

DMP11 DMR11

M8203 STATIC DIAG #2 AH-E235C-MC
CZDMSCO FICHE 1 OF 1

MAY 1980
COPYRIGHT © 79 80
MADE IN USA



A large grid of 15 columns and 20 rows of small, illegible technical diagrams or data tables. Each cell contains a small-scale version of the main diagram's content, likely representing a static diagnostic for a specific component or circuit. The diagrams are arranged in a regular grid pattern across the page.

.NLIST
.TITLE CZDMSC M8203 STATIC DIAG #2
.SBTTL PROGRAM DOCUMENT
.LIST

SEQ 0001

.REM @

IDENTIFICATION

PRODUCT CODE: AC-E234C-MC
PRODUCT NAME: CZDMSC0 M8203 STATIC DIAG #2
PRODUCT DATE: FEBRUARY 1980
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: DAVID HOFFMAN

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

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

COPYRIGHT (C) 1979, 1980 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL
DEC

PDP
DECUS

UNIBUS
DECTAPE

MASSBUS

CONTENTS

- 1.0 INTRODUCTION
- 2.0 HARDWARE REQUIREMENTS
- 3.0 PRELIMINARY PROGRAM REQUIREMENTS
- 4.0 GENERAL PROGRAM CONSIDERATIONS
 - 4.1 DIAGNOSTIC SUPERVISOR
 - 4.2 EXECUTION TIME
 - 4.3 XYDP+
 - 4.4 ACT/SLIDE
 - 4.5 APT
 - 4.6 MEMORY MANAGEMENT
 - 4.7 MEMORY PARITY OPTION
 - 4.8 ERROR LOGGING
- 5.0 PROGRAM LOAD MEDIA
- 6.0 OPERATING INSTRUCTIONS
 - 6.1 LOADING AND STARTING PROCEDURES
 - 6.1.1 LOADING PROCEDURES
 - 6.1.2 STARTING PROCEDURES
 - 6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION
 - 6.2 INITIAL DIALOGUE
 - 6.3 PROGRAM OPTIONS
 - 6.3.1 START COMMAND
 - 6.3.1.1 TESTS SWITCH
 - 6.3.1.2 PASS SWITCH
 - 6.3.1.3 FLAGS SWITCH
 - 6.3.1.4 END OF PASS SWITCH
 - 6.3.1.5 EFFECT OF START COMMAND
 - 6.3.2 RESTART COMMAND
 - 6.3.2.1 TESTS, PASS, AND FLAG SWITCHES
 - 6.3.2.2 UNITS SWITCH
 - 6.3.2.3 EFFECT OF RESTART COMMAND
 - 6.3.3 CONTINUE COMMAND
 - 6.3.3.1 PASS SWITCH
 - 6.3.3.2 FLAGS SWITCH
 - 6.3.3.3 EFFECT OF CONTINUE COMMAND
 - 6.3.4 PROCEED COMMAND
 - 6.3.4.1 FLAGS SWITCH
 - 6.3.4.2 EFFECT OF PROCEED COMMAND
 - 6.3.5 ADD COMMAND
 - 6.3.5.1 UNITS SWITCH
 - 6.3.5.2 EFFECT OF ADD COMMAND
 - 6.3.6 DROP COMMAND
 - 6.3.6.1 UNITS SWITCH
 - 6.3.6.2 EFFECT OF DROP COMMAND
 - 6.3.7 PRINT COMMAND
 - 6.3.7.1 EFFECT OF PRINT COMMAND
 - 6.3.8 DISPLAY COMMAND
 - 6.3.8.1 UNITS SWITCH
 - 6.3.8.2 EFFECT OF DISPLAY COMMAND

- 6.3.9 FLAGS COMMAND
 - 6.3.9.1 EFFECT OF FLAGS COMMAND
- 6.3.10 ZFLAGS COMMAND
 - 6.3.10.1 EFFECT OF ZFLAGS COMMAND
- 6.3.11 CONTROL CHARACTERS
- 6.3.12 HARDWARE PARAMETERS
- 6.3.13 SOFTWARE PARAMETERS
- 6.3.14 EXTENDED DISCUSSION OF P-TABLE DIALOGUE

7.0 DEVICE INFORMATION TABLES

- 8.0 TEST DESCRIPTIONS
 - 8.1 DATA PATTERNS USED

- 9.0 ERROR INFORMATION
 - 9.1 ERROR REPORTING

1.0 INTRODUCTION

THE M8203 IS A SINGLE-LINE SYNCHRONOUS LINE UNIT MODULE WHICH SUPPORTS BOTH CHARACTER

-ORIENTED (DDCMP, BSC, ETC.) AND BIT-ORIENTED (SDLC, HDLC, ETC.) PROTOCOLS. THE PURPOSE OF THIS PROGRAM IS TO PERFORM DIAGNOSTIC TESTING OF ALL M8203 LOGIC IN A RELATIVELY STATIC MANNER. THE FOLLOWING FUNCTIONS WILL BE PERFORMED: LINE UNIT REGISTER ADDRESSING, USYRT ADDRESSING, STATIC BIT INTERACTION AND READ/WRITE LOGIC TESTS, BASIC TRANSMITTER AND RECEIVER SEQUENCING AND DATA BUFFERING AND STATIC OPERATIONS IN CHARACTER AND BIT-STUFFING MODES. IN ADDITION DATA MESSAGES WILL BE SENT AT SPEEDS OF 2400 BAUD TO 1 MEGABAUD, WITH LOOPBACK IN THE USYRT, ON THE LINE UNIT AT TTL LEVEL, OR THROUGH AN EXTERNAL TEST CONNECTOR WITH A SPECIFIC MODEM INTERFACE SELECTED.

THE STATIC LOGIC TESTS WILL PROVIDE EXTENSIVE TROUBLESHOOTING CAPABILITIES, SUCH AS TIGHT SCOPE LOOPS, SWITCH OPTIONS, AND ABILITY TO 'LOCK' ONTO INTERMITTENT ERRORS. IN ADDITION TESTS WILL BE DESIGNED AND STRUCTURED TO ACHIEVE MAXIMUM FAULT RESOLUTION AND FACILITATE REPLACEMENT OF THE SMALLEST FIELD REPLACEABLE UNIT.

THIS PROGRAM WILL BE IMPLEMENTED USING THE DIAGNOSTIC SUPERVISOR AND A STRUCTURED PROGRAMMING APPROACH. BECAUSE THE DESIGN WILL CONFORM TO THE SUPERVISOR (STANDALONE VERSION) THE PROGRAM WILL BE COMPATIBLE WITH ACT, APT, XXDP+, AND SLIDE.

THROUGH DIALOGUE WITH THE OPERATOR, THE PROGRAM WILL ALLOW MODIFICATION OF DEVICE PARAMETERS, SUCH AS UNIBUS ADDRESS, VECTOR ADDRESSES AND DEVICE PRIORITY. IN ADDITION, THE OPERATOR CAN SPECIFY PARTICULAR TESTS TO BE RUN AND A VARIETY OF LOOPING, RUNNING, AND REPORTING MODES.

DEVICE ERRORS WILL BE REPORTED AS THEY OCCUR. THE REPORT WILL INCLUDE A TEST NUMBER AND DESCRIPTION OF THE ERROR, GOOD AND BAD TEST DATA, AND APPLICABLE DEVICE REGISTER CONTENTS.

2.0 HARDWARE REQUIREMENTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE M8203 STATIC LOGIC TESTS:

PDP-11/04,05,10,20,30,34,35,40,45,50,60, OR 70
16K MEMORY
CONSOLE TERMINAL
DMC-11 OR KMC-11 MICROPROCESSOR
M8203 LINE UNIT AND BC08S-1 CABLE AND BERG CONNECTORS
H3254 AND H3255 TEST CONNECTORS (IF NOT PRESENT, SOME TESTS WILL BE SKIPPED)

3.0 PRELIMINARY PROGRAM REQUIREMENTS

THIS PROGRAM OPERATES THE MICROPROCESSOR EXTENSIVELY IN ORDER TO TEST THE LINE INIT. FOR THIS REASON, THE MICROPROCESSOR DIAGNOSTIC AND SUBSYSTEM FUNCTIONAL TESTS SHOULD BE RUN FIRST, AND ANY FAULTS FOUND IN THE MICROPROCESSOR MODULE SHOULD BE REPAIRED, PRIOR TO RUNNING THE M8203 STATIC LOGIC TESTS.

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 DIAGNOSTIC SUPERVISOR

THIS PROGRAM IS COMPATIBLE WITH THE STANDALONE DIAGNOSTIC SUPERVISOR, AND MUST BE LOADED TO BE CO-RESIDENT WITH THE SUPERVISOR, OR BE PREVIOUSLY COMBINED WITH THE SUPERVISOR AND LOADED AS A SINGLE FILE. IN EITHER CASE, THE COMBINED PROGRAM WILL NOT EXCEED 16K OF MEMORY.

4.2 EXECUTION TIME

THE MAXIMUM TIME REQUIRED TO RUN THE M8203 STATIC LOGIC TESTS IS ABOUT 45 SECONDS PER PASS FOR EACH UNIT.

4.3 XXDP+

THIS PROGRAM MAY BE LOADED UNDER XXDP+, AND MAY BE RUN IN DUMP MODE OR CHAIN MODE.

4.4 ACT/SLIDE

THIS PROGRAM MAY BE LOADED UNDER ACT OR SLIDE AND MAY BE RUN IN DUMP MODE OR CHAIN MODE.

4.5 APT

THIS PROGRAM MAY BE LOADED BY THE APT SYSTEM (INCLUDING APT-RD) AND RUN IN PROGRAM MODE OR SCRIPT MODE.

4.6 MEMORY MANAGEMENT

MEMORY MANAGEMENT IS NOT UTILIZED IN THIS PROGRAM. IF IT IS INSTALLED, IT IS DISABLED BY THE PROGRAM.

4.7 MEMORY PARITY OPTION

IF PARITY MEMORY IS INSTALLED, MEMORY PARITY TRAPS ARE DISABLED BY THE PROGRAM.

4.8 ERROR LOGGING

AT THE END OF EACH PASS ON ALL UNITS, THE PROGRAM PRINTS OUT THE CUMULATIVE TOTAL NUMBER OF ERRORS SINCE THE LAST START OR RESTART COMMAND.

5.0 PROGRAM LOAD MEDIA

THIS PROGRAM CAN BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR FROM ACT, SLIDE, OR APT SYSTEMS, OR FROM ANY MEDIA SUPPORTED BY XXDP+. WHEN USING THE PAPER TAPE ABSOLUTE LOADER, THE PROGRAM SHOULD BE LOADED FIRST, FOLLOWED BY THE DIAGNOSTIC SUPERVISOR. WHEN USING XXDP+, THE DIAGNOSTIC SUPERVISOR SHOULD BE LOADED FIRST, FOLLOWED BY THE DIAGNOSTIC PROGRAM.

6.0 OPERATING INSTRUCTIONS

6.1 LOADING AND STARTING PROCEDURES

6.1.1 LOADING PROCEDURES

THIS PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER. IT MAY ALSO BE LOADED FROM ANY XXDP+ LOAD MEDIA. WHEN LOADED UNDER XXDP+, THE DIAGNOSTIC SUPERVISOR WILL BE LOADED AUTOMATICALLY.

6.1.2 STARTING PROCEDURES

THE PROGRAM STARTS AT LOCATION 200. USE STANDARD DEC PROCEDURES TO START THE PROGRAM.

6.1.3 STEPS FOR QUICK AND SIMPLE EXECUTION

THE DIAGNOSTIC CAN BE EXECUTED STANDALONE UNDER XXDP+, WITHOUT READING THE REMAINDER OF THIS DOCUMENT, AS FOLLOWS:

- A) LOAD AND START DIAGNOSTIC USING RUN COMMAND
- B) RECEIVE DIAGNOSTIC SUPERVISOR IDENTIFICATION AND PROMPT (DRS-C>)
- C) ENTER STA<CR>
- D) ANSWER HARDWARE AND SOFTWARE QUESTIONS
- E) GET END OF PASS MESSAGES OR ERROR MESSAGES
- F) TO END EXECUTION, ENTER CONTROL/C

6.2 INITIAL DIALOGUE

AFTER THE PROGRAM AND THE SUPERVISOR ARE LOADED AND THE PROGRAM IS STARTED, THE FOLLOWING IDENTIFICATION IS TYPED :

```
DRS LOADED
DIAG. RUN-TIME SERVICES
CZDMS-C-0
M8203 STATIC LOGIC TESTS - PART 2 OF 2
UNIT IS M8203
DR>
```

THE OPERATOR THEN RESPONDS BY TYPING ONE OR MORE OF THE COMMANDS DESCRIBED IN THE FOLLOWING SECTION 6.3. (FOR MORE DETAILED INFORMATION, REFER TO THE DIAGNOSTIC SUPERVISOR FUNCTIONAL SPECIFICATION).

6.3 PROGRAM OPTIONS

6.3.1 START COMMAND

```
*****
STA(RT)/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:
  <FLAG-LIST>/EOP:<INCR>
*****
```

6.3.1.1 TESTS SWITCH (/TESTS:<TEST-LIST>)

<TEST-LIST> IS A SEQUENCE OF DECIMAL NUMBERS (1:2 ETC.) OR RANGES OF DECIMAL NUMBERS (1-5:8-10 ETC.) THAT SPECIFY THE TESTS TO BE EXECUTED. THE NUMBERS ARE SEPARATED BY COLONS. THE NUMBERS RANGE FROM 1 TO THE LARGEST TEST NUMBER IN THE DIAGNOSTIC. THEY MAY BE SPECIFIED IN ANY ORDER. TESTS WILL BE EXECUTED IN NUMERICAL ORDER REGARDLESS OF THE ORDER OF SPECIFICATION. THE DEFAULT IS TO EXECUTE ALL TESTS. ON THIS AND ALL SWITCHES, THE ANGLE BRACKETS <> ARE PUNCTUATION USED IN THE DEFINITION ONLY, AND ARE NOT TO BE TYPED BY THE OPERATOR. SEE EXAMPLE AT END OF 6.3.1.5.

6.3.1.2 PASS SWITCH (/PASS:<PASS-CNT>)

<PASS-CNT> IS A DECIMAL NUMBER INDICATING THE DESIRED NUMBER OF PASSES. A PASS IS DEFINED AS THE EXECUTION OF THE FULL DIAGNOSTIC (ALL SELECTED TESTS) AGAINST ALL UNITS SUBMITTED. THE DEFAULT IS NON-ENDING EXECUTION. IN THIS CASE EXIT FROM THE PROGRAM IS ACCOMPLISHED EITHER BY TYPING A CONTROL/C OR BY OCCURANCE OF AN ERROR WITH THE HALT ON ERROR FLAG BEING SET. THE EXIT IS A RETURN TO COMMAND MODE. SEE EXAMPLE AT END OF 6.3.1.5.

6.3.1.3 FLAGS SWITCH (/FLAGS:<FLAG-LIST>)

<FLAG-LIST> IS A SEQUENCE OF ELEMENTS OF THE FORM <FLAG>, <FLAG=1>, OR <FLAG=0>, SEPARATED BY COLONS, WHERE <FLAG> HAS ONE OF THE FOLLOWING VALUES:

HOE HALT ON ERROR, CAUSING COMMAND MODE TO BE ENTERED WHEN AN ERROR IS ENCOUNTERED
 LOE LOOP ON ERROR, CAUSING THE DIAGNOSTIC TO LOOP CONTINUOUSLY WITHIN THE SMALLEST DEFINED BLOCK OF CODING (SEGMENT, SUBTEST, OR TEST) CONTAINING THE ERROR
 IER INHIBIT ERROR REPORTING
 IBF INHIBIT BASIC ERROR REPORTS
 IXE INHIBIT EXTENDED ERROR REPORTS
 PRI DIRECT ALL MESSAGES TO A LINE PRINTER
 PNT PRINT NUMBER OF TEST BEING EXECUTED
 BOE BELL ON ERROR
 UAM RUN IN UNATTENDED MODE, BYPASSING MANUAL INTERVENTION TESTS
 ISR INHIBIT STATISTICAL REPORTS
 IDU INHIBIT DROPPING OF UNITS BY DIAGNOSTIC LOOP ON TEST
 LOT LOOP ON TEST

THE FLAGS NAMED OR EQUATED TO 1 ARE SET, THOSE EQUATED TO 0 ARE CLEARED. A FLAG NOT SPECIFIED IS CLEARED. IF THE FLAGS SWITCH IS NOT GIVEN ALL FLAGS ARE CLEARED. SEE EXAMPLE AT END OF 6.3.1.5.

6.3.1.4 END OF PASS SWITCH (/EOP:<INCR>)

<INCR> IS A DECIMAL NUMBER INDICATING HOW OFTEN (IN TERMS OF PASSES) IT IS DESIRED THAT THE END OF PASS MESSAGE BE PRINTED. THE DEFAULT IS AT THE END OF EVERY PASS. SEE EXAMPLE AT END OF 6.3.1.5.

6.3.1.5 EFFECT OF START COMMAND

THE EFFECT OF THE START COMMAND IS TO INITIATE THE HARDWARE PARAMETER DIALOGUE, THE SOFTWARE PARAMETER DIALOGUE, AND THEN THE DIAGNOSTIC TESTS THEMSELVES.

THE HARDWARE PARAMETER DIALOGUE COMMENCES WITH THE QUESTION '# UNITS?' TO WHICH THE OPERATOR REPLIES WITH A DECIMAL NUMBER N FROM 1 TO 16. THE TERM 'UNIT' REFERS TO THE DEVICE TO WHICH THIS SERIES OF DIAGNOSTICS IS DEDICATED. FOLLOWING THIS ARE THE QUESTIONS WHEREBY THE P-TABLES THEMSELVES WILL BE BUILT. EACH P-TABLE IS A CORE-RESIDENT TABLE CONTAINING ALL THE HARDWARE INFORMATION FOR ONE UNIT. THE OPERATOR MUST SUPPLY N (NUMBER OF UNITS) VALUES FOR EACH QUESTION. HE MAY DO THIS BY GIVING ONE ANSWER TO EACH QUESTION (IN WHICH CASE THE SERIES OF QUESTIONS WILL BE POSED N TIMES) OR BY GIVING N VALUES, SEPARATED BY COMMAS, TO EACH QUESTION (SERIES WILL BE POSED ONCE). EACH QUESTION IS FOLLOWED BY THE RESPONSE RADIX (D FOR DECIMAL, B FOR BINARY, O FOR OCTAL, L FOR YES/NO) IN PARENTHESES AND THE DEFAULT VALUE AFTER THE PARENTHESES.

FOLLOWING THE HARDWARE QUESTIONS ARE THE SOFTWARE QUESTIONS TO BUILD THE SOFTWARE TABLES, WHICH DEFINE THE MODE (QUICK VERIFY ETC.) THAT THE DIAGNOSTIC WILL EXECUTE IN.

WHEN THE QUESTION '# UNITS?' IS ANSWERED, MEMORY STORAGE IS ALLOCATED FOR THE P-TABLES, AND IF THERE IS NOT ENOUGH TO ACCOMMODATE THEM THE MESSAGE 'TOO MANY UNITS' IS ISSUED. IN THIS CASE THE DIAGNOSTIC MUST BE EXECUTED MORE THAN ONCE TO TEST ALL UNITS.

EXAMPLE:

STA/TESTS:1:2-4:6:8-10/PASS:3/FLAGS:IER:HOE=1:UAM:LOE

THIS COMMAND WILL CAUSE THREE PASSES TO BE MADE, EACH PASS CONSISTING OF TESTS 1,2,3,4,6,8,9, AND 10 EXECUTED AGAINST ALL UNITS. THERE IS NO DIFFERENCE BETWEEN SAYING <FLAG> AND SAYING <FLAG=1>. THE NOTATION <FLAG=0> IS MEANINGFUL ONLY ON A COMMAND OTHER THAN START TO CLEAR A FLAG THAT WAS PREVIOUSLY SET. NOTE THAT ON ALL COMMANDS ONLY THE FIRST THREE LETTERS ARE SCANNED.

6.3.2 RESTART COMMAND

RES(TART)/TESTS:<TEST-LIST>/PASS:<PASS-CNT>/FLAGS:
<FLAG-LIST>/UNITS:<UNIT-LIST>

6.3.2.1 TESTS, PASS, AND FLAGS SWITCHES

<TEST-LIST>, <PASS-CNT>, AND <FLAG-LIST> ARE AS IN THE START COMMAND.

6.3.2.2 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS A SEQUENCE OF DECIMAL NUMBERS (0,1 ETC.) OR RANGES OF DECIMAL NUMBERS (0-5, 8-10 ETC.) THAT SPECIFY THE UNITS TO BE TESTED. THE NUMBERS ARE SEPARATED BY COLONS. THE NUMBERS MAY RANGE FROM 0 THRU N-1 (N IS THE NUMBER OF UNITS SPECIFIED IN THE PREVIOUS START COMMAND). THE NUMBER INDICATES THE POSITION OF THE P-TABLE AS THE DATA WAS ENTERED DURING THE HARDWARE DIAGLOGUE. THE UNITS WHICH ARE SELECTED MUST NOT HAVE BEEN DROPPED BY THE DROP COMMAND. SEE THE DISCUSSION OF ADD AND DROP COMMANDS BELOW. DEFAULT IS TO TEST ALL UNITS WHICH HAVE NOT BEEN DROPPED BY A DROP COMMAND.

6.3.2.3 EFFECT OF RESTART COMMAND

THE RESTART COMMAND DIFFERS FROM THE START COMMAND IN THAT THE P-TABLES FROM THE PREVIOUS START COMMAND (THERE MUST HAVE BEEN ONE) ARE USED, INSTEAD OF NEW ONES BEING BUILT. THE UNITS SWITCH GIVES THE ABILITY TO SELECT A SUBSET OF THESE. THE SOFTWARE DIALOGUE MAY OPTIONALLY BE REEXECUTED (OPERATOR WILL BE ASKED). THE COMMAND CAN BE USED AFTER COMMAND MODE HAS BEEN REENTERED IN ANY OF THE THREE NORMAL WAYS: A) THE REQUESTED NUMBER OF PASSES HAVE BEEN MADE B) AN ERROR WAS ENCOUNTERED WITH THE HALT ON ERROR FLAG SET C) A CONTROL/C WAS ENTERED BY THE OPERATOR.

6.3.3 CONTINUE COMMAND

CON(TINUE)/PASS:<PASS-CNT/FLAGS:<FLAG-LIST>

6.3.3.1 PASS SWITCH (/PASS:<PASS-CNT>)

<PASS-CNT> IS SAME AS IN START COMMAND, BUT THE DEFAULT IS THE UNSATISFIED PASS-CNT FROM THE PREVIOUS START OR RESTART. IF NONE REMAINS, THE DEFAULT IS NON-ENDING EXECUTION.

6.3.3.2 FLAG SWITCH (/FLAGS:<FLAG-LIST>)

<FLAG-LIST> IS SAME AS IN START COMMAND, BUT UNSPECIFIED FLAGS RETAIN THEIR CURRENT VALUE.

6.3.3.3 EFFECT OF CONTINUE COMMAND

CONTINUE MUST FOLLOW A START OR RESTART, AND COMMAND MODE MUST HAVE BEEN ENTERED DUE TO A HALT ON ERROR OR A CONTROL/C. THE EFFECT OF THE COMMAND IS TO GO TO THE BEGINNING OF THE TEST THAT WAS BEING EXECUTED WHEN THE HALT OR CONTROL/C TOOK PLACE. SOFTWARE DIALOGUE MAY OPTIONALLY BE REEXECUTED. HARDWARE PARAMETERS MAY NOT BE CHANGED.

6.3.4 PROCEED COMMAND

PRO(CEED)/FLAGS:<FLAG-LIST>

6.3.4.1 FLAGS SWITCH (/FLAGS:<FLAG-LIST>)

<FLAG-LIST> IS AS IN THE START COMMAND, BUT UNSPECIFIED FLAGS RETAIN THEIR CURRENT VALUE.

6.3.4.2 EFFECT OF PROCEED COMMAND

PROCEED MUST FOLLOW A START, RESTART, OR CONTINUE. COMMAND MODE MUST HAVE BEEN ENTERED VIA A HALT ON ERROR. THE EFFECT OF THE COMMAND IS TO BEGIN EXECUTION AT THE LOCATION FOLLOWING THE ERROR CALL. NEITHER HARDWARE NOR SOFTWARE PARAMETERS MAY BE ALTERED.

