
2894      ;
2895      ;
    
```

```

2896
2897
2898 016246          BGNMSG  TIMEXP
      016246          TIMEXP::
2899 016246          PRINTX  #TIMSGO      ;PRINT HEADER
      016246 012746 016274      MOV      #TIMSGO,-(SP)
      016252 012746 000001      MOV      #1,-(SP)
      016256 010600          MOV      SP,R0
      016260 104415          TRAP     C$PNTX
      016262 062706 000004      ADD      #4,SP
2900 016266 004737 007314      JSR      PC,PRIXOR      ;PRINT THE DATA
2901 016272          ENDMSG
      016272          L10024:
      016272 104423          TRAP     C$MSG
2902
2903
2904 016274          045      116      045  TIMSGO: .ASCIZ  '%N%A TIMER A STATUS IS IN BIT 3%N%A TIMER B STATUS IS IN BIT 2'
2905          .EVEN
    
```

```

2907                .SBTTL  BADSSR - PRINT TSSR ERRORS ON DATA TRANSFERS
2908
2909                ;+
2910                ;
2911                ;PRINT ROUTINE FOR TSSR ERRORS ON DATA TRANSFERS
2912                ;
2913                ;INPUTS:
2914                ;
2915                ;      R1      CONTENTS OF TSSR
2916                ;      R2      DATA WRITTEN (8 BITS)
2917                ;
2918                ;-
2919
2920 016374          BGNMSG  BADSSR
2921 016374          BADSSR::
2922 016374 010246   MOV      R2,-(SP)          ;SAVE DATA TRANSFERRED
2923 016376 042702 177400 BIC      #177400,R2          ;GET JUST ONE BYTE
2924 016402          PRINTB #XFERASC,R2
2925 016404 010246   MOV      R2,-(SP)
2926 016410 012746 016434 MOV      #XFERASC,-(SP)
2927 016414 010600   MOV      #2,-(SP)
2928 016416 104414   MOV      SP,R0
2929 016420 062706 000006 TRAP    C:PNTB
2930 016424 012602   ADD      #6,SP
2931 016426 004737 005264 MOV      (SP)+,R2          ;RESTORE R2
2932 016432          JSR      PC,PRITSSR        ;DECODE TSSR CONTENTS
2933 016432          ENDMSG
2934 016432          L10025:
2935 016432 104423   TRAP    C:MSG
2936 016434 045      116    045  XFERASC: .ASCIZ  '%N%A Data Transferred = %03'
2937
2938

```

```

2930          .SBTTL GLOBAL SUBROUTINES SECTION
2931          ;++
2932          ; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
2933          ; THAT ARE USED IN MORE THAN ONE TEST.
2934          ;--
2935          .SBTTL SOFINIT - SOFT INITIALIZE OF CONTROLLER
2936
2937          ;+
2938          ;
2939          ;ROUTINE TO DO A SOFT INITIALIZE OF THE CONTROLLER
2940          ;BY WRITING INTO THE TSSR REGISTER. AFTER THE INIT,
2941          ;THE TSSR REGISTER IS TESTED FOR ERRORS. ANY ERRORS
2942          ;DETECTED SHOULD BE TREATED AS DEVICE FATAL ERRORS.
2943          ;
2944          ;INPUTS:
2945          ;
2946          ;      R5      ADDRESS OF FIRST REGISTER
2947          ;
2948          ;OUTPUTS:
2949          ;
2950          ;      R0      CONTENTS OF TSSR, IF ERROR
2951          ;      CARRY   SET IF INIT WAS OKAY
2952          ;              CLEAR IF FATAL ERROR
2953          ;
2954          ;CALLING SEQUENCE:
2955          ;
2956          ;      MOV      #ADDRESS,R5
2957          ;      JSR      PC,SOFINIT
2958          ;      BCS     CONTINUE
2959          ;      ERRDF                    ;REPORT FATAL ERROR
2960          ;
2961          ;-
2962
2963 016470      SOFINIT::
2964 016470          SAVREG
2965 016474 012765 000000 000000          MOV      #0,TSSR(R5)      ; SAVE THE REGISTERS
2966 016502 004737 016744          JSR      PC,WAITF        ; DO THE INIT.
2967 016506 016500 000000          MOV      TSSR(R5),R0    ; WAIT FOR SSR
2968 016512 010004          MOV      R0,R4          ;GET THE TSSR REGISTER
2969 016514 042704 176277          BIC     #+C<HIADDR!OFL>,R4 ;START SETUP OF EXPECTED TSSR
2970 016520 052704 002200          BIS     #SSR!NBA,R4   ;CLEAR OUT UNUSED BITS
2971 016524 020400          CMP     R4,R0          ;R4 HAS EXPECTED CONTENTS
2972 016526 001402          BEQ     5$            ;ONLY EXPECTED BITS SET ?
2973 016530 000241          CLC                    ;BRANCH IF OKAY
2974 016532 000401          BR     10$           ;CLEAR THE CARRY FOR ERROR
2975 016534 000261          SEC                    ;GO TO EXIT
2976 016536 000207          RTS     PC          ;SET THE CARRY BIT
2976 016536 000207          RTS     PC          ;RETURN TO CALLER

```

```

2978          .SBTTL  CHKAMB - CHECK TSSR FOR AMBIGUITY
2979
2980          ;+
2981          ;
2982          ;THIS ROUTINE TESTS THE CONTENTS OF THE TSSR REGISTER
2983          ;FOR AMBIGUITY
2984          ;
2985          ;INPUT:
2986          ;
2987          ;      RO      CONTENTS OF TSSR
2988          ;
2989          ;OUTPUT:
2990          ;
2991          ;      RO      CONTENTS OF TSSR
2992          ;
2993          ;      CARRY   SET - NO AMBIGUITY
2994          ;              CLR - AMBIGUOUS CONTENTS
2995          ;
2996          ;-
2997
2998 016540     CHKAMB:
2999 016540     SAVREG          ;SAVE THE GENERAL REGISTERS
3000 016544     010004     MOV      RO,R4          ;CONTENTS OF TSSR
3001 016546     032700     100000     BIT      #SC,RO          ;IS BIT 15 SET ?
3002 016552     001004     BNE      5$          ;BRANCH IF YES
3003 016554     032700     174077     BIT      #+C<NBA!OFL!SSR!HIADDR>,RO ;ANY OTHER BITS SET ?
3004 016560     001023     BNE      40$          ;MUST BE AN ERROR
3005 016562     000424     BR       45$          ;RETURN WITH SUCCESS
3006 016564     032700     000200     5$:  BIT      #SSR,RO          ;IS READY BIT SET ?
3007 016570     001011     BNE      10$          ;BRANCH IF READY BIT IS SET.
3008 016572     032700     000040     BIT      #BITS,RO          ;IS FATAL ERROR BIT SET ?
3009 016576     001414     BEQ      40$          ;ERROR IF NOT
3010 016600     042704     177761     BIC      #+CTERCLS,R4      ;CLEAR ALL BUT TERMINATION CODE
3011 016604     020427     000016     CMP      R4,#16          ;ALL THREE BITS MUST BE SET
3012 016610     001007     BNE      40$          ;ERROR IF NOT SET
3013 016612     000410     BR       45$          ;OK IF ALL ARE SET
3014 016614     032700     000040     10$: BIT      #BITS,RO          ;IS FATAL ERROR BIT SET ?
3015 016620     001405     BEQ      45$          ;ERROR IF BIT IS SET WITH SSR
3016 016622     032700     000006     BIT      #BIT2!BIT1,RO    ;IS THIS A FUNCTION REJECT
3017 016626     001002     BNE      45$          ;BR, IF TSSR IS OK
3018 016630     000241     40$:  CLC          ;AMBIGUOUS CONTENTS
3019 016632     000401     BR       50$
3020 016634     000261     45$:  SEC          ;SHOW SUCCESS - NO AMBIGUITY
3021 016636     000207     50$:  RTS      PC          ;RETURN TO CALLER
3022

```

```

3024          .SBTTL ENAINT,DSBINT - ENABLE/DISABLE INTERRUPTS
3025          ;
3026          ; DEFAULT DISPLAY INTERRUPT HANDLERS.
3027          ; IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
3028          ; OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
3029          ;
3030          ;
3031          ; BIT DEFINITIONS FOR "INTMASK" AND "INTFLAG" BYTES:
3032          ;
3033          000200          IOKCKIN=BIT7          ; DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
3034          000001          IOKSTP=BIT0          ; EXPECT "STOP" INTERRUPT.
3035          ;
3036          ;INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
3037 016640          000          INTMASK: .BYTE 0
3038          ;INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
3039 016641          000          INTFLAG: .BYTE 0
3040          ;
3041          ;SAVED INTERRUPT VECTOR:
3042 016642          000000          INTVEC: .WORD 0
3043          ;SAVE CPU PC
3044 016644          000000          INTCP: .WORD 0
3045          ;
3046          ;SUBROUTINE TO ENABLE INTERRUPTS:
3047 016646          010046          ENAINT: MOV      RO,-(SP)          ;SAVE R0
3048 016650          013700          002156          MOV      IVEC,RO          ;GET POINTER TO VECTORS
3049 016654          012720          016712          MOV      #INTR,(RO)+          ;SET UP INTERRUPT VECTOR
3050 016660          012720          000340          MOV      #PRI07,(RO)+
3051 016664          012600          MOV      (SP)+,RO          ;RESTORE R0
3052 016666          011646          MOV      (SP),-(SP)
3053 016670          012766          000000          000002          MOV      #0,2(SP)          ;SET CPU TO LEVEL 0
3054 016676          000002          RTI
3055          ;
3056          ;SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 7)
3057 016700          011646          DSBINT: MOV      (SP),-(SP)
3058 016702          012766          000340          000002          MOV      #PRI07,2(SP)
3059 016710          000002          RTI
3060

```

```
3062 .SBTTL INTR - INTERRUPT HANDLERS
3063
3064 016712 BGNSRV INTR ;DEFINE INTERRUPT ENTRY
      016712 INTR::
3065 016712 012737 000001 002172 MOV #1,INTRECV ;SET FLAG TO SHOW INTERRUPT RECEIVED
3066 016720 105037 016641 CLRB INTFLAG ;CLEAR FLAG TO SAY WE GOT INTERRUPT
3067 016724 132737 000001 016640 BITB #IOKSTP,INTMASK ;EXPECTING STOP INTERRUPT?
3068 016732 001003 BNE 1$ ;BR IF YES
3069 016734 152737 000001 016641 BISB #IOKSTP,INTFLAG ;NO. SET THE ERROR FLAG.
3070
3071 ;SAVE REGISTERS, MSG BUFFER, ETC.
3072 016742 1$:
3073 016742 ENDSRV
      016742 L10026:
      016742 000002 RTI
3074
3075
```

```

3077          .SBTTL  WAITF  - WAIT FOR SUBSYSTEM READY
3078
3079          ; SUBROUTINE TO WAIT FOR THE SUBSYSTEM READY FLAG
3080
3081          ; INPUTS:
3082
3083          ;     R5      ADDRESS OF FIRST DEVICE REGISTER
3084
3085          ; OUTPUTS:
3086
3087          ;     R0      CONTENTS OF LAST TSSR READ
3088          ;     CARRY   SET - READY BIT SET
3089          ;             CLR - TIMEOUT WAITING FOR READY
3090
3091          WAITF:: BREAK          ; DO A SUPVSR BREAK FIRST.
3092          016744 104422          TRAP      C$BRK
3093          016746 012746 010000  MOV      #10000,-(SP) ;BIG MSEC TIMER
3094          016752          DELAY      1          ;DELAY 100US
3095          016752 012727 000001  MOV      #1,(PC)+
3096          016756 000000          .WORD      0
3097          016760 013727 002116  MOV      L$DLY,(PC)+
3098          016764 000000          .WORD      0
3099          016766 005367 177772  DEC      -6(PC)
3100          016772 001375          BNE      -.4
3101          016774 005367 177756  DEC      -22(PC)
3102          017000 001367          BNE      -.20
3103          017002 016500 000000  2$: MOV      TSSR(R5),R0 ;READ THE TSSR REGISTER
3104          017006 105700          TSTB     R0          ;TEST FOR READY BIT SET
3105
3106          017010 100420          BMI      3$          ; EXIT ON STOP FLAG.
3107          017012          DELAY      1          ; WAIT 100 USEC
3108          017012 012727 000001  MOV      #1,(PC)+
3109          017016 000000          .WORD      0
3110          017020 013727 002116  MOV      L$DLY,(PC)+
3111          017024 000000          .WORD      0
3112          017026 005367 177772  DEC      -6(PC)
3113          017032 001375          BNE      -.4
3114          017034 005367 177756  DEC      -22(PC)
3115          017040 001367          BNE      -.20
3116          017042 005316          DEC      (SP)        ;REDUCE DELAY COUNT
3117          017044 001356          BNE      2$          ;RETRY UNTIL TIMER EXPIRES
3118          017046 000241          CLC
3119          017050 000401          BR       4$          ; C = 0, CONTROLLER STILL RUNNING...
3120          017052 000261          3$: SEC          ;...OR HUNG-UP AFTER 300 MSEC.
3121          017054 005326          4$: DEC      (SP)+   ; C = 1, CONTROLLER IS STOPPED.
3122          017056 000207          RTS      PC         ;RESTORE STACK WITHOUT CHANGING CARRY BIT
    
```

```

3107          .SBTTL  CHKTSSR - CHECK TSSR FOR READY
3108
3109          ;*
3110          ;
3111          ;THIS ROUTINE WAITS FOR READY IN THE TSSR
3112          ;AND TESTS FOR AMBIGUOUS BIT SETTINGS IN TSSR.
3113          ;
3114          ;INPUT:
3115          ;
3116          ;      R5      ADDRESS OF CSR REGISTERS
3117          ;
3118          ;OUTPUT:
3119          ;
3120          ;      R0      CONTENTS OF TSSR
3121          ;      CARRY   SET - OKAY
3122          ;              CLR - NOT READY AMBIGUOUS, OR SC SET
3123          ;
3124          ;-
3125
3126 017060    CHKTSSR:
3127 017060    004737 016744    JSR    PC,WAITF      ;WAIT FOR READY
3128 017064    103014          BCC    20$          ;BRANCH IF TIME OUT
3129 017066    004737 016540    JSR    PC,CHKAMB     ;TSSR AMBIGOUS?
3130 017072    103006          BCC    10$          ;BR IF YES
3131 017074    032700 100000    BIT    #SC,R0       ;SPECIAL CONDITION SET?
3132 017100    001405          BEQ    15$          ;BR IF NO
3133 017102    032700 074000    BIT    #<SCE!BIE!RMR!NXM>,R0 ;ANY ERROR BITS SET?
3134 017106    001402          BEQ    15$          ;BR IF NO
3135 017110    000241          10$:  CLC           ;SET FAILURE
3136 017112    000401          BR     20$          ;
3137 017114    000261          15$:  SEC           ;SET SUCCESS
3138 017116    000207          20$:  RTS    PC     ;RETURN TO CALLER
    
```

```

3140          .SBTTL XNXM - CHECK FOR NONEXISTENT MEMORY
3141          ;+
3142          ; ROUTINE TO TEST FOR A NEXM IN THE RANGE (R1) THRU (R2).
3143          ; ON RETURN, IF "C" = 1, (R1) = NEXM ADDRESS.
3144          ;           "C" = 0, ALL ADDRESSES OK.
3145          ;
3146          ;CALL:  MOV ADR1,R1
3147          ;         MOV ADR2,R2
3148          ;         JSR PC,NXM
3149          ;         RETURN
3150          ;           ;TEST "C" AND PROCEED.
3151 017120 012737 017152 000004 XNXM:  MOV #2$,@#4           ; SET BUSERR VECTOR.
3152 017126 012737 000200 000006      MOV #PRI04,@#6
3153 017134 005003              CLR R3           ;FLAG.
3154 017136 005711 1$:  TST (R1)           ;TEST THE ADDRESS(ES).
3155          ;           ;IF ANY TRAP, CONTINUE AT 2$.
3156 017140 020102              CMP R1,R2         ;OTHERWISE, CONTINUE HERE.
3157 017142 001407              BEQ 3$           ;BR IF FINISHED (NO NEXM'S).
3158 017144 062701 000002      ADD #2,R1         ;SET NEXT ADDRESS...
3159 017150 000772              BR 1$           ;...AND CONTINUE.
3160          ;
3161 017152 005103 2$:  COM R3           ;GOT ONE, SET FLAG...
3162 017154 012716 017162      MOV #3$, (SP)
3163 017160 000002              RTI
3164 017162 3$:  CLRVEC #4           ;...AND DISMISS INTERRUPT...
3165 017162 012700 000004      MOV #4,R0         ;...AND GIVE BACK THE VECTOR.
3166 017166 104436              TRAP C$CVEC
3167 017170 005703              TST R3           ;DID WE CATCH ONE ??
3168 017172 001401              BEQ .+4         ;NO. "C" = 0, SKIP NEXT.
3169 017174 000261              SEC           ;YES. "C" = 1, (R1) = NEXM ADDR.
3170 017176 000207              RTS PC
3171
3172
3173          .SBTTL TSTLOOP - CHECK ITERATION COUNT
3174          ;+
3175          ; SUBROUTINE TO EXECUTE TEST ITERATIONS.
3176          ; EXIT WITH "C" SET IF LOOPS ALLOWED AND LOOP COUNT NON-ZERO.
3177          ; LOOP COUNTER IS SET BY "BEGIN.TEST" MACRO.
3178          ;
3179          ; CALL:  LOOPTO ARG
3180          ;
3181 017200          TSTLOOP::
3182 017200 005737 002136      TST NOITS           ; ITERATIONS INHIBITED?
3183 017204 001006              BNE 1$           ; YES.
3184 017206 005737 002152      TST QVP           ; NO.
3185 017212 100403              BMI 1$           ;LOOPS DISALLOWED IN QUICK PASS.
3186 017214 005337 002164      DEC LOOPCNT       ; BUMP LOOP COUNTER.
3187 017220 001002              BNE 2$
3188 017222 000241 1$:  CLC           ;LOOP DISALLOWED, OR DONE.
3189 017224 000401              BR 3$
3190 017226 000261 2$:  SEC           ;LOOP ENABLED.
3191 017230 000207 3$:  RTS PC
  
```

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
3219
3220

```

.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
;
;-

```

3221 017232
3222 017232 010046
3223 017234 005037 003106
3224 017240 005037 017500
3225 017244 005037 005232
3226 017250 105037 016640
3227 017254 013700 002150
3228 017260 006300
3229 017262 005737 003062
3230 017266 001430
3231 017270 100010
3232 017272 052760 160000 003130
3233 017300
017300 104455
017302 000001
017304 003636
017306 005176
3234 017310 000407
3235 017312 052760 160001 003130 3\$:
3236 017320
017320 104455
017322 000002
017324 004233
017326 000000
3237 017330 012737 177777 003060 2\$:
3238 017336
017336 013700 002150
017342 104451
3239 017344

```

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

```

	017344	104444				TRAP	C%DCLN		
3240	017346	000423				BR	5\$		
3241									
3242	017350			4\$:		RFLAGS	RO		; GET THE OPERATOR FLAGS.
	017350	104421				TRAP	C#RFLA		
3243	017352	032700	001000			BIT	#PNT,RO		; PRINT THE TEST NUMBERS?
3244	017356	001412				BEQ	1\$; BR IF NO
3245	017360	011600				MOV	(SP),RO		;GET THE ID MESSAGE
3246	017362					PRINTF	#TNAM,RO		;DISPLAY THE TEST ID
	017362	010046				MOV	RO,-(SP)		
	017364	012746	017426			MOV	#TNAM,-(SP)		
	017370	012746	000002			MOV	#2,-(SP)		
	017374	010600				MOV	SP,RO		
	017376	104417				TRAP	C#PNTF		
	017400	062706	000006			ADD	#6,SP		
3247	017404	005237	002162		1\$:	INC	TSTCNT		; BUMP TEST COUNTER.
3248	017410					SETPRI	IPRI		;PRIORITY THAT OF DEVICE
	017410	013700	002160			MOV	IPRI,RO		
	017414	104441				TRAP	C#SPRI		
3249	017416	005726			5\$:	TST	(SP)+		;FIX UP THE STACK
3250	017420	013705	002154			MOV	CSRADDR,R5		; ADDRESS OF TSV REGISTERS ON UNIBUS
3251	017424	000207				RTS	PC		
3252	017426	045	123	045	TNAM:	.ASCIZ	'#S#T#A Test'		
3253						.EVEN			

```

3255
3256
3257
3258
3259
3260 017442
      017442 104421
3261 017444 030027 020000
3262 017450 001412
3263 017452
      017452 013746 017500
      017456 012746 017502
      017462 012746 000002
      017466 010600
      017470 104417
      017472 062706 000006
3264 017476 000207
3265
3266 017500 000000
3267 017502      045      101      040
3268 017521      105      122      122
3269
3270
3271
3272
3273
3274
3275 017566 005237 017500
3276 017572 010046
3277 017574 013700 002150
3278 017600 006300
3279 017602 062700 003130
3280 017606 005210
3281 017610 032710 007777
3282 017614 001001
3283 017616 005310
3284 017620 012600
3285 017622 000207
3286
3287 017624 010046
3288 017626 013700 002150
3289 017632 006300
3290 017634 016000 003130
3291 017640 042700 170000
3292 017644 020037 002142
3293 017650 103004
3294 017652 023737 017500 002140
3295 017660 103417
3296 017662
      017662 104421
3297 017664 032700 000040
3298 017670 001013
3299 017672 012737 177777 003060
3300 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
    