NOTE THAT IF THE MESSAGE 'TOO MANY UNITS' IS ISSUED, TWO OR MORE CORE IMAGES MUST BE CREATED (WITH DIFFERENT NAMES) TO TEST ALL UNITS.

NOTE THAT ALTHOUGH THE CHAINABLE IMAGE CAN BE EXECUTED ON A 16K MACHINE, THE ORIGINAL CCI CREATION MUST BE DONE ON A LARGE MACHINE, THE EXACT SIZE BEING DEPENDENT ON WHICH UPDATE UTILITY IS USED.

6.3.5 ADD COMMAND

ADD/UNITS:<UNIT-LIST>

6.3.5.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.5.2 EFFECT OF ADD COMMAND

THE UNITS SPECIFIED ARE ADDED TO THE TEST SEQUENCE. EACH UNIT MUST HAVE A P-TABLE IN MEMORY DUE TO AN EARLIER HARDWARE DIALOGUE. THIS COMMAND MUST BE FOLLOWED BY A RESTART OR CONTINUE. THE UNITS SWITCH MUST BE SPECIFIED. THE ADD COMMAND IS MEANINGFUL ONLY FOR UNITS THAT WERE PREVIOUSLY DROPPED.

6.3.6 DROP COMMAND

DRO(P)/UNITS:<UNIT-LIST>

6.3.6.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.6.2 EFFECT OF DROP COMMAND

THE UNITS SPECIFIED WILL BE DROPPED FROM TESTING. THE UNITS WILL BE RESELECTED ONLY BY THE EXECUTION OF AN ADD OR START COMMAND. THE UNITS SWITCH MUST BE ENTERED. THIS COMMAND MUST BE FOLLOWED BY A RESTART OR A CONTINUE COMMAND.

6.3.7 PRINT COMMAND

PRI(NT)

6.3.7.1 EFFECT OF PRINT COMMAND

THE TOTAL NUMBER OF ERRORS FOR EACH UNIT SINCE THE LAST START OR RESTART COMMAND ARE PRINTED. THE ISR (INHIBIT STATISTICAL REPORTING) FLAG IS CLEARED.

6.3.8 DISPLAY COMMAND

DIS(PLAY)/UNITS:<UNIT-LIST>

6.3.8.1 UNITS SWITCH (/UNITS:<UNIT-LIST>)

<UNIT-LIST> IS AS IN THE RESTART COMMAND.

6.3.8.2 EFFECT OF DISPLAY COMMAND

THE HARDWARE P-TABLES FOR ALL UNITS UNDER TEST ARE PRINTED OUT IN THE FORMAT IN WHICH THEY WERE ENTERED. ANY UNITS THAT WERE DROPPED BY THE OPERATOR 'DROP' COMMAND ARE SO DESIGNATED.

6.3.9 FLAGS COMMAND

FLA(GS)

6.3.9.1 EFFECT OF FLAGS COMMAND

THE CURRENT SETTINGS OF ALL FLAGS ARE PRINTED.

6.3.10 ZFLAGS COMMAND

ZFL(AGS)

6.3.10.1 EFFECT OF ZFLAGS COMMAND

ALL FLAGS ARE CLEARED.

6.3.11 CONTROL CHARACTERS

A CONTROL C (C) ENTERED DURING THE EXECUTION OF A DIAGNOSTIC CAUSES A RETURN TO COMMAND MODE.

A CONTROL Z (Z) ENTERED DURING ONE OF THE THREE OPERATOR DIALOGUES- HARD CORE QUESTIONS (SEE 6.2), HARDWARE DIALOGUE (SEE 6.3.1.5), OR SOFTWARE DIALOGUE (SEE 6.3.1.5) CAUSES THE DEFAULTS TO BE TAKEN FOR THE REMAINDER OF THAT DIALOGUE.

A CONTROL O (O) ENTERED DURING THE EXECUTION OF A DIAGNOSTIC CAUSES ALL TELETYPE OUTPUT TO BE SURPRESSED FOR THE REMAINDER OF THE DIAGNOSTIC OR UNTIL ANOTHER O IS TYPED, WHICH RESTORES NORMAL TELETYPE OUTPUT.

6.3.12 HARDWARE PARAMETERS

THE FOLLOWING 8 QUESTIONS WILL BE ASKED ON A START COMMAND. THE VALUE LOCATED TO THE LEFT OF THE QUESTION MARK IS THE DEFAULT VALUE THAT WILL BE TAKEN ON A CARRIAGE RETURN RESPONSE.

1. DEVICE CSR ADDRESS : (O) 160170?

THIS IS THE ADDRESS AT WHICH THE CSR REGISTERS (SELO) RESIDE ON THE UNIBUS. THE ALLOWABLE RANGE IS 160000-177776 (OCTAL), AND THE DEFAULT VALUE IS 160170.

2. DEVICE VECTOR ADDRESS : (O) 300 ?

THIS IS THE ADDRESS OF THE INPUT INTERRUPT VECTOR FOR THIS DEVICE. THE ALLOWABLE RANGE IS 000-674 (OCTAL), AND THE DEFAULT VALUE IS 300.

3. DEVICE PRIORITY LEVEL : (O) 5 ?

THIS IS THE CPU PRIORITY AT WHICH THE INTERRUPT HANDLERS OF THIS DEVICE WILL BE EXECUTED. THE ALLOWABLE RANGE IS 0-7, AND THE DEFAULT VALUE IS 5.

4. M8203 SWITCH PACK #1 (REG 11) : (O) 0 ?

THIS IS THE EXPECTED CONTENT (OCTAL) OF SWITCH PACK #1, WHICH RESIDES IN INBUS REG 11. THE ALLOWABLE RANGE IS 000-053, AND THE DEFAULT VALUE IS 000.

5. M8203 SWITCH PACK #2 (REG 15) : (O) 0 ?

THIS IS THE EXPECTED CONTENT (OCTAL) OF SWITCH PACK #2, WHICH RESIDES IN INBUS REG 15. THE ALLOWABLE RANGE IS 000-377, AND THE DEFAULT VALUE IS 000.

6. M8203 SWITCH PACK #3 (REG 16) : (O) 0 ?

THIS IS THE EXPECTED CONTENT (OCTAL) OF SWITCH PACK #3, WHICH RESIDES IN INBUS REG 16. THE ALLOWABLE RANGE IS 000-377, AND THE DEFAULT VALUE IS 000.

7. TURNAROUND TYPE -

(0=H3254&H3255, 1=CABLE, 2=MOD LOC, 3=MOD REM, 4=NONE)
: (0) 0 ?

THIS INDICATOR TELLS THE PROGRAM WHETHER TEST CONNECTOR(S) WILL BE MOUNTED SO THAT CERTAIN TESTS CAN BE RUN IN EXTERNAL LOOPBACK MODE. THE ALLOWABLE ANSWERS ARE 0-4, AND THE DEFAULT VALUE IS 0. 0 MEANS THAT THE H3254 AND H3255 TEST CONNECTORS WILL BE USED. 1 MEANS THAT EXTERNAL TURNAROUND WILL BE PROVIDED AT THE FAR END OF A CABLE ATTACHED TO THE J1 OR J2 CONNECTOR. 2 MEANS THAT MODEM LOCAL LOOPBACK WILL BE USED, AND 3 MEANS THAT MODEM REMOTE LOOPBACK WILL BE USED. 4 MEANS THAT NO EXTERNAL TURNAROUND WILL BE PROVIDED. WHEN 0 IS SELECTED, ALL TESTS WILL BE RUN, AND IF 1-4 IS SELECTED, CERTAIN TESTS CANNOT BE RUN, AND THE PROGRAM WILL TYPE THE NUMBER(S) OF TEST(S) TO BE SKIPPED.

8. PLEASE SELECT BAUD RATE; TYPE '0' FOR 2.4K; '1' FOR 4.8K;
'2' FOR 9.6K; '3' FOR 19.2K; '4' FOR 56K; '5' FOR 250K;
'6' FOR 500K; OR '7' FOR 1 MEG BAUD : (0) 4?

THIS IS THE BAUD RATE WHICH IS SELECTED IN THE SWITCH PACK ON THE M8203. THE ALLOWABLE RANGE IS 0-7, AND THE DEFAULT VALUE IS 4 (FOR 56K).

6.3.13 SOFTWARE PARAMETERS

FOUR SOFTWARE PARAMETER QUESTIONS ARE ASKED BY THE M8203 STATIC LOGIC TESTS PROGRAM, PART 2. THESE QUESTIONS ARE THE FOLLOWING:

1. IS MAN. INTERVEN. DESIRED TO MOUNT TEST CONNECTOR(S)
(L) N?

IF THE OPERATOR ANSWERS THE QUESTION WITH Y (YES), THE PROGRAM WILL LATER PAUSE BEFORE TESTING EACH LOGICAL UNIT AND INFORM THE OPERATOR TO INSTALL THE APPROPRIATE TEST CONNECTOR(S) ON THAT UNIT, AND THEN PROCEED TO TEST THAT UNIT. IF THE OPERATOR ANSWERS N (NO) TO THE ABOVE QUESTION, THE PROGRAM WILL PERFORM TESTING ON ALL UNITS WITHOUT ALLOWING MANUAL INTERVENTION BETWEEN UNITS. IN THIS CASE, ALL TEST CONNECTOR(S) ON ALL UNITS SHOULD BE INSTALLED PRIOR TO RUNNING THE PROGRAM.

2. SHOULD SWITCH PACK AND AX3-15 PRINTOUT BE ALLOWED (L) N ?

IF THE OPERATOR ANSWERS YES, THE PROGRAM WILL ALLOW THE PRINTOUT OF SWITCH PACKS 1-3 AND MODEM INTERFACE REG AX3-15 ON ANY PASS IN WHICH THE CORRESPONDING TESTS ARE RUN. THE DEFAULT IS NO, WHICH ONLY ALLOWS THE PRINTOUT ON THE FIRST PASS AFTER LOADING, IF THE TESTS ARE RUN.

3. SHOULD SWITCH PACK TESTS BE ALLOWED (L) N ?

IF THE OPERATOR ANSWERS YES, THE PROGRAM WILL ALLOW THE READING AND COMPARISON OF SWITCH PACKS 1-3 TO VALUES ENTERED INTO THE HARDWARE P-TABLE FOR THIS UNIT, IF THE

CORRESPONDING TEST IS RUN. IF ALLOWED, SWITCH PACK ERRORS WILL BE REPORTED. THE DEFAULT IS NO, AND THE TESTS ARE NOT RUN.

4. MSG TIMER VALUE (0-177777), 0 = LONGEST TIME-OUT : (0)
0 ?

THIS VALUE CONTROLS THE DURATION OF THE RECEIVER MESSAGE TIME-OUT IN A NUMBER OF TESTS WHICH SEND AND RECEIVE MESSAGES ON THE VARIOUS MODEM INTERFACES WITH EXTERNAL LOOPBACK. THE SMALLER THE VALUE, THE LONGER THE TIME-OUT (UP TO SEVERAL SECONDS). THE DEFAULT IS 0.

6.3.14 EXTENDED DISCUSSION OF P-TABLE DIALOGUE

THE FULL CAPABILITY OF THE HARDWARE DIALOGUE IS REVEALED BY THE FOLLOWING DISCUSSION OF WHAT HAPPENS INTERNALLY.

AS SOON AS THE QUESTION '# UNITS?' IS ANSWERED (WITH THE NUMBER N, SAY) SPACE IN CORE IS ALLOCATED FOR N P-TABLES. ALL OF THE P-TABLES ARE OF THE SAME FORMAT, AND THERE IS A ONE-TO ONE CORRESPONDENCE BETWEEN THE HARDWARE PARAMETER QUESTIONS AND THE SLOTS IN THE P-TABLE FORMAT.

ON THE FIRST TRIP THRU THE QUESTIONS, ALL OF THE SLOTS IN ALL OF THE P-TABLES ARE FILLED. IF THE OPERATOR TYPES IN LESS THAN N EXPLICIT VALUES IN RESPONSE TO A PARTICULAR QUESTION, THESE VALUES ARE PLACED IN THE P-TABLES (ONE VALUE GOING INTO THE PROPER SLOT OF EACH P-TABLE BEGINNING WITH THE FIRST P-TABLE) UNTIL THE STRING OF VALUES IS EXHAUSTED. THE LAST VALUE IN THE STRING BECOMES THE NEW DEFAULT AND IS USED TO FILL THAT SLOT IN THE REMAINING P-TABLES.

ON SUBSEQUENT TRIPS THRU THE QUESTIONS, THE SAME PROCESS IS CARRIED OUT, EXCEPT THAT THE EARLIEST P-TABLE NOT TO HAVE RECEIVED AN EXPLICIT VALUE IN ANY OF ITS SLOTS NOW ASSUMES THE ROLE THAT TABLE NUMBER ONE PLAYED IN THE FIRST TRIP.

THE SERIES OF QUESTIONS IS REISSUED UNTIL AT LEAST ONE QUESTION HAS RECEIVED N EXPLICIT VALUES FROM THE OPERATOR. IN GIVING A STRING OF VALUES, COMMAS WITHOUT INTERVENING VALUES MAY BE USED TO INDICATE A REPETITION OF THE LAST NAMED VALUE.

A STRING OF VALUES MAY BE GIVEN AS A RANGE (6-10 FOR EXAMPLE). IF THE VALUES REPRESENT PURE NUMERICAL DATA, THIS SAMPLE RANGE TRANSLATES TO THE STRING 6,7,8,9,10 (AN INCREMENT OF 1). IF THE VALUES ARE ADDRESSES, THE SAMPLE RANGE TRANSLATES TO THE STRING 6,8,10 (AN INCREMENT OF 2).

NOW LET US SEE HOW WE COULD USE THESE CAPABILITIES TO CONSTRUCT A SET OF P-TABLES. ASSUME THAT WE HAVE 16 UNITS, AND THAT THERE ARE THREE HARDWARE PARAMETERS FOR EACH (THREE SLOTS IN THE P-TABLE, THREE HARDWARE QUESTIONS IN THE DIALOGUE). LET THE DESIRED VALUE FOR THE FIRST PARAMETER BE THE NUMBER 75 FOR ALL 16 TABLES. LET THE DESIRED VALUE FOR THE SECOND PARAMETER BE EQUAL TO THE UNIT NUMBER (0,1,2,...,15) EXCEPT FOR UNIT 12, WHICH SHOULD RECEIVE THE VALUE 11. LET THE DESIRED VALUE FOR THE THIRD PARAMETER BE

THE NUMBER 76 FOR THE FIRST 7 UNITS AND THE NUMBER 77 FOR THE LAST 9 UNITS.

THE FOLLOWING DIALOGUE WOULD ACCOMPLISH THIS GOAL:

UNITS (D) ? 16

UNIT 0

<QUESTION 1> ? 75

<QUESTION 2> ? 0-6

<QUESTION 3> ? 76

UNIT 7

<QUESTION 1> ?

<QUESTION 2> ? 7-11,,13-15

<QUESTION 3> ? 77

THE FIRST TIME THE SERIES IS ASKED, SLOT ONE RECEIVES A 75 IN ALL 16 TABLES. SLOT TWO RECEIVES THE VALUES 0,1,2,...,6 IN TABLES 0 THRU 6 AND A CONSTANT 6 IN TABLES 7 THRU 15. SLOT THREE RECEIVES A CONSTANT 76 IN ALL 16 TABLES.

THE SECOND TIME THRU THE SERIES, TABLES 7 THRU THE END ARE GOING TO BE AFFECTED (NOTE THAT THIS PIECE OF INFORMATION IS PRINTED OUT FOR THE THE OPERATOR IN THE FORM 'UNIT XX' AT THE BEGINNING OF EACH SERIES). QUESTION 1 IS RESPONDED TO BY A <CR>, SO SLOT ONE STAYS AT CONSTANT 75 IN TABLES 7 THRU 15, SINCE NO NEW EXPLICIT VALUES ARE TYPED IN. SLOT TWO GETS THE VALUES 7,8,9,10,11 IN TABLES 7 THRU 11, AND GETS AN 11 IN SLOT 12, AND GETS THE VALUES 13,14,15 IN TABLES 13 THRU 15. SLOT THREE GETS THE VALUE 77 IN TABLES 7 THRU 15.

THE DIALOGUE IS TERMINATED WHEN THE SOFTWARE RECOGNIZES THAT 16 EXPLICIT VALUES HAVE BEEN GIVEN FOR AT LEAST ONE QUESTION (NAMELY QUESTION 2).

7.0 DEVICE INFORMATION TABLES

* MAINTENANCE REGISTER - BSEL1

RUN = BIT7
MCLR = BIT6
STEPLU - BIT4
LULoop = BIT3
ROMO - BIT2
ROMI = BIT1
STEPMP BIT0


```

;*****
* OBUS REG 10 - TRANSMITTER BUFFER
;*****
TX7    = BIT7
TX6    = BIT6
TX5    = BIT5
TX4    = BIT4
TX3    = BIT3
TX2    = BIT2
TX1    = BIT1
TX0    = BIT0

```

```

;*****
* OBUS REG 11
;*****
OC     = BIT7
GOAH   = BIT3
ABORT  = BIT2
EOM    = BIT1
SOM    = BIT0

```

```

;*****
* OBUS REG 12
;*****
IC     = BIT7
BPOLL  = BIT6
LULP   = BIT5

```

```

;*****
* OBUS REG 13
;*****
POLL   = BIT7
DTR    = BIT6
SEIFR  = BIT5
HDX    = BIT4
MAINT1 = BIT3
MAINT2 = BIT2
SELSBY = BIT1

```

```

;*****
* OBUS REG 14
;*****
TXEN   = BIT6
DISSI  = BIT5
RDAX   = BIT4
WAX    = BIT3
ENAX   = BIT2
AX2    = BIT1
AX1    = BIT0

```

```

:*****
* OBUS REG 17
:*****
CRC2   = BIT7
CRC1   = BIT6
IDLE   = BIT5
SECA   = BIT4
STRIP  = BIT3
RDALL  = BIT2
IERR   = BIT1
DDCMP  = BIT0

```

```

:*****
* IBUS REG 10 - RECEIVER BUFFER
:*****
RX7    = BIT7
RX6    = BIT6
RX5    = BIT5
RX4    = BIT4
RX3    = BIT3
RX2    = BIT2
RX1    = BIT1
RX0    = BIT0

```

```

:*****
* IBUS REG 11
:*****
OC     - BIT7
OACT   = BIT6
SW3    = BIT5
ORDY   - BIT4
SW2    - BIT3
SW1    = BIT2
SW0    - BIT1
UNRR   - BIT0

```

```

:*****
* IBUS REG 12
:*****
IC     = BIT7
IACT   = BIT6
LULP   = BIT5
IRDY   - BIT4
OVR    = BIT3
RAB    = BIT2
EBLK   = BIT1
BCC    = BIT0

```

```

:*****
* IBUS REG 13
:*****
RING   = BIT7
DTR    = BIT6
RTS    = BIT5
HDX    = BIT4
MODR   - BIT3
CS     = BIT2
STBY   = BIT1
CARR   = BIT0

```

```

*****
* IBUS REG 14
*****
READY  = BIT7
TXEN   - BIT6
DISSI  = BIT5
RDAX   = BIT4
WAX    - BIT3
ENAX   - BIT2
AX2    - BIT1
AX1    = BIT0

```

```

*****
* IBUS REG 17
*****
SIGH   = BIT7
SIGQ   = BIT6
TXDATA = BIT5
OCOR   = BIT4
ICIR   = BIT3
TESTMD = BIT2
MCLK   = BIT1
DDCMP  = BIT0

```

```

*****
* AX0-15 - USYRT REG 0 (READ ONLY)
*****
RX7    = BIT7
RX6    = BIT6
RX5    = BIT5
RX4    = BIT4
RX3    - BIT3
RX2    = BIT2
RX1    - BIT1
RX0    = BIT0

```

```

*****
* AX0-16 - USYRT REG 1 (READ ONLY)
*****
RERR   = BIT7
ASBC2  = BIT6
ASBC1  = BIT5
ASBC0  = BIT4
ROR    = BIT3
RABT   = BIT2
REOM   = BIT1
RSOM   = BIT0

```

```

*****
* AX1-15 - USYRT REG 2
*****
TX7    = BIT7
TX6    = BIT6
TX5    = BIT5
TX4    = BIT4
TX3    = BIT3
TX2    = BIT2
TX1    = BIT1
TX0    = BIT0

```

```

*****
* AX1-16 - USYRT REG 3
*****
TERR      = BIT7
TXGA      = BIT3
TXAB      = BIT2
TEOM      = BIT1
TSOM      = BIT0

```

```

*****
* AX2-15 - USYRT REG 4
*****
SYN7      = BIT7
SYN6      = BIT6
SYN5      = BIT5
SYN4      = BIT4
SYN3      = BIT3
SYN2      = BIT2
SYN1      = BIT1
SYNO      = BIT0
SYNCH     = 226

```

```

*****
* AX2-16 - USYRT REG 5
*****
APA       = BIT7
DDC       = BIT6
STR       = BIT5
SEC       = BIT4
IDL       = BIT3
CRCTY2    = BIT2
CRCTY1    = BIT1
CRCTY0    = BIT0

```

```

*****
* AX3-15 - USYRT REG 6
*****
I422     = BIT7
XYZ       = BIT6
C32BCC   = BIT5
V35      = BIT4
INTGRL   = BIT3
C32ENB   = BIT2
OP        = BIT1
TEST     = BIT0
AX315U   = I422!XYZ!C32BCC!V35!INTGRL!OP

```

```

*****
* AX3-16 - USYRT REG 7
*****
TXLEN2   = BIT7
TXLEN1   = BIT6
TXLENO   = BIT5
RXLEN2   = BIT2
RXLEN1   = BIT1
RXLENO   = BIT0

```

TEST 1 - BIT STUFFING TEST

*
* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
* INITIATED IN BIT MODE . TWO LEADING FLAGS ARE SENT,
* FOLLOWED BY ALL SIXTEEN CHARS IN DATA PATTERN S. THIS PATTERN
* CONSISTS OF CHARACTERS WHICH REQUIRE NO BIT STUFFING AND CHARACTERS
* WHICH REQUIRE BIT STUFFING INDIVIDUALLY AND IN COMBINATION WITH
* ADJACENT CHARACTERS. ALL 16 CHARACTERS ARE READ AND COMPARED
* BY THE RECEIVER.
* PATTERN S = 000,017,036,074,170,360,037,076,174,370,077,176,374,
* 177,376,377

TEST 2 - RCV OVERRUN ERROR SET AND CLEAR TEST

*
* IN THIS TEST, A RCV OVERRUN ERROR IS FORCED IN EACH OF 2 SUBTESTS.
* IN THE FIRST, A MESSAGE IS INITIATED, 64 001 CHARS ARE SENT, AND THE
* RECEIVER IS NOT SERVICED IN RESPONSE TO THE USYRT RCV FLAG, WHICH CAUSES RCV
* OVERRUN TO SET. THEN, A CHECK IS MADE TO INSURE THAT OVRR IS NOT
* CLEARED BY THE LINE UNIT READING THE USYRT STATUS.
* THEN, IC IS SET TO CLEAR THE ERROR, AND THIS IS VERIFIED.
*
* IN THE SECOND SUBTEST, RCV OVRUN IS FORCED AGAIN, AND A MASTER CLEAR
* IS ISSUED TO CLEAR THE ERROR, AND THIS IS VERIFIED.