```

```

3301 017706 000000          .WORD 0
      017710          DODU UNITN
      017710 013700 002150  MOV UNITN,R0
      017714 104451      TRAP C$DODU
3302 017716          DOCLN
      017716 104444      TRAP C$DCLN
3303 017720 012600 2$: MOV (SP)+,R0 ; RESTORE R0
3304 017722 000207      RTS PC ; RETURN TO CALLER
3305          .SBTTL FATCHK - INC FATAL ERRORS AND CHECK FOR LIMIT
3306          ;+
3307          ;
3308          ; CHECK FATAL COUNTER, AFTER INC, FOR MORE THAN 25
3309          ; ERRORS AND IF OVER CALL UNIT DROP ROUTINE
3310          ;
3311          ;-
3312 017724          FATCHK:
3313 017724          SAVREG
3314 017730 013701 002150  MOV UNITN,R1 ;BETTER SAVE THE REGISTERS
3315 017734 006301      ASL R1 ;PICK UP THE UNIT NUMBER
3316 017736 062761 000001 003130  ADD #1,ERTABL(R1) ;MAKE IT INTO A BYTE OFFSET
3317 017744 005237 002170      INC FATFLG ;ADD 1 TO THE PROPER UNIT'S ERROR COUNTER
3318 017750 023727 002170 000031  CMP FATFLG,#25. ;BUMP FATAL ERROR COUNTER
3319 017756 002406      BLT 9$ ;CHECK AGAINST 25
3320 017760          RFLAGS RO ;BR, IF LESS THAN 25 ERRORS
      017760 104421      TRAP C$RFLA ;READ THE FLAGS INTO RO
3321 017762 032700 040000  BIT #BIT14,R0 ;BR, IF LOOP ON ERROR IS SET
3322 017766 001002      BNE 9$ ;OTHERWISE NEVER BE ABLE TO SCOPE ETC.
3323 017770 004737 017776  JSR PC,CKDROP ;DROP UNIT IF ALLOWED
3324 017774 000207 9$: RTS PC ;RETURN ETC.
3325          ;
3326          ;
3327          ;

```

```
3329 .SBTTL CKDROP - CHECK IF UNIT SHOULD BE DROPPED
3330 ;+
3331 ; CHECK IF UNIT SHOULD BE DROPPED
3332 ;-
3333 CKDROP: MOV RO, -(SP)
3334 FORCERROR 1$, NOTSSR
3335 RFLAGS RO
020010 104421 TRAP C$RFLA
3336 020012 032700 000040 BIT #IDU, RO
3337 020016 001010 BNE 1$
3338 020020 011600 MOV (SP), RO
3339 020022 012737 177777 003060 MOV #-1, DUFLG
3340 020030 DODU UNITN
020030 013700 002150 MOV UNITN, RO
020034 104451 TRAP C$DODU
3341 020036 DOCLN ;ABORT THE PASS
020036 104444 TRAP C$DCLN
3342 020040 012600 1$: MOV (SP)+, RO
3343 020042 000207 RTS PC
3344
3345
3346
3347
```

```
3348 .SBTTL CONFIG - DETERMINE CONFIGURATION OF SYSTEM
3349 ;
3350 ; SUBROUTINE - DETERMINE CONFIGURATION OF TK-25 SYSTEM.
3351 ;
3352 CONFIG:
3353 020044 004737 016470 JSR PC, SOFINIT
3354 020050 000207 RTS PC
3355
3356
3357
```

```
3359 .SBTTL KTON,KTOFF - ENABLE/DISABLE MEMORY MANAGEMENT
3360 ;
3361 ; SUBROUTINE - ENABLE MEM MGT.
3362 ;
3363 020052 005737 003100 KTON: TST KTFLG ; GOT KT?
3364 020056 001403 BEQ 1$ ; NO.
3365 020060 012737 000001 177572 MOV #1,SRO ; YES. ENABLE KT11.
3366 020066 000207 1$: RTS PC
3367
3368
3369
3370 ;
3371 ; SUBROUTINE - DISABLE MEM MGT.
3372 ;
3373 020070 005737 003100 KTOFF: TST KTFLG ; GOT KT11?
3374 020074 001405 BEQ 1$ ; NO.
3375 020076 000240 NOP
3376 020100 000240 NOP
3377 020102 012737 000000 177572 MOV #0,SRO ; DISABLE KT.
3378 020110 000207 1$: RTS PC
3379
3380
```

3382
 3383
 3384
 3385
 3386
 3387
 3388
 3389
 3390
 3391
 3392
 3393
 3394
 3395
 3396
 3397
 3398
 3399
 3400
 3401 020112
 3402 020112
 3403 020116 005737 003100
 3404 020122 001433
 3405 020124 010102
 3406 000006
 3407
 3408
 3409
 3410 020156 042701 000177
 3411 020162 020137 003100
 3412 020166 103011
 3413 020170 010137 172354
 3414 020174 042702 160000
 3415 020200 062702 140000
 3416 020204 010200
 3417 020206 000261
 3418 020210 000401
 3419 020212 000241
 3420 020214 000207
 3421

.SBTTL SETMAP - SETUP PAR6 MAPPING

```

;+
;
; THIS ROUTINE SETS UP KERNEL PAR6 TP HANDLE
; AN 18 BIT ADDRESS. THE OFFSET INTO THE PAGE
; IS RETURNED BIASED TO PAR6.
;
; INPUTS:
;
;     R0      HIGH ORDER ADDRESS BITS
;     R1      LOW ORDER ADDRESS BITS
;
; OUTPUTS:
;
;     R0      OFFSET INTO BLOCK WITH PAR6 BIAS (I.E. THE ADDRESS)
;     CARRY   SET IF SUCCESS
;             CLR IF ERROR
;
;-
SETMAP:
    SAVREG                ;SAVE R1-R4 UNTIL NEXT RETURN
    TST    KTFLG          ;SYSTEM HAVE ABOVE 28K?
    BEQ    10$            ;BR IF NO
    MOV    R1,R2          ;SAVE LOW ORDER BITS
    .REPT  6
    ASR    R0              ;CONVERT WORD ADDRESS TO 32W BLOCKS
    ROR    R1              ;MAKE IT DOUBLE PRECISION
    .ENDR
    BIC    #177,R1        ;ALINE FOR LOWER 4K BOUNDARY
    CMP    R1,KTFLG      ;HIGHER THAN EXISTING MEMORY?
    BHIS   10$           ;BR IF YES
    MOV    R1,#KIPAR6    ;SETUP MAPPING REGISTER PAR6
    BIC    #160000,R2    ;SETUP DISPLACEMENT IN PAGE
    ADD    #140000,R2    ;ADD IN PAR6 BIAS
    MOV    R2,R0         ;RETURN IN R0
    SEC                    ;SET SUCCESS
    BR     15$           ;
    10$:  CLC              ;SET FAILURE
    15$:  RTS    PC      ;RETURN
  
```

```

3423 .SBTTL FILLMEM - FILL MEMORY WITH BACKGROUND PATTERN
3424 ;+
3425 ; FILL MEMORY WITH A BACKGROUND PATTERN
3426 ;
3427 ; INPUTS:
3428 ;
3429 ; RO = BACKGROUND PATTERN
3430 ; FREE = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
3431 ; KTFLG = SET TO HIGHEST MEMORY LOCATION IF > 28K.
3432 ;
3433 ; OUTPUTS:
3434 ;
3435 ; NONE
3436 ;-
3437 ;
3438 020216 FILLMEM:
3439 020216 SAVREG ;SAVE R1-R5 UNTIL NEXT RETURN
3440 020222 004737 020070 JSR PC,KTOFF ;DISABLE KT.
3441 020226 010003 MOV RO,R3 ;COPY TEST PATTERN
3442 020230 013701 003072 MOV FREE,R1 ;GET FIRST FREE LOCATION
3443 020234 013702 003074 MOV FRESIZ,R2 ;SIZE OF FREE SPACE BELOW 28K.
3444 020240 010321 10$: MOV R3,(R1)+ ;STORE A BACKGROUND WORD
3445 020242 005302 DEC R2 ;DONE ALL MEMORY IN FREE SPACE?
3446 020244 003375 BGT 10$ ;BR IF NO
3447 020246 005737 003100 TST KTFLG ; GOT KT?
3448 020252 001452 BEQ 55$ ; NO. GET OUT.
3449 020254 004737 020052 JSR PC,KTON ; YES. ENABLE KT.
3450 020260 005000 CLR RO ;HIGH ORDER ADDRESS START
3451 020262 013701 003104 MOV PST32W,R1 ;GET >28K START ADDRESS (IN 32W BLOCKS)
3452 000006 .REPT 6
3453 CLC ;CLEAR C BIT
3454 ROL R1 ;CONVERT BLOCKS TO WORDS
3455 ROL RO ;MAKE IT DOUBLE PRECISION
3456 .ENDR
3457 020332 004737 020112 JSR PC,SETMAP ;SETUP PAR6 MAPPING REGISTER
3458 020336 010320 30$: MOV R3,(R0)+ ;STORE TEST PATTERN IN >28K ADDRESS
3459 020340 020027 160000 CMP RO,#160000 ;END OF PAR6 MAPPING AREA?
3460 020344 103774 BLO 30$ ;BR IF NO
3461 020346 162700 020000 SUB #20000,R0 ;BACKUP INTO PAR6 MAPPING BEGIN
3462 020352 062737 000200 172354 ADD #200,#KIPAR6 ;POINT TO NEXT 4K BLOCK >28K.
3463 020360 023737 172354 003100 CMP #KIPAR6,KTFLG ;END OF MEMORY?
3464 020366 001402 BEQ 50$ ;BR IF YES
3465 020370 000137 020336 JMP 30$ ;KEEP GOING ON ETC.
3466 020374 004737 020070 50$: JSR PC,KTOFF ; DISABLE KT.
3467 020400 000207 55$: RTS PC
3468
3469
    
```

```

3471          .SBTTL  CMPMEM - COMPARE MEMORY TO BACKGROUND PATTERN
3472          ;*
3473          ; COMPARE MEMORY WITH A BACKGROUND PATTERN
3474          ;
3475          ; INPUTS:
3476          ;
3477          ;     RO = BACKGROUND PATTERN
3478          ;     FREE  = FIRST LOCATION AVAILABLE TO DIAGNOSTIC
3479          ;     KTFLG  = SET TO HIGHEST MEMORY LOCATION IF > 28K.
3480          ;
3481          ; OUTPUTS:
3482          ;
3483          ;     CARRY  - SET IF NO ERROR
3484          ;     CARRY  - CLR IF ERROR
3485          ;
3486          ; IMPLICIT OUTPUTS:
3487          ;
3488          ;     ERRHI  - ERROR HIGH ADDRESS
3489          ;     ERRLO  - ERROR LOW ADDRESS
3490          ;     EXPD   - EXPECTED DATA
3491          ;     RECV   - RECEIVED DATA
3492          ;-
3493          CMPMEM:
3494          SAVREG          ;SAVE R1-R5 UNTIL NEXT RETURN
3495          MOV            R0,R3          ;COPY TEST PATTERN
3496          JSR            PC,KTOFF      ;DISABLE KT.
3497          MOV            FREE,R1        ;GET FIRST FREE LOCATION
3498          MOV            FRESIZ,R2     ;SIZE OF FREE SPACE BELOW 28K.
3499          10$:          CMP            R3,(R1)  ;FREE SPACE LOCATION EQUAL TO EXPD?
3500          BEQ            15$          ;BR IF YES
3501          MOV            R1,ERRLO     ;SAVE ADDRESS IN ERROR
3502          CLR            ERRHI        ;NO HIGH ADDRESS
3503          MOV            R3,EXPD      ;SAVE EXPD FOR ERROR REPORT
3504          MOV            (R1),RECV    ;SAVE RECV FOR ERROR REPORT
3505          BR            50$          ;
3506          15$:          TST            (R1)+  ;POINT TO NEXT ADDRESS
3507          DEC            R2          ;DONE ALL MEMORY IN FREE SPACE?
3508          BGT            10$          ;BR IF NO
3509          TST            KTFLG        ; GOT KT?
3510          BEQ            55$          ; NO. GET OUT.
3511          JSR            PC,KTON      ; YES. ENABLE KT.
3512          CLR            R0          ;HIGH ORDER ADDRESS START
3513          MOV            PST32W,R1    ;GET >28K START ADDRESS (IN 32W BLOCKS)
3514          .REPT          6
3515          ROL            R1          ;CONVERT BLOCKS TO WORDS
3516          ROL            R0          ;MAKE IT DOUBLE PRECISION
3517          .ENDR
3518          BIC            #177,R1     ;ALINE 4K BOUNDARY
3519          MOV            R0,-(SP)     ;SAVE HIGH ORDER
3520          MOV            R1,-(SP)     ;SAVE LOW ORDER
3521          JSR            PC,SETMAP    ;SETUP PAR6 MAPPING REGISTER
3522          MOV            R0,R4        ;COPY ADDRESS BIASED TO PAR6
3523          MOV            (SP)+,R1    ;RESTORE LOW ORDER IN NON PAR6 FORMAT
3524          MOV            (SP)+,R0    ;RESTORE HIGH ORDER IN NON PAR6 FORMAT
3525          30$:          CMP            R3,(R4)  ;ABOVE 28K LOCATION EQUAL EXPD?
3526          BEQ            32$          ;BR IF YES
3527          MOV            R0,ERRHI    ;SAVE HIGH ORDER IN ERROR

```



```

3549          .SBTTL  REGSAV  - SAVE R1-R5 ON STACK
3550          ;*
3551          ;
3552          ;ROUTINE TO
3553          ;SAVE R1 THROUGH R5 ON THE STACK
3554          ;
3555          ;CALLING SEQUENCE:
3556          ;
3557          ;       JSR      R5,REGSAV
3558          ;
3559          ;THIS IS A COOROUTINE WHICH TRANSFER CONTROL BACK TO
3560          ;THE CALLING ROUTINE. AT THE END OF THE CALLING ROUTINE,
3561          ;THE RTS PC RETURNS CONTROL TO THIS ROUTINE TO RESTORE
3562          ;REGISTERS.
3563          ;
3564          ;THIS ROUTINE SHOULD ONLY BE CALLED FROM ROUTINES WHICH ARE
3565          ;CALLED VIA A JSR PC INSTRUCTION
3566          ;
3567          ;-
3568
3569          REGSAV:
3570          BREAK          ;LOOK FOR CNTL C
3571          TRAP          C$BRK
3572          MOV           R4,-(SP)
3573          MOV           R3,-(SP)
3574          MOV           R2,-(SP)
3575          MOV           R1,-(SP)
3576          MOV           R5,-(SP)
3577          MOV           10.(SP),R5
3578          JSR          PC,@(SP)+
3579          MOV           (SP)+,R1
3580          MOV           (SP)+,R2
3581          MOV           (SP)+,R3
3582          MOV           (SP)+,R4
3583          MOV           (SP)+,R5
3584          BREAK          ;LOOK FOR CNTL C
3585          TRAP          C$BRK
3586          RTS          PC
    
```

000012

```

3587          .SBTTL  GETPAT  - GET 8 BIT PATTERN FROM OPERATOR
3588          ;+
3589          ;
3590          ;ROUTINE TO REQUEST AN 8 BIT DATA PATTERN FROM THE OPERATOR
3591          ;
3592          ;INPUTS:
3593          ;
3594          ;      NONE.
3595          ;
3596          ;OUTPUTS:
3597          ;
3598          ;      RO      OCTAL NUMBER FROM THE OPERATOR
3599          ;
3600          ;CALLING SEQUENCE:
3601          ;
3602          ;      JSR      PC,GETPAT
3603          ;
3604          ;-
3605
3606 020722      GETPAT::
3607 020722          SAVREG          ;SAVE THE GENERAL REGISTERS
3608 020726      1$:      GMANID  DATASC,PATDAT,0,377,0,377,NO
3609          020726      104443      TRAP      C$GMAN
3610          020730      000406      BR        10000$
3611          020732      020756      .WORD    PATDAT
3612          020734      000022      .WORD    T$CODE
3613          020736      020760      .WORD    DATASC
3614          020740      000377      .WORD    377
3615          020742      000000      .WORD    T$LOLIM
3616          020744      000377      .WORD    T$HILIM
3617          020746      10000$:
3618          020746      103367      BNCOMPLETE 1$      ;RETRY IF ERROR
3619          020750      013700      BCC      1$
3620          020754      000207      MOV      PATDAT,RO      ;DATA PATTERN FROM OPERATOR
3621          020756          020756      RTS      PC          ;RETURN TO CALLER
3622
3623          ;+
3624          ;LOCAL DATA AREA
3625          ;-
3626
3627          020756      000000      PATDAT: .WORD    0          ;TEMPORARY STORAGE FOR DATA
3628          020760      105      116      124      DATASC: .ASCIZ  'ENTER DATA PATTERN'
3629          .EVEN
  
```

```

3621          .SBTTL  GETSEL  - ISSUE MENU AND GET OPERATOR RESPONSE
3622          ;+
3623          ;ROUTINE TO ISSUE A MENU AND GET
3624          ;THE OPERATOR'S RESPONSE.
3625          ;
3626          ;INPUTS:
3627          ;
3628          ;      RO      ADDRESS OF ASCIZ STRING OF MENU
3629          ;      R1      MAXIMUM ALLOWABLE OPERATOR RESPONSE
3630          ;
3631          ;OUTPUTS:
3632          ;
3633          ;      RO      NUMBER OF THE OPERATOR'S SELECTION
3634          ;-
3635 021004    GETSEL::
3636 021004          SAVREG          ;SAVE GENERAL REGISTERS
3637 021010    010002          MOV      RO,R2          ;SAVE THE MENU ADDRESS
3638 021012    010203          1$:  MOV      R2,R3          ;START OF MENU STRING
3639 021014    005713          2$:  TST      (R3)          ;END OF ASCII ?
3640 021016    001412          BEQ      3$          ;BRANCH IF ALL LINES DISPLAYED
3641 021020          PRINTF  #SELASC,(R3)+ ;DISPLAY THE MENU
3642 021020    012346          MOV      (R3)+,-(SP)
3643 021022    012746    021170          MOV      #SELASC,-(SP)
3644 021026    012746    000002          MOV      #2,-(SP)
3645 021032    010600          MOV      SP,RO
3646 021034    104417          TRAP    C$PNTF
3647 021036    062706    000006          ADD      #6,SP
3648 021042    000764          BR      2$
3649 021044          3$:  GMANID  MENASC,MENRES,D,-1,0,-1,NO
3650 021044    104443          TRAP    C$GMAN
3651 021046    000406          BR      10001$
3652 021050    021224          .WORD  MENRES
3653 021052    000042          .WORD  T$CODE
3654 021054    021175          .WORD  MENASC
3655 021056    177777          .WORD  -1
3656 021060    000000          .WORD  T$LOLIM
3657 021062    177777          .WORD  T$HILIM
3658 021064          10001$:
3659 021064          BNCOMPLETE  1$          ;RETRY IF ERROR
3660 021064    103352          BCC      1$
3661 021066    013700    021224          MOV      MENRES,RO          ;GET THE OPERATOR'S REPLY
3662 021072    020001          CMP      RO,R1          ;COMPARE TO MAXIMUM ALLOWED
3663 021074    101411          BLOS    5$          ;BRANCH IF OK
3664 021076          PRINTF  #MENERR          ;DISPLAY ERROR MESSAGE
3665 021076    012746    021122          MOV      #MENERR,-(SP)
3666 021102    012746    000001          MOV      #1,-(SP)
3667 021106    010600          MOV      SP,RO
3668 021110    104417          TRAP    C$PNTF
3669 021112    062706    000004          ADD      #4,SP
3670 021116    000735          BR      1$          ;RETRY
3671 021120    000207          5$:  RTS      PC          ;RETURN TO CALLER
3672 021122    045      116    045  MENERR: .ASCIZ  '%N%A *** Menu Selection Too Large ***'
3673 021170    045      116    045  SELASC: .ASCIZ  '%N%T'
3674 021175    105      156    164  MENASC: .ASCIZ  'Enter Menu Selection: '
3675 021224    000000          MENRES: .WORD  0
    
```

```

3657 .SBTTL CHKMAN - CHECK MANUAL INTERVENTION LEGALITY
3658
3659 ;+
3660 ;ROUTINE TO TEST FOR MANUAL INTERVENTION LEGALITY.
3661 ;
3662 ;INPUT:
3663 ;
3664 ; NONE.
3665 ;
3666 ;OUTPUT:
3667 ;
3668 ; CARRY 0 MANUAL INTERVENTION NOT ALLOWED
3669 ; 1 MANUAL INTERVENTION IS OK
3670 ;
3671 ;SIDE EFFECTS:
3672 ;
3673 ; A MESSAGE IS DISPLAYED WARNING THAT TEST IS
3674 ; NOT EXECUTED IF MANUAL INTERVENTION IS NOT
3675 ; ALLOWED.
3676 ;
3677 ;-
3678
3679 021226 CHKMAN::
3680 021226 SAVREG ;SAVE THE REGISTERS
3681 021232 MANUAL ;SEE IF MANUAL INTERVENTION OK
3681 021232 104450 TRAP C$MANI
3682 021234 BCOMPLETE 1$ ;BRANCH IF ALLOWED
3682 021234 103411 BCS 1$
3683 021236 PRINTF #NOMAN ;PRINT THE WARNING MESSAGE
3683 021236 012746 021262 MOV #NOMAN,-(SP)
3683 021242 012746 000001 MOV #1,-(SP)
3683 021246 010600 MOV SP,R0
3683 021250 104417 TRAP C$PNTF
3683 021252 062706 000004 ADD #4,SP
3684 021256 000241 CLC ;CLEAR CARRY FOR ERROR
3685 021260 000207 1$: RTS PC ;RETURN
3686
3687 021262 045 116 045 NOMAN: .ASCIZ '%N%A *** Manual Intervention not Allowed - Test Aborted ***'
3688 .even
  
```

```
3690          .SBTTL  ENVIRN  - SETUP FREE DIAGNOSTIC SPACE
3691          ;
3692          ; SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
3693          ;
3694 021356      ENVIRN: MEMORY  R0
                021356 104431  TRAP    C$MEM
3695 021360 010037 003072      MOV    R0,FREE      ; GET 1ST FREE ADDRESS...
3696 021364 062737 000002 003072  ADD    #2,FREE
3697 021372 011037 003074      MOV    (R0),FRESIZ  ;...AND WORD COUNT.
3698 021376 162737 000004 003074  SUB    #4,FRESIZ
3699 021404 013702 002012      MOV    L$UNIT,R2   ; GET NUMBER OF UNITS
3700 021410 162737 000007 003074 10$:  SUB    #7,FRESIZ   ; TAKE AWAY 7 WORDS PER UNIT
3701 021416 005302      DEC    R2
3702 021420 001373      BNE    10$
3703 021422 013700 003072      MOV    FREE,R0    ;GET FIRST FREE ADDRESS
3704 021426 063700 003074      ADD    FRESIZ,R0  ;POINT TO LAST FREE ADDRESS
3705 021432 162700 000002      SUB    #2,R0      ;BACKUP 1 WORD
3706 021436 010037 003076      MOV    R0,FREEHI  ;STORE LAST FREE ADDRESS
3707 021442 000207      RTS    PC        ;RETURN
3708
```

```

3710          .SBTTL  KTINIT  - SETUP KT11 MEMORY MANAGEMENT REGISTERS
3711          ;+
3712          ;
3713          ;ROUTINE TO INIT KT-11
3714          ;
3715          ;-
3716
3717 021444      KTINIT:
3718 021444 005037 003100      CLR      KTFLG          ; INIT >28K MEMORY FLAG
3719 021450 005037 003102      CLR      KTENABLE       ; INIT TEST >28K FLAG
3720 021454 023727 002120 001577  CMP      L$HIME,#1577    ; GOT ENOUGH MEMORY (>28K)?
3721 021462 101444          BLOS     9$              ; NO.
3722 021464 013700 000004      MOV      @#ERRVEC,RO     ; SAVE OLD ERR VEC PTR.
3723 021470 012737 021562 000004  MOV      #2$,@#ERRVEC   ; SET ERR VEC PTR.
3724 021476 005737 177572      TST      @#SRO          ; GOT KT11?
3725 021502 000240          NOP                    ; (TRAP IF NO).
3726 021504 013737 002120 003100  MOV      L$HIME,KTFLG   ; YES. SET KT FLAG.
3727 021512 042737 000177 003100  BIC      #177,KTFLG     ;
3728 021520 010037 000004      MOV      RO,@#ERRVEC   ; RESTORE OLD ERR VEC PTR.
3729 021524 005000          CLR      RO            ; RO = AR DATA.
3730 021526 012701 172340      MOV      #KIPAR,R1     ; R1 = KI REGS PTR.
3731 021532 012761 077406 177740 1$:  MOV      #77406,-40(R1) ; SET DESCRIPTOR REG.
3732 021540 010021          MOV      RO,(R1)+      ; SET KIPAR REG.
3733 021542 062700 000200      ADD      #200,RO       ; BUMP AR DATA BY "4K".
3734 021546 020027 002000      CMP      RO,#2000     ; AT "I/O"?
3735 021552 001367          BNE     1$            ; NO.
3736 021554 012741 177600      MOV      #177600,-(R1) ; YES. SET KTPAR7 FOR I/O.
3737 021560 000405          BR      9$
3738
3739 021562 012716 021570      2$:  MOV      #6$, (SP)     ; SET UP RETURN
3740 021566 000002          RTI                    ; RTI TO NEXT LOCATION
3741
3742 021570 010037 000004      6$:  MOV      RO,@#ERRVEC   ; RESTORE OLD ERR VEC PTR.
3743
3744 021574 000207      9$:  RTS      PC
3745          ;          .IIF DF ONEFILE,          .PAGE
3754
3755
3761
    
```

```
3763  
3764 021576 .SBTTL PROTECTION TABLE  
      021576 BGNPROT  
3765 021576 177777 177777 177777 L$PROT::  
3766 021606 .WORD -1, -1, -1, -1 ;NO DEVICE PROTECTION REQUIRED.  
3767 ENDPROT
```

```

3769                                     .SBTTL INITIALIZE SECTION
3770
3771                                     ;++
3772                                     ;THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
3773                                     ;AT THE BEGINNING OF EACH PASS.
3774                                     ;
3775                                     ;IF "START" OR "RESTART", SET QUICK-PASS FLAG AND BUS-INIT.
3776                                     ;IF "CONTINUE", NOTHING IS REQUIRED.
3777                                     ;
3778                                     ;--
3779                                     ;+
3780                                     ;INSERT TEMPORARY JUMP TO ODT
3781                                     ;-
3782 021606                                BGNINIT
    021606                                L$INIT::
3783 021606                                40$:
3784 021606 012737 005672 002146          MOV     #EPRT1,EPRTSW      ;SET UP PRIMARY MESSAGE FOR REPLACEMENT
3785 021614 005037 003106                  CLR     SIFLAG            ;CLEAR "SOFT INIT" FLAG
3786 021620 005037 003102                  CLR     KTENABLE         ;CLEAR TEST ABOVE 28K FLAG
3787 021624 005037 002246                  CLR     RAMSIZ           ;CLEAR RAM SIZE FOR RAMERR ROUTINE
3788 021630                                READEF  #EF.CONTINUE
    021630 012700 000036                    MOV     #EF.CONTINUE,RO
    021634 104447                            TRAP   C$REFG
3789 021636                                BNCOMPLETE 1$
    021636 103023                            BCC    1$
3790 021640 023737 002150 002012          CMP     UNITN,L$UNIT      ;UNIT IN RANGE?
3791 021646 103064                            BHIS   4$                 ;BR IF NO.
3792 021650 005737 003060                  TST    DUFLG             ;DROPPED UNIT?
3793 021654 100466                            BMI    NXTU              ;BR IF YES
3794 021656 013701 002150                  MOV     UNITN,R1
3795 021662 006301                            ASL    R1
3796 021664 005761 003130                  TST    ERTABL(R1)
3797 021670 001512                            BEQ    SETU
3798 021672 032761 040000 003130          BIT     #BIT14,ERTABL(R1) ;DROPPED?
3799 021700 001054                            BNE    NXTU
3800 021702                                EXIT    INIT              ;DO NOTHING IF "CONTINUE".
    021702 104432                            TRAP   C$EXIT
    021704 000412                            .WORD  L10030-.
3801 021706                                1$:
    021706 012700 000035                    READEF  #EF.NEW
    021712 104447                            MOV     #EF.NEW,RO
    021714 103046                            TRAP   C$REFG
3802 021714                                BNCOMPLETE NXTU          ;TAKE NEXT UNIT IF NOT NEW PASS.
    021714 103046                            BCC    NXTU
3803 021716                                READEF  #EF.START
    021716 012700 000040                    MOV     #EF.START,RO
    021722 104447                            TRAP   C$REFG
3804 021724                                BCOMPLETE 2$
    021724 103404                            BCS    2$
3805 021726                                READEF  #EF.RESTART
    021726 012700 000037                    MOV     #EF.RESTART,RO
    021732 104447                            TRAP   C$REFG
3806 021734                                BNCOMPLETE 31$
    021734 103025                            BCC    31$
3807 021736                                2$:
3808 021736                                BRESET                                ;1ST PASS, BUS-INIT...
    021736 104433                            TRAP   C$RESET          ;BUS RESET.
3809 021740 005037 002162                  CLR     TSTCNT           ;NUMBER OF TESTS RUN IN PASS
    
```

```

3810 021744 005037 002170          CLR    FATFLG      ;RESET FLAG TO ZERO "FATAL ERRORS"
3811 021750 005037 003332          CLR    SKIPT      ;CLEAR THE SUBTEST "SKIPPER"
3812 021754          20$:
3813 021754 012737 177777 002152      MOV    #-1,QVP    ;...QUICK VERIFY...
3814 021762 004737 021356          JSR    PC,ENVIRN  ;SET ENVIRONMENT.
3815 021766 004737 021444          JSR    PC,KTINIT  ;INITIALIZE KT MEMORY MANAGEMENT
3816 021772 012700 003130          MOV    #ERTABL,RO
3817 021776 005020 30$:          CLR    (RO)+      ;CLEAR THE ERROR TABLE
3818 022000 020027 003330          CMP    RO,#ERTABE
3819 022004 103774          BLO   30$
3820 022006 000404          BR    4$
3821 022010 005037 002152 31$:          CLR    QVP
3822 022014 000137 022064          JMP    PASRPT     ;GO REPORT THE STATUS
3823
3824 022020 4$:
3825 022020 012737 177777 002150 NEWPAS: MOV    #-1,UNITN  ;INIT UNIT NUMBER...
3826 022026 005037 002166          CLR    DEVCNT    ;CLEAR COUNT OF DEVICES RUNNING
3827 022032          NXTU:          BREAK
3828 022034 104422          TRAP  C$BRK
3829 022040 005237 002150          INC    UNITN
3830 022046 023737 002150 002012      CMP    UNITN,L$UNIT ;...AND SET NEXT UNIT NUMBER.
3831 022050 012737 177777 003060      BLO   SETU
3832 022056 000401          MOV    #-1,DUFLG
3833 022060          BR    11$
3834 022062 104444          DOCLN          ;ABORT, NO MORE UNITS.
3835 022064 000240          TRAP  C$DCLN
3836 022064 023727 002012 000001 11$:          NOP
3837 022072 101752          PASRPT:        CMP    L$UNIT,#1  ;HOW MANY UNITS SELECTED?
3838 022074 005737 002166          BLOS  NEWPAS    ;BR IF ONLY 1
3839 022100 001747          TST   DEVCNT    ;ARE ANY STILL RUNNING?
3840 022102          BEQ   NEWPAS    ;BR IF NO
3841 022104 104421          RFLAGS        RO
3842 022110 032700 000100          TRAP  C$RFLA
3843          BIT    #ISR,RO  ;SHOULD WE PRINT STATISTICS
3844 022112          BNE  NEWPAS    ;BR IF NO
3845 022112 104424          DORPT
3846 022114 000741          TRAP  C$DRPT
3847          BR    NEWPAS
3848 022116          10$:
3849 022124          SETU:          GPHARD UNITN,RO  ;GET UNIT N P-TABLE POINTER.
3850 022126 013700 002150          MOV    UNITN,RO
3851 022132 005237 002166          TRAP  C$GPHRD
3852 022136 012001          BNCOMPLETE NXTU ;BR IF UNIT NOT AVAILABLE.
3853 022140 010137 002154          BCC   NXTU
3854          CLR    DUFLG  ;CLEAR "DROPPED" FLAG.
3855 022144 012001          INC    DEVCNT
3856 022146 011002          MOV    (RO)+,R1 ;GET 1ST REGISTER ADDRESS.
3857 022150 010237 002160          MOV    R1,CSRADDR ;ADDRESS OF REGISTERS OF UNIT UNDER TEST
3858 022154 010137 002156          MOV    (RO),R2  ;GET VECTOR ADDRESS.
3859 022160 012721 016712          MOV    R2,IPRI  ;GET INTERRUPT PRIORITY
3859          MOV    R1,IVEC ;SET INTERRUPT PRIORITY.
3859          MOV    #INTR,(R1)+ ;SET INTERRUPT VECTOR POINTER...
3859          ;...VECTOR...
    
```

```

3860 022164 010221          MOV     R2,(R1)+      ;...AND PRIORITY.
3861
3862 022166          1$:
3863          ;         TST     QVP          ;1ST PASS ??
3864          ;         BEQ     5$          ;NO, SKIP THE PASS 1 STUFF.
3865
3866          ;
3867          ;1ST PASS, CHECK THAT DEVICE ADDRESSES ARE VALID, AND
3868          ;THAT THE DISPLAY STATUS IS PROPERLY INITIALIZED.
3869          ;
3870 022166 013701 002150          MOV     UNITN,R1
3871 022172 006301          ASL     R1
3872 022174 052761 100000 003130  BIS     #BIT15,ERTABL(R1) ;SAY DEVICE RUNNING
3873 022202 005037 005232          CLR     EXTA          ;CLEAR ERROR EXTENSION FLAG.
3874 022206 023727 002012 000001  CMP     L$UNIT,#1      ;ARE WE TESTING MULTIPLE UNITS?
3875 022214 101416          BLOS   10$           ;BR IF NO.
3876 022216          RFLAGS  R0          ;YES -- GET OPERATOR FLAGS.
          022216 104421          TRAP   C$RFLA
3877 022220 032700 001000          BIT     #PNT,R0        ;SHOULD WE PRINT UNIT #?
3878 022224 001412          BEQ     10$           ;BR IF NOT.
3879 022226          PRINTF #PUNIT,UNITN ;PRINT THE UNIT #
          022226 013746 002150          MOV     UNITN,-(SP)
          022232 012746 022320          MOV     #PUNIT,-(SP)
          022236 012746 000002          MOV     #2,-(SP)
          022242 010600          MOV     SP,R0
          022244 104417          TRAP   C$PNTF
          022246 062706 000006          ADD     #6,SP
3880 022252          10$:
3881 022252 005037 003062          CLR     NODEV
3882 022256 013701 002154          MOV     CSRADDR,R1    ;ADDRESS OF FIRST REGISTER
3883 022262 010102          MOV     R1,R2          ;START OF REGISTERS
3884 022264 062702 000000          ADD     #TSSR,R2      ;ADDRESS OF TSSR REGISTER
3885 022270 004737 017120          JSR     PC,XNXM        ;TEST BOTH CONTROLLER REGISTERS...
3886 022274 103005          BCC    2$             ;...AND BR IF ALL OK,
3887 022276 010137 003062          MOV     R1,NODEV      ;FLAG DEVICE AS NON-EXISTENT
3888 022302 012737 177777 003060  MOV     #-1,DUFLG     ;DROP THIS UNIT.
3889 022310          2$:
3890          ;
3891          ;FINALLY, SET CPU PRIORITY AND WE'RE DONE.
3892          ;
3893 022310          5$:          SETPRI  #PRI00        ;ENABLE INTERRUPTS.
          022310 012700 000000          MOV     #PRI00,R0
          022314 104441          TRAP   C$SPRI
3894 022316          ENDINIT
          022316          L10030:
          022316 104411          TRAP   C$INIT
3895
3896 022320 045 116 045 PUNIT: .ASCIZ /%N%N%A***** TESTING UNIT %D2%A *****/
3897          .EVEN
    
```

.SBTTL ADD AND DROP UNITS SECTIONS

```

3899
3900
3901
3902
3903
3904
3905
3906 022366
      022366
3907 022366 010001
3908 022370 006301
3909 022372 052761 100000 003130
3910 022400 042761 040000 003130
3911 022406
      022406 010046
      022410 012746 022434
      022414 012746 000002
      022420 010600
      022422 104417
      022424 062706 000006
3912 022430
      022430 000167
      022432 000026
3913 022434 045 116 045 1$:
3914
3915
3916 022462
      022462
      022462 104452
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928 022464
      022464
3929 022464 012737 177777 003060
3930 022472 010001
3931 022474 006301
3932 022476 052761 140000 003130
3933 022504 000240 000240 000240
3934 022512
      022512 010046
      022514 012746 022540
      022520 012746 000002
      022524 010600
      022526 104417
      022530 062706 000006
3935 022534
      022534 000167
      022536 000030

```

```

      ; THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
      ; TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
      ; OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
      ;--
      BGNAU
L$AU::
      MOV     RO,R1           ; GET UNIT TO BE ADDED (RO)
      ASL     R1              ; MAKE IT A WORD INDEX
      BIS     #100000,ERTABL(R1) ; SET THE "ACTIVE" BIT
      BIC     #40000,ERTABL(R1) ; CLEAR THE "DROPPED" BIT
      PRINTF  #1$,RO
      MOV     RO,-(SP)
      MOV     #1$,-(SP)
      MOV     #2$,-(SP)
      MOV     SP,RO
      TRAP   C$PNTF
      ADD     #6,SP
      EXIT   AU
      .WORD  J$JMP
      .WORD  L10031-2-.
      .ASCIZ /%N%A UNIT %D%A ADDED/
      .EVEN
      ENDAU                   ; UNUSED.
L10031:
      TRAP   C$AU

```

```

      ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
      ; TO BE REMOVED FROM THE TEST LIST.
      ;
      ; SUPVSR DOES THE "DROPPING". THIS IS JUST TO TELL THE MAN.
      ; "DROPPED" UNITS ARE RE-SELECTED ON OPERATOR "STA" OR "ADD"
      ; COMMAND, OTHERWISE REMAIN INACTIVE. THE "DISPLAY" COMMAND
      ; WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
      ; WHICH ARE STILL ACTIVE.
      ; UPON ENTRY, RO CONTAINS THE UNIT TO BE DROPPED.
      ;
      BGNDU
L$DU::
      MOV     #-1,DUFLG
      MOV     RO,R1
      ASL     R1
      BIS     #140000,ERTABL(R1) ; SAY DROPPED
      240,240,240 ; ??????????
      PRINTF  #1$,RO
      MOV     RO,-(SP)
      MOV     #1$,-(SP)
      MOV     #2$,-(SP)
      MOV     SP,RO
      TRAP   C$PNTF
      ADD     #6,SP
      EXIT   DU
      .WORD  J$JMP
      .WORD  L10032-2-.

```

```

3936 022540 045 116 045 1$: .ASCIZ /%N%A UNIT %D%A DROPPED/
3937 .EVEN
3938 022570 ENDDU
022570 L10032:
022570 104453 TRAP C$DU
3939 ;++
3940 ; AUTO-DROP CODE SECTION.
3941 ;--
3942 022572 BGNAUTO
022572 L$AUTO::
3943 022572 012703 000550 MOV #360.,R3 ;ENOUGH TIME FOR 2400' REEL TO REWIND
3944 022576 004737 016744 JSR PC,WAITF ;WAIT FOR SSR TO SET
3945 022602 103420 BCS 20$ ;LEAVE WHEN SSR IS SET
3946 022604 DELAY 250. ;WAIT FOR .25 SECONDS
022604 012727 000372 MOV #250.,(PC)+
022610 000000 .WORD 0
022612 013727 002116 MOV L$DLY,(PC)+
022616 000000 .WORD 0
022620 005367 177772 DEC -6(PC)
022624 001375 BNE .-4
022626 005367 177756 DEC -22(PC)
022632 001367 BNE .-20
3947 022634 005303 DEC R3 ;BUMP COUNTER DOWN
3948 022636 001357 BNE 10$ ;KEEP GOING
3949 022640 004737 017776 JSR PC,CKDROP ;TRY AND DROP UNIT
3950 022644 20$: ENDAUTO ; UNUSED.
3951 022644 L10033:
022644 104461 TRAP C$AUTO
  
```

.SBTTL CLEAN-UP AND REPORT CODING SECTIONS

```

3953
3954
3955
3956
3957
3958
3959
3960 022646          BGNCLN
      022646
3961 022646 005737 003060
3962 022652 100407
3963
3964
3965 022654 013705 002154
3966 022660 012765 000000 000000
3967 022666 004737 016744
3968 022672
3969 022672
      022672
      022672 104412
3970
3971
3972
3973
3974 022674
      022674
3975 022674
      022674 012746 023136
      022700 012746 000001
      022704 010600
      022706 104416
      022710 062706 000004
3976 022714 010246
3977 022716 010346
3978 022720 010446
3979 022722 012704 003130
3980 022726 005003
3981 022730 011402
3982 022732 001467
3983 022734 100066
3984 022736 032702 040000
3985 022742 001015
3986 022744 042702 170000
3987 022750
      022750 010246
      022752 010346
      022754 012746 023173
      022760 012746 000003
      022764 010600
      022766 104416
      022770 062706 000010
3988 022774 000446
3989 022776 020227 160000
3990 023002 001012
3991 023004
      023004 010346
      023006 012746 023255
    
```

```

; **
; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS
; EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).
; USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.
; --
      BGNCLN
L$CLEAN::
      TST      DUFLG          ; "DROPPED" FLAG IS SET ON...
      BMI      1$            ; ...AND GROSS CONTROLLER FAULT...
                                ; ...DON'T TRY TO XCT CLEANUP CODE.
      MOV      CSRADDR,R5    ; ADDRESS OF TSV REGISTERS ON UNIBUS
      MOV      #0,TSSR(R5)  ; DO SOFT INIT
      JSR      PC,WAITF
1$:
2$:      ENDCLN
L10034:  TRAP      C$CLEAN
; **
; THE REPORT CODING SECTION CONTAINS THE
; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
; --
      BGNRPT
L$RPT::
      PRINTS  #DEVSUM
      MOV      #DEVSUM,-(SP)
      MOV      #1,-(SP)
      MOV      SP,R0
      TRAP    C$PNTS
      ADD      #4,SP
      MOV      R2,-(SP)
      MOV      R3,-(SP)
      MOV      R4,-(SP)
      MOV      #ERTABL,R4    ; GET START OF ERROR TABLE.
      CLR      R3            ; CLEAR UNIT NUMBER
1$:      MOV      (R4),R2    ; GET ERROR TABLE ENTRY & TEST IT.
      BEQ      4$            ; ZERO IF UNIT NOT RUN
      BPL      4$
      BIT      #BIT14,R2    ; WAS UNIT DROPPED?
      BNE      2$            ; BR IF YES
      BIC      #+C7777,R2   ; GET ERROR COUNT FIELD
      PRINTS  #DEVONL,R3,R2 ; PRINT
      MOV      R2,-(SP)
      MOV      R3,-(SP)
      MOV      #DEVONL,-(SP)
      MOV      #3,-(SP)
      MOV      SP,R0
      TRAP    C$PNTS
      ADD      #10,SP
      BR      4$
2$:      CMP      R2,#160000 ; WAS UNIT NON-EXISTENT?
      BNE      3$            ; BR IF NO
      PRINTS  #DEVNXR,R3
      MOV      R3,-(SP)
      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
3992 023026 000431      BR     4$
3993 023030 020227 160001      3$:  CMP    R2,#160001      ; WAS UNIT NOT READY AT STARTUP?
3994 023034 001012      BNE   30$              ; BR IF NO.
3995 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
3996 023060 000414      BR     4$
3997 023062 042702 170000      30$: BIC    #+C7777,R2
3998 023066      PRINTS #DEVDR0,R3,R2
023066 010246      MOV    R2,-(SP)
023070 010346      MOV    R3,-(SP)
023072 012746 023420      MOV    #DEVDR0,-(SP)
023076 012746 000003      MOV    #3,-(SP)
023102 010600      MOV    SP,R0
023104 104416      TRAP  C:PNTS
023106 062706 000010      ADD    #10,SP
3999 023112 062704 000002      4$:  ADD    #2,R4
4000 023116 005203      INC    R3
4001 023120 020427 003330      CMP    R4,#ERTABE
4002 023124 103701      BLO   1$
4003 023126 012604      MOV    (SP)+,R4
4004 023130 012603      MOV    (SP)+,R3
4005 023132 012602      MOV    (SP)+,R2
4006 023134      ENDRPT              ; UNUSED.
023134      L10035:
023134 104425      TRAP  C:RPT
4007
4008
4009 023136      045      116      045  DEVSUM: .ASCIZ /#N#ADEVICE STATUS SUMMARY:#N/
4010 023173      045      101      040  DEVONL: .ASCIZ /#A UNIT #D3#A CONTROLLER READY, ERRORS = #D#N/
4011 023255      045      101      040  DEVNXR: .ASCIZ /#A UNIT #D3#A DROPPED, NON-EXISTENT REGISTER#N/
4012 023337      045      101      040  DEVNRD: .ASCIZ /#A UNIT #D3#A DROPPED, NOT READY AT STARTUP#N/
4013 023420      045      101      040  DEVDR0: .ASCIZ /#A UNIT #D3#A DROPPED, ERRORS = #D#N/
4014
4017
4024
4030
4038
    
```

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
 4077
 4078

.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.