TEST 3 - ABORT SEQUENCE TEST

*
* SET BIT MODE, CRC, AND ENABLE THE DEVICE FOR
* TRANSMIT AND RECEIVE. SEND 2 FLAGS AND 4 DATA CHARS (001).
* AS THE FIRST DATA CHAR IS BEING TRANSMITTED,
* SET THE ABORT BIT (REG 11).
* ON THE RECEIVER SIDE, CHECK FOR RECEPTION OF THE FIRST DATA CHAR
* AND THEN THE SETTING OF RAB AND REOM A CHAR TIME LATER.
* ALSO, CHECK FOR IACT = 0. THEN, CHECK THAT RAB
* IS CLEARED BY READING THE USYRT STATUS, TRANSMITTING A NEW MSG,
* RECEIVING THE FIRST CHAR (003) AND CHECKING FOR RAB CLEARED.
*
* REPEAT THE ABOVE SEQUENCE, SET IC, AND CHECK THAT
* THIS CLEARS RAB.
*
* REPEAT THE ABOVE SEQUENCE, ISSUE MASTER CLEAR, CHECK THAT THIS
* CLEARS RAB.

```

*****
; TEST 4 - ABORT AND IDLE FLAGS TEST
*
* TRANSMIT THE SAME ABORT SEQUENCE AS IN THE PREVIOUS TEST, BUT
* WITH THE IDLE BIT SET. CHECK THAT FLAGS ARE SENT AND RECEIVED
* (NOT ABORT CHARACTERS) BY VERIFYING THAT RAB DOES
* NOT SET, AND THAT THE MESSAGE TERMINATES WITH EBLK = 1.
*****

```

```

*****
; TEST 5 - TRANSMITTER UNDERRUN ERROR, IDLE ABORT CHARS, BIT MODE
*
* A MESSAGE IS INITIATED IN BIT MODE, 4 001 CHARS ARE SENT, AND THE TRANSMITTER
* IS NOT SERVICED IN RESPONSE TO THE LAST TX FLAG, WHICH CAUSES TX
* UNDERRUN ERROR TO SET. ON THE RECEIVER SIDE, CHECK THAT THE DATA
* CHAR IS RECEIVED, AND THAT 8 CYCLES LATER THE RAB BIT SETS, AND
* THE DEVICE IDLES ABORT CHARACTERS.
*****

```

```

*****
; TEST 6 - RECEIVER DISABLE TEST
*
* TRANSMIT AND RECEIVE ARE ENABLED IN BIT MODE, AND 2 FLAGS
* ARE SENT, FOLLOWED BY 5 252 DATA CHARS. AFTER THE SECOND DATA CHAR HAS BEGUN
* TO BE RECEIVED, IC IS SET.
* THEN, THE PROGRAM CHECKS THAT A USYRT RCV FLAG IS NOT GENERATED, AND
* THE RECEIVER DATA PATH STOPS OPERATING IN THE MIDDLE OF THE CHAR.
*****

```

```

*****
; TEST 7 - ASSEMBLED BIT COUNT TEST
*
* THE FOLLOWING SEQUENCE IS PERFORMED 8 TIMES, EACH TIME USING A
* DIFFERENT TX CHAR LENGTH (FROM 2 TO 8 BITS) AND A RCV CHAR LENGTH = 8
* BITS :
* A MESSAGE IS INITIATED IN BIT MODE, NO CRC.
* 2 FLAGS ARE SENT, FOLLOWED BY 3 000 DATA CHARACTERS AND A
* TERMINATING FLAG. AFTER THE RECEIVER HAS RECEIVED THE MESSAGE, AX0-16
* IS READ TO RETRIEVE THE ASSEMBLED BIT COUNT. THIS COUNT IS CHECKED TO INSURE
* THAT IT IS CORRECT FOR THE TX CHAR LENGTH USED IN THAT TRANSMISSION.
*****

```

```

;*****
TEST 8 - SECONDARY STATION ADDRESS BIT TEST
*
* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
* BIT MODE, AND THE SECA BIT (REG 17) IS SET.
* 2 FLAGS ARE SENT, FOLLOWED BY 252, 000, AND A TERMINATING FLAG.
* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
* RECEIVED.
*
* NEXT, THE SECONDARY STATION ADDRESS BITS IN AX2-15 ARE LOADED
* WITH THE FIRST WORD OF DATA PATTERN T. 2 FLAGS ARE SENT,
* FOLLOWED BY THE FIRST WORD OF DATA PATTERN T, A 000 CHAR,
* AND A TERMINATING FLAG.
* THEN, THE RCV'D DATA IS CHECKED TO MAKE SURE THAT THE SEC STATION
* ADDRESS IS RCV'D AS THE FIRST DATA CHAR, FOLLOWED BY 000.
*
* THEN, THE SUBTEST IS REPEATED FOR EACH OF THE REMAINING WORDS OF
* DATA PATTERN T.
* PATTERN T = 000,125,252,176,177
;*****

```

```

;*****
TEST 9 - RDALL (ALL PARTIES ADDRESS) BIT TEST
*
* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
* BIT MODE, AND THE SECA BIT IS SET.
* 2 FLAGS ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
* RECEIVED.
* NEXT, THE RDALL BIT IN REG 17 IS SET TO 1. 2 FLAGS
* ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
* THEN, THE REC'D DATA IS CHECKED TO MAKE SURE THAT 377
* IS REC'D AS THE FIRST DATA CHAR, FOLLOWED BY 125.
;*****

```

```

;*****
TEST 10 - INSERT ERROR (IERR) BIT TEST - CHAR MODE, NO CRC
*
* THE LINE UNIT IS PLACED IN DDCMP MODE WITH NO ERROR DETECTION, AND 2
* SYNCHS, A 000 CHAR, A 377 CHAR, AND 2 SYNCHS ARE LOADED INTO THE
* TRANSMITTER SILO. THEN, THE LU IS CLOCKED UNTIL THE 2ND BIT OF THE 000
* CHAR IS ABOUT TO BE SENT AND THE IERR BIT IS SET FOR A CLOCK TIME AND
* THEN CLEARED. IN THE SAME WAY, IERR IS SET PRIOR TO THE SENDING OF THE 4TH
* AND 5TH BITS OF THE 000 CHAR. IT IS ALSO SET FOR THE SENDING OF THE FIRST
* 4 BITS OF THE 377 CHAR. THE PROGRAM READS THE FIRST RCV'D CHAR FROM AX0
* AND CHECKS IT TO BE 032, AND READS THE 2ND CHAR AND CHECKS IT TO BE 377.
* THEN, A MASTER CLEAR IS DONE TO IDLE THE DEVICE.
;*****

```

```

*****
TEST 11 - SWITCH PACK PRINTOUT AND TEST
*
* - READ AND PRINT SWITCH PACK 1 :
* THE PROGRAM READS REG 11 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
* SWITCHES ARE IN BITS 1,2,3,5.
*
* - READ AND PRINT SWITCH PACK 2 :
* THE PROGRAM READS REG 15 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
* SWITCHES ARE IN BITS 0-7.
*
* - READ AND PRINT SWITCH PACK 3 :
* THE PROGRAM READS REG 16 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
* (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
* THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
* SWITCHES ARE IN BITS 0-7.
*****

```

```

*****
TEST 12 - REG AX3-15 PRINTOUT
*
* IN THIS TEST, REG AX3-15 IS READ AND THE CONTENTS PRINTED OUT IF DESIRED BY
* THE OPERATOR, AS INDICATED IN THE SOFTWARE P-TABLE. THE DEFAULT IS TO NOT
* PRINT THE REG.
*****

```

```

*****
TEST 13 - CRC GENERATION TEST
*
* - CRC-16, CHAR MODE:
* THE FOLLOWING MESSAGE IS SENT IN DDCMP MODE WITH CRC-16 SELECTED -
* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOB AND STEPLU
* TO CLOCK THE DATA. AT THE END OF THE MESSAGE THE
* PROGRAM CHECKS FOR BCC = 1 (IN REG 12) INDICATING NO ERROR.
*
* - CRC-CCITT - 1'S PRESET:
* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT
* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC = 0, INDICATING NO ERROR.
*
* - CRC-CCITT - 0'S PRESET:
* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT
* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC = 0, INDICATING NO ERROR.
*****

```



```

*****
TEST 14 - CRC ERROR DETECTION TEST
*
* - CRC-16, CHAR MODE :
* THE FOLLOWING MESSAGE IS SENT IN DDCMP MODE, WITH CRC-16 SELECTED -
* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOB AND STEPLU
* TO CLOCK THE DATA. JUST BEFORE THE FIRST BIT OF THE LAST 000 CHAR IS SENT,
* THE IERR BIT IS SET IN REG 17 TO CAUSE A 1 TO BE SENT, INTRODUCING A DATA
* ERROR. AT THE END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 0, INDICATING
* AN ERROR.
*
* - CRC-CCITT - 1'S PRESET :
* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT THE
* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 1, INDICATING AN ERROR.
*
* - CRC-CCITT - 0'S PRESET :
* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT THE
* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 1, INDICATING AN ERROR.
*****

```

```

*****
TEST 15 - VRC PARITY GENERATION TEST
*
* SUBTEST 1 - TEST OF CORRECT ODD VRC PARITY GENERATION :
* THE LINE UNIT IS PLACED IN CHAR MODE, WITH ODD VRC AND 7-BIT CHARS SELECTED.
* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)
* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.
* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 1 AND FOR THE
* LAST 4 CHARS IT SHOULD = 0.
*
* SUBTEST 2 - TEST OF CORRECT EVEN VRC PARITY GENERATION :
* THE LINE UNIT IS PLACED IN CHAR MODE, WITH EVEN VRC AND 7-BIT CHARS SELECTED.
* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)
* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.
* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 0 AND FOR THE
* LAST 4 CHARS IT SHOULD = 1.
*
* DATA PATTERN Q = 000,120,125,137,040,052,057,177
*****

```

```

;*****
TEST 16 - VRC ERROR DETECTION TEST
*
* SUBTEST 1 - FORCING OF BCC USING ODD VRC
* THE LINE UNIT IS PLACED IN CHAR MODE WITH ODD VRC AND 7-BIT CHARS SELECTED.
* THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER
* 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM
* CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING
* NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE
* RECEIVED (INDICATING AN ERROR).
*
* SUBTEST 2 - FORCING OF BCC USING EVEN VRC
* THE LINE UNIT IS PLACED IN CHAR MODE WITH EVEN VRC AND 7-BIT CHARS SELECTED.
* THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER
* 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM
* CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING
* NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE
* RECEIVED (INDICATING AN ERROR).
*
* DATA PATTERN R = 000,100,120,124,164,172,176,177.000,100,120,124,164,
*                   172,176.
;*****

```

```

;*****
TEST 17 - INTEGRAL MODEM INTERFACE TEST - CHAR MODE, CRC
*
* THE INTEGRAL MODEM IS SELECTED BY THE PROGRAM IN AX3-15, AND A
* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
* ON THE LINE UNIT OR AT THE CABLE. THE MESSAGE CONSISTS OF
* 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
* SKIPPED FOR THAT UNIT.
;*****

```

```

;*****
TEST 18 - V.35 MODEM INTERFACE TEST - CHAR MODE, CRC
*
* THE V.35 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A
* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,
* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF
* 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
* SKIPPED FOR THAT UNIT.
;*****

```

```

*****
TEST 19 - RS 232C AND RS 423 MODEM INTERFACE TEST - CHAR MODE, CRC
*
* THE RS232C RS423 (XYZ) MODEM INTERFACE IS SELECTED BY THE PROGRAM IN
* AX3-15, AND A MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURN-
* AROUND CONNECTOR ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,
* OR A MODEM TEST MODE. THE MESSAGE CONSISTS
* OF 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE
* P-TABLE FOR THE CURRENT UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS
* PROVIDED, THE TEST WILL BE SKIPPED FOR THAT UNIT.
*****

```

```

*****
TEST 20 - RS 422 MODEM INTERFACE TEST - CHAR MODE, CRC
*
* THE RS 422 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A
* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,
* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF
* 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
* SKIPPED FOR THAT UNIT.
*****

```

```

*****
TEST 21 - HALF-DUPLEX BIT (HALF DUPX) TEST
*
* THIS TEST VERIFIES THAT SETTING HALF-DUPLEX BIT IN REG 13 DOES NOT INHIBIT
* LOADING OF THE USYRT TRANSMITTER FROM THE TRANSMITTER SILO.
* A MASTER CLEAR IS ISSUED, DDCMP MODE IS ENTERED, AND THE HALF DUPX
* BIT IN REG 13 IS SET. A MESSAGE IS LOADED INTO THE TX SILO
* CONSISTING OF 2 SYNCHS, 000,125,252,377,000, AND 2 MORE SYNCHS.
* THE LINE UNIT IS THEN CLOCKED EXTENSIVELY, AND THE TX SILO IS CHECKED TO
* BE UNLOADED (ALL CHARS SHOULD HAVE BEEN REMOVED) AND THE RECEIVER
* IS MONITORED TO INSURE THAT NO RCV FLAGS ARE GENERATED.
*****

```

```

*****
TEST 22 - HALF-DUPLEX RCV DISABLED TEST WITH SILOS DISABLED
*
* THIS TEST SENDS A MESSAGE IN HDX, CHAR MODE, WITH NO ERROR DETECTION, AND
* THE SILOS DISABLED. THE MSG CONSISTS OF 2 SYNCHS AND 2 000 CHARS.
* THE DATA IS SENT WITH LULOOP SET FOR TTL DATA LOOPBACK. THE PROGRAM CHECKS
* THAT THE RECEIVER NEVER BECOMES ACTIVE, BECAUSE THE RCV CLOCK IS INHIBITED
* WHEN THE HDX BIT IS SET.
*****

```

TEST 23 - INTERACTION OF MODEM CONTROL BITS

- * THIS TEST WILL BE RUN ONLY IF THE P-TABLE FOR THIS UNIT INDICATES THAT THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED. OTHERWISE, THE TEST WILL BE SKIPPED FOR THE UNIT.
- * THE FOLLOWING SUBTESTS ARE PERFORMED:
 - * - A MASTER CLEAR IS DONE AND REG 13 IS READ AND CHECKED FOR INITIALIZED STATE, WITH LULOOK SET TO 1. THEN, LULOOK IS CLEARED AND REG 13 IS READ AND CHECKED FOR THE PROPER STATE, WITH LULOOK CLEARED.
 - * REG 13 IS THEN LOADED WITH 0'S, AND READ AND CHECKED FOR THE INITIALIZED STATE.
 - * REG 17 IS THEN READ AND CHECKED FOR INITIALIZED STATE.
 - * - RUN IS SET IN BSEL1, AND REG 13 IS READ AND CHECKED FOR RING SET.
 - * - POLL IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR SIGO SET.
 - * - BPOLL IS SET IN REG 12, ONLY TO LIGHT THE LED FOR THIS SIGNAL.
 - * - DTR IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR DTR AND MODR SET.
 - * - SELFR IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR SIGR SET.
 - * - HDX IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR HDX SET.
 - * - MAINT1 IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR TEST MODE SET.
 - * - SELSBY IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR STBY SET.
 - * - A MASTER CLEAR IS DONE, 2 TSOM'S ARE LOADED INTO THE TX SILO, THE LINE UNIT IS Clocked UNTIL THE TRANSMITTER IS ACTIVE, AND REG 13 IS READ AND CHECKED FOR RTS, CS, CARR SET.

TEST 24 - DATA TEST - BIT MODE, NO ERR DET

- * A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH ERROR DETECTION INHIBITED. THE MESSAGE CONSISTS OF 5 FLAGS, PAT A REPEATED 2 TIMES, AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED, THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
- * IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE TEST WILL NOT BE RUN.
- * PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
375,373,367,357,337,277,177
- * 8-BIT CHARACTERS ARE USED.




```

*****
TEST 25 - DATA TEST - CHAR MODE, NO ERR DET
*
* A MESSAGE IS INITIATED IN CHAR MODE, WITH ERROR DETECTION
* INHIBITED. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
*             375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.
*****

```

```

*****
TEST 26 - DATA TEST - BIT MODE, CRC-CCITT-1
*
* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH CRC-CCITT-1 ERROR
* DETECTION. THE MESSAGE CONSISTS OF 5 FLAGS, PAT A REPEATED 2 TIMES,
* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
*             375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.
*****

```

```

*****
TEST 27 - DATA TEST - BIT MODE, CRC-CCITT-0
*
* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH CRC-CCITT-0 ERROR
* DETECTION. THE MESSAGE CONSISTS OF 5 FLAGS, PAT A REPEATED 2 TIMES,
* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
*             375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.
*****

```

```

*****
TEST 28 - DATA TEST - CHAR MODE, CRC-16
*
* A MESSAGE IS INITIATED IN CHAR MODE, WITH CRC-16 ERROR
* DETECTION. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
*             375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.
*****

```

```

*****
TEST 29 - DATA TEST - CHAR MODE, ODD VRC
*
* A MESSAGE IS INITIATED IN CHAR MODE, WITH ODD VRC ERROR DETECTION
* SELECTED. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
*             375,373,367,357,337,277,177
* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).
*****

```

```

*****
TEST 30 - DATA TEST - CHAR MODE, EVEN VRC
*
* A MESSAGE IS INITIATED IN CHAR MODE, WITH EVEN VRC ERROR DETECTION
* SELECTED. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,
* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
*             375,373,367,357,337,277,177
* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).
*****

```

```

*****
TEST 31 - CONTIGUOUS ONES IN SEC. STA. ADRS. MODE, BIT MODE
*
* IN THIS TEST, A MESSAGE CONSISTING OF 5 ONES CHARS (377 OCT)
* IS SENT IN SECONDARY STATION ADDRESS MODE, WITH THE STATION ADRS
* FOR THIS LINE = 377. THE PROGRAM CHECKS FOR CORRECT RECEPTION OF
* THE FIRST CHARACTER (STATION ADDRESS) AND THE REMAINING 4
* ONES CHARACTERS (DATA). THIS TEST EXERCISES THE SECONDARY STATION
* ADDRESS LOGIC, AND CHECKS THAT THE SEC. STA. ADRS. CAN BE BIT-STUFFED
* AND TRANSMITTED AND RECEIVED CORRECTLY.
*****

```

```

*****
TEST 32 - DDCMP MESSAGE TEST - CHAR MODE
*
* IN THIS TEST, THREE USYRT MESSAGES ARE SENT TO SIMULATE A DDCMP HEADER,
* DDCMP DATA MESSAGE, AND THE START OF A NEW DDCMP HEADER.
* FIRST, THE DATA IN PATTERN A IS TRANSMITTED AND RECEIVED
* AND THEN CRC (CRC-16) IS SENT, FOLLOWED BY THE DATA IN PATTERN A
* AGAIN AND THE CRC ON THAT DATA, AND FINALLY THE DATA IN 'MSG1' IS
* SENT WITH ITS CORRESPONDING CRC.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
*             375,373,367,357,337,277,177
* MSG1 - SYNCH,SYNCH,SYNCH,SYNCH,000,125,252,377,SYNCH,SYNCH
*****

```

8.1 DATA PATTERNS USED

***** DATA PATTERN A *****

PATA:

.BYTE 125
.BYTE 252
.BYTE 000
.BYTE 377
.BYTE 001
.BYTE 002
.BYTE 004
.BYTE 010
.BYTE 020
.BYTE 040
.BYTE 100
.BYTE 200
.BYTE 376
.BYTE 375
.BYTE 373
.BYTE 367
.BYTE 357
.BYTE 337
.BYTE 277
.BYTE 177

***** DATA PATTERN Q *****

PATQ:

.BYTE 000
.BYTE 120
.BYTE 125
.BYTE 137
.BYTE 040
.BYTE 052
.BYTE 057
.BYTE 177

***** DATA PATTERN R *****

PATR:

.BYTE 000
.BYTE 100
.BYTE 120
.BYTE 124
.BYTE 164
.BYTE 172
.BYTE 176
.BYTE 177
.BYTE 000
.BYTE 100
.BYTE 120
.BYTE 124
.BYTE 164
.BYTE 172
.BYTE 176

***** DATA PATTERN S *****

PATS:

.BYTE 000
.BYTE 017
.BYTE 036
.BYTE 074
.BYTE 170

```

.BYTE 360
.BYTE 037
.BYTE 076
.BYTE 174
.BYTE 370
.BYTE 077
.BYTE 176
.BYTE 374
.BYTE 177
.BYTE 376
.BYTE 377

```

***** DATA PATTERN T *****

```

PATT: .BYTE 000
       .BYTE 125
       .BYTE 252
       .BYTE 176
       .BYTE 177

```

9.0 ERROR INFORMATION

9.1 ERROR REPORTING

ERRORS ARE REPORTED BY THE PROGRAM AS THEY OCCUR (IF NOT INHIBITED). THE REPORT CONFORMS TO THE DIAGNOSTIC SUPERVISOR ERROR REPORT FORMAT, AND CONSISTS OF A DESCRIPTION OF THE ERROR, THE TEST NUMBER, SUBTEST NUMBER, PC OF THE ERROR CALL, DEVICE ADDRESS, AND BASIC AND EXTENDED ERROR INFORMATION.

THE FOLLOWING EXAMPLE PROVIDES A TYPICAL ERROR REPORT, WHICH DESCRIBES AN "IRDY NOT SET" ERROR, AND PROVIDES THE PC OF THE ERROR CALL AND THE PC OF THE CALL TO THE SUBROUTINE REPORTING IT, THE FAILING REGISTER NAME, AND DEVICE REGISTER CONTENTS :

```

CZDMS DVC FTL ERR 00017 ON UNIT 00 TST 034 SUB 000 PC: 006210
IRDY NOT SET
PC OF SUBR CALL: 030044
DEVICE CSR ADDRESS : 160170

```

FAILING REG: INBUS/OUTBUS REG 12

LINE UNIT INBUS REGS:

```

REG10  REG11  REG12  REG13
000    120    000    257
REG14  REG15  REG16  REG17
024    377    377    035

```

LINE UNIT EXTENDED REGS:

```

AX0-15 AX0-16 AX1-15 AX1-16
000    000    000    000
AX2-15 AX2-16 AX3-15 AX3-16
000    000    000    000

```


FOR OTHER ERRORS, THE REPORT MAY BE MORE EXTENSIVE, AND
REQUIRE ADDITIONAL DATA TO BE REPORTED.