```

4079 023470          BGNTST
      023470
4080 023470 005037 002170          CLR   FATFLG          ;CLEAR FATAL ERROR FLAG
4081 023474 012737 005672 002146  MOV   #EPRT1,EPRTSW  ;SET UP ERROR MESSAGE SWITCH
4082 023502 005037 003100          CLR   KTFLG          ;HOLD OFF KT11
4087 023506 012700 023704          MOV   #TST1ID,R0     ;ASCII MESSAGE TO IDENTIFY TEST
4088 023512 004737 017232          JSR   PC,TSTSETUP    ;DO INITIAL TEST SETUP
4089 023516 012737 000005 002164  MOV   #5.,LOOPCNT    ;PERFORM 5 ITERATIONS
4090 023524          T1LOOP:
4091 023524 005003          CLR   R3             ;USE R3 AS FATAL ERROR FLAG
4092
4093 023526          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      023526
      023526 104402          T1.1:          TRAP   C$BSUB
4094
4095 023530          BRESET          ;ISSUE A BUS RESET
      023530 104433          TRAP   C$RESET
4096 023532 004737 016744          JSR   PC,WAITF       ;WAIT FOR READY
  
```



```

4112 023574 005703          TST      R3          ;DID WE HAVE FATAL ERROR ?
4113 023576 001402          BEQ      20$         ;BRANCH IF NOT
4114 023600 004737 017776          JSR      PC,CKDROP  ;GO DROP THIS UNIT, IF ALLOWED
4115 023604 005003          CLR      R3          ;RESET FATAL ERROR FLAG
4116
4117
4118 023606          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      023606          T1.2:
      023606 104402          TRAP      C$BSUB
4119
4120 023610 005065 000000          CLR      TSSR(R5)   ;WRITE TO ISSUE A SOFT RESET
4121 023614 004737 016744          JSR      PC,WAITF   ;WAIT FOR READY TO SET
4122 023620 016501 000000          MOV      TSSR(R5),R1 ;GET REGISTER TSSR DATA
4123 023624 010102          MOV      R1,R2     ;START SETUP OF EXPECTED TSSR
4124 023626 042702 176277          BIC      #+C<HIADDR!OFL>,R2 ;CLEAR OUT UNUSED BITS
4125 023632 052702 002200          BIS      #SSR!NBA,R2 ;R4 HAS EXPECTED CONTENTS
4126 023636 020102          CMP      R1,R2     ;COMPARE EXPECTED TO RECEIVED
4127 023640 001405          BEQ      10$         ;BRANCH IF COMPARE
4131 023642          ERRDF      ERRNO,SFIERR,SFFMSG ;REPORT A FATAL ERROR
      023642 104455          TRAP      C$ERDF
      023644 000146          .WORD    102
      023646 003550          .WORD    SFIERR
      023650 011566          .WORD    SFFMSG
4132 023652 005203          INC      R3          ;SET THE ERROR FLAG
4133 023654          10$:
4134 023654          ENDSUB          ;//////////////// END SUBTEST //////////////////
      023654 104403          L10040:
      023654          TRAP      C$ESUB
4135
4136
4137 023656 005703          TST      R3          ;FATAL ERROR DETECTED ?
4138 023660 001402          BEQ      20$         ;BRANCH IF NOT
4139 023662 004737 017776          JSR      PC,CKDROP  ;SEE IF TIME TO DROP UNIT
4140 023666 004737 017200          JSR      PC,TSTLOOP ;SHOULD WE DO ITERATIONS ?
4141 023672 103002          BCC      40$         ;BRANCH IF NOT
4142 023674 000137 023524          JMP      T1LOOP     ;LOOP UNTIL COUNT EXPIRED
4143 023700          40$:
      023700 104432          EXIT      TST       ;ALL DONE THIS TEST
      023702 000022          TRAP      C$EXIT
      023702          .WORD    L10036-.
4144
4145
4146          ;+
4147          ;LOCAL TEXT MESSAGES FOR TEST
4148          ;-
4149 023704          111 156 151 TST1ID: .ASCIZ 'Initialization'
4150          .EVEN
4151 023724          ENDTST
      023724          L10036:
      023724 104401          TRAP      C$ETST
4152
4153
    
```

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
 4200
 4201
 4202
 4203 023726
 023726
 4204 023726 005037 002170
 4205 023732 012737 005672 002146
 4206 023740 005037 003100
 4207
 4208 023744
 023744
 023744 104402
 4209

.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.

BGNTST

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

BGNSUB

T2::
 ;//////////////// BEGIN SUBTEST //////////////////
 T2.1: TRAP C\$BSUB

```

4214 023746 012700 024672      MOV      #TST2ID,R0      ;ASCII MESSAGE TO IDENTIFY TEST
4215 023752 004737 017232      JSR      PC,TSTSETUP    ;DO INITIAL TEST SETUP
4216 023756 012737 000002 002164  T2LOOP:  MOV      #2,LOOPCNT     ;PERFORM 2 ITERATIONS
4217 023764
4218 023764 004737 016470      JSR      PC,SOFINIT     ;DO INITIALIZE ON CONTROLLER
4219 023770 103405                BCS      20$            ;BR IF INIT WAS OK
4223 023772 010001                MOV      R0,R1          ;CONTENTS OF TSSR REGISTER
4224 023774                ERRDF   ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
                                TRAP      C$ERDF
                                .WORD    201
                                .WORD    SFIERR
                                .WORD    SFIMSG
                                023774 104455
                                023776 000311
                                024000 003550
                                024002 011506
4225 024004 012704 000002      20$:   MOV      #2,R4          ;SET RAM ADDRESS AT TWO
4226 024010                25$:
4227
4228 024010 110402                MOVB     R4,R2          ;EXPECTED DATA FROM WRAP-AROUND
4229 024012 110465 177777      MOVB     R4,TSDBH(R5)   ;LOAD ADDRESS INTO TSDB
4230 024016 110265 177776      MOVB     R2,TSDBL(R5)   ;LOADS DATA INTO RAM LOCATION
4231 024022 116501 177776      MOVB     TSBAL(R5),R1   ;READS WRAP DATA
4232 024026 120102                CMPB     R1,R2          ;DOES WRITTEN(WRAP) = READ
4233 024030 001404                BEQ      30$            ;BR IF OK, THEY ARE EQUAL
4237 024032                ERRHRD  ERRNO,TSBAM2,EXPREC ;DATA NOT WRAPPED CORRECTLY
                                TRAP      C$ERHRD
                                .WORD    202
                                .WORD    TSBAM2
                                .WORD    EXPREC
                                024032 104456
                                024034 000312
                                024036 024530
                                024040 016170
4238 024042                30$:
4239
4240 024042 005204                INC      R4              ;NEXT ADDRESS
4241 024044 020427 000400      CMP      R4,#400        ;END OF RAM MEMORY CHECK
4242 024050 001357                BNE     25$            ;LOOP TILL ALL RAM WRITTEN
4243 024052 005002                CLR     R2              ;CLEAR OUT R2 HIGH BITS
4244 024054 005304                DEC     R4              ;SET BACK TO 377
4245 024056 110402                40$:   MOVB     R4,R2          ;GET DATA PATTERN BACK IN SHAPE
4246 024060 110465 177777      MOVB     R4,TSDBH(R5)   ;LOAD UP RAM ADDRESS POINTER
4247 024064 116501 177776      MOVB     TSBAL(R5),R1   ;READ RAM CONTENTS BACK
4248 024070 120102                CMPB     R1,R2          ;CHECK WITH DATA WRITTEN
4249 024072 001404                BEQ     45$            ;BR IF OK, DATA IN = DATA OUT
4253 024074                ERRHRD  ERRNO,TSBAM2,EXPREC ;WRITTEN DATA NOT = TO READ
                                TRAP      C$ERHRD
                                .WORD    203
                                .WORD    TSBAM2
                                .WORD    EXPREC
                                024074 104456
                                024076 000313
                                024100 024530
                                024102 016170
4254 024104                45$:   CKLOOP                ;SCOPE LOOP
                                TRAP      C$CLP1
4255 024106 005304                DEC     R4              ;DROP DATA COUNTER (PATTERN)
4256 024110 022704 000002      CMP     #2,R4          ;AT LOC TWO YET
4257 024114 001360                BNE     40$            ;BR, IF NOT AT TWO YET
4258
4259 024116                ENDSUB                ;////////////////// END SUBTEST ////////////////////
                                L10042:
                                TRAP      C$ESUB
                                024116 104403
4260

```

```

4262
4263 024120          BGNSUB          ;////////// BEGIN SUBTEST //////////
      024120          ;              T2.2:          TRAP      C$BSUB
      024120 104402          ;
4264          ;          TEST 2, SUBTEST 2
4265          ;
4266          ;
4267          ;          THIS SUBTEST WRITES RAM WITH ALL ZEROS
4268          ;          THEN WALKS AN ALL ONES WORD DOWN THROUGH MEMORY
4269          ;
4270 024122 004737 016470      JSR      PC,SOFINIT          ;DO INITIALIZE ON CONTROLLER
4271 024126 103405          BCS      20$          ;BR IF INIT WAS OK
4275 024130 010001          MOV      R0,R1          ;CONTENTS OF TSSR REGISTER
4276 024132          ERRDF  ERRNO,SFIERR,SFIMSG ;FATAL ERROR TSSR WAS NOT OK
      024132 104455          ;              TRAP      C$ERDF
      024134 000314          ;              .WORD  204
      024136 003550          ;              .WORD  SFIERR
      024140 011506          ;              .WORD  SFIMSG
4277 024142 005002          20$:    CLR      R2          ;TEST DATA = 0
4278 024144 012704 000002      MOV      #2,R4          ;STARTING RAM ADDRESS = 2
4279 024150          25$:
4280
4281 024150 110465 177777      MOVB    R4,TSDBH(R5)      ;LOAD ADDRESS INTO TSDB
4282 024154 110265 177776      MOVB    R2,TSDBL(R5)      ;LOADS DATA INTO RAM LOCATION
4283 024160 116501 177776      MOVB    TSBAL(R5),R1      ;READS WRAP DATA
4284 024164 120102          CMPB    R1,R2          ;DOES WRITTEN(WRAP) = READ ?
4285 024166 001404          BEQ     30$          ;BR IF OK, THEY ARE EQUAL
4289 024170          ERRHRD  ERRNO,TSBAM2,EXPREC ;DATA NOT WRAPPED CORRECTLY
      024170 104456          ;              TRAP      C$ERHRD
      024172 000315          ;              .WORD  205
      024174 024530          ;              .WORD  TSBAM2
      024176 016170          ;              .WORD  EXPREC
4290 024200          30$:
4291
4292 024200 005204          INC      R4          ;NEXT ADDRESS
4293 024202 020427 000400      CMP     R4,#400          ;END OF RAM MEMORY CHECK
4294 024206 001360          BNE     25$          ;BR, MORE RAM TO GO
4295
4296 024210 005304          35$:    DEC     R4          ;SET BACK TO 377
4297 024212 005002          CLR     R2          ;SET TO ALL ZEROS
4298 024214          40$:
4299 024214 110465 177777      MOVB    R4,TSDBH(R5)      ;LOAD UP THE ADDRESS FOR RAM
4300 024220 116501 177776      MOVB    TSBAL(R5),R1      ;READ THE RAM CONTENTS BACK
4301 024224 005002          CLR     R2          ;LOOKING FOR 000000 (EXPECTED)
4302 024226 120102          CMPB    R1,R2          ;BOTH SHOULD BE 00000000 BINARY
4303 024230 001404          BEQ     43$          ;BR, IF DATA IS GOOD
4307 024232          ERRHRD  ERRNO,TSBAM3,EXPREC ;CHARACTERISTICS DATA NOT CORRECT
      024232 104456          ;              TRAP      C$ERHRD
      024234 000316          ;              .WORD  206
      024236 024612          ;              .WORD  TSBAM3
      024240 016170          ;              .WORD  EXPREC
4308 024242 012702 000377      43$:    MOV     #000377,R2      ;SET ALL ONES WORD
4309 024246 110465 177777      MOVB    R4,TSDBH(R5)      ;LOAD UP RAM ADDRESS POINTER
4310 024252 110265 177776      MOVB    R2,TSDBL(R5)      ;WRITE DATA INTO RAM
4311 024256 116501 177776      MOVB    TSBAL(R5),R1      ;READ RAM CONTENTS BACK
4312 024262 120102          CMPB    R1,R2          ;CHECK WITH DATA WRITTEN
4313 024264 001404          BEQ     45$          ;BR IF OK, DATA IN = DATA OUT
    
```

```
4317 024266          ERRHRD  ERRNO,TSBAM2,EXPREC  ;WRITTEN DATA NOT = TO READ
      024266 104456          TRAP C$ERHRD
      024270 000317          .WORD 207
      024272 024530          .WORD TSBAM2
      024274 016170          .WORD EXPREC
4318 024276          45$: CKLOOP  ;SCOPE LOOP
      024276 104406          TRAP C$CLP1
4319 024300 005304          ;DROP RAM ADDRESS POINTER
4320 024302 022704 000002  ;AT LOC 2 YET
4321 024306 001342          ;BR, IF NOT AT TWO YET
4322
4323 024310          ENDSUB  ;////////// END SUBTEST ////////////
      024310          L10043:
      024310 104403          TRAP C$ESUB
4324
```


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
4458
4459
4460

.SBTTL TEST 3: COMMAND REJECT

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

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

4652 025626 052525 .WORD 052525

4653

4654

4655

4656

4657

4658

;*
;LOCAL TEXT MESSAGES FOR TEST
;*

4659	025630	103	157	155	T3NBA:	.ASCIZ	'Command Not Rejected'
4660	025655	103	157	156	T3SSR:	.ASCIZ	'Contents of TSSR Incorrect After Write Packet'
4661	025733	125	156	145	T3INT:	.ASCIZ	'Unexpected Interrupt Received On Write Packet'
4662	026011	105	170	160	T3NINT:	.ASCIZ	'Expected Interrupt Not Received On Write Packet'
4663	026071	111	156	143	T3TSBA:	.ASCIZ	'Incorrect TSBA Address After Packet Write'
4664	026143	103	157	155	TST3ID:	.ASCIZ	'Command Reject'

4665

4666 026162

026162

026162 104401

.EVEN
ENDTST

L10045: TRAP C\$ETST


```

4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875 026774          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      026774          ;              T4.3:
      026774 104402          TRAP      C$BSUB
4876
4877 026776          SETPRI  #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
      026776 012700 000000          MOV      #PRI00,R0
      027002 104441          TRAP      C$SPRI
4878 027004 012703 027314          ;START OF TEST DATA FOR SUBTEST
4879 027010 012704 027250          ;GET THE ADDRESS OF COMMAND PACKET
4880 027014 004737 030146          ;RESTORE PACKET TO STARTING VALUES
4881
4882
4883 027020 004737 016470          JSR      PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
4884 027024 103405          BCS     10$
4888 027026 010001          MOV     R0,R1          ;BR IF SOFT INIT = OK
4889 027030          ERRDF  ERRNO,SFIERR,SFIMSG ;SAVE CONTENTS OF TSSR
      027030 104455          ;DEVICE FATAL ERROR DURING INIT
      027032 000634          TRAP    C$ERDF
      027034 003550          .WORD  412
      027036 011506          .WORD  SFIERR
4890 027040 005037 002172          .WORD  SFIMSG
4891 027044 052737 000001 027260 10$: CLR      INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
4892 027052 010465 177776          BIS     #1,T4DATA          ;MAKE ADDRESS ODD
4893 027056 004737 016744          MOV     R4,TSDB(R5)        ;SET THE PACKET ADDRESS
4894 027062 103405          JSR     PC,WAITF          ;WAIT FOR SSR TO SET
4895 027064 010001          BCS     15$
4899 027066          MOV     R0,R1          ;BR IF CARRY SET (GOOD RETURN)
      027066 104455          ;SAVE CONTENTS OF TSSR
      027070 000635          ERRDF  ERRNO,T4SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      027072 027656          TRAP   C$ERDF
      027074 011520          .WORD  413
4900 027076          .WORD  T4SSR
      027076 104406 15$: CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      027100          TRAP   C$CLP1
4901 027100          ESCAPE SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      027102 000116          TRAP   C$ESCAPE
      027104 005737 002172          .WORD  L10053-.
4902 027110 001404          TST     INTRECV          ;DID AN INTERRUPT OCCUR ?
4903 027112          BEQ     22$
4907 027112          ERRHRD ERRNO,T4INT,PKTSSR ;BRANCH IF NOT
      027112 104456          TRAP   C$ERHRD
      027114 000636          .WORD  414
      027116 027745          .WORD  T4INT
      027120 011520          .WORD  PKTSSR
4908 027122 016501 000000          22$: MOV     TSSR(R5),R1        ;GET THE CONTENTS OF TSSR
4909 027126 012702 102206          MOV     #SC!SSR!TSREJ!NBA,R2 ;EXPECTED CONTENTS OF TSSR
4910 027132 032701 000100          BIT     #OFL,R1          ;IS OFF-LINE BIT SET ?
4911 027136 001402          BEQ     25$          ;BRANCH IF NOT OFF-LINE
  
```

TEST 4: WRITE CHARACTERISTICS

```

4912 027140 052702 000100
4913 027144 020201
4914 027146 001414
4915 027150 010100
4916 027152
4917 027162 020027 002000
4918 027166 001404
4922 027170
    027170 104456
    027172 000637
    027174 027560
    027176 011520
4923 027200
    027200 104406
4924 027202 032701 002000
4925 027206 001004
4929 027210
    027210 104456
    027212 000640
    027214 027330
    027216 011520
4930
4931 027220
    027220
    027220 104403
4932
4933 027222 005737 002170
4934 027226 001402
4935 027230 004737 017776
4936 027234
4937 027234
    027234 104432
    027236 000756
4938
4939
4940
4941
4942
4944 027240
4946 027250
4947 027250 100004
4948 027252 027260
4949 027254 000000
4950 027256 000010
4951
4952 027260
4953 027260 027274
4954 027262 000000
4955 027264 000016
4956 027266 000000
4957 027270
4958
4959 027270 000000 000000
4960 027274
4961
4962
4963

```

```

25$:  BIS      #0FL,R2      ;SET OFF-LINE IN EXPECTED DATA
      CMP      R2,R1      ;DOES EXPECTED MATCH RECEIVED ?
      BEQ      30$        ;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
      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$ESUB
60$:  TST      FATFLG     ;ANY FATAL ERRORS ?
      BEQ      60$        ;BRANCH IF NOT
      JSR      PC,CKDROP  ;TRY TO DROP THE UNIT
      EXIT      TST       ;ALL DONE THIS TEST
                                TRAP   C$EXIT
                                .WORD  L10050-.
;+
;LOCAL STORAGE FOR THIS TEST
;-
      .BLKB    10-<.-TUV2A&7>
T4PACKET:
      .WORD    100004
      .WORD    T4DATA
      .WORD    0
      .WORD    8.
      ;COMMAND PACKET FOR TEST
      ;WRITE CHARACTERISTICS COMMAND, WITH ACK
      ;ADDRESS OF CHARACTERISTICS BLOCK
      ;STARTING VALUE OF BLOCK SIZE
T4DATA:
      .WORD    T4BFR
      .WORD    0
      .WORD    14.
      .WORD    0
      ;CHARACTERISTICS DATA BLOCK
      ;ADDRESS OF MESSAGE BUFFER
      ;LENGTH OF MESSAGE BUFFER
T4SP:
      .WORD    0,0
T4BFR:  .BLKW    8.
      ;SPACE
      ;MESSAGE BUFFER
;+
;

```

```
4964 ;TEST DATA FOR SUBTEST TWO
4965 ;
4966 ;DATA HAS FORMAT:
4967 ;
4968 ; 1ST WORD OFFSET TO TEST WORD IN PACKET
4969 ; 2ND WORD BITS TO SET FOR TEST
4970 ;
4971 ;-
4972
4973 027314 T42DATA:
4974 027314 000000 037140 .WORD 0,BIT5!BIT6!BIT9!BIT10!BIT11!BIT12!BIT13
4975 027320 000002 000001 .WORD 2,BIT0
4976 027324 000004 100100 .WORD 4,BIT6!BIT15
4977 027330 T42DONE=.
4978
4979
4980 ;*
4981 ;LOCAL TEXT MESSAGES FOR TEST ;
4982 ;-
4983
4984 027330 116 102 101 T42NBA: .ASCIZ 'NBA Not Set On Rejected WRITE CHARACTERISTICS'
4985 027406 127 122 111 T4NBA: .ASCIZ 'WRITE CHARACTERISTICS Command Not Accepted'
4986 027461 127 122 111 T42REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
4987 027560 127 122 111 T44REJ: .ASCIZ 'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
4988 027656 103 157 156 T4SSR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
4989 027745 125 156 145 T4INT: .ASCIZ 'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
4990 030034 111 156 143 T4TSBA: .ASCIZ 'Incorrect TSBA Address After WRITE CHARACTERISTICS'
4991 030117 127 162 151 TST4ID: .ASCIZ 'Write Characteristics'
4992 .EVEN
4993
```

4995
4996
4997
4998
4999
5000
5001
5002 030146
5003 030146
5004 030152 012701 027250
5005 030156 012721 100004
5006 030162 012721 027260
5007 030166 005021
5008 030170 012721 000010
5009 030174 012721 027274
5010 030200 005021
5011 030202 012721 000020
5012 030206 005021
5013 030210 005011
5014 030212 000207
5015 030214
030214
030214 104401
5016

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

T4REST:

SAVREG ;SAVE THE REGISTERS
MOV #T4PACKET,R1 ;START OF THE PACKET
MOV #100004,(R1)+ ;WRITE CHARACTERISTICS WITH ACK
MOV #T4DATA,(R1)+ ;ADDRESS OF CHAR DATA BLOCK
CLR (R1)+ ;EXTENDED ADDRESS
MOV #8,(R1)+ ;SIZE OF DATA BLOCK IN BYTES
MOV #T4BFR,(R1)+ ;ADDRESS OF MESSAGE BUFFER
CLR (R1)+ ;LENGTH OF MESSAGE BUFFER
CLR (R1)
RTS PC ;RETURN
ENDTST

L10050: TRAP C\$ETST

.SBTTL TEST 5: VOLUME CHECK

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

THE TEST PROCEEDS AS FOLLOWS:

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

```

5052 030216          BGNTST
      030216
5053 030216 005037 002170          CLR    FATFLG          ;CLEAR FATAL ERROR FLAG
5054 030222 012737 005672 002146  MOV    #EPRT1,EPRTSW  ;SET UP ERROR MESSAGE SWITCH
5055 030230 005037 003100          CLR    KTFLG          ;HOLD OFF KT11
5060 030234 012700 031407          MOV    #TSTSID,R0     ;ASCII MESSAGE TO IDENTIFY TEST
5061 030240 004737 017232          JSR    PC,TSTSETUP   ;DO INITIAL TEST SETUP
5062 030244 012737 000002 002164  MOV    #2.,LOOPCNT   ;PERFORM 2 ITERATIONS
5063 030252          TSLOOP:
5064
5065 030252 012704 030720          MOV    #TSPACKET,R4  ;PACKET FOR WRITE CHARACTERISTICS
5066 030256 012702 030742          MOV    #TSBFR,R2     ;ADDRESS OF THE MESSAGE BUFFER
5067 030262 012762 052525 000006  MOV    #052525,XSTO(R2) ;SET XSTATO TO KNOWN VALUE
5068 030270 004737 016470          JSR    PC,SOFINIT   ;DO SOFT INIT OF CONTROLLER
5069 030274 103405          BCS   10$           ;BR IF SOFT INIT = OK
5073 030276 010001          MOV    R0,R1        ;SAVE CONTENTS OF TSSR
5074 030300          ERRDF  ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
      030300 104455          TRAP   C:ERDF
      030302 000765          .WORD 501
      030304 003550          .WORD SFIERR
      030306 011506          .WORD SFIMSG
5075 030310 042714 040000          10$: BIC    #BIT14,(R4)   ;CLEAR THE CVC BIT
5076 030314 010465 177776          MOV    R4,TSDB(R5)  ;SET THE PACKET ADDRESS FOR WRITE CHAR
  
```

```

5077 030320 004737 017060      JSR    PC,CHKTSSR      ;WAIT FOR SSR TO SET
5078 030324 103405              BCS    15$             ;BR IF CARRY SET (GOOD RETURN)
5079 030326 010001              MOV    R0,R1          ;SAVE CONTENTS OF TSSR
5083 030330 104455              ERRDF  ERRNO,T5SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP    C$ERDF
                                .WORD   502
                                .WORD   T5SSR
                                .WORD   PKTSSR
5084 030340 104406      15$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP    C$CLP1
5085 030342 104410      ESCAPE TST          ;EXIT IF FATAL ERROR
                                TRAP    C$ESCAPE
                                .WORD   L10054-.
5086 030346 016203 000006      MOV    XST0(R2),R3    ;STORE STATUS FOR A WHILE
5087 030352 020327 052525      CMP    R3,#052525    ;CHECK FOR XSTATO OVER WRITTEN (GOOD!)
5088 030356 001006              BNE    20$           ;BR, IF XSTATO HAS BEEN UPDATED
5089 030360 016501 000000      MOV    TSSR(R5),R1   ;PICK UP TSSR FOR ERROR PRINTOUT
5093 030364 104456              ERRHRD ERRNO,T5NMSG,PKTSSR ;"NO MESSAGE PACKET RETURNED"
                                TRAP    C$ERHRD
                                .WORD   503
                                .WORD   T5NMSG
                                .WORD   PKTSSR
5094 030374 032762 000020 000006 20$: BIT    #XSOVCK,XST0(R2) ;IS VOLUME CHECK CLEAR IN XSTO ?
5095 030402 001006              BNE    23$           ;OKAY IF VOLUME CHECK IS CLEARED
5099 030404 016501 000000      MOV    TSSR(R5),R1   ;CONTENTS OF TSSR FOR ERROR REPORT
5100 030410 104456              ERRHRD ERRNO,T5VCK2,PKTMES ;VOLUME CHECK NOT SET
                                TRAP    C$ERHRD
                                .WORD   504
                                .WORD   T5VCK2
                                .WORD   PKTMES
5101 030420 104406      23$:  CKLOOP          ;LOOP ON ERROR ?
                                TRAP    C$CLP1
5102 030422 010465 177776      MOV    R4,TSDB(R5)   ;SET THE PACKET ADDRESS FOR WRITE CHAR
5103 030426 004737 017060      JSR    PC,CHKTSSR    ;WAIT FOR SSR TO SET
5104 030432 103405              BCS    25$             ;BR IF CARRY SET (GOOD RETURN)
5105 030434 010001              MOV    R0,R1          ;SAVE CONTENTS OF TSSR
5109 030436 104455              ERRDF  ERRNO,T5SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                TRAP    C$ERDF
                                .WORD   505
                                .WORD   T5SSR
                                .WORD   PKTSSR
5110 030446 104406      25$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
                                TRAP    C$CLP1
5111 030450 104410      ESCAPE TST          ;EXIT IF FATAL ERROR
                                TRAP    C$ESCAPE
                                .WORD   L10054-.
5112 030454 026203 000006      CMP    XST0(R2),R3    ;THE XSTO SHOULD NOT HAVE CHANGED
5113 030460 001406              BEQ    27$           ;OKAY IF VOLUME CHECK IS SET
5117 030462 016501 000000      MOV    TSSR(R5),R1   ;CONTENTS OF TSSR FOR ERROR REPORT
5118 030466 104456              ERRHRD ERRNO,T5NVCK,PKTMES ;VOLUME CHECK NOT SET
                                TRAP    C$ERHRD
                                .WORD   506
                                .WORD   T5NVCK
                                .WORD   PKTMES
5119 030476 104406      27$:  CKLOOP          ;LOOP ON ERROR ?
                                TRAP    C$CLP1
5120 030500 032762 000020 000006 30$: BIT    #XSOVCK,XST0(R2) ;IS VOLUME CHECK SET IN XSTO ?
    
```

```

5121 030506 001006
5125 030510 016501 000000
5126 030514 104456
      030514 000773
      030516 031055
      030520 011574
5127 030524 33$: CKLOOP
      030524 104406
5128 030526 052714 040000
5129 030532 010465 177776
5130 030536 004737 017060
5131 030542 103405
5132 030544 010001
5136 030546
      030546 104455
      030550 000774
      030552 031221
      030554 011520
5137 030556 35$: CKLOOP
      030556 104406
5138 030560
      030560 104410
      030562 000642
5139 030564 032762 000020 000006
5140 030572 001406
5144 030574 016501 000000
5145 030600
      030600 104456
      030602 000775
      030604 030762
      030606 011574
5146 030610 40$: CKLOOP
      030610 104406
5147 030612 042714 040000
5148 030616 010465 177776
5149 030622 004737 017060
5150 030626 103405
5151 030630 010001
5155 030632
      030632 104455
      030634 000776
      030636 031221
      030640 011520
5156 030642 45$: CKLOOP
      030642 104406
5157 030644
      030644 104410
      030646 000556
5158 030650 032762 000020 000006
5159 030656 001406
5163 030660 016501 000000
5164 030664
      030664 104456
      030666 000777
      030670 031131
      030672 011574

      BNE 33$
      MOV TSSR(R5),R1
      ERRHRD ERRNO,T5VCK2,PKTMES
      ;OKAY IF VOLUME CHECK IS SET
      ;CONTENTS OF TSSR FOR ERROR REPORT
      ;VOLUME CHECK NOT SET
      TRAP C$ERHRD
      .WORD 507
      .WORD T5VCK2
      .WORD PKTMES
      ;LOOP ON ERROR ?
      TRAP C$CLP1
      ;SET THE CVC BIT
      ;SET THE PACKET ADDRESS FOR WRITE CHAR
      ;WAIT FOR SSR TO SET
      ;BR IF CARRY SET (GOOD RETURN)
      ;SAVE CONTENTS OF TSSR
      ;DEVICE FATAL SSR FAILED TO SET
      TRAP C$ERDF
      .WORD 508
      .WORD T5SSR
      .WORD PKTSSR
      ;LOOP ON ERROR, IF FLAG SET
      TRAP C$CLP1
      ;EXIT IF FATAL ERROR
      TRAP C$ESCAPE
      .WORD L10054-.
      BIT #XSOVCK,XST0(R2)
      BEQ 40$
      MOV TSSR(R5),R1
      ERRHRD ERRNO,T5VCK,PKTMES
      ;IS VOLUME CHECK CLEAR IN XSTO ?
      ;OKAY IF VOLUME CHECK IS CLEARED
      ;CONTENTS OF TSSR FOR ERROR REPORT
      ;VOLUME CHECK NOT CLEARED
      TRAP C$ERHRD
      .WORD 509
      .WORD T5VCK
      .WORD PKTMES
      ;LOOP ON ERROR ?
      TRAP C$CLP1
      ;CLEAR THE CVC BIT
      ;SET THE PACKET ADDRESS FOR WRITE CHAR
      ;WAIT FOR SSR TO SET
      ;BR IF CARRY SET (GOOD RETURN)
      ;SAVE CONTENTS OF TSSR
      ;DEVICE FATAL SSR FAILED TO SET
      TRAP C$ERDF
      .WORD 510
      .WORD T5SSR
      .WORD PKTSSR
      ;LOOP ON ERROR, IF FLAG SET
      TRAP C$CLP1
      ;EXIT IF FATAL ERROR
      TRAP C$ESCAPE
      .WORD L10054-.
      BIT #XSOVCK,XST0(R2)
      BEQ 50$
      MOV TSSR(R5),R1
      ERRHRD ERRNO,T5NVCK,PKTMES
      ;IS VOLUME CHECK CLEAR IN XSTO ?
      ;OKAY IF VOLUME CHECK IS CLEARED
      ;CONTENTS OF TSSR FOR ERROR REPORT
      ;VOLUME CHECK NOT CLEARED
      TRAP C$ERHRD
      .WORD 511
      .WORD T5NVCK
      .WORD PKTMES
    
```



```

5365
5366
5367
5368
5369
5370
5371
5372
5373
5374 032126          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
      032126          ;              T6.3:
      032126 104402          TRAP      C$BSUB
5375
5376 032130          SETPRI  #PRI00          ;LOWER PRIORITY TO ALLOW INTERRUPTS
      032130 012700 000000          MOV      #PRI00,R0
      032134 104441          TRAP      C$SPRI
5377 032136 012703 032372          MOV      #T62DATA,R3          ;START OF TEST DATA FOR SUBTEST
5378 032142 012704 032330          MOV      #T6PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
5379 032146 004737 033426          JSR      PC,T6REST          ;RESTORE PACKET TO STARTING VALUES
5380
5381
5382 032152 004737 016470          JSR      PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
5383 032156 103405          BCS     10$          ;BR IF SOFT INIT = OK
5387 032160 010001          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
5388 032162          ERRDF  ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL ERROR DURING INIT
      032162 104455          TRAP      C$ERDF
      032164 001141          .WORD    609
      032166 003550          .WORD    SFIERR
      032170 011506          .WORD    SFIMSG
5389 032172 005037 002172          CLR      INTRECV          ;CLEAR INTERRUPT RECEIVED FLAG
5390 032176 052737 000001 032340 10$:  BIS     #1,T6DATA          ;MAKE ADDRESS ODD
5391 032204 010465 177776          MOV      R4,TSDB(R5)          ;SET THE PACKET ADDRESS
5392 032210 004737 016744          JSR      PC,WAITF          ;WAIT FOR SSR TO SET
5393 032214 103405          BCS     15$          ;BR IF CARRY SET (GOOD RETURN)
5394 032216 010001          MOV      R0,R1          ;SAVE CONTENTS OF TSSR
5398 032220          ERRDF  ERRNO,T6SSR,PKTSSR          ;DEVICE FATAL SSR FAILED TO SET
      032220 104455          TRAP      C$ERDF
      032222 001142          .WORD    610
      032224 033047          .WORD    T6SSR
      032226 011520          .WORD    PKTSSR
5399 032230          15$:  CKLOOP          ;LOOP ON ERROR, IF FLAG SET
      032230 104406          TRAP      C$CLP1
5400 032232          ESCAPE  SUB          ;BY-PASS SUBTEST IF FATAL ERROR
      032232 104410          TRAP      C$ESCAPE
      032234 000056          .WORD    L10060-.
5401 032236 005737 002172          TST     INTRECV          ;DID AN INTERRUPT OCCUR ?
5402 032242 001004          BNE     22$          ;BRANCH IF YES
5406 032244          ERRHRD  ERRNO,T6NINT,PKTSSR
      032244 104456          TRAP      C$ERHRD
      032246 001143          .WORD    611
      032250 033136          .WORD    T6NINT
      032252 011520          .WORD    PKTSSR
5407 032254 016501 000000          22$:  MOV      TSSR(R5),R1          ;GET THE CONTENTS OF TSSR
5408 032260 012702 102206          MOV      #SC!SSR!TSREJ!NBA,R2          ;EXPECTED CONTENTS OF TSSR
5409 032264 032701 000100          BIT     #OFL,R1          ;IS OFF-LINE BIT SET ?
5410 032270 001402          BEQ     25$          ;BRANCH IF NOT OFF-LINE
5411 032272 052702 000100          BIS     #OFL,R2          ;SET OFF-LINE IN EXPECTED DATA
  
```

```
5412 032276 020201          25$:  CMP      R2,R1          ;DOES EXPECTED MATCH RECEIVED ?
5413 032300 001404          BEQ      30$          ;OKAY IF MATCH
5417 032302          ERRHRD  ERRNO,T64REJ,PKTSSR ;COMMAND NOT REJECTED
      032302 104456          TRAP    C$ERHRD
      032304 001144          .WORD  612
      032306 032653          .WORD  T64REJ
      032310 011520          .WORD  PKTSSR
5418 032312          30$:
5419
5420 032312          ENDSUB          ;////////////////// END SUBTEST ////////////////////
      032312          L10060:
5421 032312 104403          TRAP    C$ESUB
```

```

5423
5424 032314          EXIT   TST           ;ALL DONE THIS TEST
      032314 104432
      032316 001162          TRAP   C$EXIT
                                .WORD L10055-.
5425
5426
5427      ;+
5428      ;LOCAL STORAGE FOR THIS TEST
5429      ;-
5431 032320          .BLKB   10-<.-TUV2A&7>
5433 032330          T6PACKET:          ;COMMAND PACKET FOR TEST
5434 032330 100204          .WORD   100204          ;WRITE CHAR COMMAND, WITH IE, ACK
5435 032332 032340          .WORD   T6DATA          ;ADDRESS OF CHARACTERISTICS BLOCK
5436 032334 000000          .WORD   0
5437 032336 000010          .WORD   8.           ;STARTING VALUE OF BLOCK SIZE
5438
5439 032340          T6DATA:           ;CHARACTERISTICS DATA BLOCK
5440 032340 032352          .WORD   T6BFR          ;ADDRESS OF MESSAGE BUFFER
5441 032342 000000          .WORD   0
5442 032344 000016          .WORD   14.          ;LENGTH OF MESSAGE BUFFER
5443 032346 000000 000000 .WORD   0,0
5444
5445 032352          T6BFR:   .BLKW   8.           ;MESSAGE BUFFER
5446
5447      ;+
5448      ;
5449      ;TEST DATA FOR SUBTEST TWO
5450      ;
5451      ;DATA HAS FORMAT:
5452      ;
5453      ;           1ST WORD      OFFSET TO TEST WORD IN PACKET
5454      ;           2ND WORD      BITS TO SET FOR TEST
5455      ;
5456      ;-
5457
5458 032372          T62DATA:
5459 032372 000000 036140          .WORD   0,BIT5!BIT6!BIT6!BIT10!BIT11!BIT12!BIT13
5460 032376 000002 000001          .WORD   2,BIT0
5461 032402 000004 100100          .WORD   4,BIT6!BIT15
5462          T62DONE=.
5463
5464
5465      ;+
5466      ;LOCAL TEXT MESSAGES FOR TEST
5467      ;-
5468
5469 032406          127      122      111  T6NBA:  .ASCIZ  'WRITE CHARACTERISTICS Command Not Accepted'
5470 032461          127      122      111  T62REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Non-Zero Unused Fields'
5471 032560          127      122      111  T63REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Invalid Data Count'
5472 032653          127      122      111  T64REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Invalid Block Address'
5473 032751          127      122      111  T65REJ: .ASCIZ  'WRITE CHARACTERISTICS Not Rejected With Invalid Buffer Length'
5474 033047          103      157      156  T6SSR:  .ASCIZ  'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
5475 033136          105      170      160  T6NINT: .ASCIZ  'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
5476 033227          125      156      145  T6INT:  .ASCIZ  'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
5477 033316          111      156      143  T6TSBA: .ASCIZ  'Incorrect TSBA Address After WRITE CHARACTERISTICS'
5478 033401          103      157      155  TST6ID: .ASCIZ  'Completion Interrupt'
5479          .EVEN
  
```

```
5481
5482
5483
5484
5485
5486
5487
5488 033426
5489 033426
5490 033432 012701 032330
5491 033436 012721 100204
5492 033442 012721 032340
5493 033446 005021
5494 033450 012721 000010
5495 033454 012721 032352
5496 033460 005021
5497 033462 012721 000016
5498 033466 005021
5499 033470 005011
5500 033472 005037 032352
5501 033476 000207
5502 033500
      033500
      033500 104401

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

T6REST:
      SAVREG
      MOV #T6PACKET,R1 ;SAVE THE REGISTERS
      MOV #100204,(R1)+ ;START OF THE PACKET
      MOV #T6DATA,(R1)+ ;WRITE CHARACTERISTICS WITH ACK, IE
      CLR (R1)+ ;ADDRESS OF CHAR DATA BLOCK
      MOV #8,(R1)+ ;EXTENDED ADDRESS
      MOV #T6BFR,(R1)+ ;SIZE OF DATA BLOCK IN BYTES
      CLR (R1)+ ;ADDRESS OF MESSAGE BUFFER
      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
```



```

5982
5983      ;+
5984      ;LOCAL STORAGE FOR THIS TEST
5985      ;-
5987      .BLKB   10-<.-TUV2A&7>
5989      035640
5990      035640 100204
5991      035642 035650
5992      035644 000000
5993      035646 000010
5994
5995      035650
5996      035650 035662
5997      035652 000000
5998      035654 000016
5999      035656 000000 000000
6000
6001      035662
6002
6003
6004
6005      ;+
6006      ;TEST DATA FOR SUBTEST FOUR
6007      ;
6007      035702
6008      035702 100204
6009      035704 035712
6010      035706 000000
6011      035710 000010
6012
6013      035712
6014      035712 035724
6015      035714 000000
6016      035716 000016
6017      035720 000000 000000
6018
6019      035724
6020
6021
6022      ;+
6023      ;LOCAL TEXT MESSAGES FOR TEST
6024      ;-
6025
6026      035744 115 145 163 T7MBF: .ASCIZ 'Message Buffer Modified after MESSAGE BUFFER RELEASE Command'
6027      036041 116 102 101 T7NBA: .ASCIZ 'NBA Not Clear After WRITE CHARACTERISTICS Command'
6028      036123 116 102 101 T7NNBA: .ASCIZ 'NBA Set After MESSAGE BUFFER RELEASE Command'
6029
6030      036200 103 157 156 T7SSRM: .ASCIZ 'Contents Of TSSR Incorrect After Message Buffer Release'
6031      036270 103 157 156 T7SSR: .ASCIZ 'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
6032      036357 105 170 160 T7NINT: .ASCIZ 'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
6033      036450 125 156 145 T7INT: .ASCIZ 'Unexpected Interrupt Received On WRITE CHARACTERISTICS'
6034      036537 102 141 163 TST7ID: .ASCIZ 'Basic Packet Protocol'
6035      .EVEN
6036
  
```

6038
 6039
 6040
 6041
 6042
 6043
 6044
 6045 036566
 6046 036566
 6047 036572 012701 035640
 6048 036576 012721 100204
 6049 036602 012721 035650
 6050 036606 005021
 6051 036610 012721 000010
 6052 036614 012721 035662
 6053 036620 005021
 6054 036622 012721 000016
 6055 036626 005021
 6056 036630 005011
 6057 036632 005037 035662
 6058 036636 000207
 6059
 6060
 6061
 6062
 6063
 6064
 6065 036640
 6066 036640
 6067 036644 012701 035702
 6068 036650 012721 100204
 6069 036654 012721 035712
 6070 036660 005021
 6071 036662 012721 000010
 6072 036666 012721 035724
 6073 036672 005021
 6074 036674 012721 000016
 6075 036700 005021
 6076 036702 005011
 6077 036704 005037 035724
 6078 036710 000207
 6079 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


```

6251 037436          EXIT   TST           ;ALL DONE THIS TEST
      037436 104432
      037440 000770          TRAP   C$EXIT
                                .WORD  L10066-.