IF EXTENDED ERROR INFORMATION HAD BEEN INHIBITED USING THE
IXE FLAG PRIOR TO RUNNING THE TEST, THE ABOVE ERROR WOULD
HAVE BEEN REPORTED IN THE FOLLOWING SHORTENED FORM :

CZDMS DVC FTL ERR 00017 ON UNIT 00 TST 034 SUB 000 PC:006210
IRDY NOT SET
PC OF SUBR CALL: 030044
DEVICE CSR ADDRESS : 160170

FAILING REG: INBUS/OUTBUS REG 12

@

```
2194  
2203          002000          .TITLE CZDMSC M8203 STATIC TESTS #2  
2204                                     .=2000  
2205  
2206  
2207  
2208  
2209  
2210  
2211 002000          .MCALL SVC  
2212                                     SVC ; INITIALIZE SUPERVISOR MACROS  
2213  
2214  
2215  
2216  
2217 002000          BGNMOD LU2MOD  
2218  
2219  
2220          000001          $LSTIN= 1  
2221          000001          $LSTTAG= 1  
2222          000001          SVCINS= 1 ; LIST INSTRUCTIONS, SHIFTED RIGHT  
2223          000001          SVCTST= 1 ; LIST TEST TAGS, SHIFTED RIGHT  
2224          000001          SVCSUB= 1 ; LIST SUBTEST TAGS, SHIFTED RIGHT  
2225          000001          SVCGBL= 1 ; LIST GLOBAL TAGS, SHIFTED RIGHT  
2226          000001          SVCTAG= 1 ; LIST OTHER TAGS, SHIFTED RIGHT  
2227  
2228          : CHANGE THE VALUES OF THE SVC... SYMBOLS TO BE ZERO IF YOU WISH  
2229          : TO ALIGN THE MACRO CALLS AND THEIR EXPANSIONS. CHANGE THE  
2230          : SYMBOLS TO BE MINUS-ONE TO NOT LIST THE EXPANSIONS. YOU MAY  
2231          : CHANGE THE SYMBOLS AT ANY POINT IN YOUR PROGRAM.  
2232  
2233
```

2235
2236
2237
2238
2239
2240
2241
2242
2244
2245
2246
2247
2248
2250
2251
(4)
(4)
(4)
(4)
(4)
(4)
(6)
(6)
(5)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)
(4)
(5)

002000

002000
002000 103
002001 132
002002 104
002003 115
002004 123
002005 000
002006 000
002007 000
002010
002010 103
002011
002011 060
002012
002012 000000
002014
002014 000055
002016
002016 036372
002020
002020 037334
002022
002022 002226
002024
002024 002256
002026
002026 040004
002030
002030 000000
002032
002032 000000
002034
002034 000000
002036
002036 000000
002040
002040 002124
002042
002042 000000
002044
002044 000000
002044

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

POINTER BGNSW,BGNSFT,BGNAU,BGNDU

:XX
: IF ANY OPTIONAL POINTERS ARE TO BE USED IN THE 'HEADER', CHANGE
: 'POINTER' TO CONTAIN THE CORRECT ARGUMENTS. IF ALL OPTIONAL
: POINTERS ARE TO BE USED, CHANGE 'POINTER' TO BE 'POINTER ALL'.
:XX

HEADER CZDMS,C,0,45.,0

LSNAME::
 .ASCII /C/
 .ASCII /Z/
 .ASCII /D/
 .ASCII /M/
 .ASCII /S/
 .BYTE 0
 .BYTE 0
 .BYTE 0
LSREV::
 .ASCII /C/
LSDEPO::
 .ASCII /O/
LSUNIT::
 .WORD 0
LSTIML::
 .WORD 45.
LSHPCP::
 .WORD LSHARD
LSSPCP::
 .WORD LSSOFT
LSHPTP::
 .WORD LSHW
LSSPTP::
 .WORD LSSW
LSLADP::
 .WORD LSLAST
L\$STA::
 .WORD 0
L\$CO::
 .WORD 0
LSDTYP::
 .WORD 0
L\$APT::
 .WORD 0
LSDTTP::
 .WORD 0
L\$PRIO::
 .WORD L\$DISPATCH
L\$ENVI::
 .WORD 0
 .WORD 0

(5) 002046
(4) 002046 000000
(5) 002050
(4) 002050 003
(3) 002051 003
(5) 002052
(4) 002052 000000
(5) 002054 000000
(5) 002056
(4) 002056 000000
(5) 002060
(4) 002060 003162
(5) 002062
(4) 002062 000000
(5) 002064
(4) 002064 000000
(5) 002066
(4) 002066 000000
(5) 002070
(4) 002070 023164
(5) 002072
(4) 002072 023102
(5) 002074
(4) 002074 000000
(5) 002076
(4) 002076 003170
(5) 002100
(4) 002100 104035
(5) 002102
(4) 002102 000000
(5) 002104
(4) 002104 022046
(5) 002106
(4) 002106 023100
(5) 002110
(4) 002110 023020
(5) 002112
(4) 002112 022040
(5) 002114
(4) 002114 000000
(5) 002116
(4) 002116 000000
(5) 002120
(4) 002120 000000

L\$EXP1:: .WORD 0
L\$MREV:: .BYTE C\$REVISION
.BYTE C\$EDIT
L\$EF:: .WORD 0
.WORD 0
L\$SPC:: .WORD 0
L\$DEVP:: .WORD L\$DVTYP
L\$REPP:: .WORD 0
L\$EXP4:: .WORD 0
L\$EXP5:: .WORD 0
L\$AUT:: .WORD L\$AU
L\$DUT:: .WORD L\$DU
L\$LUN:: .WORD 0
L\$DESP:: .WORD L\$DESC
L\$LOAD:: EMT E\$LOAD
L\$ETP:: .WORD 0
L\$ICP:: .WORD L\$INIT
L\$CCP:: .WORD L\$CLEAN
L\$ACP:: .WORD L\$AUTO
L\$PRT:: .WORD L\$PROT
L\$TEST:: .WORD 0
L\$DLY:: .WORD 0
L\$HIME:: .WORD 0

2252
2254
2255
2256
2258
2259
2260
2261
2262
2263
2264
2265

:XX
: CHANGE THE 'HEADER' TO CONTAIN THE PROPER ARGUMENTS.
:XX

.EVEN

2288
2289
2290
2291
2292
2293
2294
2295
2296
(3)
(3)
(3)
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
(3)
2311
2312
2313
2314
2315

002224 000013
002224
002226
002226
002226 000007
002230 160170
002232 000300
002234 005000
002236 000003
002240 000056
002242 000000
002244 000000
002246 000000
002250 000004
002252 000001
002254
002254

.SBTTL DEFAULT HARDWARE P-TABLE

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

BGNHW DFPTBL
7
160170
300
5000
3
056
000
000
0
4
1

.WORD L10000-LSHW/2
LSHW::
DFPTBL::
:MICROPROCESSOR TYPE = M8207
:DMC11 OR KMC11 CSR UNIBUS ADDRESS
:DMC11 OR KMC11 INTERRUPT VECTOR
:DMC11 OR KMC11 INTERRUPT PRIORITY LEVEL = 5
:LINE UNIT = M8203
:SWITCH PACK #1 (REG 11)
:SWITCH PACK #2 (REG 15)
:SWITCH PACK #3 (REG 16)
:H3254&H3255 USED
:BAUD RATE = 56 K
:RUN SWITCH ON MICROPROCESSOR IS ON

L10000:

2317
2318
2319
2320
2321
2322
2323
2324
(3)
(3)
(3)
2325
2326
2327
2328
2329
2330
2331
2332
(3)
2333
2334
2335
2336
2337
2338

.SBTTL SOFTWARE P-TABLE

:/
:/ THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
:/ PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
:/

BGNSW SFPTBL

.WORD L10001-L\$SW/2

L\$SW::
SFPTBL::

MIFLAG: .WORD 0 ; =1 IF MAN. INTERVENTION DESIRED, =0 IF NOT
PRNFLG: .WORD 0 ; =1 IF SW PACK AND AX3-15 PRINTOUT ALLOWED ALWAYS
SWIFLG: .WORD 0 ; =1 IF SWITCH PACK VERIFICATION TEST SHOULD BE RUN
TCOUNT: .WORD 0 ; INITIAL MSG TIME-OUT VALUE (0=LONGEST TIME-OUT)

ENDSW

L10001:


```
(1) 000036 EF.CONTINUE== 30. ; CONTINUE COMMAND WAS ISSUED
(1) 000035 EF.NEW== 29. ; A NEW PASS HAS BEEN STARTED
(1) 000034 EF.PWR== 28. ; A POWER-FAIL/POWER-UP OCCURRED
(1)
(1)
(1)
(1)
(1)
(1)
(1) 000340 PRI07== 340
(1) 000300 PRI06== 300
(1) 000240 PRI05== 240
(1) 000200 PRI04== 200
(1) 000140 PRI03== 140
(1) 000100 PRI02== 100
(1) 000040 PRI01== 40
(1) 000000 PRI00== 0
(1)
(1) ; OPERATOR FLAG BITS
(1)
(1) 000004 EVL== 4
(1) 000010 LOT== 10
(1) 000020 ADR== 20
(1) 000040 IDU== 40
(1) 000100 ISR== 100
(1) 000200 UAM== 200
(1) 000400 BOE== 400
(1) 001000 PNT== 1000
(1) 002000 PRI== 2000
(1) 004000 IXE== 4000
(1) 010000 IBE== 10000
(1) 020000 IER== 20000
(1) 040000 LOE== 40000
(1) 100000 HOE== 100000
```

2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383

```
::*****
;* PROGRAM EVENT FLAG DEFINITIONS
::*****
```

```
::*****
;* MAINTENANCE REGISTER - BSEL1
::*****
```

```
RUN = BIT7
MCLR = BIT6
STEPLU = BIT4
LULOOP = BIT3
ROMO = BIT2
ROMI = BIT1
STEPMP = BIT0
```

```
::*****
```

```
2384 ;* OBUS REG 10 - TRANSMITTER BUFFER
2385 :*****
2386 000200 TX7 = BIT7
2387 000100 TX6 = BIT6
2388 000040 TX5 = BIT5
2389 000020 TX4 = BIT4
2390 000010 TX3 = BIT3
2391 000004 TX2 = BIT2
2392 000002 TX1 = BIT1
2393 000001 TX0 = BIT0
2394
2395 :*****
2396 ;* OBUS REG 11
2397 :*****
2398 000200 OC = BIT7
2399 000010 GOAH = BIT3
2400 000004 ABORT = BIT2
2401 000002 EOM = BIT1
2402 000001 SOM = BIT0
2403
2404 :*****
2405 ;* OBUS REG 12
2406 :*****
2407 000200 IC = BIT7
2408 000100 BPOLL = BIT6
2409 000040 LULP = BIT5
2410
2411 :*****
2412 ;* OBUS REG 13
2413 :*****
2414 000200 POLL = BIT7
2415 000100 DTR = BIT6
2416 000040 SELFR = BIT5
2417 000020 HDX = BIT4
2418 000010 MAINT1 = BIT3
2419 000004 MAINT2 = BIT2
2420 000002 SELSBY = BIT1
2421
2422 :*****
2423 ;* OBUS REG 14
2424 :*****
2425 000100 TXEN = BIT6
2426 000040 DISSI = BIT5
2427 000020 RDAX = BIT4
2428 000010 WAX = BIT3
2429 000004 ENAX = BIT2
2430 000002 AX2 = BIT1
2431 000001 AX1 = BIT0
2432
2433 :*****
2434 ;* OBUS REG 17
2435 :*****
2436 000200 CRC2 = BIT7
2437 000100 CRC1 = BIT6
2438 000040 IDLE = BIT5
2439 000020 SECA = BIT4
```

```
2440      000010      STRIP  = BIT3
2441      000004      RDALL  = BIT2
2442      000002      IERR   = BIT1
2443      000001      DDCMP  = BIT0
2444
2445      ::*****
2446      :* IBUS REG 10 - RECEIVER BUFFER
2447      ::*****
2448      000200      RX7    = BIT7
2449      000100      RX6    = BIT6
2450      000040      RX5    = BIT5
2451      000020      RX4    = BIT4
2452      000010      RX3    = BIT3
2453      000004      RX2    = BIT2
2454      000002      RX1    = BIT1
2455      000001      RX0    = BIT0
2456
2457      ::*****
2458      :* IBUS REG 11
2459      ::*****
2460      000200      OC      = BIT7
2461      000100      OACT   = BIT6
2462      000040      SW3    = BIT5
2463      000020      ORDY   = BIT4
2464      000010      SW2    = BIT3
2465      000004      SW1    = BIT2
2466      000002      SW0    = BIT1
2467      000001      UNRR   = BIT0
2468
2469      ::*****
2470      :* IBUS REG 12
2471      ::*****
2472      000200      IC      = BIT7
2473      000100      IACT   = BIT6
2474      000040      LULP   = BIT5
2475      000020      IRDY   = BIT4
2476      000010      OVRR   = BIT3
2477      000004      RAB    = BIT2
2478      000002      EBLK   = BIT1
2479      000001      BCC    = BIT0
2480
2481      ::*****
2482      :* IBUS REG 13
2483      ::*****
2484      000200      RING   = BIT7
2485      000100      DTR    = BIT6
2486      000040      RTS    = BIT5
2487      000020      HDX    = BIT4
2488      000010      MODR   = BIT3
2489      000004      CS     = BIT2
2490      000002      STBY   = BIT1
2491      000001      CARR   = BIT0
2492
2493      ::*****
2494      :* IBUS REG 14
2495      ::*****
```

2496	000200	READY	=	BIT7
2497	000100	TXEN	=	BIT6
2498	000040	DISSI	=	BIT5
2499	000020	RDAX	=	BIT4
2500	000010	WAX	=	BIT3
2501	000004	ENAX	=	BIT2
2502	000002	AX2	=	BIT1
2503	000001	AX1	=	BIT0
2504				
2505		:*****		
2506		:* IBUS REG 17		
2507		:*****		
2508	000200	SIGR	=	BIT7
2509	000100	SIGO	=	BIT6
2510	000040	TXDATA	=	BIT5
2511	000020	OCOR	=	BIT4
2512	000010	ICIR	=	BIT3
2513	000004	TESTMD	=	BIT2
2514	000002	MCLK	=	BIT1
2515	000001	DDCMP	=	BIT0
2516				
2517		:*****		
2518		:* AX0-15 - USYRT REG 0 (READ ONLY)		
2519		:*****		
2520	000200	RX7	=	BIT7
2521	000100	RX6	=	BIT6
2522	000040	RX5	=	BIT5
2523	000020	RX4	=	BIT4
2524	000010	RX3	=	BIT3
2525	000004	RX2	=	BIT2
2526	000002	RX1	=	BIT1
2527	000001	RX0	=	BIT0
2528				
2529		:*****		
2530		:* AX0-16 - USYRT REG 1 (READ ONLY)		
2531		:*****		
2532	000200	RERR	=	BIT7
2533	000100	ASBC2	=	BIT6
2534	000040	ASBC1	=	BIT5
2535	000020	ASBC0	=	BIT4
2536	000010	ROR	=	BIT3
2537	000004	RABT	=	BIT2
2538	000002	REOM	=	BIT1
2539	000001	RSOM	=	BIT0
2540				
2541		:*****		
2542		:* AX1-15 - USYRT REG 2		
2543		:*****		
2544	000200	TX7	=	BIT7
2545	000100	TX6	=	BIT6
2546	000040	TX5	=	BIT5
2547	000020	TX4	=	BIT4
2548	000010	TX3	=	BIT3
2549	000004	TX2	=	BIT2
2550	000002	TX1	=	BIT1
2551	000001	TX0	=	BIT0

```
2552
2553      :*****
2554      :* AX1-16 - USYRT REG 3
2555      :*****
2556      000200      TERR      = BIT7
2557      000010      TXGA      = BIT3
2558      000004      TXAB      = BIT2
2559      000002      TEOM      = BIT1
2560      000001      TSOM      = BIT0
2561
2562      :*****
2563      :* AX2-15 - USYRT REG 4
2564      :*****
2565      000200      SYN7      = BIT7
2566      000100      SYN6      = BIT6
2567      000040      SYN5      = BIT5
2568      000020      SYN4      = BIT4
2569      000010      SYN3      = BIT3
2570      000004      SYN2      = BIT2
2571      000002      SYN1      = BIT1
2572      000001      SYN0      = BIT0
2573      000226      SYNCH     = 226
2574
2575      :*****
2576      :* AX2-16 - USYRT REG 5
2577      :*****
2578      000200      APA      = BIT7
2579      000100      DDC      = BIT6
2580      000040      STR      = BIT5
2581      000020      SEC      = BIT4
2582      000010      IDL      = BIT3
2583      000004      CRCTY2   = BIT2
2584      000002      CRCTY1   = BIT1
2585      000001      CRCTY0   = BIT0
2586
2587      :*****
2588      :* AX3-15 - USYRT REG 6
2589      :*****
2590      000200      I422     = BIT7
2591      000100      XYZ      = BIT6
2592      000040      C32BCC   = BIT5
2593      000020      V35      = BIT4
2594      000010      INTGRL   = BIT3
2595      000004      C32ENB   = BIT2
2596      000002      OP       = BIT1
2597      000001      TEST     = BIT0
2598      000372      AX315U   = I422!XYZ!C32BCC!V35!INTGRL!OP
2599
2600      :*****
2601      :* AX3-16 - USYRT REG 7
2602      :*****
2603      000200      TXLEN2   = BIT7
2604      000100      TXLEN1   = BIT6
2605      000040      TXLEN0   = BIT5
2606      000004      RXLEN2   = BIT2
2607      000002      RXLEN1   = BIT1
```


2608 000001 RXLENO - BIT0

2609
2610
2611
2612
2613
2614

 : * TX CONTROL BITS DEFINED ON WORD BASIS
 :*****

2615
2616 004000 TXGOA = BIT11
2617 002000 TXABT = BIT10
2618 001000 TXEOM = BIT9
2619 000400 TXSOM = BIT8

2620
2621
2622
2623
2624
2625
2626

 : * RCV CONTROL BITS DEFINED ON WORD BASIS
 :*****

2627
2628
2629 004000 RXOVR = BIT11
2630 002000 RXABT = BIT10
2631 001000 RXEBL = BIT9
2632 000400 RXBCC = BIT8

2633
2634
2635
2636
2637

 : * ADDRESS EQUATES FOR REGISTER STORAGE TABLE (LUREG:)
 :*****

2638
2639
2640 002266 LUR10 = LUREG+0 ;LINE UNIT IBUS REG 10
2641 002270 LUR11 = LUREG+2 ;LINE UNIT IBUS REG 11
2642 002272 LUR12 = LUREG+4 ;LINE UNIT IBUS REG 12
2643 002274 LUR13 = LUREG+6 ;LINE UNIT IBUS REG 13
2644 002276 LUR14 = LUREG+10 ;LINE UNIT IBUS REG 14
2645 002300 LUR15 = LUREG+12 ;LINE UNIT IBUS REG 15
2646 002302 LUR16 = LUREG+14 ;LINE UNIT IBUS REG 16
2647 002304 LUR17 = LUREG+16 ;LINE UNIT IBUS REG 17
2648 002306 AX0.15 = LUREG+20 ;USYRT REG 0
2649 002310 AX0.16 = LUREG+22 ;USYRT REG 1
2650 002312 AX1.15 = LUREG+24 ;USYRT REG 2
2651 002314 AX1.16 = LUREG+26 ;USYRT REG 3
2652 002316 AX2.15 = LUREG+30 ;USYRT REG 4
2653 002320 AX2.16 = LUREG+32 ;USYRT REG 5
2654 002322 AX3.15 = LUREG+34 ;USYRT REG 6
2655 002324 AX3.16 = LUREG+36 ;USYRT REG 7

2656
2657
2658
2659

2660
2661 100000 CHPCHK = BIT15

2662 100000
2663 BCCCHK = BIT15

2664 100000
2665
2666 100000
2667
2668
2669
2670
2671
2672
2673
2674
2675 021000
2676 122000
2677
2678
2679
2680
2681 000001
2682 000002
2683
2684
2685
2686
2687

CRCCHK = BIT15
TCCHK = BIT15

;* MICROINSTRUCTION DEFINITIONS

MVIOX = 021000 ;MOVE IBUS TO OBUS*
MVIXO = 122000 ;MOVE IBUS* TO OBUS

***** ERROR1 BIT FLAG DEFINITIONS *****

RRDYTO = BIT0
WRDYTO = BIT1

```
2689 .SBTTL GLOBAL DATA SECTION
2690
2691 :////////////////////////////////////////////////////////////////////
2692 :/ THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
2693 :/ IN MORE THAN ONE TEST.
2694 :////////////////////////////////////////////////////////////////////
2695
2696 :*****
2697 :* STORAGE FOR DEVICE REGISTERS
2698 :*****
2699 002266 000020 LUREG: .BLKW 16.
2700
2701 :*****
2702 :* MISCELLANEOUS STORAGE
2703 :*****
2704 002326 000000 SCRACH: .WORD 0 ;GEN'L PURPOSE SCRATCH WORD
2705 002330 000000 LOGDEV: .WORD 0 ;LOGICAL DEVICE NUMBER
2706 002332 000000 PSTACK: .WORD 0 ;CONTAINS BASE LEVEL PROGRAM STACK POINTER
2707 002334 000000 PRIOR: .WORD 0 ;CPU PRIORITY FOR PRINTOUT
2708 002336 000000 SUBRPC: .WORD 0 ;PC OF SUBR CALL FOR ERROR REPORTS
2709 002340 000000 INTFLG: .WORD 0 ;INTERRUPT RECEIVED FLAGS
2710 ; BIT 0 FOR TX, BIT 1 FOR RCV
2711 002342 000000 ERRFLG: .WORD 0 ;SUBROUTINE ERROR FLAG
2712 002344 000000 TIMFLG: .WORD 0 ;EVENT TIME-OUT FLAG
2713 002346 000000 RETADR: .WORD 0 ;SUBR ERROR RETURN ADDRESS
2714 002350 000000 REDBYT: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM LU REG
2715 002352 000000 WRIBYT: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO LU REG
2716 002354 000000 RAX15: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM REG 15
2717 002356 000000 RAX16: .WORD 0 ;LO BYTE CONTAINS BYTE READ FROM REG 16
2718 002360 000000 WAX15: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 15
2719 002362 000000 WAX16: .WORD 0 ;LO BYTE CONTAINS BYTE TO LOAD INTO REG 16
2720 002364 000000 REGNUM: .WORD 0 ;NUMBER (10-17) OF LINE UNIT REG BEING TESTED
2721 002366 000000 AXNUM: .WORD 0 ;NUMBER (0-7) OF EXTENDED REG BYTE BEING TESTED
2722 002370 000000 GOODAT: .WORD 0 ;STORAGE FOR EXPECTED DATA
2723 002372 000000 BADDAT: .WORD 0 ;STORAGE FOR ACTUAL DATA
2724 002374 000000 LOADAT: .WORD 0 ;CONTAINS TEST DATA LOADED INTO REG
2725 002376 000000 FRSTIM: .WORD 0 ;FLAG=0 IF PROGRAM JUST LOADED
2726 002400 000000 FRSPAS: .WORD 0 ;FLAG=0 IF FIRST PASS AFTER LOAD
2727 002402 000000 STARES: .WORD 0 ;FLAG=0 IF FIRST TIME THRU AFTER STA OR RES
2728 002404 000000 SAVE4: .WORD 0 ;SAVE LOC 4 HERE (ERROR TRAP VECTOR)
2729 002406 000000 SAVE6: .WORD 0 ;SAVE LOC 6 HERE (ERROR TRAP VECTOR)
2730 002410 000000 ERROR1: .WORD 0 ;SUBR ERROR BIT FLAGS (DEF'D IN GLOBAL EQUATES)
2731 002412 000000 TXWORD: .WORD 0 ;BITS 0-11 CONTAIN DATA TO LOAD INTO TX SILO
2732 002414 000000 RXWORD: .WORD 0 ;BITS 0-11 CONTAIN DATA READ FROM RCV SILO
2733 002416 000000 DISILO: .WORD 0 ;CONTAINS CURRENT STATE OF DISSI IN BIT 5
2734 002420 000000 CHPTYP: .WORD 0 ;USYRT CHIP TYPE, =0 FOR SIG, ELSE =1
2735 002422 000000 MODINT: .WORD 0 ;MODEM INTERFACE SELECTION
2736 002424 000000 SAVLEN: .WORD 0 ;SAVED TX AND RCV CHAR LENGTHS
2737 002426 000000 DEVMAP: .WORD 0 ;BIT MAP OF ACTIVE DEVICES
2738 002430 000000 DEVPTR: .WORD 0 ;DEVICE MAP BIT POINTER
2739 002432 000000 UNIT: .WORD 0 ;CONTAINS UNIT NUMBER (1 TO N)
2740 002434 000000 TSTNUM: .WORD 0 ;CONTAINS TEST NUMBER FOR SOME TESTS
2741
2742 :***** CURRENT DEVICE PARAMETERS *****
2743 002436 160170 MPCR: .WORD 160170 ;POINTER TO MICROPROCESSOR CSR'S
2744 002440 160171 BSEL1: .WORD 160171 ;POINTER TO BSEL1
```

```
2745 002442
2746 002442 160174
2747 002444 160176
2748 002446 000300
2749 002450 000304
2750 002452 000240
2751 002454 000000
2752 002456 000000
2753 002460 000000
2754 002462 000000
2755 002464 000004
2756
2757
2758 002466 000
2759 002467 000
2760 002470 000
2761 002471 000
2762 002472 000
2763 002473 000
2764 002474 000
2765 002475 000
2766
2767
2768 002476 000000
2769 002500 000000
2770 002502 000000
2771 002504 000000
2772 002506 000000
2773 002510 000000
2774 002512 000000
2775 002514 000000
2776
2777
2778 002516 000000
2779 002520 000000
2780 002522 000000
2781 002524 000000
2782 002526 000000
2783 002530 000000
2784 002532 000000
2785 002534 000000
2786
2787
2788 002536
2789 002536 000
2790 002537 056
2791 002540 000
2792 002541 257
2793 002542 100
2794 002543 377
2795 002544 377
2796 002545 306
2797
2798 002546 200
2799
2800
```

```
BSEL4:
SEL4: .WORD 160174 ;POINTER TO SEL4
SEL6: .WORD 160176 ;POINTER TO SEL6
MPIVEC: .WORD 300 ;MICROPROCESSOR INPUT INTERRUPT VECTOR
MPOVEC: .WORD 304 ;MICROPROCESSOR OUTPUT INTERRUPT VECTOR
MPRIOR: .WORD 240 ;MICROPROCESSOR DEVICE PRIORITY
LUSWI1: .WORD 0 ;LINE UNIT SWITCH PACK #1
LUSWI2: .WORD 0 ;LINE UNIT SWITCH PACK #2
LUSWI3: .WORD 0 ;LINE UNIT SWITCH PACK #3
TSTCON: .WORD 0 ;TEST CONNECTOR INDICATOR
BDRATE: .WORD 4 ;BAUD RATE

;***** STORAGE FOR DATA READ IN ADDRESS TESTS *****
REDDAT: .BYTE 0
        .BYTE 0
        .BYTE 0
        .BYTF 0
        .BYTE 0
        .BYTE 0
        .BYTE 0
        .BYTE 0
        .BYTE 0

;:***** GEN'L PURPOSE SCRATCH STORAGE *****
REG0: .WORD 0
REG1: .WORD 0
REG2: .WORD 0
REG3: .WORD 0
REG4: .WORD 0
REG5: .WORD 0
REG6: .WORD 0
REG7: .WORD 0

;:***** SCRATCH STORAGE FOR MESSAGE REPORTING *****
TMP0: .WORD 0
TMP1: .WORD 0
TMP2: .WORD 0
TMP3: .WORD 0
TMP4: .WORD 0
TMP5: .WORD 0
TMP6: .WORD 0
TMP7: .WORD 0

;***** INBUS LU REG BIT MASKS FOR UNPREDICTABLE BITS *****
UPBITS: .BYTE 000 ;MASK FOR REG 10
        .BYTE 056 ;MASK FOR REG 11
        .BYTE 000 ;MASK FOR REG 12
        .BYTE 257 ;MASK FOR REG 13
        .BYTE 100 ;MASK FOR REG 14
        .BYTE 377 ;MASK FOR REG 15
        .BYTE 377 ;MASK FOR REG 16
        .BYTE 306 ;MASK FOR REG 17

R14NRW: .BYTE 200 ;REG 14 NON-R/W BITS

;***** MASKS FOR EXTENDED REGISTER NON-READ/WRITE BITS *****
```