6252
6253
6254      ;+
6255      ;LOCAL STORAGE FOR THIS TEST
6256      ;-

6258 037442          .BLKB  10-<.-TUV2A&7>
6260 037450          T8PACKET:
6261 037450 100204          .WORD  100204          ;COMMAND PACKET FOR TEST
6262 037452 037460          .WORD  T8DATA          ;WRITE CHAR COMMAND, WITH IE, ACK
6263 037454 000000          .WORD  0              ;ADDRESS OF CHARACTERISTICS BLOCK
6264 037456 000010          .WORD  8.              ;STARTING VALUE OF BLOCK SIZE
6265
6266 037460          T8DATA:
6267 037460 037472          .WORD  T8BFR          ;CHARACTERISTICS DATA BLOCK
6268 037462 000000          .WORD  0              ;ADDRESS OF MESSAGE BUFFER
6269 037464 000016          .WORD  14.            ;LENGTH OF MESSAGE BUFFER
6270 037466 000000 000000  .WORD  0,0
6271
6272 037472          T8BFR:  .BLKW  8.          ;MESSAGE BUFFER
6273
6274
6276 037512          .BLKB  10-<.-TUV2A&7>
6278 037520          T8PK2:
6279 037520 100204          .WORD  100204          ;COMMAND PACKET FOR TEST
6280 037522 037530          .WORD  T8DTA          ;WRITE CHAR COMMAND, WITH IE, ACK
6281 037524 000000          .WORD  0              ;ADDRESS OF CHARACTERISTICS BLOCK
6282 037526 000010          .WORD  8.              ;STARTING VALUE OF BLOCK SIZE
6283
6284 037530          T8DTA:
6285 037530 037542          .WORD  T8BF2          ;CHARACTERISTICS DATA BLOCK
6286 037532 000000          .WORD  0              ;ADDRESS OF MESSAGE BUFFER
6287 037534 000016          .WORD  14.            ;LENGTH OF MESSAGE BUFFER
6288 037536 000000 000000  .WORD  0,0
6289
6290 037542          T8BF2:  .BLKW  8.          ;MESSAGE BUFFER
6291
6292
6293
6294
6295      ;+
6296      ;LOCAL TEXT MESSAGES FOR TEST
6297      ;-

6298 037562          111    116    111  T8NBA:  .ASCIZ  'INITIALIZE Command Not Accepted'
6299 037622          111    116    111  T82REJ: .ASCIZ  'INITIALIZE Not Rejected With Non-Zero Mode Field'
6300 037703          107    105    124  T83REJ: .ASCIZ  'GET STATUS Not Accepted'
6301 037733          107    105    124  T84REJ: .ASCIZ  'GET STATUS Not Rejected With Non-Zero Mode Field'
6302 040014          103    157    156  T8SSR:  .ASCIZ  'Contents of TSSR Incorrect After INITIALIZE'
6303 040070          103    157    156  T8SR2:  .ASCIZ  'Contents of TSSR Incorrect After GET STATUS'
6304 040144          105    170    160  T8NINT: .ASCIZ  'Expected Interrupt Not Received On INITIALIZE'
6305 040222          111    156    143  T8TSBA: .ASCIZ  'Incorrect TSBA Address After INITIALIZE'
6306 040272          116    157    156  T8T8ID: .ASCIZ  'Non-Tape Motion Commands'
6307          .EVEN
6308
    
```

```
6310
6311
6312
6313
6314
6315
6316
6317
6318 040324
6319 040324
6320 040330 012701 037450
6321 040334 012721 100213
6322 040340 005021
6323 040342 005021
6324 040344 005021
6325 040346 005021
6326 040350 005021
6327 040352 005021
6328 040354 005021
6329 040356 005011
6330 040360 005037 037472
6331 040364 000207
6332
6333
6334
6335
6336
6337
6338
6339 040366
6340 040366
6341 040372 012701 037450
6342 040376 012721 100217
6343 040402 005021
6344 040404 005021
6345 040406 005021
6346 040410 005021
6347 040412 005021
6348 040414 005021
6349 040416 005021
6350 040420 005011
6351 040422 005037 037472
6352 040426 000207
6353 040430
      040430
      040430 104401

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;INITIALIZE COMMAND
;
;-
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 T8BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
  RTS PC ;RETURN

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;GET STATUS COMMAND
;
;-
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 T8BFR ;CLEAR 1ST LOC IN MESSAGE BUFFER
  RTS PC ;RETURN
  ENDTST

L10066: TRAP C$ETST
```

6356
 6357
 6358
 6359
 6360
 6361
 6362
 6363
 6364
 6365
 6366
 6367
 6368
 6369
 6370
 6371
 6372
 6373
 6374
 6375
 6376
 6377
 6378
 6379
 6380
 6381
 6382
 6383
 6384
 6385
 6390
 6391
 6392
 6393
 6394

.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
;
; --
  
```

BGNTST

```

          T9::
          ;CLEAR FATAL ERROR FLAG
          ;SET UP ERROR MESSAGE SWITCH
          ;HOLD OFF KT11
          ;ASCII MESSAGE TO IDENTIFY TEST
          ;DO INITIAL TEST SETUP
          ;PERFORM 2 ITERATIONS
          ;LOOP ON TEST LABEL
  
```

040432
 040432 005037 002170
 040436 012737 005672 002146
 040444 005037 003100
 040450 012700 042100
 040454 004737 017232
 040460 012737 000002 002164

T9LOOP:

```

6396 .SBTTL TEST 9: SUBTEST 1: GET STATUS SELECTED LOCATIONS
6397 ;**
6398 ; TEST 9: SUBTEST 1:
6399 ;
6400 ; SUBTEST DESCRIPTION:
6401 ;
6402 ; This subtest verifies the controller can fetch a get status
6403 ; command from all available memory locations.
6404 ; Two word blocks are tested one at a time by first setting
6405 ; all available memory to a background pattern of 125252.
6406 ; A Get Status command is then executed to various addresses in
6407 ; each available memory 4k word block. The various addresses
6408 ; are determined by floating a 1 then a 0 through the address bits.
6409 ;
6410 ; TEST STEPS:
6411 ;
6412 ; BEGIN
6413 ; Fill Memory with background pattern of 125252
6414 ; Write to TSSR to soft initialize
6415 ; Do a WRITE CHARACTERISTICS to setup a message buffer
6416 ;
6417 ; REPEAT FOR SELECTED VALID ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
6418 ; BEGIN
6419 ; Get a valid modulo-4 test address
6420 ; Do a GET STATUS command from the test address
6421 ; END
6422 ; END
6423 ;--
6424
6425 040466 BGNSUB ;////////////////// BEGIN SUBTEST ////////////////////
        040466 ; T9.1: TRAP C$BSUB
        040466 104402
6426
6427 ;Fill Memory with background pattern of 125252
6428 040470 012700 125252 MOV #125252,R0 ;BACKGROUND DATA
6429 040474 004737 020216 JSR PC,FILLMEM ;FILL MEMORY WITH BACKGROUND DATA
6430
6431 ;Write to TSSR to soft initialize
6432 040500 004737 016470 JSR PC,SOFINIT ;DO SOFT INIT OF CONTROLLER
6433 040504 103405 BCS 15$ ;BR IF SOFT INIT = OK
6434 040506 NEXT.ERRNO
6435 040506 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
6436 040510 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL ERROR DURING INIT
        040510 104455 TRAP C$ERDF
        040512 001605 .WORD 901
        040514 003550 .WORD SFIERR
        040516 011506 .WORD SFIMSG
6437
6438 ;Do a WRITE CHARACTERISTICS to setup a message buffer
6439 040520 15$:
6440 040520 012704 041670 MOV #T9PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
6441 040524 004737 043436 JSR PC,T9SWRT ;RESTORE PACKET TO STARTING VALUES
6442 040530 005037 003102 CLR KTENABLE ;TURN OFF KT-11
6443 040534 010465 177776 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS
6444 040540 004737 017060 JSR PC.CHKTSSR ;WAIT FOR SSR TO SET
6445 040544 FORCERROR 17$
6446 040560 103405 BCS 20$ ;BR IF SSR SET IN CHKTSSR
    
```

```

6447 040562 010001          MOV     R0,R1          ;SAVE CONTENTS OF TSSR
6448 040564                NEXT.ERRNO
6449 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
6450
6451
6452          ;Verify a Get Status can be fetched from each address
6453          ;Get a valid modulo-4 test address
        ;Do a GET STATUS command from the test address
6454 040574 005037 002170      20$:   CLR     FATFLG          ;CLEAR FATAL ERROR FLAG
6455 040600 005037 041740          CLR     T9KT           ;TEST ABOVE 28K SWITCH
6456 040604 012702 041744          MOV     #T9BLK,R2      ;POINT TO TEST PATTERN TABLE
6457 040610
        T91LOOP:
6458 040610 005037 003102          CLR     KTENABLE      ;TURN OFF ABOVE 28K TEST FLAG
6459 040614 012201                MOV     (R2)+,R1      ;GET TEST PATTERN ADDRESS
6460 040616 005000                CLR     R0            ;ASSUME NO TEST ABOVE 28K
6461 040620 005737 041740          TST     T9KT          ;TEST ABOVE 28K THIS TIME?
6462 040624 001407                BEQ     25$           ;BR IF NO
6463 040626 016200 177776          MOV     -2(R2),R0     ;GET TEST PATTERN AGAIN
6464 040632 042700 177774          BIC     #+C<A1716>,R0 ;SAVE 18 BIT ADDRESS ONLY
6465 040636 012737 000001 003102  MOV     #1,KTENABLE   ;TURN ON ABOVE 28K TEST FLAG
6466 040644 004737 042746      25$:   JSR     PC,T9CONVERT ;CONVERT TEST PATTERN TO TEST ADDRESS
6467 040650 103034                BCC     65$           ;BR IF INVALID PACKET ADDRESS
6468 040652 013704 041734          MOV     T9LOADD,R4    ;COPY CURRENT PACKET LOW ADDRESS
6469 040656 013703 041732          MOV     T9HIADD,R3    ;COPY CURRENT PACKET HIGH ADDRESS
6470 040662 004737 043504          JSR     PC,T9SETGET   ;SETUP CURRENT PACKET TO GET STATUS
6471 040666 042703 177774          BIC     #+C<A1716>,R3 ;SAVE ADDRESS BITS 17+16
6472 040672 050304                BIS     R3,R4         ;SETUP 18 BIT PACKET ADDRESS
6473 040674 004737 020070          JSR     PC,KTOFF      ;TURN OFF KT-11
6474 040700 010465 177776          MOV     R4,TSDDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
6475 040704 004737 017060          JSR     PC,CHKTSSR   ;WAIT FOR SSR TO SET
6476 040710
        FORCERROR 32$
6477 040724 103405                BCS     40$           ;BR IF SSR SET IN CHKTSSR
6478 040726 010001                MOV     R0,R1         ;SAVE CONTENTS OF TSSR
6479 040730                NEXT.ERRNO
6480 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
6481 040740                40$:   CKLOOP          ;LOOP ON ERROR, IF FLAG SET
        040740 104406                TRAP   C$CLP1
6482 040742
        65$:
6483 040742                FORCEXIT 80$
6484 040752 020227 042076          CMP     R2,#T9TBE     ;DONE ALL TSTBLK TEST PATTERNS?
6485 040756 103002                BHIS   70$           ;BR IF YES
6486 040760 000137 040610          JMP     T91LOOP       ;DO ANOTHER MODULO- 4 ADDRESS
6487 040764 005737 041740      70$:   TST     T9KT          ;DONE ABOVE 28K TESTING TOO?
6488 040770 003012                BGT     80$           ;BR IF YES
6489 040772 005737 003100          TST     KTFLG        ;ANY MEMORY ABOVE 28K ON SYSTEM?
6490 040776 001407                BEQ     80$           ;BR IF NO
6491 041000 012737 000001 041740  MOV     #1,T9KT       ;SET SWITCH
6492 041006 012702 041744          MOV     #T9BLK,R2    ;RESET TEST PATTERN TABLE
6493 041012 000137 040610          JMP     T91LOOP       ;DO ABOVE 28K TESTING
6494 041016 004737 020070      80$:   JSR     PC,KTOFF      ;TURN OFF KT11
    
```

6495 041022
041022
041022 104403
6496 041024 005737 002170
6497 041030 001402
6498 041032 004737 017776
6499 041036
6500

100\$:

ENDSUB

TST FATFLG
BEQ 100\$
JSR PC,CKDROP

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

```

6502          .SBTTL TEST 9: SUBTEST 2: MESSAGE PACKETS TO SELECTED LOCATIONS
6503          ;**
6504          ; TEST 9: SUBTEST 2:
6505          ;
6506          ; SUBTEST DESCRIPTION:
6507          ;
6508          ; This subtest verifies the controller can deposit message packets
6509          ; to all available memory locations.
6510          ; First all available memory is set to a background pattern
6511          ; of 125252.
6512          ; Write Characteristics commands are then executed with message
6513          ; buffer addresses set to various addresses in each available
6514          ; memory location.
6515          ; The various addresses are determined by floating a 1 then a 0
6516          ; through the address bits.
6517          ;
6518          ; TEST STEPS:
6519          ;
6520          ; BEGIN
6521          ; Fill Memory with background pattern of 125252
6522          ; Write to TSSR to soft initialize
6523          ; Do a WRITE CHARACTERISTICS to setup a message buffer to compare
6524          ;
6525          ; REPEAT FOR SELECTED ADDRESSES IN DIAGNOSTIC FREE SPACE AND ABOVE 32K
6526          ; BEGIN
6527          ; Get a valid modulo-4 test address
6528          ; Set the packet message buffer to the TEST ADDRESS
6529          ; Do a WRITE CHARACTERISTICS
6530          ; Restore the test message buffer to background pattern
6531          ; END
6532          ; END
6533          ;--
6534
6535          BGNSUB          ;//////////////// BEGIN SUBTEST //////////////////
6536          041036          T9.2:          TRAP          C$BSUB
6537          041036          104402
6538          ;Fill Memory with background pattern of 125252
6539          041040 012700 125252          MOV          #125252,R0          ;BACKGROUND DATA
6540          041044 004737 020216          JSR          PC,FILLMEM          ;FILL MEMORY WITH BACKGROUND DATA
6541          ;Write to TSSR to soft initialize
6542          041050 004737 016470          JSR          PC,SOFINIT          ;DO SOFT INIT OF CONTROLLER
6543          041054 103405          BCS          15$          ;BR IF SOFT INIT = OK
6544          041056          NEXT.ERRNO
6545          041056 010001          MOV          R0,R1          ;SAVE CONTENTS OF TSSR
6546          041060          ERRDF          ERRNO,SFIERR,SFIMSG          ;DEVICE FATAL ERROR DURING INIT
6547          041060 104455          TRAP          C$ERDF
6548          041062 001610          .WORD          904
6549          041064 003550          .WORD          SFIERR
6550          041066 011506          .WORD          SFIMSG
6551          ;Do a WRITE CHARACTERISTICS to setup a message buffer to compare
6552          041070          15$:          MOV          #T9PACKET,R4          ;GET THE ADDRESS OF COMMAND PACKET
6553          041074 004737 043436          JSR          PC,T9SWRT          ;SET PACKET TO WRITE CHARACTERISTICS
6554          041100 004737 020070          JSR          PC,KTOFF          ;TURN OFF KT-11
  
```

```

6553 041104 010465 177776      MOV      R4,TSDB(R5)      ;SET THE PACKET ADDRESS
6554 041110 004737 017060      JSR      PC,CHKTSSR      ;WAIT FOR SSR TO SET
6555 041114      FORCERROR 17$
6556 041130 103405      BCS     20$              ;BR IF SSR SET IN CHKTSSR
6557 041132 010001      MOV     RO,R1            ;SAVE CONTENTS OF TSSR
6558 041134      NEXT.ERRNO
6559 041134      17$:  ERRDF  ERRNO,T9WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      041134 104455      TRAP   C$ERDF
      041136 001611      .WORD  905
      041140 042202      .WORD  T9WRTSSR
      041142 011520      .WORD  PKTSSR

6560
6561      ;Get a valid modulo-4 test address
6562      ;Set the packet message buffer to the test address
6563      ;Do a WRITE CHARACTERISTICS
6564 041144 005037 002170      20$:  CLR      FATFLG      ;CLEAR FATAL ERROR FLAG
6565 041150 012703 041744      MOV     #T9BLK,R3      ;POINT TO TEST PATTERN TABLE
6566 041154      T92LOOP:
6567 041154 012301      MOV     (R3)+,R1        ;GET TEST PATTERN ADDRESS
6568 041156 010100      MOV     R1,R0          ;GET ADDRESS ALL "18 BITS"
6569 041160 042700 177774      BIC     #177774,R0     ;LEAVE ONLY A17 AND A16
6570 041164 042701 000001      BIC     #1,R1          ;ALWAYS ON A WORD BOUNDRY
6571 041170 004737 043140      JSR     PC,T9CT2       ;CONVERT TEST PATTERN TO TEST ADDRESS
6572 041174 103402      BCS     25$              ;BR IF VALID MESSAGE BUFFER ADDRESS
6573 041176 000137 041274      JMP     150$           ;GET ANOTHER TEST PATTERN TO TRY
6574 041202 012704 041670      25$:  MOV     #T9PACKET,R4 ;SET THE COMMAND PACKET ADDRESS
6575 041206 004737 043436      JSR     PC,T9SWRT      ;SETUP T9PACKET TO WRITE CHAR.
6576 041212 013737 041734 041700  MOV     T9LOADD,T9DATA ;SETUP LOW ORDER MESSAGE BUFFER ADD.
6577 041220 013737 041732 041702  MOV     T9HIADD,T9DATA+2 ;SETUP HIGH ORDER MESSAGE BUFFER ADD.
6578 041226 004737 020070      JSR     PC,KTOFF       ;TURN OFF KT-11
6579 041232 010465 177776      MOV     R4,TSDB(R5)   ;SET THE PACKET ADDRESS TO EXECUTE
6580 041236 004737 017060      JSR     PC,CHKTSSR    ;WAIT FOR SSR TO SET
6581 041242      FORCERROR 32$
6582 041256 103405      BCS     50$              ;BR IF SSR SET IN CHKTSSR
6583 041260 010001      MOV     RO,R1            ;SAVE CONTENTS OF TSSR
6584 041262      NEXT.ERRNO
6585 041262      32$:  ERRDF  ERRNO,T9WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
      041262 104455      TRAP   C$ERDF
      041264 001612      .WORD  906
      041266 042202      .WORD  T9WRTSSR
      041270 011520      .WORD  PKTSSR

6586 041272      50$:  CKLOOP      ;LOOP ON ERROR, IF FLAG SET
      041272 104406      TRAP   C$CLP1

6587 041274      150$:
6588 041274      FORCEXIT 160$
6589 041304 020327 042076      CMP     R3,#T9TBE     ;DONE ALL TST9BLK TEST PATTERNS?
6590 041310 103002      BHIS   160$           ;BR IF YES
6591 041312 000137 041154      JMP     T92LOOP       ;DO ANOTHER MODULO- 4 ADDRESS
6592 041316 004737 020070      160$: JSR     PC,KTOFF       ;TURN OFF KT11
6593 041322      ENDSUB              ;////////////////// END SUBTEST ////////////////////
      041322      L10073:
      041322 104403      TRAP   C$ESUB
6594 041324 005737 002170      TST     FATFLG        ;ANY FATAL ERRORS ?
6595 041330 001402      BEQ    180$           ;BRANCH IF NOT
6596 041332 004737 017776      JSR     PC,CKDROP     ;TRY TO DROP THE UNIT
6597 041336      180$:
    
```

```

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

```

6650 041404 005037 003102          CLR      KTENABLE          ;TURN OFF ABOVE 28K TEST FLAG
6651 041410 012301                   MOV      (R3)+,R1          ;GET TEST PATTERN ADDRESS
6652 041412 010100                   MOV      R1,R0             ;GET ADDRESS ALL "18 BITS"
6653 041414 042700 177774           BIC      #177774,R0        ;LEAVE ONLY A17 AND A16
6654 041420 042701 000003           BIC      #3,R1             ;GET RID OF A17 AND A16
6655 041424 005737 041740           TST      T9KT              ;TEST ABOVE 28K THIS TIME?
6656 041430 001407                   BEQ      25$               ;BR IF NO
6657 041432 016300 177776           MOV      -2(R3),R0         ;GET TEST PATTERN AGAIN
6658 041436 042700 177774           BIC      #C<A1716>,R0      ;SAVE 18 BIT ADDRESS ONLY
6659 041442 012737 000001 003102   MOV      #1,KTENABLE       ;TURN ON ABOVE 28K TEST FLAG
6660 041450 004737 042746           JSR      PC,T9CONVERT      ;CONVERT TEST PATTERN TO TEST ADDRESS
6661 041454 103402                   BCS      30$               ;BR IF VALID TEST ADDRESS
6662 041456 000137 041560           JMP      60$               ;GET NEXT TEST PATTERN
6663                                     ;Set the test packet characteristics data pointer to the test address
6664 041462 012704 041670           30$:   MOV      #T9PACKET,R4  ;GET THE ADDRESS OF COMMAND PACKET
6665 041466 004737 043436           JSR      PC,T9SWRT         ;RESTORE PACKET TO STARTING VALUES
6666 041472 013764 041734 000002   MOV      T9LOADD,PKLOW(R4) ;STORE CHAR. DATA PTR LOW ADDRESS
6667 041500 013764 041732 000004   MOV      T9HIADD,PKHI(R4) ;STORE CHAR. DATA PTR HIGH ADDRESS
6668 041506 004737 043546           JSR      PC,T9CHAR         ;STORE EXPECTED DATA IN DATA BLOCK
6669                                     ;Do a WRITE CHARACTERISTIC command
6670 041512 004737 020070           JSR      PC,KTOFF          ;TURN OFF KT-11
6671 041516 010465 177776           MOV      R4,TSDB(R5)       ;SET THE PACKET ADDRESS TO EXECUTE
6672 041522 004737 017060           JSR      PC,CHKTSSR        ;WAIT FOR SSR TO SET
6673 041526                   FORCERROR 32$
6674 041542 103405                   BCS      40$               ;BR IF SSR SET IN CHKTSSR
6675 041544 010001                   MOV      R0,R1             ;SAVE CONTENTS OF TSSR
6676 041546                   NEXT,ERRNO
6677 041546           32$:   ERRDF  ERRNO,T9WRTSSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
                                           TRAP    C#ERDF
                                           .WORD  908
                                           .WORD  T9WRTSSR
                                           .WORD  PKTSSR
6678 041556           40$:   CKLOOP                    ;LOOP ON ERROR, IF FLAG SET
                                           TRAP    C#CLP1
6679 041560           60$:
6680 041560 020327 042076           CMP      R3,#T9TBE         ;DONE ALL TSTBLK TEST PATTERNS?
6681 041564 103002                   BHIS    65$                ;BR IF YES
6682 041566 000137 041404           JMP      T93LOOP           ;DO ANOTHER MODULO- 4 ADDRESS
6683 041572 005737 041740           65$:   TST      T9KT         ;DONE ABOVE 28K TESTING TOO?
6684 041576 003012                   BGT     70$                ;BR IF YES
6685 041600 005737 003100           TST     KTFLG              ;ANY MEMORY ABOVE 28K ON SYSTEM?
6686 041604 001407                   BEQ     70$                ;BR IF NO
6687 041606 012737 000001 041740   MOV     #1,T9KT           ;SET SWITCH
6688 041614 012703 041744           MOV     #T98LK,R3         ;RESET TEST PATTERN TABLE
6689 041620 000137 041404           JMP     T93LOOP           ;DO ABOVE 28K TESTING
6690 041624 004737 020070           70$:   JSR      PC,KTOFF      ;TURN OFF KT11
6691 041630                   ENDSUB                    ;////////// END SUBTEST ////////////
                                           L10074:
6692 041632 005737 002170           TST     FATFLG            ;ANY FATAL ERRORS ?
                                           TRAP    C#ESUB
6693 041636 001402                   BEQ     75$                ;BRANCH IF NOT
6694 041640 004737 017776           JSR     PC,CKDROP         ;TRY TO DROP THE UNIT
6695 041644           75$:
6696 041644 004737 017200           100$:  JSR     PC,TSTLOOP     ;SHOULD WE DO ITERATIONS?
6697 041650 103002                   BCC     105$              ;BR IF NO
6698 041652 000137 040466           JMP     T9LOOP            ;LOOP UNTIL ITERATION COUNT DONE
6699 041656           105$:
    