2801	002547		ANBITS:		
2802	002547	377	.BYTE	377	:MASK FOR AX0-15
2803	002550	377	.BYTE	377	:MASK FOR AX0-16
2804	002551	000	.BYTE	000	:MASK FOR AX1-15
2805	002552	360	.BYTE	360	:MASK FOR AX1-16
2806	002553	000	.BYTE	000	:MASK FOR AX2-15
2807	002554	000	.BYTE	000	:MASK FOR AX2-16
2808	002555	004	.BYTE	004	:MASK FOR AX3-15
2809	002556	030	.BYTE	030	:MASK FOR AX3-16

2810					
2811			:***** DATA PATTERN A *****		
2812	002557		PATA:		
2813	002557	125	.BYTE	125	
2814	002560	252	.BYTE	252	
2815	002561	000	.BYTE	000	
2816	002562	377	.BYTE	377	
2817	002563	001	.BYTE	001	
2818	002564	002	.BYTE	002	
2819	002565	004	.BYTE	004	
2820	002566	010	.BYTE	010	
2821	002567	020	.BYTE	020	
2822	002570	040	.BYTE	040	
2823	002571	100	.BYTE	100	
2824	002572	200	.BYTE	200	
2825	002573	376	.BYTE	376	
2826	002574	375	.BYTE	375	
2827	002575	373	.BYTE	373	
2828	002576	367	.BYTE	367	
2829	002577	357	.BYTE	357	
2830	002600	337	.BYTE	337	
2831	002601	277	.BYTE	277	
2832	002602	177	.BYTE	177	

2833					
2834			:***** DATA PATTERN B *****		
2835	002603		PATB:		
2836	002603	000	.BYTE	000	
2837	002604	000	.BYTE	000	
2838	002605	040	.BYTE	040	
2839	002606	100	.BYTE	100	
2840	002607	220	.BYTE	220	
2841	002610	000	.BYTE	000	
2842	002611	000	.BYTE	000	
2843	002612	051	.BYTE	051	

2844					
2845					
2846			:***** DATA PATTERN Q *****		
2847	002613	000	PATQ:	.BYTE	000
2848	002614	120	.BYTE	120	
2849	002615	125	.BYTE	125	
2850	002616	137	.BYTE	137	
2851	002617	040	.BYTE	040	
2852	002620	052	.BYTE	052	
2853	002621	057	.BYTE	057	
2854	002622	177	.BYTE	177	

2855					
2856			:***** DATA PATTERN R *****		

2857	002623	000	PATR:	.BYTE	000
2858	002624	100		.BYTE	100
2859	002625	120		.BYTE	120
2860	002626	124		.BYTE	124
2861	002627	164		.BYTE	164
2862	002630	172		.BYTE	172
2863	002631	176		.BYTE	176
2864	002632	177		.BYTE	177
2865	002633	000		.BYTE	000
2866	002634	100		.BYTE	100
2867	002635	120		.BYTE	120
2868	002636	124		.BYTE	124
2869	002637	164		.BYTE	164
2870	002640	172		.BYTE	172
2871	002641	176		.BYTE	176

2872			:***** DATA PATTERN S *****		
2873			PATS:	.BYTE	000
2874	002642	000		.BYTE	017
2875	002643	017		.BYTE	036
2876	002644	036		.BYTE	074
2877	002645	074		.BYTE	170
2878	002646	170		.BYTE	360
2879	002647	360		.BYTE	037
2880	002650	037		.BYTE	076
2881	002651	076		.BYTE	174
2882	002652	174		.BYTE	370
2883	002653	370		.BYTE	077
2884	002654	077		.BYTE	176
2885	002655	176		.BYTE	374
2886	002656	374		.BYTE	177
2887	002657	177		.BYTE	376
2888	002660	376		.BYTE	377
2889	002661	377			

2890			:***** DATA PATTERN T *****		
2891			PATT:	.BYTE	000
2892	002662	000		.BYTE	125
2893	002663	125		.BYTE	252
2894	002664	252		.BYTE	176
2895	002665	176		.BYTE	177
2896	002666	177			

2897			ENDPAT:		
2898	002667		.EVEN		
2899		002670			

2900			:*** TEST MESSAGES TO BE TRANSMITTED ***		
2901			MSG1:	TXSOM	
2902				TXSOM	
2903				000	
2904				125	
2905				252	
2906				377	
2907	002670	000400			
2908	002672	000400			
2909	002674	000000			
2910	002676	000125			
2911	002700	000252			
2912	002702	000377			

2913	002704	000000	000
2914	002706	001000	TXEOM
2915	002710	001000	TXEOM
2916	002712	001000	TXEOM
2917	002714	001000	TXEOM
2918			
2919	002716	000400	MSG2: TXSOM
2920	002720	000400	TXSOM
2921	002722	000000	000
2922	002724	000377	377
2923	002726	001000	TXEOM
2924	002730	001000	TXEOM
2925			
2926	002732	000001	MSG3: 001
2927	002734	000001	001
2928	002736	000001	001
2929	002740	000001	001
2930	002742	002000	TXABT
2931	002744	000400	TXSOM
2932	002746	000400	TXSOM
2933	002750	000003	003
2934	002752	000003	003
2935	002754	000003	003
2936	002756	000003	003
2937	002760	000003	003
2938			
2939			
2940			
2941			
2942			
2943			
2944	002762	000100	
2945			
2946			
2947			
2948			
2949			
2950			
2951			
2952			
2953			
2954			

*** RECEIVED DATA BUFFER (64. WORDS) ***
RCVBUF: .BLKW 64.

2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046

.SBTTL GLOBAL SUBROUTINES

:/ THE GLOBAL SUBROUTINES ARE CALLED BY MORE THAN ONE TEST
:/

* STPCLK - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR TO
* EXECUTE AN INSTRUCTION WHICH IS PASSED IN THE WORD FOLLOWING THE CALL.

STPCLK:
BISB #ROMO!ROMI,@BSEL1 ;SET ROMO, ROMI BITS IN BSEL1
MOV @ (SP),@SEL6 ;PUT INSTRUCTION INTO SEL6
BISB #ROMO!ROMI!STEPMP,@BSEL1 ;SET ROMO, ROMI, STEPMP IN BSEL1
BICB #ROMO!ROMI!STEPMP,@BSEL1 ;CLEAR ROMO, ROMI, STEPMP IN BSEL1
ADD #2,(SP) ;FIX UP RETURN PC
RTS PC ;RETURN

* MSTCLR - THIS SUBROUTINE ISSUES A MASTER CLEAR AND SETS LULOOP

MSTCLR:
MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;SAVE LU REG. NO.
MOVB #MCLR,@BSEL1 ;SET MASTER CLEAR BIT
BICB #RUN!MCLR,@BSEL1 ;CLEAR RUN AND MCLR BITS
MOV #20.,R1 ;INITIALIZE STALL COUNTER
2\$: NOP ;STALL IN LOOP FOR SEVERAL MICRO-SEC
DEC R1
BNE 2\$
BISB #LULOOP,@BSEL1 ;SET LU LOOP
MOV #13,REGNUM ;SET LU REG NO. = 13
CLR WRIBYT
JSR PC,WRITLU ;CLEAR REG 13
MOV (SP)+,REGNUM ;RESTORE LU REG NO.
MOV (SP)+,R1 ;RESTORE R1
CLR SAVLEN ;CLEAR SAVED CHAR LENGTH FROM SETUP
RTS PC ;RETURN

* READLU - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR
* TO EXECUTE AN INSTRUCTION WHICH READS THE LINE UNIT REG WHOSE
* NUMBER IS PASSED IN REGNUM, INTO REDBYT.

```
3047 003372  
3048 003372 010146  
3049 003374 013701 002364  
3050 003400 006301  
3051 003402 006301  
3052 003404 006301  
3053 003406 006301  
3054 003410 052701 000004  
3055 003414 052701 021000  
3056 003420 010137 003430  
3057 003424 004737 003240  
3058 003430 000000  
3059 003432 117737 177004 002350  
3060 003440 105037 002351  
3061 003444 012601  
3062 003446 000207  
3063  
3064  
3065  
3066  
3067  
3068  
3069  
3070  
3071  
3072  
3073 003450  
3074 003450 010146  
3075 003452 013701 002364  
3076 003456 052701 000100  
3077 003462 052701 122000  
3078 003466 010137 003510  
3079 003472 105037 002353  
3080 003476 113777 002352 176736  
3081 003504 004737 003240  
3082 003510 000000  
3083 003512 012601  
3084 003514 000207  
3085  
3086  
3087  
3088  
3089  
3090  
3091  
3092  
3093  
3094 003516 010146  
3095 003520 013746 002364  
3096 003524 012701 002266  
3097 003530 012737 000010 002364  
3098 003536 004737 003372  
3099 003542 113721 002350  
3100 003546 105021  
3101 003550 005237 002364  
3102 003554 023727 002364 000020
```

```
READLU:  
MOV R1,-(SP) ;SAVE R1  
MOV REGNUM,R1 ;GET LINE UNIT REG NUMBER  
ASL R1 ;SHIFT INTO SOURCE BITS 4-7  
ASL R1  
ASL R1  
ASL R1  
BIS #4,R1 ;SET DESTINATION = BSEL4  
BIS #MVIOX,R1 ;SET REST OF MOVE INSTRUCTION  
MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT  
JSR PC,STPCLK ;EXECUTE MOVE INSTRUCTION  
2$: .WORD 0 ;INSTRUCTION GOES HERE  
MOV B @BSEL4,REDBYT ;GET LU REG CONTENTS INTO REDBYT  
CLRB REDBYT+1 ;CLR HI BYTE OF STORAGE  
MOV (SP)+,R1 ;RESTORE R1  
RTS PC ;RETURN
```

```
*****  
* WRITLU - THIS SUBROUTINE FORCES THE DMC11 OR KMC11 MICROPROCESSOR TO  
* EXECUTE AN INSTRUCTION WHICH LOADS THE BYTE CONTAINED IN WRIBYT  
* INTO THE LU REG WHOSE NUMBER IS PASSED IN REGNUM,  
*****
```

```
WRITLU:  
MOV R1,-(SP) ;SAVE R1  
MOV REGNUM,R1 ;GET LINE UNIT REG NUMBER  
BIS #100,R1 ;SET SOURCE = BSEL4  
BIS #MVIXO,R1 ;SET REST OF MOVE INSTRUCTION  
MOV R1,2$ ;SET INSTRUCTION AS SUBROUTINE ARGUMENT  
CLRB WRIBYT+1 ;CLR HI BYTE OF STORAGE  
MOV B WRIBYT,@BSEL4 ;LOAD BYTE INTO BSEL4  
JSR PC,STPCLK ;EXECUTE MOVE INSTRUCTION  
2$: .WORD 0  
MOV (SP)+,R1 ;RESTORE R1  
RTS PC ;RETURN
```

```
*****  
* GETREG - THIS SUBROUTINE READS THE LINE UNIT REGISTERS 10-17 INTO THE  
* REGISTER STORAGE TABLE (LUREG:).  
*****
```

```
GETREG: MOV R1,-(SP) ;SAVE R1  
MOV REGNUM,-(SP) ;SAVE CURRENT REG NO.  
MOV #LUR10,R1 ;INIT POINTER TO REG STORAGE TABLE  
MOV #10,REGNUM ;INIT LU REG NO. TO 10  
3$: JSR PC,READLU ;READ A LINE UNIT REG  
MOV B REDBYT,(R1)+ ;PUT BYTE READ INTO TABLE  
CLRB (R1)+ ;CLEAR UPPER BYTE OF TABLE ENTRY  
INC REGNUM ;INCREMENT REG NO.  
CMP REGNUM,#20 ;SEE IF ALL REGS READ YET
```

3103 003562 002765
 3104 003564 012637 002364
 3105 003570 012601
 3106 003572 000207

```
BLT 3$ ;BR IF NOT
MOV (SP)+,REGNUM ;RESTORE CURRENT REG NO.
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN
```

3107
 3108
 3109
 3110
 3111

```
*****
* LOOPIN - THIS SUBROUTINE PLACES THE MICROPROCESSOR IN A LOOP ON AN
* INSTRUCTION, BY MOVING THE INSTRUCTION FROM THE WORD FOLLOWING THE CALL
* INTO SEL6, AND SETTING RUN AND ROMI IN BSEL1. THE SUBROUTINE RETURNS
* WITH THE MICROPROCESSOR STUCK IN THE LOOP, AND IF IT IS DESIRED TO
* TERMINATE THE LOOP, THE PDP-11 PROGRAM MUST CLEAR THE RUN BIT IN
* BSEL1, OR CALL SUBROUTINE MSTCLR TO DO THIS.
*****
```

3112
 3113
 3114
 3115
 3116
 3117
 3118
 3119

```
LOOPIN:
BISB #ROMO!ROMI,@BSEL1 ;SET ROMO, ROMI BITS IN BSEL1
MOV @ (SP),@SEL6 ;PUT MICROINSTRUCTION INTO SEL6
BISB #RUN!ROMO!ROMI,@BSEL1 ;SET RUN, ROMO, ROMI IN BSEL1
ADD #2,(SP) ;FIX UP RETURN PC
RTS PC ;RETURN WITH MICROPROCESSOR STUCK IN SINGLE
; INSTRUCTION LOOP
```

3120 003574
 3121 003574 152777 000006 176636
 3122 003602 017677 000000 176634
 3123 003610 152777 000206 176622
 3124 003616 062716 000002
 3125 003622 000207

3126
 3127
 3128
 3129
 3130
 3131

```
*****
* READAX - THIS SUBROUTINE READS THE USYRT REG PAIR WHOSE NUMBER (0-3)
* IS PASSED IN BITS 1,2 OF AXNUM ON ENTRY, AND RETURNS THE BYTES READ IN
* RAX15 AND RAX16. IF THE LINE UNIT DOES NOT RESPOND WITH READY IN REG 14,
* RRDYTO BIT IS SET IN ERROR1 ON RETURN.
*****
```

3132
 3133
 3134
 3135
 3136
 3137

```
READAX: MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;STORE CURRENT REG NO.
BIC #RRDYTO,ERROR1 ;CLEAR ERROR BIT
MOV #14,REGNUM ;SET LU REG NO. = 14
MOVB AXNUM,WRIBYT ;SET UP AX REG NO. BITS
ASR WRIBYT
BISB #RDAX!ENAX,WRIBYT ;SET UP BITS TO LOAD INTO REG 14
BIS DISILO,WRIBYT ;SET PROPER STATE OF DISSI BIT
JSR PC,WRITLU ;SET RDAX AND ENAX IN REG 14
CLR R1 ;INIT TIMER
6$: JSR PC,READLU ;READ REG 14
BITB #READY,REDBYT ;SEE IF READY BIT SET IN REG 14 YET
BNE 9$ ;BR IF READY SET
INC R1 ;INCR TIMER
BNE 6$ ;BR IF TIMER DIDN'T TIME OUT YET
BIS #RRDYTO,ERROR1 ;SET ERROR FLAG FOR TIME OUT ON READ RDY
BR 12$ ;BR TO RETURN
9$: MOV #15,REGNUM ;SET REG NO. = 15
JSR PC,READLU ;READ REG 15
MOVB REDBYT,RAX15 ;STORE REG AX-15
CLRB RAX15+1 ;CLR HI BYTE OF STORAGE
```

3138 003624 010146
 3139 003626 013746 002364
 3140 003632 042737 000001 002410
 3141 003640 012737 000014 002364
 3142 003646 113737 002366 002352
 3143 003654 006237 002352
 3144 003660 152737 000024 002352
 3145 003666 053737 002416 002352
 3146 003674 004737 003450
 3147 003700 005001
 3148 003702 004737 003372
 3149 003706 132737 000200 002350
 3150 003714 001006
 3151 003716 005201
 3152 003720 001370
 3153 003722 052737 000001 002410
 3154 003730 000424
 3155 003732 012737 000015 002364
 3156 003740 004737 003372
 3157 003744 113737 002350 002354
 3158 003752 105037 002355

```
3159 003756 012737 000016 002364      MOV      #16,REGNUM      ;SET REG NO. = 16
3160 003764 004737 003372          JSR      PC,READLU      ;READ REG 16
3161 003770 113737 002350 002356      MOV      REDBYT,RAX16   ;STORE REG AX-16
3162 003776 105037 002357          CLRB    RAX16+1        ;CLR HI BYTE OF STORAGE
3163 004002 012637 002364      12$:    MOV      (SP)+,REGNUM ;RESTORE CURRENT REG NO.
3164 004006 012601          MOV      (SP)+,R1      ;RESTORE R1
3165 004010 000207          RTS      PC            ;RETURN
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176 004012 010146          WRITAX: MOV      R1,-(SP) ;SAVE R1
3177 004014 013746 002364          MOV      REGNUM,-(SP) ;SAVE CURRENT REG NO.
3178 004020 042737 000002 002410      BIC      #WRDYTO,ERROR1 ;CLEAR ERROR BIT
3179 004026 012737 000014 002364      MOV      #14,REGNUM    ;SET LU REG NO. = 14
3180 004034 113737 002366 002352      MOV      AXNUM,WRIBYT ;SET AX REG NO. BITS
3181 004042 006237 002352          ASR      WRIBYT
3182 004046 053737 002416 002352      BIS      DISILO,WRIBYT ;SET PROPER STATE OF DISSI BIT
3183
3184 004054 004737 003450          JSR      PC,WRITLU     ;SET AX NO. BITS IN REG 14
3185 004060 012737 000015 002364      MOV      #15,REGNUM    ;SET REG NO. = 15
3186 004066 105037 002361          CLRB    WAX15+1       ;CLR HI BYTE OF STORAGE
3187 004072 113737 002360 002352      MOV      WAX15,WRIBYT ;SET UP BYTE TO WRITE INTO REG 15
3188 004100 004737 003450          JSR      PC,WRITLU     ;WRITE BYTE INTO REG 15
3189 004104 005237 002364          INC     REGNUM        ;SET REG NO. = 16
3190 004110 105037 002363          CLRB    WAX16+1       ;CLR HI BYTE OF STORAGE
3191 004114 113737 002362 002352      MOV      WAX16,WRIBYT ;SET UP BYTE TO WRITE INTO REG 16
3192 004122 004737 003450          JSR      PC,WRITLU     ;WRITE BYTE INTO REG 16
3193 004126 012737 000014 002364      MOV      #14,REGNUM    ;SET REG NO. = 14
3194 004134 113737 002366 002352      MOV      AXNUM,WRIBYT ;SET AX REG NO. BITS
3195 004142 006237 002352          ASR      WRIBYT
3196 004146 152737 000014 002352      BIS      #ENAX!WAX,WRIBYT ;SET UP BITS TO LOAD INTO REG 14
3197 004154 053737 002416 002352      BIS      DISILO,WRIBYT ;SET PROPER STATE OF DISSI BIT
3198 004162 004737 003450          JSR      PC,WRITLU     ;SET ENAX AND WAX IN REG 14
3199 004166 005001          CLR     R1            ;INIT PROGRAM TIMER
3200 004170 004737 003372 002350      6$:    JSR      PC,READLU     ;READ REG 14
3201 004174 132737 000200          BITB    #READY,REDBYT ;SEE IF READY BIT SET IN REG 14 YET
3202 004202 001005          BNE     9$           ;BR IF READY SET
3203 004204 005201          INC     R1            ;INCR TIMER
3204 004206 001370          BNE     6$           ;BR IF TIMER DIDN'T TIME OUT YET
3205 004210 052737 000002 002410      9$:    BIS      #WRDYTO,ERROR1 ;SET ERROR FLAG BIT FOR TIME OUT ON WRITE RDY
3206 004216 012637 002364          MOV      (SP)+,REGNUM ;RESTORE CURRENT REG NO.
3207 004222 012601          MOV      (SP)+,R1      ;RESTORE R1
3208 004224 000207          RTS      PC            ;RETURN
3209
3210
3211
3212
3213
3214
```

3215
 3216
 3217
 3218 004226 010146
 3219 004230 013746 002366
 3220 004234 012737 015212 002516
 3221 004242 032737 000001 002366
 3222 004250 001403
 3223 004252 012737 015215 002516
 3224 004260 004737 003516
 3225 004264 142777 000010 176146
 3226 004272 012701 002306
 3227 004276 005037 002366
 3228 004302 004737 003624
 3229 004306 113721 002354
 3230 004312 105021
 3231 004314 113721 002356
 3232 004320 105021
 3233 004322 062737 000002 002366
 3234 004330 023727 002366 000010
 3235 004336 002761
 3236 004340 012637 002366
 3237 004344 012601
 3238 004346 013737 002366 002520
 3239 004354 006237 002520
 3240 004360 000207
 3241
 3242
 3243
 3244
 3245
 3246
 3247
 3248
 3249
 3250
 3251
 3252
 3253
 3254 004362 013746 002364
 3255 004366 013746 002336
 3256 004372 005737 002336
 3257 004376 001006
 3258 004400 016637 000004 002336
 3259 004406 162737 000004 002336
 3260 004414 012737 000011 002364
 3261 004422 004737 003372
 3262 004426 032776 000001 000004
 3263 004434 001413
 3264 004436 132737 000020 002350
 3265 004444 001022
 3266 004446 004737 004226
 3267
 3268 004452
 (4) 004452 104455
 (5) 004454 000007

;* GETALL - THIS SUBROUTINE READS THE LINE UNIT REGS 10-17 AND THE EXTENDED
 :* REGISTERS AX0-AX3 INTO REGISTER STORAGE TABLE (LUREG:).
 :*****

```

GETALL: MOV R1,-(SP) ;SAVE R1
        MOV AXNUM,-(SP) ;SAVE CURRENT AX REG BYTE NO.
        MOV #DH5,TMPO ;SET AX LO BYTE NO.
        BIT #BIT0,AXNUM ;SEE IF LO OR HI BYTE
        BEQ 1$ ;BR IF LO BYTE
        MOV #DH6,TMPO ;SET AX HI BYTE NO.
1$: JSR PC,GETREG ;READ AND STORE REGS 10-17
    BICB #LULOOP,@BSEL1 ;CLEAR LULOOP
    MOV #AX0.15,R1 ;INIT POINTER TO REG STORAGE TABLE
    CLR AXNUM ;INIT AX REG BYTE NO. TO 0
3$: JSR PC,READAX ;READ 2 AX REG BYTES
    MOVB RAX15,(R1)+ ;PUT LO BYTE READ INTO TABLE
    CLRB (R1)+ ;CLEAR UPPER BYTE OF TABLE ENTRY
    MOVB RAX16,(R1)+ ;PUT HI BYTE READ INTO TABLE
    CLRB (R1)+ ;CLEAR UPPER BYTE OF TABLE ENTRY
    ADD #2,AXNUM ;INCR AX REG BYTE NO.
    CMP AXNUM,#10 ;SEE IF ALL REGS READ YET
    BLT 3$ ;BR IF NOT
    MOV (SP)+,AXNUM ;RESTORE CURRENT AX REG BYTE NO.
    MOV (SP)+,R1 ;RESTORE R1
    MOV AXNUM,TMP1
    ASR TMP1 ;GET EXTENDED REG NO. FOR PRINTOUT
    RTS PC ;RETURN
  
```

 :* OSIRDY - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF ORDY (REG 11)
 :* AND OCOR (REG 17) AND REPORTS AN ERROR IF EITHER IS NOT PROPERLY SET
 :* AS PASSED IN BIT 0 (ORDY) AND BIT 1 (OCOR) OF THE WORD FOLLOWING THE
 :* CALL.
 :* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST, AT THE ADDRESS IN
 :* RETADR.
 :*****

```

OSIRDY: MOV REGNUM,-(SP) ;SAVE LU REG NO.
        MOV SUBRPC,-(SP)
        TST SUBRPC ;SEE IF THIS IS A NESTED CALL
        BNE 1$ ;BR IF YES
        MOV 4(SP),SUBRPC
        SUB #4,SUBRPC ;GET PC OF SUBROUTINE CALL
1$: MOV #11,REGNUM ;SET REG NO. TO 11
    JSR PC,READLU ;READ REG 11
    BIT #BIT0,@4(SP) ;GET EXPECTED STATE OF ORDY
    BEQ 3$ ;BR IF EXPECTED ORDY = 0
    BITB #ORDY,REDBYT ;SEE IF ORDY = 1
    BNE 9$ ;BR IF ORDY = 1
    JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT ORDY NOT SET
ERRDF 7,EM7,ERR4
  
```

TRAP C\$ERDF
 .WORD 7

```
(5) 004456 013324  
(5) 004460 016376 .WORD EM7  
3269 004462 000451 .WORD ERR4  
3270 004464 132737 000020 002350 3$: BR 16$ ;TAKE ERROR RETURN  
3271 004472 001407 BEQ 9$ ;SEE IF ORDY = 0  
3272 004474 004737 004226 JSR PC,GETALL ;BR IF ORDY = 0  
3273 ;REPORT ORDY NOT CLEARED ;GET REGS FOR PRINTOUT  
3274 004500 ERRDF 8,EM8,ERR4  
(4) 004500 104455 TRAP C$ERDF  
(5) 004502 000010 .WORD 8  
(5) 004504 013341 .WORD EM8  
(5) 004506 016376 .WORD ERR4  
3275 004510 000436 BR 16$ ;TAKE ERROR RETURN  
3276 004512 012737 000017 002364 9$: MOV #17,REGNUM ;SET REG NO. = 17  
3277 004520 004737 003372 JSR PC,READLU ;READ LU REG 17  
3278 004524 132776 000002 000004 BITB #BIT1,24(SP) ;GET EXPECTED STATE OF OCOR  
3279 004532 001413 BEQ 12$ ;BR IF EXPECTED OCOR = 0  
3280 004534 132737 000020 002350 BITB #OCOR,REDBYT ;SEE IF OCOR = 1  
3281 004542 001031 BNE 20$ ;BR IF OCOR = 1  
3282 004544 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT  
3283 ;REPORT OCOR NOT SET  
3284 004550 ERRDF 9,EM9,ERR4  
(4) 004550 104455 TRAP C$ERDF  
(5) 004552 000011 .WORD 9  
(5) 004554 013362 .WORD EM9  
(5) 004556 016376 .WORD ERR4  
3285 004560 000412 BR 16$ ;TAKE ERROR RETURN  
3286 004562 132737 000020 002350 12$: BITB #OCOR,REDBYT ;SEE IF OCOR = 0  
3287 004570 001416 BEQ 20$ ;BR IF OCOR = 0  
3288 004572 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT  
3289 ;REPORT OCOR NOT CLEARED  
3290 004576 ERRDF 10,EM10,ERR4  
(4) 004576 104455 TRAP C$ERDF  
(5) 004600 000012 .WORD 10  
(5) 004602 013377 .WORD EM10  
(5) 004604 016376 .WORD ERR4  
3291 004606 016637 000002 002364 16$: MOV 2(SP),REGNUM ;RESTORE LU REG NO.  
3292 004614 013706 002332 MOV PSTACK,SP ;RESTORE STACK POINTER TO BASE LEVEL  
3293 004620 013746 002346 MOV RETADR,-(SP) ;FIX ERROR RETURN PC  
3294 004624 000407 BR 23$  
3295 004626 062766 000002 000004 20$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC  
3296 004634 012637 002336 MOV (SP)+,SUBRPC  
3297 004640 012637 002364 MOV (SP)+,REGNUM ;RESTORE LU REG NO.  
3298 004644 000207 23$: RTS PC ;RETURN  
3299  
3300  
3301  
3302  
3303  
3304  
3305 ;*****  
3306 ;* WAIT50 - THIS SUBROUTINE STALLS FOR AT LEAST 50 MICRO-SEC, AND THEN RETURNS.  
3307 ;*****  
3307 004646 010146 WAIT50: MOV R1,-(SP) ;SAVE R1  
3308 004650 012701 000310 MOV #200.,R1 ;INIT COUNTER  
3309 004654 005301 3$: DEC R1 ;DECREMENT COUNTER  
3310 004656 001376 BNE 3$ ;BR IF NOT DONE YET
```


3311 004660 012601
3312 004662 000207

MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN

3313
3314
3315
3316
3317

3318

;* STALL - THIS SUBROUTINE STALLS FOR ABOUT A MICRO-SEC.

3319

3320

3321 004664 000240
3322 004666 000240
3323 004670 000240
3324 004672 000207

STALL: NOP
NOP
NOP
RTS PC

3325

3326

3327

3328

3329

3330

3331

3332

;* LDTXSI - THIS SUBROUTINE LOADS THE TX SILO (REGS 10,11) WITH THE DATA PASSED
;* IN BITS 0-11 OF TXWORD.

3333

3334

3335 004674 013746 002364
3336 004700 042737 170000 002412
3337 004706 012737 000011 002364
3338 004714 113737 002413 002352
3339 004722 004737 003450
3340 004726 012737 000010 002364
3341 004734 113737 002412 002352
3342 004742 004737 003450
3343 004746 012637 002364
3344 004752 000207

LDTXSI: MOV REGNUM, -(SP) ;SAVE LU REG NO.
BIC #170000, TXWORD ;CLEAR UNUSED BITS
MOV #11, REGNUM ;SET REG NO. = 11
MOVB TXWORD+1, WRIBYT ;SET DATA TO BE WRITTEN INTO REG 11
JSR PC, WRITLU ;LOAD DATA INTO REG 11
MOV #10, REGNUM ;SET REG NO. = 10
MOVB TXWORD, WRIBYT ;SET DATA TO BE WRITTEN INTO REG 10
JSR PC, WRITLU ;LOAD DATA INTO REG 10
MOV (SP)+, REGNUM ;RESTORE LU REG NO.
RTS PC ;RETURN

3345

3346

3347

3348

3349

3350

3351

3352

3353

3354

3355

;* STPLU - THIS SUBROUTINE CLOCKS THE LINE UNIT FOR THE NO. OF CYCLES PASSED
;* IN BITS 0-14 OF THE WORD FOLLOWING THE CALL.
;* IF BIT 15 = 1, A CHECK IS MADE TO DETERMINE IF THE USYRT CHIP TYPE
;* REQUIRES DECREMENTING THE NO. OF CYCLES BY 1.

3356 004754 010146
3357 004756 017601 000002
3358 004762 001426
3359 004764 100006
3360 004766 042701 100000
3361 004772 005737 002420
3362 004776 001401
3363 005000 005301
3364 005002 152777 000010 175430 2\$
3365 005010 152777 000020 175422 3\$
3366 005016 004737 004664

STPLU: MOV R1, -(SP) ;SAVE R1
MOV @2(SP), R1 ;GET DESIRED NO. OF CYCLES
BEQ 6\$;IF DESIRED CYCLES = 0, RETURN
BPL 2\$;BR IF CHIP TYPE CHECK NOT NECESSARY
BIC #BIT15, R1 ;CLEAR FLAG BIT
TST CHPTYP ;SEE IF SIG USYRT
BEQ 2\$;BR IF YES
DEC R1 ;DECREMENT CYCLE COUNT
BISB #LULOOP, @BSEL1 ;SET LU LOOP BIT
BISB #STEPLU, @BSEL1 ;SET THE STEPLU BIT (CLOCK THE TRANSMITTER)
JSR PC, STALL ;STALL

```

3367 005022 142777 000020 175410      BICB  #STEPLU,@BSEL1 ;CLEAR THE STEPLU BIT (CLOCK THE RECEIVER)
3368 005030 004737 004664      JSR   PC,STALL      ;STALL
3369 005034 005301                DEC   R1             ;DECREMENT CYCLE COUNTER
3370 005036 001364                BNE   3$            ;BR IF NOT DONE YET
3371 005040 062766 000002 000002 6$:  ADD   #2,2(SP)      ;FIX UP RETURN PC
3372 005046 012601                MOV   (SP)+,R1     ;RESTORE R1
3373 005050 000207                RTS   PC            ;RETURN
3374
3375
3376
3377
3378
3379

```

```

:*****
:* OACTIV - THIS SUBROUTINE CHECKS FOR THE PROPER STATE OF OACT (REG 11) AND
:* REPORTS AN ERROR IF IT IS NOT PROPERLY SET TO THE STATE OF BIT 0 IN THE
:* WORD FOLLOWING THE CALL.
:*****

```

```

3384 005052 013746 002364      OACTIV: MOV   REGNUM,-(SP) ;SAVE LU REG NO.
3385 005056 013746 002336      MOV   SUBRPC,-(SP)
3386 005062 005737 002336      TST   SUBRPC        ;SEE IF THIS IS A NESTED CALL
3387 005066 001006                BNE   1$            ;BR IF YES
3388 005070 016637 000004 002336  MOV   4(SP),SUBRPC
3389 005076 162737 000004 002336  SUB   #4,SUBRPC     ;GET PC OF SUBROUTINE CALL
3390 005104 012737 000011 002364  1$:  MOV   #11,REGNUM   ;SET REG NO. = 11
3391 005112 004737 003372      JSR   PC,READLU    ;READ REG 11
3392 005116 032776 000001 000004  BIT   #BIT0,@4(SP) ;GET EXPECTED STATE OF OACT
3393 005124 001413                BEQ   3$            ;BR IF EXPECTED OACT = 0
3394 005126 132737 000100 002350  BITB  #OACT,REDBYT ;SEE IF OACT = 1
3395 005134 001031                BNE   9$            ;BR IF OACT = 1
3396 005136 004737 004226      JSR   PC,GETALL    ;GET REGS FOR PRINTOUT
3397
3398      ;REPORT OACT NOT SET
ERRDF 11,EM11,ERP4

```

```

TRAP  C$ERDF
.WORD 11
.WORD EM11
.WORD ERR4

```

```

(4) 005142 104455
(5) 005144 000013
(5) 005146 013420
(5) 005150 016376
3399 005152 000412                BR    6$            ;TAKE ERROR RETURN
3400 005154 132737 000100 002350 3$:  BITB  #OACT,REDBYT ;SEE IF OACT = 0
3401 005162 001416                BEQ   9$            ;BR IF OACT = 0
3402 005164 004737 004226      JSR   PC,GETALL    ;GET REGS FOR PRINTOUT
3403
3404      ;REPORT OACT NOT CLEARED
ERRDF 12,EM12,ERR4

```

```

TRAP  C$ERDF
.WORD 12
.WORD EM12
.WORD ERR4

```

```

(4) 005170 104455
(5) 005172 000014
(5) 005174 013435
(5) 005176 016376
3405 005200 016637 000002 002364 6$:  MOV   2(SP),REGNUM ;RESTORE LU REG NO.
3406 005206 013706 002332      MOV   PSTACK,SP   ;RESTORE PROGRAM STACK TO BASE LEVEL
3407 005212 013746 002346      MOV   RETADR,-(SP) ;FIX UP ERROR RETURN PC
3408 005216 000407                BR    12$           ;
3409 005220 062766 000002 000004 9$:  ADD   #2,4(SP)     ;FIX UP ERROR-FREE RETURN PC
3410 005226 012637 002336      MOV   (SP)+,SUBRPC ;RESTORE LU REG NO.
3411 005232 012637 002364      MOV   (SP)+,REGNUM ;RESTORE LU REG NO.
3412 005236 000207                RTS   PC            ;RETURN
3413
3414

```

```

3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428 005240 010146
3429 005242 013746 002364
3430 005246 013746 002366
3431 005252 016637 000006 002336
3432 005260 162737 000004 002336
3433 005266 004737 003276
3434 005272 004737 004362
3435 005276 000001
3436 005300 004737 005052
3437 005304 000000
3438 005306 012737 000004 002366
3439 005314 117637 000006 002360
3440 005322 012737 000400 002412
3441 005330 113737 002360 002412
3442 005336 005037 002362
3443 005342 004737 004012
3444 005346 012737 000017 002364
3445 005354 062766 000002 000006
3446 005362 117637 000006 002352
3447 005370 004737 003450
3448 005374 004737 004674
3449 005400 004737 004674
3450 005404 004737 004646
3451 005410 004737 004362
3452 005414 000003
3453 005416 004737 005052
3454 005422 000000
3455 005424 005001
3456 005426 012737 000011 002364
3457 005434 152777 000010 174776 6$:
3458 005442 152777 000020 174770
3459 005450 004737 004664
3460 005454 004737 003372
3461 005460 132737 000100 002350
3462 005466 001014
3463 005470 142777 000020 174742
3464 005476 004737 004664
3465 005502 005201
3466 005504 020127 000003
3467 005510 002751
3468 005512 004737 005052
3469 005516 000001
3470 005520 012737 000017 002364 9$:

```

```

:*****
:* INITRN - THIS SUBROUTINE INITIATES TRANSMISSION OF A MESSAGE, BY DOING A
:* MASTER CLEAR, LOADING AX2-15 AND REG 17 WITH THE DATA PASSED IN THE 2
:* WORDS FOLLOWING THE CALL, LOADING 2 SOM CHARS INTO THE TX SILO, AND
:* CLOCKING THE LINE UNIT UNTIL THE FIRST SYNCH OR FLAG HAS BEEN SERIALIZED
:* IN THE USYRT. THE PROGRAM MONITORS ORDY,OCOR, AND OACT FOR VALID STATES,
:* THROUGHOUT THE PROCESS.
:* IF THE SUBROUTINE DETECTS AN ERROR, A RETURN IS MADE TO THE TEST, AT THE
:* ADDRESS CONTAINED IN RETADR.
:*****
INITRN: MOV R1,-(SP) ;SAVE R1
MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV AXNUM,-(SP) ;SAVE AX BYTE NO.
MOV 6(SP),SUBRPC
SUB #4,SUBRPC ;GET PC OF SUBR CALL
JSR PC,MSTCLR ;ISSUE A MASTER CLEAR
JSR PC,OSIRDY ;CHECK ORDY=1, OCOR=0
1
JSR PC,OACTIV ;CHK OACT=0
0
MOV #4,AXNUM ;SET AX BYTE NO. = 4 FOR AX2
MOVB @6(SP),WAX15 ;SET DATA BYTE TO LOAD INTO AX2-15
MOV #TXSOM,TXWORD ;SET TSOM BIT
MOVB WAX15,TXWORD ;SET SYNCH CHAR
CLR WAX16
JSR PC,WRITAX ;LOAD AX2
MOV #17,REGNUM ;SET REG NO. = 17
ADD #2,6(SP) ;INCR POINTER TO NEXT DATA BYTE
MOVB @6(SP),WRIBYT ;SET DATA BYTE TO LOAD INTO REG 17
JSR PC,WRITLU ;LOAD REG 17
JSR PC,LDTXSI ;LOAD THE SILO WITH SOM CHAR
JSR PC,LDTXSI ;LOAD ANOTHER SOM INTO SILO
JSR PC,WAIT50 ;WAIT FOR DATA TO RIPPLE
JSR PC,OSIRDY ;CHK ORDY=1, OCOR=1
3
JSR PC,OACTIV ;CHK FOR OACT = 0
0
CLR R1 ;INIT CYCLE COUNTER
MOV #11,REGNUM ;SET LU REG NO. = 11
6$: BISB #LULOOP,@BSEL1 ;SET LINE UNIT LOOP BIT
BISB #STEPLU,@BSEL1 ;SET CLOCK BIT
JSR PC,STALL ;STALL FOR MICRO-SEC
JSR PC,READLU ;READ REG 11
BITB #OACT,REDBYT ;SEE IF OACT = 1 YET
BNE 9$ ;BR IF OACT = 1
BICB #STEPLU,@BSEL1 ;CLEAR CLOCK BIT
JSR PC,STALL ;STALL FOR A MICRO-SEC
INC R1 ;INCR CYCLE COUNT
CMP R1,#3 ;SEE IF 3 CYCLES DONE YET
BLT 6$ ;BR IF NOT
JSR PC,OACTIV ;CHK FOR OACT = 1
1
9$: MOV #17,REGNUM ;SET REG NO. = 17

```

3471	005526	005037	002420		CLR	CHPTYP	:CLEAR USYRT CHIP INDICATOR
3472	005532	004737	003372		JSR	PC,READLU	:READ REG 17
3473	005536	132737	000020	002350	BITB	#OCOR,REDBYT	:CHK FOR OCOR CLEARED YET
3474	005544	001403			BEQ	12\$:BR IF YES - IT IS SIG CHIP
3475	005546	012737	000001	002420	MOV	#1,CHPTYP	:SET INDICATOR FOR OTHER CHIP TYPE
3476	005554	142777	000020	174656	BICB	#STEPLU,@BSEL1	:CLEAR CLOCK BIT
3477	005562	004737	004664		JSR	PC,STALL	:STALL FOR MICRO-SEC
3478	005566	004737	004362		JSR	PC,OSIRDY	:CHK FOR ORDY = 1, OCOR = 0
3479	005572	000001			1		
3480	005574	062766	000002	000006	ADD	#2,6(SP)	:FIX UP RETURN PC
3481	005602	012637	002366		MOV	(SP)+,AXNUM	:RESTORE AX BYTE NO.
3482	005606	012637	002364		MOV	(SP)+,REGNUM	:RESTORE LU REG NO.
3483	005612	012601			MOV	(SP)+,R1	:RESTORE R1
3484	005614	005037	002336		CLR	SUBRPC	:CLEAR SUBR CALL PC
3485	005620	000207			RTS	PC	:RETURN

3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499

```

:*****
:* TXCHAR - THIS SUBROUTINE INITIATES TRANSMISSION OF A CHARACTER, BY LOADING
:* THE TX SILO WITH DATA PASSED IN BITS 0-11 OF THE WORD FOLLOWING THE CALL
:* AND CLOCKS THE LINE UNIT WITH THE NUMBER OF CYCLES PASSED IN BITS 0-14
:* OF THE SECOND WORD FOLLOWING THE CALL. IF BIT 15 = 1, A CHK IS MADE TO
:* DETERMINE IF THE USYRT CHIP TYPE REQUIRES DECREMENTING THE NO. OF CYCLES
:* BY 1. THE PROGRAM CHECKS FOR VALID STATES OF ORDY,
:* OCOR, AND OACT THROUGHOUT THE PROCESS.
:* IF AN ERROR IS DETECTED, A RETURN IS MADE TO THE TEST, AT THE ADDRESS
:* CONTAINED IN RETADR.
:*****

```

3501							
3502	005622	010146			TXCHAR: MOV	R1,-(SP)	:SAVE R1
3503	005624	010246			MOV	R2,-(SP)	:SAVE R2
3504	005626	016637	000004	002336	MOV	4(SP),SUBRPC	
3505	005634	162737	000004	002336	SUB	#4,SUBRPC	:GET PC OF SUBR CALL
3506	005642	017637	000004	002412	MOV	@4(SP),TXWORD	:GET DATA TO BE TRANSMITTED
3507	005650	004737	004674		JSR	PC,LDTXSI	:LOAD THE TX SILO WITH THE DATA
3508	005654	004737	004646		JSR	PC,WAIT50	:WAIT FOR DATA TO RIPPLE DOWN SILO
3509	005660	062766	000002	000004	ADD	#2,4(SP)	:INCR POINTER
3510	005666	005001			CLR	R1	:INIT CYCLE COUNT
3511	005670	017602	000004		MOV	@4(SP),R2	:GET DESIRED NO. OF CYCLES
3512	005674	005702			TST	R2	:SEE IF CHIP TYPE CHK SHOULD BE MADE
3513	005676	100006			BPL	9\$:BR IF NOT
3514	005700	042702	100000		BIC	#BIT15,R2	:CLEAR FLAG BIT
3515	005704	005737	002420		TST	CHPTYP	:SEE IF SIG USYRT
3516	005710	001401			BEQ	9\$:BR IF YES
3517	005712	005302			DEC	R2	:DECREMENT NO. OF CYCLES
3518	005714	004737	005052		9\$: JSR	PC,OACTIV	:CHK OACT = 1
3519	005720	000001			1		
3520	005722	020102			CMP	R1,R2	:SEE IF REQUIRED CYCLES DONE YET
3521	005724	001410			BEQ	12\$:BR IF YES
3522	005726	004737	004362		JSR	PC,OSIRDY	:CHK ORDY=1, OCOR=1
3523	005732	000003			3		
3524	005734	004737	004754		JSR	PC,STPLU	:STEP LU ONE CYCLE
3525	005740	000001			1		
3526	005742	005201			INC	R1	:INCR CYCLE COUNT

```

3527 005744 000763
3528 005746 004737 004362
3529 005752 000001
3530 005754 062766 000002 000004
3531 005762 005037 002336
3532 005766 012602
3533 005770 012601
3534 005772 000207
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548 005774 013746 002364
3549 006000 013746 002336
3550 006004 005737 002336
3551 006010 001006
3552 006012 016637 000004 002336
3553 006020 162737 000004 002336
3554 006026 012737 000012 002364
3555 006034 004737 003372
3556 006040 032776 000002 000004
3557 006046 001413
3558 006050 132737 000020 002350
3559 006056 001022
3560 006060 004737 004226
3561
3562 006064
(4) 006064 104455
(5) 006066 000021
(5) 006070 013456
(5) 006072 016376
3563 006074 000451
3564 006076 132737 000020 002350
3565 006104 001407
3566 006106 004737 004226
3567
3568 006112
(4) 006112 104455
(5) 006114 000022
(5) 006116 013473
(5) 006120 016376
3569 006122 000436
3570 006124 012737 000017 002364
3571 006132 004737 003372
3572 006136 132776 000001 000004
3573 006144 001413
3574 006146 132737 000010 002350

12$: BR 9$
JSR PC,OSIRDY ;CHK ORDY=1, OCOR=0
1
ADD #2,4(SP) ;FIX UP RETURN PC
CLR SUBRPC ;CLEAR SUBR CALL PC
MOV (SP)+,R2 ;RESTORE R2
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN

:*****
:* ISIRDY - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF ICIR (REG 17)
:* AND IRDY (REG 12) AND REPORTS AN ERROR IF EITHER IS NOT PROPERLY SET
:* AS PASSED IN BIT 0 (ICIR) AND BIT 1 (IRDY) OF THE WORD FOLLOWING THE
:* CALL.
:* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS
:* IN RETADR.
:*****
ISIRDY: MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV SUBRPC,-(SP)
TST SUBRPC ;SEE IF THIS IS A NESTED CALL
BNE 1$ ;BR IF YES
MOV 4(SP),SUBRPC
SUB #4,SUBRPC ;GET PC OF SUBR CALL
1$: MOV #12,REGNUM ;SET REG NO. TO 12
JSR PC,READLU ;READ REG 12
BIT #BIT1,24(SP) ;GET EXPECTED STATE OF IRDY
BEQ 3$ ;BR IF EXPECTED IRDY = 0
BITB #IRDY,REDBYT ;SEE IF IRDY = 1
BNE 9$ ;BR IF IRDY = 1
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT IRDY NOT SET
ERRDF 17,EM17,ERR4

TRAP C$ERDF
.WORD 17
.WORD EM17
.WORD ERR4

3$: BR 16$ ;TAKE ERROR EXIT
BITB #IRDY,REDBYT ;SEE IF IRDY = 0
BEQ 9$ ;BR IF IRDY = 0
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT IRDY NOT CLEARED
ERRDF 18,EM18,ERR4

TRAP C$ERDF
.WORD 18
.WORD EM18
.WORD ERR4

9$: BR 16$ ;TAKE ERROR RETURN
MOV #17,REGNUM ;SET REG NO. = 17
JSR PC,READLU ;READ REG 17
BITB #BIT0,24(SP) ;GET EXPECTED STATE OF ICIR
BEQ 12$ ;BR IF EXPECTED ICIR = 0
BITB #ICIR,REDBYT ;SEE IF ICIR = 1