```

```

6700 041656 004737 020070      JSR      PC,KTOFF      ;TURN OFF MEMORY MANAGEMENT
6701 041662                      EXIT     TST           ;ALL DONE THIS TEST
      041662 104432
      041664 001724
6702
6703
6704
6705      ;*
6706      ;LOCAL STORAGE FOR THIS TEST
6707      ;-
6709 041666                      .BLKB   10-<.-TUV2A&7>
6711 041670                      T9PACKET:
6712 041670 100004                .WORD   100004      ;COMMAND PACKET FOR TEST
6713 041672 041700                .WORD   T9DATA      ;WRITE CHARACTERISTICS COMMAND, WITH ACK
6714 041674 000000                .WORD   0           ;ADDRESS OF CHARACTERISTICS BLOCK
6715 041676 000010                .WORD   8           ;STARTING VALUE OF BLOCK SIZE
6716
6717 041700                      T9DATA:
6718 041700 041712                .WORD   T9BFR      ;CHARACTERISTICS DATA BLOCK
6719 041702 000000                .WORD   0           ;LOW ADDRESS OF MESSAGE BUFFER
6720 041704 000016                .WORD   14.        ;HIGH ORDER OF MESSAGE BUFFER
6721 041706 000000 000000        .WORD   0,0       ;LENGTH OF MESSAGE BUFFER
6722
6723 041712                      T9BFR:  .BLKW   8.   ;MESSAGE BUFFER
6724
6725 041732 000000                T9HIADD: .WORD   0   ;HIGH ADDRESS
6726 041734 000000                T9LOADD: .WORD   0   ;LOW ADDRESS
6727 041736 000000                T9PAR6:  .WORD   0   ;ADDRESS IN PAR FORMAT
6728 041740 000000                T9KT:    .WORD   0   ;TEST ABOVE 28K SWITCH
6729 041742 000000                T94TST: .WORD   0   ;ADDRESS TEST BIT
6730
6731      ;*
6732      ;
6733      ;TABLE OF ADDRESSES
6734      ;
6735      ;-
6735 041744 000001                T9BLK:  .WORD   000001
6736 041746 000002                .WORD   000002
6737 041750 000003                .WORD   000003
6738 041752 000005                .WORD   000005
6739 041754 000006                .WORD   000006
6740 041756 000007                .WORD   000007
6741 041760 000011                .WORD   000011
6742 041762 000012                .WORD   000012
6743 041764 000013                .WORD   000013
6744 041766 000021                .WORD   000021
6745 041770 000022                .WORD   000022
6746 041772 000023                .WORD   000023
6747 041774 000041                .WORD   000041
6748 041776 000042                .WORD   000042
6749 042000 000043                .WORD   000043
6750 042002 000101                .WORD   000101
6751 042004 000102                .WORD   000102
6752 042006 000103                .WORD   000103
6753 042010 000201                .WORD   000201
6754 042012 000202                .WORD   000202
6755 042014 000203                .WORD   000203
6756 042016 000401                .WORD   000401
    
```

6757	042020	000402	.WORD	000402
6758	042022	000403	.WORD	000403
6759	042024	001001	.WORD	001001
6760	042026	001002	.WORD	001002
6761	042030	001003	.WORD	001003
6762	042032	002001	.WORD	002001
6763	042034	002002	.WORD	002002
6764	042036	002003	.WORD	002003
6765	042040	004001	.WORD	004001
6766	042042	004002	.WORD	004002
6767	042044	004003	.WORD	004003
6768	042046	010001	.WORD	010001
6769	042050	010002	.WORD	010002
6770	042052	010003	.WORD	010003
6771	042054	020001	.WORD	020001
6772	042056	020002	.WORD	020002
6773	042060	020003	.WORD	020003
6774	042062	040001	.WORD	040001
6775	042064	040002	.WORD	040002
6776	042066	040003	.WORD	040003
6777	042070	100001	.WORD	100001
6778	042072	100002	.WORD	100002
6779	042074	100003	.WORD	100003
6780	042076	177777	T9TBE: .WORD	177777

6781 ;*
6782 ;LOCAL TEXT MESSAGES FOR TEST
6783 ;-
6784

6785	042100	104	115	101	T9T9ID: .ASCIZ	'DMA Memory Addressing'
6786	042126	103	157	156	T9GETSSR: .ASCIZ	'Contents of TSSR Incorrect After GET STATUS'
6787	042202	103	157	156	T9WRTSSR: .ASCIZ	'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
6788	042271	115	145	163	T9MSGBUF: .ASCIZ	'Message Buffer Contents Incorrect After WRITE CHARACTERISTICS'
6789	042367	102	141	143	T9BKGND: .ASCIZ	'Background Pattern Disturbed By WRITE CHARACTERISTICS'
6790	042455	105	170	160	T9NINT: .ASCIZ	'Expected Interrupt Not Received On WRITE CHARACTERISTICS'
6791	042546	127	162	151	T9DPR: .ASCIZ	'Write Characteristic data in ram does not match expected'
6792	042637	124	123	123	T9NXM: .ASCIZ	'TSSR NXM bit failed to set when non-existent memory address specifi

ed'

6793 .EVEN
6794

```

6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808
6809
6810
6811
6812
6813
6814
6815
6816 042746
6817 042746
6818 042752 005037 041734
6819 042756 005037 041732
6820 042762 005037 041736
6821 042766 042701 170000
6822 042772 010005
6823 042774 004737 020070
6824 043000 013702 003072
6825 043004 062702 000020
6826 043010 060102
6827 043012 042702 000003
6828 043016 013703 003076
6829 043022 162703 000020
6830 043026 010237 041734
6831 043032 010237 041736
6832 043036 020203
6833 043040 101007
6834 043042 020237 003072
6835 043046 103007
6836 043050 005737 003102
6837 043054 001004
6838 043056 000424
6839 043060 162702 000020
6840 043064 000754
6841 043066
6842 043066 005737 003102
6843 043072 001420
6844 043074 005737 003100
6845 043100 001413
6846 043102 004737 020052
6847 043106 010500
6848 043110 010037 041732
6849 043114 010201
6850 043116 004737 020112
6851 043122 010037 041736
6852 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
    
```

6853	043130	000241	90\$:	CLC			;CLR C BIT FOR FAILURE
6854	043132	000401		BR	105\$:
6855	043134	000261	100\$:	SEC			;SET SUCCESS
6856	043136	000207	105\$:	RTS	PC		;RETURN
6857							
6858							

6860
 6861
 6862
 6863
 6864
 6865
 6866
 6867
 6868
 6869
 6870
 6871
 6872
 6873
 6874
 6875
 6876
 6877
 6878
 6879
 6880
 6881 043140
 6882 043140
 6883 043144 005037 041734
 6884 043150 005037 041732
 6885 043154 005037 041736
 6886 043160 042701 170000
 6887 043164 010005
 6888 043166 004737 020070
 6889 043172 013702 003072
 6890 043176 062702 000020
 6891 043202 060102
 6892 043204 013703 003076
 6893 043210 162703 000020
 6894 043214 010237 041734
 6895 043220 010237 041736
 6896 043224 020203
 6897 043226 101007
 6898 043230 020237 003072
 6899 043234 103007
 6900 043236 005737 003102
 6901 043242 001004
 6902 043244 000424
 6903 043246 162702 000020
 6904 043252 000754
 6905 043254
 6906 043254 005737 003102
 6907 043260 001420
 6908 043262 005737 003100
 6909 043266 001413
 6910 043270 004737 020052
 6911 043274 010500
 6912 043276 010037 041732
 6913 043302 010201
 6914 043304 004737 020112
 6915 043310 010037 041736
 6916 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
;   C BIT = 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   #1C<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
  
```

6917 043316 000241	90\$:	CLC			;CLR C BIT FOR FAILURE
6918 043320 000401		BR	105\$;
6919 043322 000261	100\$:	SEC			;SET SUCCESS
6920 043324 000207	105\$:	RTS	PC		;RETURN
6921					

```

6923      ;+
6924      ;
6925      ;ROUTINE TO READ THE FIRST 2 BYTES FROM RAM
6926      ;MEMORY AND COMPARE THIS DATA TO A COMMAND PACKET.
6927      ;
6928      ;INPUT:
6929      ;
6930      ;      R4      ADDRESS OF THE COMMAND PACKET
6931      ;      R5      FIRST DEVICE UNIBUS ADDRESS
6932      ;
6933      ;OUTPUT:
6934      ;
6935      ;      CARRY   SET - RAM MATCHES PACKET
6936      ;              CLR - RAM DOES NOT MATCH PACKET
6937      ;
6938      ;IMPLICIT OUTPUT:
6939      ;
6940      ;      THE TABLE RAMDATA IS FILLED WITH THE
6941      ;      DATA HELD IN RAM.
6942      ;      RAMSIZ  SET TO 2 FOR PRAMPKT ROUTINE
6943      ;
6944      ;SIDE EFFECTS:
6945      ;
6946      ;      THE SUBSYSTEM IS LEFT IN MAINTENANCE MODE
6947      ;
6948      ;-
6949      T9CKRAM::
6950      SAVREG
6951      MOV      #RAMDATA,R1      ;SAVE THE GENERAL REGISTERS
6952      MOV      #RMPKTBEG,R2    ;ADDRESS TO SAVE THE RAM DATA
6953      CLR      R3              ;BYTE ADDRESS OF FIRST RAM DATA
6954      JSR      PC,CHKTSSR      ;CLEAR THE ERROR FLAG
6955      MOVB    #0,TSDB(R5)      ;WAIT FOR SSR
6956      JSR      PC,CHKTSSR      ;SET MAINTENANCE MODE
6957      MOV     R2,TSDB(R5)      ;WAIT FOR SSR TO SET
6958      JSR      PC,CHKTSSR      ;SELECT NEXT RAM ADDRESS
6959      MOVB    TSBA(R5),(R1)    ;WAIT FOR SSR TO SET
6960      CMPB    (R1)+,(R4)+      ;READ THE RAM DATA
6961      BEQ     20$              ;COMPARE TO EXPECTED
6962      INC     R3              ;BRANCH IF OK
6963      INC     R2              ;SET ERROR FLAG
6964      CMP     R2,#RMPKTBEG+2  ;ADDRESS OF NEXT RAM LOCATION
6965      BLT     10$              ;DONE 2 BYTES?
6966      TST     R3              ;BR IF NO
6967      BEQ     30$              ;WAS AN ERROR FOUND ?
6968      CLC     CLC              ;BRANCH IF NOT
6969      BR      50$              ;CLEAR CARRY TO SHOW ERROR
6970      SEC     SEC              ;AND EXIT
6971      MOV     #2,RAMSIZ        ;SHOW GOOD COMPARE
6972      RTS     PC              ;SETUP RAMSIZ
6973      ;+
6974      ;
6975      ;ROUTINE TO SETUP PACKET TO WRITE CHARACTERISTICS
6976      ;-
6977      T9SWRT:
6978      SAVREG
6979      ;SAVE THE REGISTERS
    
```

```

6980 043442 012701 041670      MOV      #T9PACKET,R1      ;START OF THE PACKET
6981 043446 012721 100004      MOV      #100004,(R1)+    ;WRITE CHARACTERISTICS WITH ACK
6982 043452 012721 041700      MOV      #T9DATA,(R1)+   ;ADDRESS OF CHAR DATA BLOCK
6983 043456 005021              CLR      (R1)+           ;EXTENDED ADDRESS
6984 043460 012721 000010      MOV      #8.,(R1)+       ;SIZE OF DATA BLOCK IN BYTES
6985 043464 012721 041712      MOV      #T9BFR,(R1)+   ;ADDRESS OF MESSAGE BUFFER
6986 043470 005021              CLR      (R1)+
6987 043472 012721 000016      MOV      #14.,(R1)+     ;LENGTH OF MESSAGE BUFFER
6988 043476 005021              CLR      (R1)+
6989 043500 005011              CLR      (R1)
6990 043502 000207              RTS       PC              ;RETURN
6991
6992      ;+
6993      ;
6994      ;ROUTINE TO SETUP A GET STATUS COMMAND PACKET AT CURRENT PACKET ADDRESS
6995      ;
6996      ;      R3      HIGH ORDER PACKET ADDRESS
6997      ;      R4      LOW ORDER PACKET ADDRESS
6998      ;      NOTE: R3 IS IGNORED IF KTENABLE FLAG CLEAR
6999      ;
7000      ;-
7001 043504      T9SETGET:
7002 043504      SAVREG              ;SAVE THE REGISTERS
7003 043510 010401      MOV      R4,R1          ;GET LOW ORDER ADDRESS
7004 043512 005737 003102      TST      KTENABLE      ;TESTING ABOVE 28K?
7005 043516 001404      BEQ      10$           ;BR IF NO
7006 043520 010300      MOV      R3,RO         ;GET HIGH ORDER ADDRESS
7007 043522 004737 020112      JSR      PC,SETMAP     ;RETURN ADDRESS BIASED TO PAR6 IN RO
7008 043526 010001      MOV      RO,R1         ;GET ADDRESS
7009 043530 012700 000017      10$:  MOV      #P.GETSTATUS,RO ;GET STATUS COMMAND CODE NO IE
7010 043534 052700 100000      BIS      #P.ACK,RO     ;SET ACK
7011 043540 010021      MOV      RO,(R1)+      ;STORE GET STATUS IN PACKET
7012 043542 005021      CLR      (R1)+        ;CLEAR UNUSED WORD
7013 043544 000207      RTS       PC          ;RETURN
7014
7015      ;+
7016      ;
7017      ;ROUTINE TO SETUP A CHARACTERISTIC DATA BLOCK AT A TEST ADDRESS
7018      ;
7019 043546      T9CHAR:
7020 043546      SAVREG              ;SAVE R1-R5 UNTIL NEXT RETURN
7021 043552 012700 041700      MOV      #T9DATA,RO    ;GET T9PACKET DATA POINTER
7022 043556 013701 041734      MOV      T9LOADD,R1   ;ASSUME NOT ABOVE 28K
7023 043562 005737 003102      TST      KTENABLE      ;TESTING ABOVE 28K?
7024 043566 001402      BEQ      10$           ;BR IF NO
7025 043570 013701 041736      MOV      T9PAR6,R1    ;SET TEST ADDRESS ABOVE 28K
7026 043574 012021      10$:  MOV      (RO)+,(R1)+    ;STORE DATA WORD 1
7027 043576 012021      MOV      (RO)+,(R1)+    ;STORE DATA WORD 2
7028 043600 012021      MOV      (RO)+,(R1)+    ;STORE DATA WORD 3
7029 043602 012021      MOV      (RO)+,(R1)+    ;STORE DATA WORD 4
7030 043604 012021      MOV      (RO)+,(R1)+    ;STORE DATA WORD 5
7031 043606 000207      RTS       PC          ;RETURN
7032 043610      ENDTST
      043610
      043610 104401
    
```

L10071: TRAP C\$ETST

```

7034 .SBTTL TEST 10: INITIALIZE AFTER WRITE CHARACTERISTICS
7035
7036 ;*
7037 ; TEST DESCRIPTION:
7038 ;
7039 ; This test verifies that a Hardware Initialize command
7040 ; invoked after a Write Characteristics command sets up
7041 ; the Command, Message and Characteristic image blocks
7042 ; in the controller ram correctly.
7043 ;
7044 ; TEST STEPS:
7045 ;
7046 ; REPEAT FOR LOOPCNT
7047 ; BEGIN
7048 ; Do WRITE CHARACTERISTICS command.
7049 ; If the NBA bit in the TSSR register is NOT=0 then Print Error.
7050 ; Write to TSSR register to soft initialize the controller
7051 ; If controller RAM 310-377 NOT=0 then Print Error
7052 ; END
7053 ;--
7054
7055 043612 BGNTST
7056 043612
7056 043612 005037 002170 CLR FATFLG ;CLEAR FATAL ERROR FLAG
7057 043616 012737 005672 002146 MOV #EPRT1,EPRTSW ;SET UP ERROR MESSAGE SWITCH
7058 043624 005037 003100 CLR KTLG ;HOLD OFF KT11
7059 043630 012700 044262 MOV #TST10ID,R0 ;ASCII MESSAGE TO IDENTIFY TEST
7060 043634 004737 017232 JSR PC,TSTSETUP ;DO INITIAL TEST SETUP
7061 043640 012737 000002 002164 MOV #2.,LOOPCNT ;PERFORM 2 ITERATIONS
7062 043646
7063 043646 004737 044536 JSR PC,T10REST ;SET PACKET TO START-UP VALUES
7064 043652
7065 043652 012703 002732 MOV #TSTBLK+10.,R3 ;START OF TEST DATA
7066 043656 012704 044220 MOV #T10PACKET,R4 ;GET THE ADDRESS OF COMMAND PACKET
7067 043662 012764 000010 000006 MOV #8.,PKBCNT(R4) ;START WITH MINIMUM ALLOWABLE VALUE
7068 043670
7069 043670 004737 016470 JSR PC,SOFINIT ;WRITE TO TSSR TO SOFT INITIALIZE
7070 043674 103405 BCS 10$ ;BR IF SOFT INIT OKAY
7071 043676 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
7072 043700 ERRDF ERRNO,SFIERR,SFIMSG ;DEVICE FATAL DURING INIT
7073 043700 104455 TRAP C$ERDF
7074 043702 001750 .WORD 1000
7075 043704 003550 .WORD SFIERR
7076 043706 011506 .WORD SFIMSG
7077
7078
7079 ;Do WRITE CHARACTERISTICS command.
7080 043710 005037 002170 10$: CLR FATFLG ;CLEAR FATAL ERROR FLAG
7081 043714 010465 177776 MOV R4,TSDB(R5) ;SET THE PACKET ADDRESS TO EXECUTE
7082 043720 004737 017060 JSR PC,CHKTSSR ;WAIT FOR SSR TO SET
7083 043724 FORCERROR 12$ ;@DFORCE ERROR IF FORCER=1
7084 043740 103407 BCS 15$ ;BR IF CARRY SET (GOOD RETURN)
7085 043742 010001 MOV R0,R1 ;SAVE CONTENTS OF TSSR
7086 043744 NEXT.ERRNO
7087 043744 12$: ERRDF ERRNO,T10SSR,PKTSSR ;DEVICE FATAL SSR FAILED TO SET
7088 043744 104455 TRAP C$ERDF
7089 043746 001751 .WORD 1001
7090 043750 044447 .WORD T10SSR
    
```



```
044150 000434
7128                                     .WORD L10075-.
7129 044152 005204
7130 044154 020427 000377      82$:  INC R4          ;LOOK AT NEXT RAM LOC.
7131 044160 001351      .CMP R4,#377      ;AT TOP OF RAM ADDRESS SPACE
7132                                     BNE 60$          ;BRANCH TILL ALL MEMORY TESTED
7133
7134 044162 005737 002170      TST FATFLG      ;ANY FATAL ERRORS ?
7135 044166 001402      BEQ 160$        ;BRANCH IF NOT
7136 044170 004737 017776      JSR PC,CKDROP  ;TRY TO DROP THE UNIT
7137 044174 004737 017200      160$: JSR PC,TSTLOOP ;DONE ALL ITERATIONS?
7138 044200 103002      BCC 165$       ;BR IF YES
7139 044202 000137 043646      JMP T10LOOP    ;LOOP UNTIL ITERATION COUNT DONE
7140 044206
7141 044206      EXIT TST
      044206 104432
      044210 000374
7142
7143      TRAP C$EXIT
      .WORD L10075-.
```

```

7145
7146      ;+
7147      ;LOCAL STORAGE FOR THIS TEST
7148      ;-
7150 044212      .BLKB  10-<.-TUV2A&7>
7152 044220      T1OPACKET:
7153 044220 100004      .WORD  100004      ;COMMAND PACKET FOR TEST
7154 044222 044230      .WORD  T1ODATA      ;WRITE CHARACTERISTICS COMMAND, WITH ACK
7155 044224 000000      .WORD  0      ;ADDRESS OF CHARACTERISTICS BLOCK
7156 044226 000010      .WORD  8.      ;STARTING VALUE OF BLOCK SIZE
7157
7158 044230      T1ODATA:
7159 044230 044242      .WORD  T1OBFR      ;CHARACTERISTICS DATA BLOCK
7160 044232 000000      .WORD  0      ;ADDRESS OF MESSAGE BUFFER
7161 044234 000016      .WORD  14.      ;LENGTH OF MESSAGE BUFFER
7162 044236 000000 000000      .WORD  0,0
7163
7164 044242      T1OBFR: .BLKW  8.      ;MESSAGE BUFFER
7165      ;LOCAL TEXT MESSAGES FOR TEST
7166      ;-
7167
7168 044262      111      156      151  TST10ID: .ASCIZ  'Initialization After WRITE CHARACTERISTICS'
7169 044335      111      156      143  T1OMEM: .ASCIZ  'Incorrect RAM Data After Init'
7170      .EVEN
7171 044374      127      122      111  T1ONBA: .ASCIZ  'WRITE CHARACTERISTICS Command Not Accepted'
7172 044447      103      157      156  T1OSSR: .ASCIZ  'Contents of TSSR Incorrect After WRITE CHARACTERISTICS'
7173
  