```

```
3575 006154 001031  
3576 006156 004737 004226  
3577  
3578 006162  
(4) 006162 104455  
(5) 006164 000023  
(5) 006166 013514  
(5) 006170 016376  
3579 006172 000412  
3580 006174 132737 000010 002350 12$:  
3581 006202 001416  
3582 006204 004737 004226  
3583  
3584 006210  
(4) 006210 104455  
(5) 006212 000024  
(5) 006214 013531  
(5) 006216 016376  
3585 006220 016637 000002 002364 16$:  
3586 006226 013706 002332  
3587 006232 013746 002346  
3588 006236 000407  
3589 006240 062766 000002 000004 20$:  
3590 006246 012637 002336  
3591 006252 012637 002364  
3592 006256 000207 23$:  
3593  
3594  
3595  
3596  
3597  
3598  
3599  
3600  
3601  
3602  
3603  
3604  
3605 006260 013746 002364  
3606 006264 013746 002336  
3607 006270 005737 002336  
3608 006274 001006  
3609 006276 016637 000004 002336  
3610 006304 162737 000004 002336  
3611 006312 012737 000012 002364 1$:  
3612 006320 004737 003372  
3613 006324 032776 000001 000004  
3614 006332 001413  
3615 006334 132737 000100 002350  
3616 006342 001031  
3617 006344 004737 004226  
3618  
3619 006350  
(4) 006350 104455  
(5) 006352 000025  
(5) 006354 013552
```

```
      BNE      20$          ;BR IF ICIR = 1  
      JSR     PC,GETALL    ;GET REGS FOR PRINTOUT  
;REPORT ICIR NOT SET  
      ERRDF  19,EM19,ERR4  
  
      TRAP   C$ERDF  
      .WORD  19  
      .WORD  EM19  
      .WORD  ERR4  
  
      BR      16$          ;TAKE ERROR RETURN  
      BITB   #ICIR,REDBYT ;SEE IF ICIR = 0  
      BEQ    20$          ;BR IF ICIR = 0  
      JSR     PC,GETALL    ;GET REGS FOR PRINTOUT  
;REPORT ICIR NOT CLEARED  
      ERRDF  20,EM20,ERR4  
  
      TRAP   C$ERDF  
      .WORD  20  
      .WORD  EM20  
      .WORD  ERR4  
  
      MOV     2(SP),REGNUM ;RESTORE LU REG NO.  
      MOV     PSTACK,SP   ;RESTORE STACK POINTER TO BASE LEVEL  
      MOV     RETADR,-(SP) ;FIX ERROR RETURN PC  
      BR      23$  
      ADD     #2,4(SP)     ;FIX UP ERROR-FREE RETURN PC  
      MOV     (SP)+,SUBRPC  
      MOV     (SP)+,REGNUM ;RESTORE LU REG NO.  
      RTS     PC          ;RETURN  
  
;*****  
;* IACTIV - THIS SUBROUTINE CHECKS FOR THE PROPER STATE OF IACT (REG 12) AND  
;* REPORTS AN ERROR IF IT IS NOT PROPERLY SET TO THE STATE OF BIT 0 IN THE  
;* WORD FOLLOWING THE CALL.  
;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS IN  
;* RETADR.  
;*****  
IACTIV: MOV     REGNUM,-(SP) ;SAVE LU REG NO.  
        MOV     SUBRPC,-(SP)  
        TST     SUBRPC      ;SEE IF THIS IS A NESTED CALL  
        BNE    1$          ;BR IF YES  
        MOV     4(SP),SUBRPC  
        SUB     #4,SUBRPC    ;GET PC OF SUBR CALL  
        MOV     #12,REGNUM  ;SET REG NO. = 12  
        JSR     PC,READLU   ;READ REG 12  
        BIT     #BIT0,24(SP) ;GET EXPECTED STATE OF IACT  
        BEQ    3$          ;BR IF EXPECTED IACT = 0  
        BITB   #IACT,REDBYT ;SEE IF IACT = 1  
        BNE    9$          ;BR IF IACT = 1  
        JSR     PC,GETALL    ;GET REGS FOR PRINTOUT  
;REPORT IACT NOT SET  
        ERRDF  21,EM21,ERR4  
  
      TRAP   C$ERDF  
      .WORD  21  
      .WORD  EM21
```

```

(5) 006356 016376                                     .WORD  ERR4
3620 006360 000412
3621 006362 132737 000100 002350 3$:  BR      6$      ;TAKE ERROR EXIT
;SEE IF IACT = 0
3622 006370 001416                                     ;BR IF IACT = 0
3623 006372 004737 004226                                     ;GET REGS FOR PRINTOUT
3624                                     ;REPORT IACT NOT CLEARED
3625 006376                                     ERRDF  22,EM22,ERR4
(4) 006376 104455                                     TRAP  C$ERDF
(5) 006400 000026                                     .WORD  22
(5) 006402 013567                                     .WORD  EM22
(5) 006404 016376                                     .WORD  ERR4
3626 006406 016637 000002 002364 6$:  MOV      2(SP),REGNUM ;RESTORE LU REG NO.
3627 006414 013706 002332 ;RESTORE PROGRAM STACK TO BASE LEVEL
3628 006420 013746 002346 ;FIX UP ERROR RETURN PC
3629 006424 000407
3630 006426 062766 000002 000004 9$:  ADD      #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
3631 006434 012637 002336 ;RESTORE LU REG NO.
3632 006440 012637 002364 ;RETURN
3633 006444 000207 12$:  MOV      (SP)+,SUBRPC
3634                                     MOV      (SP)+,REGNUM
3635                                     RTS      PC
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645 006446 013746 002366
3646 006452 013746 002336
3647 006456 005737 002336
3648 006462 001006
3649 006464 016637 000004 002336
3650 006472 162737 000004 002336
3651 006500 012737 000001 002366 1$:  MOV      #4,SUBRPC ;GET PC OF SUBR CALL
3652 006506 004737 003624 ;SET AX BYTE NO. FOR AX0-16
3653 006512 032776 000001 000004 ;READ AX0
3654 006520 001413 ;GET EXPECTED STATE OF RSOM
3655 006522 132737 000001 002356 ;BR IF EXPECTED RSOM = 0
3656 006530 001022 ;SEE IF RSOM = 1
3657 006532 004737 004226 ;BR IF RSOM = 1
3658 ;REPORT RSOM NOT SET
3659 006536                                     ;GET REGS FOR PRINTOUT
(4) 006536 104455                                     ERRDF  29,EM29,ERR6
(5) 006540 000035                                     TRAP  C$ERDF
(5) 006542 013631                                     .WORD  29
(5) 006544 017566                                     .WORD  EM29
3660 006546 000444                                     .WORD  ERR6
3661 006550 132737 000001 002356 3$:  BR      16$     ;TAKE ERROR EXIT
3662 006556 001407 ;SEE IF RSOM = 0
3663 006560 004737 004226 ;BR IF RSOM = 0
3664 ;REPORT RSOM NOT CLEARED
3665 006564                                     JSR      PC,GETALL ;GET REGS FOR PRINTOUT
(4) 006564 104455                                     ERRDF  28,EM28,ERR6
TRAP  C$ERDF

```

```

*****
;* RSEOM - THIS SUBROUTINE CHECKS FOR THE PROPER STATES OF RSOM AND REOM IN
;* AX0-16, AND REPORTS AN ERROR IF EITHER IS NOT SET TO THE STATE PASSED IN BITS
;* 0,1, RESPECTIVELY, OF THE WORD FOLLOWING THE CALL.
;* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST AT THE ADDRESS IN RETADR.
*****
RSEOM:

```

```

(5) 006566 000034
(5) 006570 013610
(5) 006572 017566
3666 006574 000431
3667 006576 132776 000002 000004 9$: BITB #BIT1,24(SP) ;TAKE ERROR RETURN
3668 006604 001413 BEQ 12$ ;GET EXPECTED STATE OF REOM
3669 006606 132737 000002 002356 BITB #REOM,RAX16 ;BR IF EXPECTED REOM = 0
3670 006614 001031 BNE 20$ ;SEE IF REOM = 1
3671 006616 004737 004226 JSR PC,GETALL ;BR IF REOM = 1
3672 ;REPORT REOM NOT SET ;GET REGS FOR PRINTOUT
3673 006622 ERRDF 31,EM31,ERR6
(4) 006622 104455 TRAP C$ERDF
(5) 006624 000037 .WORD 31
(5) 006626 013667 .WORD EM31
(5) 006630 017566 .WORD ERR6
3674 006632 000412
3675 006634 132737 000002 002356 12$: BR 16$ ;TAKE ERROR RETURN
3676 006642 001416 BEQ 20$ ;SEE IF REOM = 0
3677 006644 004737 004226 JSR PC,GETALL ;BR IF REOM = 0
3678 ;REPORT REOM NOT CLEARED ;GET REGS FOR PRINTOUT
3679 006650 ERRDF 30,EM30,ERR6
(4) 006650 104455 TRAP C$ERDF
(5) 006652 000036 .WORD 30
(5) 006654 013646 .WORD EM30
(5) 006656 017566 .WORD ERR6
3680 006660 016637 000002 002366 16$: MOV 2(SP),AXNUM ;RESTORE AX BYTE NO.
3681 006666 013706 002332 MOV PSTACK,SP ;RESTORE STACK POINTER TO BASE LEVEL
3682 006672 013746 002346 MOV RETADR,-(SP) ;FIX ERROR RETURN PC
3683 006676 000407 BR 23$
3684 006700 062766 000002 000004 20$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
3685 006706 012637 002336 MOV (SP)+,SUBRPC
3686 006712 012637 002366 MOV (SP)+,AXNUM ;RESTORE AX BYTE NO.
3687 006716 000207 23$: RTS PC ;RETURN
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697 006720 013746 002364
3698 006724 012737 000012 002364
3699 006732 004737 003372
3700 006736 113737 002350 002415
3701 006744 042737 170000 002414
3702 006752 012737 000010 002364
3703 006760 004737 003372
3704 006764 113737 002350 002414
3705 006772 012637 002364
3706 006776 000207
3707
3708
3709
3710

```

 * RDRXSI - THIS SUBROUTINE READS THE RCV SILO (REGS 10,12) AND RETURNS THE
 * SILO ENTRY IN BITS 0-11 OF RXWORD.

```

RDRXSI: MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV #12,REGNUM ;SET REG NO. = 12
JSR PC,READLU ;READ LU REG 12
MOVB REDBYT,RXWORD+1 ;GET HI BITS OF SILO ENTRY
BIC #170000,RXWORD ;CLEAR UNUSED BITS
MOV #10,REGNUM ;SET REG NO. = 10
JSR PC,READLU ;READ REG 10
MOVB REDBYT,RXWORD ;GET LOW BITS OF SILO ENTRY
MOV (SP)+,REGNUM ;RESTORE LU REG NO.
RTS PC ;RETURN

```


3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766

007000	010146		
007002	010346		
007004	013746	002364	
007010	016637	000006	002336
007016	162737	000004	002336
007024	012737	000012	002364
007032	005001		
007034	017603	000006	
007040	062703	000003	
007044	005776	000006	
007050	001414		
007052	004737	006260	
007056	000000		
007060	004737	005774	
007064	000001		
007066	004737	006446	
007072	000000		
007074	004737	004754	6\$:
007100	000001		
007102	004737	004646	8\$:
007106	005201		
007110	004737	003372	
007114	132737	000020	002350
007122	001005		
007124	020103		
007126	002762		
007130	004737	005774	
007134	000003		
007136	020176	000006	9\$:
007142	002003		
007144	004737	005774	
007150	000001		
007152	004737	006260	12\$:
007156	000001		
007160	004737	005774	
007164	000003		
007166	062766	000002	000006
007174	012637	002364	
007200	012603		
007202	012601		
007204	005037	002336	
007210	000207		

```

*****
* RCV1ST - THIS SUBROUTINE RECEIVES THE FIRST CHAR OF A MESSAGE, AND MONITORS
* STATUS OF THE RECEIVER. FIRST, A CHECK IS MADE FOR IACT = 0, IRDY = 0,
* ICIR = 1, AND RSOM = 0. THEN, THE LINE UNIT IS CLOCKED USING
* STEPLU UNTIL IRDY = 1. THE PROGRAM CHECKS FOR THIS TO OCCUR WITHIN 3
* CYCLES AFTER THE NO. OF CYCLES PASSED IN THE WORD FOLLOWING THE CALL.
* IF AN ERROR OCCURS, A RETURN IS MADE TO THE TEST, AT THE ADDRESS
* CONTAINED IN RETADR.
*****
RCV1ST: MOV R1,-(SP) ;SAVE R1
MOV R3,-(SP) ;SAVE R3
MOV REGNUM,-(SP) ;SAVE LU REG NO.
MOV 6(SP),SUBRPC
SUB #4,SUBRPC ;GET PC OF SUBROUTINE CALL
MOV #12,REGNUM ;SET LU REG NO. = 12
CLR R1 ;INIT CYCLE COUNT TO 0
MOV @6(SP),R3 ;GET CYCLE COUNT LIMIT
ADD #3,R3
TST @6(SP) ;SEE IF DESIRED CYCLES = 0
BEQ 8$ ;BR IF YES
JSR PC,IACTIV ;CHK FOR IACT = 0
0
JSR PC,ISIRDY ;CHK FOR ICIR = 1, IRDY = 0
1
JSR PC,RSEOM ;CHK RSOM = 0, REOM = 0 IN AX0-16
0
6$: JSR PC,STPLU ;CLOCK LU FOR 1 CYCLE
1
8$: JSR PC,WAIT50 ;ALLOW SILO DATA TO RIPPLE
INC R1 ;INCREMENT CYCLE COUNT
JSR PC,READLU ;READ REG 12
BITB #IRDY,REDBYT ;SEE IF IRDY = 1 YET
BNE 9$ ;BR IF IRDY = 1
CMP R1,R3 ;SEE IF LIMIT EXCEEDED
BLT 6$ ;BR IF NOT YET
JSR PC,ISIRDY ;CHK FOR ICIR = 1, IRDY = 1
3
9$: CMP R1,@6(SP) ;SEE IF LESS THAN REQUIRED CYCLES
BGE 12$ ;BR IF NOT
JSR PC,ISIRDY ;CHK FOR ICIR = 1, IRDY = 0
1
12$: JSR PC,IACTIV ;CHK FOR IACT = 1
1
JSR PC,ISIRDY ;CHK FOR ICIR = 1, IRDY = 1
3
ADD #2,6(SP) ;FIX UP RETURN PC
MOV (SP)+,REGNUM ;RESTORE LU REG NO.
MOV (SP)+,R3 ;RESTORE R3
MOV (SP)+,R1 ;RESTORE R1
CLR SUBRPC ;CLEAR SUBR CALL PC
RTS PC ;RETURN

```

```

3767
3768
3769
3770
3771
3772
3773
3774 007212 013746 002364
3775 007216 012737 000017 002364
3776 007224 017637 000002 002352
3777 007232 152737 000002 002352
3778 007240 004737 003450
3779 007244 062766 000002 000002
3780 007252 017637 000002 007264
3781 007260 004737 004754
3782 007264 000000
3783 007266 142737 000002 002352
3784 007274 004737 003450
3785 007300 062766 000002 000002
3786 007306 012637 002364
3787 007312 000207
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802 007314 010146
3803 007316 013746 002364
3804 007322 016637 000004 002336
3805 007330 162737 000004 002336
3806 007336 017601 000004
3807 007342 042701 170000
3808 007346 004737 006720
3809 007352 023727 002424 000347
3810 007360 001005
3811 007362 042701 000200
3812 007366 042737 000200 002414
3813 007374 120137 002414
3814 007400 001445
3815 007402 005037 002370
3816 007406 110137 002370
3817 007412 005037 002372
3818 007416 113737 002414 002372
3819 007424 012737 000011 002364
3820 007432 004737 003372
3821 007436 132737 000001 002350
3822 007444 001410
  
```

```

*****
* STPERR - THIS SUBROUTINE LOADS THE CONTENTS OF THE FIRST WORD FOLLOWING THE
* CALL INTO REG 17, AND SETS THE IERR BIT, AND CLOCKS THE LINE UNIT
* FOR THE NO. OF CYCLES PASSED IN THE 2ND WORD FOLLOWING THE CALL. THEN,
* IT RESTORES REG 17 TO ITS ORIGINAL CONTENTS, CLEARING THE IERR BIT.
*****
  
```

```

STPERR: MOV     REGNUM, -(SP)      ;SAVE LU REG NO.
        MOV     #17, REGNUM      ;SET LU REG NO. = 17
        MOV     @2(SP), WRIBYT
        BISB    #IERR, WRIBYT
        JSR     PC, WRITLU       ;SET IERR BIT IN REG 17
        ADD     #2, 2(SP)        ;INCREMENT SUBR ARGUMENT POINTER
        MOV     @2(SP), 3$      ;GET DESIRED NO. OF CYCLES
        JSR     PC, STPLU       ;CLOCK LU FOR DESIRED NO. OF CYCLES
3$:     .WORD    0               ;NO. OF CYCLES GOES HERE
        BICB    #IERR, WRIBYT
        JSR     PC, WRITLU       ;CLEAR IERR BIT IN REG 17
        ADD     #2, 2(SP)        ;FIX UP RETURN PC
        MOV     (SP)+, REGNUM    ;RESTORE LU REG NO.
        RTS     PC              ;RETURN
  
```

```

*****
* CKDATA - THIS SUBROUTINE READS THE RCV SILO AND COMPARES THE SILO ENTRY
* TO BITS 0-11 OF THE FIRST WORD FOLLOWING THE CALL. IF THERE IS A
* MISMATCH, THE ERROR IS REPORTED AND A RETURN IS MADE TO THE TEST AT THE
* ADDRESS CONTAINED IN RETADR. IF BIT 15 = 0 IN THE FIRST WORD
* FOLLOWING THE CALL, THE SUBROUTINE WILL NOT CHECK THE BCC BIT (SILO
* BIT 8). IF THERE ARE NO ERRORS, THE LINE UNIT IS CLOCKED FOR THE
* NUMBER OF CYCLES PASSED IN THE SECOND WORD FOLLOWING THE CALL.
*****
  
```

```

CKDATA: MOV     R1, -(SP)        ;SAVE R1
        MOV     REGNUM, -(SP)    ;SAVE LU REG NO.
        MOV     4(SP), SUBRPC
        SUB     #4, SUBRPC       ;GET PC OF SUBR CALL
        MOV     @4(SP), R1       ;GET EXPECTED SILO ENTRY
        BIC     #170000, R1      ;CLEAR UNUSED BITS FOR COMPARE
        JSR     PC, RDRXSI      ;READ RCV SILO
        CMP     SAVLEN, #TXLEN2!TXLEN1!TXLEN0!RXLEN2!RXLEN1!RXLEN0
        BNE     4$              ;BR IF CHAR LENGTH NOT = 7
        BIC     #BIT7, R1        ;MASK OFF 8TH BIT
        BIC     #BIT7, RXWORD
        CMPB    R1, RXWORD       ;COMPARE EXPECTED BITS 0-7 TO ACTUAL
        BEQ     6$              ;BR IF MATCH
        CLR     GOODAT
        MOVB    R1, GOODAT       ;GET EXPECTED DATA
        CLR     BADDAT
        MOVB    RXWORD, BADDAT   ;GET ACTUAL DATA
        MOV     #11, REGNUM      ;SET REG NO. = 11
        JSR     PC, READLU      ;READ REG 11
        BITB    #UNRR, REDBYT    ;SEE IF TX UNDERRUN ERROR
        BEQ     5$              ;BR IF NOT, TO REPORT DATA ERROR
  