```

```
7175
7176
7177
7178
7179
7180
7181
7182
7183
7184 044536
7185 044536
7186 044542 012701 044220
7187 044546 012721 100004
7188 044552 012721 044230
7189 044556 005021
7190 044560 012721 000010
7191 044564 012721 044242
7192 044570 005021
7193 044572 012721 000016
7194 044576 005021
7195 044600 005011
7196 044602 000207
7197 044604
      044604
      044604 104401
```

```

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;
;-
.EVEN
T10REST:
      SAVREG
      MOV #T10PACKET,R1 ;SAVE THE REGISTERS
      MOV #100004,(R1)+ ;START OF THE PACKET
      MOV #T10DATA,(R1)+ ;WRITE CHARACTERISTICS WITH ACK
      CLR (R1)+ ;ADDRESS OF CHAR DATA BLOCK
      MOV #8,(R1)+ ;EXTENDED ADDRESS
      MOV #T10BFR,(R1)+ ;SIZE OF DATA BLOCK IN BYTES
      CLR (R1)+ ;ADDRESS OF MESSAGE BUFFER
      MOV #14,(R1)+ ;LENGTH OF MESSAGE BUFFER
      CLR (R1)+
      CLS (R1)
      RTS PC ;RETURN
      ENDTST
```

L10075: TRAP C#ETST


```
7305 045504 000000          .WORD 0
7306 045506 000010          .WORD 8.          ;STARTING VALUE OF BLOCK SIZE
7307
7308
7309 045510          T11DTA:          ;SELECT DATA BLOCK
7310 045510 045076          .WORD T11BFR      ;ADDRESS OF MESSAGE BUFFER
7311 045512 000000          .WORD 0
7312 045514 000400          .WORD 256.        ;LENGTH OF MESSAGE BUFFER
7313 045516 000000 000000  .WORD 0,0
7314
7315
7316          ;+
7317          ;LOCAL TEXT MESSAGES FOR TEST
7318          ;-
7319
7320 045522          127      122      111  T11NBA: .ASCIZ 'WRITE SUBSYSTEM MEMORY Command Not Accepted'
7321 045576          105      170      160  T11NINT: .ASCIZ 'Expected Interrupt Not Received On WRITE SUBSYSTEM MEMORY'
7322 045670          102      141      163  TST11ID: .ASCIZ 'Basic WRITE SUBSYSTEM MEMORY Command'
7323          .EVEN
7324
```

```

7326
7327
7328
7329
7330
7331
7332
7333
7334 045736
7335 045736
7336 045742 012701 045060
7337 045746 012721 100206
7338 045752 012721 045070
7339 045756 005021
7340 045760 012721 000006
7341 045764 005021
7342 045766 005021
7343 045770 005011
7344 045772 000207
7345
7346
7347 045774
7348 045774
7349 046000 012701 045500
7350 046004 012721 100204
7351 046010 012721 045510
7352 046014 005021
7353 046016 012721 000010
7354 046022 012721 045076
7355 046026 005021
7356 046030 012721 000400
7357 046034 005021
7358 046036 005011
7359 046040 005037 045076
7360 046044 000207
7361 046046
      046046
      046046 104401
  
```

```

;+
;
;ROUTINE TO RESTORE COMMAND PACKET TO START-UP (DEFAULT) VALUES
;WRITE SUBSYSTEM MEMORY COMMAND
;
;-
  
```

```

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

```

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

```

L10076: TRAP C$ETST
  
```

.SBTTL HARDWARE PARAMETER CODING SECTION

```

7363
7364
7365
7366
7367
7368
7369
7370
7371
7372
7373 046050
      046050 000010
      046052
7374
7375 046052
      046052 000031
      046054 046072
      046056 160000
      046060 177776
7376 046062
      046062 001031
      046064 046121
      046066 000000
      046070 000776
7377
7378 046072
      046072
7379 046072 104 105 126 HPM1: .ASCIZ 'DEVICE ADDRESS (TSSR) '
7380 046121 111 116 124 HPM2: .ASCIZ 'INTERRUPT VECTOR '
7381 046145 111 116 124 HPM3: .ASCIZ 'INTERRUPT PRIORITY '
7382
7383
    
```

```

; **
; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
; WITH THE OPERATOR.
; --
    
```

```

      BGNHRD
      .WORD L10100-L$HARD/2
      L$HARD::
      ;GET TSBA/TSDB REGISTER ADDRESS.
      .WORD T$CODE
      .WORD HPM1
      .WORD T$LOLIM
      .WORD T$HILIM
      GPRMA HPM2,2,0,0,776,YES ;GET VECTOR ADDRESS.
      .WORD T$CODE
      .WORD HPM2
      .WORD T$LOLIM
      .WORD T$HILIM
      ; GPRMD HPM3,4,0,340,0,7,YES ;GET INTERRUPT PRIORITY.
      ENDHRD
      .EVEN
      L10100:
    
```


ADDSSR 011612 G
 ADR = 000020 G
 AMBTSS 006156
 ASSEMB= 000010
 A1716 = 000003
 BADDAT 003110 G
 BADSSR 016374 G
 BAR = 174402
 BENBSW 002174 G
 BIE = 040000
 BIT0 = 000001 G
 BIT00 = 000001 G
 BIT01 = 000002 G
 BIT02 = 000004 G
 BIT03 = 000010 G
 BIT04 = 000020 G
 BIT05 = 000040 G
 BIT06 = 000100 G
 BIT07 = 000200 G
 BIT08 = 000400 G
 BIT09 = 001000 G
 BIT1 = 000002 G
 BIT10 = 002000 G
 BIT11 = 004000 G
 BIT12 = 010000 G
 BIT13 = 020000 G
 BIT14 = 040000 G
 BIT15 = 100000 G
 BIT2 = 000004 G
 BIT3 = 000010 G
 BIT4 = 000020 G
 BIT5 = 000040 G
 BIT6 = 000100 G
 BIT7 = 000200 G
 BIT8 = 000400 G
 BIT9 = 001000 G
 BOE = 000400 G
 BRINIT 004355
 BSELO = 000000
 BSEL1 = 000001
 CHKAMB 016540
 CHKMAN 021226 G
 CHKTSS 017060
 CKDROP 017776
 CKEMAX 017624
 CKMSG 011032 G
 CKMSG2 011152 G
 CKRAM 010354 G
 CKRAM2 010730 G
 CMPMEM 020402
 CONFIG 020044
 COUNT 002254 G
 CSR = 174400
 CSRADD 002154 G
 CTAB 003116 G
 CTABE 003130 G
 CTABM 003116 G

C\$AU = 000052
 C\$AUTO= 000061
 C\$BRK = 000022
 C\$BSEG= 000004
 C\$BSUB= 000002
 C\$CEFG= 000045
 C\$CLCK= 000062
 C\$CLEA= 000012
 C\$CLOS= 000035
 C\$CLP1= 000006
 C\$CVEC= 000036
 C\$DCLN= 000044
 C\$DODU= 000051
 C\$DRPT= 000024
 C\$DU = 000053
 C\$EDIT= 000003
 C\$ERDF= 000055
 C\$ERHR= 000056
 C\$ERRO= 000060
 C\$ERSF= 000054
 C\$ERSO= 000057
 C\$ESCA= 000010
 C\$ESEG= 000005
 C\$ESUB= 000003
 C\$ETST= 000001
 C\$EXIT= 000032
 C\$GETB= 000026
 C\$GETW= 000027
 C\$GMAN= 000043
 C\$GPHR= 000042
 C\$GPLO= 000030
 C\$GPRI= 000040
 C\$INIT= 000011
 C\$INLP= 000020
 C\$MANI= 000050
 C\$MEM = 000031
 C\$MSG = 000023
 C\$OPEN= 000034
 C\$PNTB= 000014
 C\$PNTF= 000017
 C\$PNTS= 000016
 C\$PNTX= 000015
 C\$QIO = 000377
 C\$RDBU= 000007
 C\$REFG= 000047
 C\$RESE= 000033
 C\$REVI= 000003
 C\$RFLA= 000021
 C\$RPT = 000025
 C\$SEFG= 000046
 C\$SPRI= 000041
 C\$SVEC= 000037
 C\$TPRI= 000013
 DAR = 174404
 DATA 002256 G
 DATAFL 014710
 DATASC 020760

DEBUGM 011304
 DEVCNT 002166 G
 DEVDR0 023420
 DEVNRD 023337
 DEVNXR 023255
 DEVONL 023173
 DEVSUM 023136
 DFPTBL 002124 G
 DIAGMC= 000000
 DICEA = 000001
 DLCYL = 000177
 DLDER= 100200
 DLERR = 177730
 DLGETS= 000004
 DLRDHD= 000010
 DLRDNH= 000016
 DLSR = 000013
 DLUN = 000006
 DSBINT 016700
 DUAD12 004541
 DUFLG 003060 G
 DUMMY 003030
 EF.CON= 000036 G
 EF.NEW= 000035 G
 EF.PWR= 000034 G
 EF.RES= 000037 G
 EF.STA= 000040 G
 EMAXDU 017521
 EN = 000000
 ENAINT 016646
 ENVIRN 021356
 EPRTSW 002146 G
 EPRT1 005672
 EPRT2 005672
 EPRT3 005672
 ERCM 011405
 ERRHI 002202 G
 ERRK 017500
 ERRLO 002204 G
 ERRNO = 002120
 ERRVEC= 000004 G
 ERTABE 003330
 ERTABL 003130
 ESUM 017502
 EVL = 000004 G
 EXBCNT= 000010
 EXPBRE 016176 G
 EXPD 002176 G
 EXPGOT 004431
 EXPGT2 004465
 EXPMSG 002266 G
 EXPREC 016170 G
 EXTA 005232
 EXTEND 005230
 E\$END = 002100
 E\$LOAD= 000035
 FATCHK 017724

FATERR= 000060
 FATFLG 002170 G
 FERCM 011374
 FIFEXP 011642 G
 FIF1MS 011714
 FIF2MS 011763
 FILLME 020216
 FNOINT 004113
 FORCER 002144 G
 FREE 003072 G
 FREEHI 003076
 FRESIZ 003074 G
 FUSI 004015
 F\$AU = 000015
 F\$AUTO= 000020
 F\$BGN = 000040
 F\$CLEA= 000007
 F\$DU = 000016
 F\$END = 000041
 F\$HARD= 000004
 F\$HW = 000013
 F\$INIT= 000006
 F\$JMP = 000050
 F\$MOD = 000000
 F\$MSG = 000011
 F\$PROT= 000021
 F\$PWR = 000017
 F\$RPT = 000012
 F\$SEG = 000003
 F\$SOFT= 000005
 F\$SRV = 000010
 F\$SUB = 000002
 F\$SW = 000014
 F\$TEST= 000001
 GDDAT 003112 G
 GERRMA 002142 G
 GETPAT 020722 G
 GETSEL 021004 G
 G\$CNT0= 000200
 G\$DELM= 000372
 G\$DISP= 000003
 G\$EXCP= 000400
 G\$HILI= 000002
 G\$LOLI= 000001
 G\$NO = 000000
 G\$OFFS= 000400
 G\$OFSI= 000376
 G\$PRMA= 000001
 G\$PRMD= 000002
 G\$PRML= 000000
 G\$RADA= 000140
 G\$RADB= 000000
 G\$RADD= 000040
 G\$RADL= 000120
 G\$RADO= 000020
 G\$XFER= 000004
 G\$YES = 000010

HIADDR= 001400
 HIMEM = 007776
 HOE = 100000 G
 HPM1 046072
 HPM2 046121
 HPM3 046145
 IBE = 010000 G
 IDU = 000040 G
 IER = 020000 G
 IFAULT 004154
 INCERK 017566
 INTCPC 016644
 INTFLA 016641
 INTMAS 016640
 INTR 016712 G
 INTREC 002172 G
 INTVEC 016642
 INTX 004176
 IOKCKI= 000200
 IOKSTP= 000001
 IPRI 002160 G
 ISR = 000100 G
 IVEC 002156 G
 IXE = 004000 G
 I\$AU = 000041
 I\$AUTO= 000041
 I\$CLN = 000041
 I\$DU = 000041
 I\$HRD = 000041
 I\$INIT= 000041
 I\$MOD = 000040
 I\$MSG = 000041
 I\$PROT= 000040
 I\$PTAB= 000041
 I\$PWR = 000041
 I\$RPT = 000041
 I\$SEG = 000041
 I\$SETU= 000041
 I\$SFT = 000041
 I\$SRV = 000041
 I\$SUB = 000041
 I\$TST = 000041
 J\$JMP = 000167
 KIPAR0= 172340
 KIPAR1= 172342
 KIPAR2= 172344
 KIPAR3= 172346
 KIPAR4= 172350
 KIPAR5= 172352
 KIPAR6= 172354
 KIPAR7= 172356
 KIPDR0= 172300
 KIPDR1= 172302
 KIPDR2= 172304
 KIPDR3= 172306
 KIPDR4= 172310
 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 046200 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	PRMSG 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	PR1ASC 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 046072	PASRPT 022064	PW.NOP= 000000
L\$AUT 002070 G	L10007 011610	L10101 046214	PATCH 046420 G	PW.NO1= 000023
L\$AUTO 022572 G	L10010 011626	L10102 046430	PATDAT 020756	PW.RDE= 000024
L\$CCP 002106 G	L10011 011640	L10104 046436	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.NO0= 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 046372 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	PRBREC 016166	P.OPP = 020000
L\$HARD 046052 G	L10036 023724	MS.RSD= 000001	PRBTOT 016117	P.POSI= 000010
L\$HIME 002120 C	L10037 023572	MS.RSF= 000020	PRBYTE 015616 G	P.READ= 000001
L\$HPCP 002016 G	L10040 023654	MS.RST= 000010	PRI = 002000 G	P.SWB = 010000
L\$HPTP 002022 G	L10041 024676	NBA = 002000	PRIADD 007532	P.WRIT= 000005
L\$HW 002124 G	L10042 024116	NEWPAS 022020	PRIAO 007602	P.WRTC= 000004
L\$ICP 002104 G	L10043 024310	NODEV 003062 G	PRIBXO 007164 G	P.WRTS= 000006
L\$INIT 021606 G	L10044 024510	NOINIT 004233	PRIEQU 007432	QVP 002152 G
L\$LADP 002026 G	L10045 026162	NOINTR 004117	PRIPKT 006712 G	RAMASC 013616
L\$LAST 046424 G	L10046 025240	NOITS 002136 G	PRIRAM 007440	RAMDAT 002206 G
L\$LOAD 002100 G	L10047 025560	NOMAN 021262	PRITAD 007646	RAMER 010456 G
L\$LUN 002074 G	L10050 030214	NP.IR = 000200	PRITSS 005264	RAMERR 016204 G
L\$MREV 002050 G	L10051 026506	NP.L00= 000040	PRITO 007716	RAMEXP 016224 G
L\$NAME 002000 G	L10052 026772	NP.OUT= 000100	PRIXOR 007314 G	RAMFHR 014362
L\$PRIO 002042 G	L10053 027220	NP.WRP= 000020	PRI00 = 000000 G	RAMFOR 007470
L\$PROT 021576 G	L10054 031424	NSI 004050	PRI01 = 000040 G	RAMHLD 010640
L\$PRT 002112 G	L10055 033500	NSINIT 004305	PRI02 = 000100 G	RAMIOP 010644
L\$REPP 002062 G	L10056 031700	NUL 004425	PRI03 = 000140 G	RAMPD 010715

SYMBOL TABLE

RAMR5H	010642	S0.IDB=	000010	TSSX	003716	T\$\$AU =	010031	T3PACK	025620
RAMSIZ	002246 G	S0.IFB=	000002	TSTBLK	002720 G	T\$\$AUT=	010033	T3SSR	025655
RAMTAD	016212 G	S0.IFP=	000001	TSTCNT	002162 G	T\$\$CLE=	010034	T3TSBA	026071
RBPCRA	014755	S0.ILD=	000020	TSTEND	017442	T\$\$DAT=	010104	T3.1	024734
RCVHIA	002250 G	S0.ION=	000040	TSTFLA	002260 G	T\$\$DU =	010032	T3.2	025254
RCVLOA	002252 G	S0.IRD=	000100	TSTL00	017200 G	T\$\$HAR=	010100	T4	026164 G
RDERR	005104	S0.IRW=	000004	TSTPTR	002262 G	T\$\$HW =	010000	T4BFR	027274
READ =	000014	S0.ISP=	000200	TSTSET	017232 G	T\$\$INI=	010030	T4DATA	027260
READY =	000001	S1.ICE=	002000	TST1ID	023704	T\$\$MSG=	010025	T4INT	027745
RECMG	002432 G	S1.IEO=	010000	TST10I	044262	T\$\$PC =	000001	T4LOOP	026220
RECV	002200 G	S1.IFM=	001000	TST11I	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.I1R=	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
RMSGGE=	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 G
SEEK =	000006	TCOCOD	006220	T\$ARGC=	000003	T10DAT	044230	T5BFR	030742
SELASC	021170	TEMP1	003064 G	T\$CODE=	001130	T10L00	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=	046436	T10RES	044536	T5PACK	030720
SFFMSG	011566 G	TFCASC	006060	T\$GMAN=	000000	T10SSR	044447	T5SSR	031221
SFHERR	003603	TIMEXP	016246 G	T\$HILI=	000776	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 G
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\$NSO =	000000	T11DTA	045510	T6LOOP	031462
SOFINI	016470 G	TPB =	177566	T\$NS1 =	000005	T11L00	044642	T6NBA	032406
SPACE	007760 G	TPS =	177564	T\$NS2 =	000002	T11NBA	045522	T6NINT	033136
SPM1	046214	TRANST	002134 G	T\$NS3 =	000003	T11NIN	045576	T6PACK	032330
SPM4	046260	TSBA =	177776 G	T\$PCNT=	000000	T11PAC	045060	T6REST	033426
SPM6	046310	TSBAH =	177777 G	T\$PTAB=	010103	T11PK2	045500	T6SSR	033047
SPM7	046340	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\$SEK0=	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 G
SVCTST=	000001	TSSRFO	005736	T\$TSTM=	177777	T3NBA	025630	T7BFR	035662
S\$LSYM=	010000	TSSRH =	000001 G	T\$TSTS=	000001	T3NINT	026011	T7BUFR	035724

SYMBOL TABLE

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	XS0BOT= 000002	X2.BUF= 000100
T7RST 036566	T9BLK 041744	USI 004021	XS0CON 015022	X2.EXT= 000200
T7RT2 036640	T9CHAR 043546	WAITF 016744 G	XS0EOT= 000001	X2.OPM= 100000
T7SSR 036270	T9CKRA 043326 G	WC.IFA= 000200	XS0IE = 000040	X2.RCE= 040000
T7SSRM 036200	T9CONV 042746	WC.IFE= 000002	XS0ILA= 000400	X2.REV= 000077
T7.1 033536	T9CT2 043140	WC.IGO= 000001	XS0ILC= 001000	X2.SPA= 035400
T7.2 034122	T9DATA 041700	WC.IRE= 000010	XS0LET= 020000	X2.UNI= 000007
T7.3 034514	T9DPR 042546	WC.IRW= 000004	XS0MOT= 000200	X2.WCF= 002000
T7.4 035404	T9GETS 042126	WC.IOT= 000100	XS0NEF= 002000	X3.DCK= 000010
T8 036714 G	T9HIF 041732	WC.IIT= 000040	XS0ONL= 000100	X3.MBZ= 000006
T8BFR 037472	T9KT 041740	WC.I5R= 000020	XS0PED= 000010	X3.MDE= 177400
T8BF2 037542	T9LOAD 041734	WF.IED= 000010	XS0RLL= 010000	X3.OPI= 000100
T8DATA 037460	T9LOOP 040466	WF.IER= 000004	XS0RLS= 040000	X3.REV= 000040
T8DTA 037530	T9MSGB 042271	WF.IHI= 000200	XS0TMK= 100000	X3.RIB= 000001
T8LOOP 036750	T9NINT 042455	WF.IRE= 000040	XS0VCK= 000020	X3.SPA= 000200
T8NBA 037562	T9NXM 042637	WF.IWF= 000020	XS0WLE= 004000	X3.TRF= 000020
T8NINT 040144	T9PACK 041670	WF.IWR= 000100	XS0WLK= 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	WRTCHR 010152 G	XS3CON 015201	X4.TSM= 020000
T8RT2 040366	T9TBE 042076	WRTERR 005011	XXCOMM 003070 G	X4.WRC= 000377
T8SR2 040070	T9WRTS 042202	WRTMSG 004754	X\$ALWA= 000000	

. ABS. 046436 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 35592 WORDS (140 PAGES)

DYNAMIC MEMORY: 20060 WORDS (77 PAGES)

ELAPSED TIME: 00:09:31

CZTKEB.BIC.CZTKEB/-SP=SVC/ML.CZTKEB