```

3823	007446	004737	004226						
3824									
3825	007452								
(4)	007452	104455						TRAP	C\$ERDF
(5)	007454	000066						.WORD	54
(5)	007456	014561						.WORD	EM54
(5)	007460	016376						.WORD	ERR4
3826	007462	000137	010144						
3827	007466	012737	000010	002364	5\$:	JMP	36\$;TAKE ERROR EXIT
3828	007474	004737	004226			MOV	#10,REGNUM		;SET REG NO. = 10
3829						JSR	PC,GETALL		;GET REGS FOR PRINTOUT
3830	007500					;REPORT	PCV'D DATA MISCOMPARE		
(4)	007500	104455				ERRDF	34,EM34,ERR8		
(5)	007502	000042						TRAP	C\$ERDF
(5)	007504	013704						.WORD	34
(5)	007506	020676						.WORD	EM34
3831	007510	000137	010144					.WORD	ERR8
3832	007514	000301				JMP	36\$;TAKE ERROR EXIT
3833	007516	012737	000012	002364	6\$:	SWAB	R1		
3834	007524	120137	002415			MOV	#12,REGNUM		;SET LU REG NO. FOR ERROR REPORTS
3835	007530	001002				CMPB	R1,RXWORD+1		;COMPARE EXPECTED SILO BITS 8-11 TO ACTUAL
3836	007532	000137	010120			BNE	7\$;BR IF MISMATCH
3837	007536	005037	002370			JMP	22\$;CONTINUE
3838	007542	110137	002370			CLR	GOODAT		
3839	007546	005037	002372			MOVB	R1,GOODAT		;SET EXPECTED DATA
3840	007552	113737	002415	002372		CLR	BADDAT		
3841	007560	032776	100000	000004		MOVB	RXWORD+1,BADDAT		;SET ACTUAL DATA
3842	007566	001433				BIT	#BCCCHK,@4(SP)		;SEE IF BCC SHOULD BE IGNORED
3843	007570	132701	000001			BEQ	10\$;BR IF YES
3844	007574	001014				BITB	#BCC,R1		;SEE IF EXPECTED BIT = 1
3845	007576	132737	000001	002415		BNE	8\$;BR IF YES
3846	007604	001424				BITB	#BCC,RXWORD+1		;SEE IF ACTUAL BIT = 0
3847	007606	004737	004226			BEQ	10\$;BR IF YES
3848						JSR	PC,GETALL		;GET REGS FOR PRINTOUT
3849	007612					;REPORT	BCC NOT CLEARED		
(4)	007612	104455				ERRDF	35,EM35,ERR8		
(5)	007614	000043						TRAP	C\$ERDF
(5)	007616	013732						.WORD	35
(5)	007620	020676						.WORD	EM35
3850	007622	000137	010144					.WORD	ERR8
3851	007626	132737	000001	002415	8\$:	JMP	36\$;TAKE ERROR EXIT
3852	007634	001010				BITB	#BCC,RXWORD+1		;SEE IF ACTUAL BIT = 1
3853	007636	004737	004226			BNE	10\$;BR IF YES
3854						JSR	PC,GETALL		;GET REGS FOR PRINTOUT
3855	007642					;REPORT	BCC NOT SET		
(4)	007642	104455				ERRDF	36,EM36,ERR8		
(5)	007644	000044						TRAP	C\$ERDF
(5)	007646	013752						.WORD	36
(5)	007650	020676						.WORD	EM36
3856	007652	000137	010144					.WORD	ERR8
3857	007656					JMP	36\$;TAKE ERROR EXIT
3858	007656	132701	000002			BITB	#EBLK,R1		;SEE IF EXPECTED BIT = 1
3859	007662	001014				BNE	12\$;BR IF YES
3860	007664	132737	000002	002415		BITB	#EBLK,RXWORD+1		;SEE IF ACTUAL BIT = 0
3861	007672	001424				BEQ	14\$;BR IF YES
3862	007674	004737	004226			JSR	PC,GETALL		;GET REGS FOR PRINTOUT

```
3863 ;REPORT EBLK NOT CLEARED
3864 007700 ERRDF 37,EM37,ERR8
(4) 007700 104455
(5) 007702 000045 TRAP C$ERDF
(5) 007704 013766 .WORD 37
(5) 007706 020676 .WORD EM37
3865 007710 000137 010144 JMP 36$ ;TAKE ERROR EXIT
3866 007714 132737 000002 002415 12$: BITB #EBLK,RXWORD+1 ;SEE IF ACTUAL BIT = 1
3867 007722 001010 BNE 14$ ;BR IF YES
3868 007724 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
3869 ;REPORT EBLK NOT SET
3870 007730 ERRDF 38,EM38,ERR8
(4) 007730 104455 TRAP C$ERDF
(5) 007732 000046 .WORD 38
(5) 007734 014007 .WORD EM38
(5) 007736 020676 .WORD ERR8
3871 007740 000137 010144 JMP 36$ ;TAKE ERROR EXIT
3872 007744 14$: BITB #RAB,R1 ;SEE IF EXPECTED BIT = 1
3873 007744 132701 000004 BNE 16$ ;BR IF YES
3874 007750 001014 BITB #RAB,RXWORD+1 ;SEE IF ACTUAL BIT = 0
3875 007752 132737 000004 002415 BEQ 18$ ;BR IF YES
3876 007760 001424 JSR PC,GETALL ;GET REGS FOR PRINTOUT
3877 007762 004737 004226 ;REPORT RAB NOT SET
3878 ERRDF 39,EM39,ERR8
3879 007766
(4) 007766 104455 TRAP C$ERDF
(5) 007770 000047 .WORD 39
(5) 007772 014024 .WORD EM39
(5) 007774 020676 .WORD ERR8
3880 007776 000137 010144 JMP 36$ ;TAKE ERROR EXIT
3881 010002 132737 000004 002415 16$: BITB #RAB,RXWORD+1 ;SEE IF ACTUAL BIT = 1
3882 010010 001010 BNE 18$ ;BR IF YES
3883 010012 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
3884 ;REPORT RAB NOT SET
3885 010016 ERRDF 40,EM40,ERR8
(4) 010016 104455 TRAP C$ERDF
(5) 010020 000050 .WORD 40
(5) 010022 014044 .WORD EM40
(5) 010024 020676 .WORD ERR8
3886 010026 000137 010144 JMP 36$ ;TAKE ERROR EXIT
3887 010032 18$: BITB #OVRR,R1 ;SEE IF EXPECTED BIT = 1
3888 010032 132701 000010 BNE 20$ ;BR IF YES
3889 010036 001014 BITB #OVRR,RXWORD+1 ;SEE IF ACTUAL BIT = 0
3890 010040 132737 000010 002415 BEQ 22$ ;BR IF YES
3891 010046 001424 JSR PC,GETALL ;GET REGS FOR PRINTOUT
3892 010050 004737 004226 ;REPORT OVRR NOT CLEARED
3893 ERRDF 41,EM41,ERR8
3894 010054
(4) 010054 104455 TRAP C$ERDF
(5) 010056 000051 .WORD 41
(5) 010060 014060 .WORD EM41
(5) 010062 020676 .WORD ERR8
3895 010064 000137 010144 JMP 36$ ;TAKE ERROR EXIT
3896 010070 132737 000010 002415 20$: BITB #OVRR,RXWORD+1 ;SEE IF ACTUAL BIT = 1
3897 010076 001010 BNE 22$ ;BR IF YES
3898 010100 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
```

```

3899 ;REPORT OVRR NOT SET
3900 (4) 010104 104455 ERRDF 42,EM42,ERR8
(5) 010106 000052 TRAP C$ERDF
(5) 010110 014101 .WORD 42
(5) 010112 020676 .WORD EM42
3901 010114 000137 010144 .WORD ERR8
3902 010120
3903 010120 062766 000002 000004 22$: JMP 36$ ;TAKE ERROR EXIT
3904 010126 017637 000004 010140 ADD #2,4(SP) ;INCR SUBROUTINE ARGUMENT POINTER
3905 010134 004737 004754 MOV @4(SP),24$ ;GET DESIRED CYCLE COUNT
3906 010140 000000 24$: JSR PC,STPLU ;CLOCK LU FOR DESIRED CYCLES
3907 010142 000407 .WORD 0
3908 010144 011637 002364 36$: BR 38$ ;TAKE ERROR-FREE EXIT
3909 010150 013706 002332 MOV (SP),REGNUM ;RESTORE LU REG NO.
3910 010154 013746 002346 MOV PSTACK,SP ;RESTORE PROGRAM STACK TO BASE LEVEL
3911 010160 000406 BR RETADR,-(SP) ;FIX UP ERROR RETURN PC
3912 010162 062766 000002 000004 38$: ADD #2,4(SP) ;FIX UP ERROR-FREE RETURN PC
3913 010170 012637 002364 MOV (SP)+,REGNUM ;RESTORE LU REG NO.
3914 010174 012601 MOV (SP)+,R1 ;RESTORE R1
3915 010176 005037 002336 40$: CLR SUBRPC ;CLEAR SUBROUTINE PC
3916 010202 000207 RTS PC ;RETURN
3917
3918
3919
3920
3921
3922
3923 ;*****
3924 ;* LODATA - THIS SUBROUTINE LOADS THE TRANSMITTER SILO WITH 5 SOM'S, THE DATA
3925 ;* IN PATTERN A REPEATED 2 TIMES (40 CHARS), AND 2 EOM'S. IN ADDITION, THE
3926 ;* DATA CHARS ARE ALSO LOADED INTO THE RECEIVED MSG BUFFER (RCVBUF:), AS
3927 ;* EXPECTED DATA FOR LATER COMPARISON.
3928 ;*****
3928 010204 010146 LODATA: MOV R1,-(SP) ;SAVE R1
3929 010206 010346 MOV R3,-(SP) ;SAVE R3
3930 010210 010446 MOV R4,-(SP) ;SAVE R4
3931 010212 004737 010746 JSR PC,LODSIL ;LOAD 5 SOM'S INTO TX SILO
3932 010216 000400 TXSOM
3933 010220 000005 5
3934 010222 012701 000002 MOV #2,R1 ;INIT COUNTER
3935 010226 012704 002762 MOV #RCVBUF,R4 ;GET POINTER TO RCV BUF
3936 010232 012703 002557 3$: MOV #PATA,R3 ;GET POINTER TO DATA PATTERN
3937 010236 005037 002412 6$: CLR TXWORD
3938 010242 112337 002412 MOV (R3)+,TXWORD ;GET A DATA CHAR
3939 010246 013724 002412 MOV TXWORD,(R4)+ ;LOAD A DATA CHAR INTO RCV BUF
3940 010252 004737 004674 JSR PC,LDTXSI ;LOAD DATA CHAR INTO TX SILO
3941 010256 020327 002603 CMP R3,#PATB ;SEE IF AT END OF PATTERN A YET
3942 010262 103765 BLO 6$ ;BR IF NOT YET
3943 010264 005301 DEC R1 ;DECREMENT COUNTER
3944 010266 001361 BNE 3$ ;BR IF NOT DONE YET
3945 010270 052764 100400 177776 BIS #CRCCHK!RXBCC,-2(R4) ;SET UP TO CHK BCC = 1 ON LAST DATA CHAR
3946 010276 012737 001000 002412 MOV #TXEOM,TXWORD
3947 010304 004737 004674 JSR PC,LDTXSI ;LOAD AN EOM INTO TX SILO
3948 010310 004737 004674 JSR PC,LDTXSI ;LOAD ANOTHER EOM
3949 010314 012604 MOV (SP)+,R4 ;RESTORE R4
3950 010316 012603 MOV (SP)+,R3 ;RESTORE R3

```

3951	010320	012601		
3952	010322	000207		
3953				
3954				
3955				
3956				
3957				
3958				
3959				
3960				
3961				
3962				
3963	010324	013746	002366	
3964	010330	013746	002364	
3965	010334	012737	000004	002366
3966	010342	017637	000004	002360
3967	010350	005037	002362	
3968	010354	004737	004012	
3969	010360	012737	000017	002364
3970	010366	062766	000002	000004
3971	010374	017637	000004	002352
3972	010402	004737	003450	
3973	010406	012737	000006	002366
3974	010414	062766	000002	000004
3975	010422	017637	000004	002360
3976	010430	062766	000002	000004
3977	010436	017637	000004	002362
3978	010444	013737	002362	002424
3979	010452	142777	000010	171760
3980	010460	004737	004012	
3981	010464	152777	000010	171746
3982	010472	062766	000002	000004
3983	010500	012637	002364	
3984	010504	012637	002366	
3985	010510	005037	002336	
3986	010514	000207		
3987				
3988				
3989				
3990				
3991				
3992				
3993				
3994				
3995				
3996				
3997	010516	010146		
3998	010520	010246		
3999	010522	017601	000004	
4000	010526	062766	000002	000004
4001	010534	017602	000004	
4002	010540	062766	000002	000004
4003	010546	012137	002412	
4004	010552	004737	004674	
4005	010556	005302		
4006	010560	001372		

```

MOV      (SP)+,R1      :RESTORE R1
RTS      PC            :RETURN

:*****
:* SETUP - THIS SUBROUTINE LOADS THE FIRST WORD AFTER THE CALL INTO AX2-15
:* (SYNCH CHAR), LOADS THE SECOND WORD AFTER THE CALL INTO REG 17
:* LOADS THE THIRD WORD INTO AX3-15, AND LOADS THE FOURTH WORD INTO AX3-16.
:*****
SETUP:  MOV      AXNUM,-(SP)      :SAVE AX BYTE NO.
        MOV      REGNUM,-(SP)    :SAVE LU REG NO.
        MOV      #4,AXNUM        :SET AX BYTE NO. FOR AX2
        MOV      @4(SP),WAX15
        CLR      WAX16
        JSR      PC,WRITAX      :SET SYNCH CHAR IN AX2-15, CLEAR AX2-16
        MOV      #17,REGNUM      :SET LU REG NO. = 17
        ADD      #2,4(SP)        :INCREMENT ARGUMENT POINTER
        MOV      @4(SP),WRIBYT
        JSR      PC,WRITLU      :LOAD REG 17
        MOV      #6,AXNUM        :SET AX BYTE NO. FOR AX3
        ADD      #2,4(SP)        :INCREMENT ARGUMENT POINTER
        MOV      @4(SP),WAX15
        ADD      #2,4(SP)        :INCREMENT ARGUMENT POINTER
        MOV      @4(SP),WAX16
        MOV      WAX16,SAVLEN    :STORE TX AND RCV CHAR LENGTHS
        BICB    #LULOOP,@BSEL1  :CLEAR LULOOP
        JSR      PC,WRITAX      :LOAD AX3-15, AX3-16
        BISB    #LULOOP,@BSEL1  :SET LULOOP
        ADD      #2,4(SP)        :FIX RETURN PC
        MOV      (SP)+,REGNUM    :RESTORE LU REG NO.
        MOV      (SP)+,AXNUM     :RESTORE AX BYTE NO.
        CLR      SUBRPC         :CLEAR SUBROUTINE PC STORAGE
        RTS      PC            :RETURN

:*****
:* LODMSG - THIS SUBROUTINE LOADS THE NO. OF WORDS PASSED IN THE SECOND WORD
:* FOLLOWING THE CALL FROM THE MSG BUFFER WHOSE ADDRESS IS IN THE FIRST
:* WORD FOLLOWING THE CALL, INTO THE TRANSMITTER SILO.
:*****
LODMSG: MOV      R1,-(SP)        :SAVE R1
        MOV      R2,-(SP)        :SAVE R2
        MOV      @4(SP),R1       :GET MSG POINTER INTO R1
        ADD      #2,4(SP)        :INCR ARG POINTER
        MOV      @4(SP),R2       :GET WORD COUNT INTO R2
        ADD      #2,4(SP)        :FIX UP RETURN PC
6$:     MOV      (R1)+,TXWORD     :GET NEXT MSG WORD
        JSR      PC,LDIXSI      :LOAD A WORD INTO TX SILO
        DEC      R2             :DECR COUNT
        BNE     6$             :BR IF NOT DONE YET

```

4007 010562 004737 004646
4008 010566 012602
4009 010570 012601
4010 010572 000207

JSR PC, WAIT50 ;WAIT FOR SILO TO RIPPLE
MOV (SP)+, R2 ;RESTORE R2
MOV (SP)+, R1 ;RESTORE R1
RTS PC ;RETURN

4011
4012
4013
4014
4015

4016
4017
4018
4019
4020
4021

* LDDBYTS - THIS SUBROUTINE LOADS THE NO. OF BYTES PASSED IN THE SECOND WORD
* FOLLOWING THE CALL FROM THE MSG BUFFER WHOSE ADDRESS IS IN THE FIRST
* WORD FOLLOWING THE CALL, INTO THE LOW BYTE OF THE TX SILO. FOR EACH
* BYTE LOADED, A 0 IS LOADED INTO THE HI 4 BITS OF THE TX SILO.

4022 010574 010146
4023 010576 010246
4024 010600 017601 000004
4025 010604 062766 000002 000004
4026 010612 017602 000004
4027 010616 062766 000002 000004
4028 010624 112137 002412
4029 010630 105037 002413
4030 010634 004737 004674
4031 010640 005302
4032 010642 001370
4033 010644 004737 004646
4034 010650 012602
4035 010652 012601
4036 010654 000207

LDDBYTS: MOV R1, -(SP) ;SAVE R1
MOV R2, -(SP) ;SAVE R2
MOV @4(SP), R1 ;GET DATA POINTER INTO R1
ADD #2, 4(SP) ;INCR ARGUMENT POINTER
MOV @4(SP), R2 ;GET BYTE COUNT INTO R2
ADD #2, 4(SP) ;FIX UP RETURN PC
6\$: MOV (R1)+, TXWORD ;GET NEXT DATA BYTE
CLRB TXWORD+1 ;CLEAR HI BYTE
JSR PC, LDTXSI ;LOAD A SILO ENTRY
DEC R2 ;DECR BYTE COUNT
BNE 6\$;BR IF NOT DONE YET
JSR PC, WAIT50 ;WAIT FOR SILO TO RIPPLE
MOV (SP)+, R2 ;RESTORE R2
MOV (SP)+, R1 ;RESTORE R1
RTS PC ;RETURN

4037
4038
4039
4040
4041

4042
4043
4044
4045
4046

* LDMSG1 - THIS SUBROUTINE LOADS THE TRANSMITTER SILO WITH MSG1, AND LOADS
* THE DATA CHARS INTO THE RCV MSG BUFFER (RCVBUF:), AS EXPECTED DATA
* FOR LATER COMPARISON.

4047 010656 010146
4048 010660 010246
4049 010662 004737 010746
4050 010666 000400
4051 010670 000003
4052 010672 004737 010516
4053 010676 002670
4054 010700 000010
4055 010702 012701 002674
4056 010706 012702 002762
4057 010712 012122
4058 010714 020127 002706
4059 010720 103774
4060 010722 052762 100400 177776
4061 010730 012722 000160
4062 010734 012722 000034

LDMSG1: MOV R1, -(SP) ;SAVE R1
MOV R2, -(SP) ;SAVE R2
JSR PC, LODSIL ;LOAD 3 SOM'S INTO TX SILO
TXSOM 3
JSR PC, LODMSG ;LOAD MSG1 INTO TX SILO (WITH 2 SOM'S, 1 EOM)
MSG1 8.
MOV #MSG1+4, R1 ;GET POINTER TO MSG1
MOV #RCVBUF, R2 ;GET POINTER TO MSG BUF
3\$: MOV (R1)+, (R2)+ ;LOAD A CHAR INTO MSG BUF
CMP R1, #MSG1+14. ;SEE IF DID LAST DATA CHAR YET
BLO 3\$;BR IF NOT
BIS #CRCCHK!RXBCC, -2(R2) ;SET EXPECTED BCC
MOV #160, (R2)+ ;LOAD HI CRC BYTE
MOV #034, (R2)+ ;LOAD LO CRC BYTE

4063 010740 012602
4064 010742 012601
4065 010744 000207
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076 010746 010146
4077 010750 017637 000002 002412
4078 010756 062766 000002 000002
4079 010764 017601 000002
4080 010770 004737 004674
4081 010774 005301
4082 010776 001374
4083 011000 004737 004646
4084 011004 062766 000002 000002
4085 011012 012601
4086 011014 000207
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4110 011016 013746 002364
4111 011022 013746 002366
4112 011026 010246
4113 011030 016637 000006 002336
4114 011036 162737 000004 002336
4115 011044 012737 000333 002422
4116 011052 032776 100000 000006
4117 011060 001405
4118 011062 005737 002462

```
MOV (SP)+,R2 ;RESTORE R2
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN

:*****
;* LODSIL - THIS SUBROUTINE REPEATEDLY LOADS THE DATA PASSED IN THE FIRST WORD
;* FOLLOWING THE CALL INTO THE TX SILO, FOR THE NO. OF TIMES PASSED IN THE
;* SECOND WORD FOLLOWING THE CALL.
:*****
LODSIL: MOV R1,-(SP) ;SAVE R1
MOV @2(SP),TXWORD ;GET DATA
ADD #2,2(SP) ;INCR ARGUMENT POINTER
MOV @2(SP),R1 ;GET COUNT
3$: JSR PC,LDTXSI ;LOAD WORD INTO TX SILO
DEC R1 ;DECR COUNT
BNE 3$ ;BR IF NOT ALL LOADED YET
JSR PC,WAIT50 ;ALLOW SILO DATA TO RIPPLE
ADD #2,2(SP) ;FIX UP RETURN PC
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN
```

```
:*****
;* CKLPBK - THIS SUBROUTINE DETERMINES IF THE TEST CALLING IT CAN BE RUN. THE
;* TEST PASSES THE DESIRED MODEM INTERFACE TYPE IN THE WORD FOLLOWING THE
;* CALL, AND IF THE PROPER EXTERNAL LOOPBACK HAS BEEN PROVIDED BY THE
;* OPERATOR FOR THAT INTERFACE, AND IF THE BAUD RATE IS CORRECT, A NORMAL
;* RETURN IS MADE TO RUN THE TEST. IF NOT, A RETURN IS MADE TO THE TEST,
;* AT THE ADDRESS IN RETADR (RETADR CONTAINS THE TEST EXIT ADDRESS, SO
;* THE TEST GETS SKIPPED).
;* IF BIT 15 IS SET IN THE WORD FOLLOWING THE CALL, THE TEST WILL NOT
;* BE RUN UNLESS THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED. IF THE
;* TEST IS TO BE RUN, THE SUBROUTINE RETURNS THE MODEM SELECT BITS FOR
;* AX3-15 IN MODINT. IF NECESSARY, THE SUBROUTINE WILL SET MAINT1, MAINT2,
;* OR DTR.
;* IF THE PROGRAM PASSES '0' IN THE WORD FOLLOWING THE CALL, THE SUBROUTINE
;* WILL ATTEMPT TO RUN WHICHEVER MODEM INTERFACE IS SELECTED BY CABLE,
;* SWITCH, OR TEST CONNECTOR. IF SUCCESSFUL, THE SELECTED INTERFACE WILL
;* BE PASSED BACK TO THE TEST IN MODINT.
:*****
CKLPBK: MOV REGNUM,-(SP) ;SAVE REG NO.
MOV AXNUM,-(SP) ;SAVE AX BYTE NO.
MOV R2,-(SP) ;SAVE R2
MOV 6(SP),SUBRPC
SUB #4,SUBRPC ;GET PC OF SUBR CALL
MOV #I422!XYZ!V35!INTGRL!OP!TEST,MODINT ;INIT MODEM SELECT BITS
BIT #TCCHK,@6(SP) ;SEE IF H3254,5 CHECK IS DESIRED
BEQ 1$ ;BR IF NOT
TST TSTCON ;SEE IF H3254,5 INSTALLED
```



```

4119 011066 001074          BNE      7$          ;BR IF NOT, TO SKIP TEST
4120 011070 000137 012122  JMP      32$         ;BR TO RUN TEST
4121          ;IF NO EXTERNAL LPBK, SKIP TEST
4122 011074 023727 002462 000004 1$:  CMP      TSTCON,#4    ;SEE IF NO LPBK
4123 011102 001466          BEQ      7$          ;BR IF NO LPBK, TO SKIP TEST
4124 011104 142777 000010 171326  BICB    #LULOOK,#SEL1 ;CLEAR LULOOK
4125 011112 012737 000006 002366  MOV      #6,AXNUM    ;SET AX BYTE NO. FOR AX3-15
4126 011120 004737 003624          JSR      PC,READAX   ;READ AX3-15
4127
4128          ;*** SEE IF AN INTERFACE IS REQUESTED ***
4129
4130 011124 027627 000006 000010          CMP      @6(SP),#INTGRL ;SEE IF INTEGRAL MODEM REQUESTED
4131 011132 001422          BEQ      4$          ;BR IF INTGRL MODEM REQUESTED
4132 011134 027627 000006 000020          CMP      @6(SP),#V35    ;SEE IF V.35 REQUESTED
4133 011142 001512          BEQ      10$         ;BR IF V.35 REQUESTED
4134 011144 027627 000006 000200          CMP      @6(SP),#1422  ;SEE IF 422 REQUESTED
4135 011152 001002          BNE      2$          ;BR IF 422 NOT REQUESTED
4136 011154 000137 011454          JMP      14$         ;422 REQUESTED
4137 011160 027627 000006 000100 2$:  CMP      @6(SP),#XYZ   ;SEE IF XYZ REQUESTED
4138 011166 001002          BNE      3$          ;BR IF XYZ NOT REQUESTED
4139 011170 000137 011510          JMP      17$         ;XYZ REQUESTED
4140 011174 000137 011602          3$:  JMP      21$         ;NONE REQUESTED, FIND AN INTERFACE TO TEST
4141
4142          ;SEE IF INTEGRAL MODEM CAN BE RUN
4143 011200 005737 002462          4$:  TST      TSTCON      ;SEE IF H3254 AND H3255 USED
4144 011204 001050          BNE      8$          ;BR IF NOT
4145 011206 023727 002464 000004 5$:  CMP      BDRATE,#4    ;SEE IF BAUD RATE > OR = 56K
4146 011214 002405          BLT      6$          ;BR IF BAUD RATE TOO SLOW FOR INTGRL MODM
4147 011216 042737 000010 002422  BIC      #INTGRL,MODINT ;ASSERT INTEGRAL MODEM
4148 011224 000137 012122          JMP      32$         ;GO TO RUN TEST
4149          ;PRINT 'BAUD RATE INCORRECT'
4150          6$:
4151 011230 023727 002402 000001          CMP      STARES,#1    ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
4152 011236 001026          BNE      40$        ;BR IF NOT, TO SKIP PRINTING
4153 011240          PRINTF  #FMT25
(7) 011240 012746 013155          MOV      #FMT25,-(SP)
(6) 011244 012746 000001          MOV      #1,-(SP)
(3) 011250 010600          MOV      SP,R0
(4) 011252 104417          TRAP    C$PNTF
(4) 011254 062706 000004          ADD     #4,SP
4154          ;PRINT 'TEST XX NOT RUN'
4155          7$:
4156 011260 023727 002402 000001          CMP      STARES,#1    ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
4157 011266 001012          BNE      40$        ;BR IF NOT, TO SKIP PRINTING
4158 011270          PRINTF  #FMT19,TSTNUM
(8) 011270 013746 002434          MOV      TSTNUM,-(SP)
(7) 011274 012746 013124          MOV      #FMT19,-(SP)
(6) 011300 012746 000002          MOV      #2,-(SP)
(3) 011304 010600          MOV      SP,R0
(4) 011306 104417          TRAP    C$PNTF
(4) 011310 062706 000006          ADD     #6,SP
4159 011314 013766 002346 000006 40$: MOV      RETADR,6(SP) ;SET TEST EXIT ADDRESS FOR ERRORS
4160 011322 000137 012130          JMP      33$         ;GO TO SKIP TEST
4161 011326 032737 000010 002354 8$:  BIT      #INTGRL,RAX15 ;SEE IF INTEGRAL MODEM IS SELECTED
4162 011334 001324          BNE      5$          ;BR IF YES, TO CHECK BAUD RATE
4163          ;PRINT 'MODEM INTERFACE NOT SELECTED'

```

```

4164 011336
4165 011336 023727 002402 000001 9$:      CMP      STARES,#1      ;SEE IF THIS IS FIRST PASS SINCE STA OR RES
4166 011344 001363      BNE      40$           ;BR IF NOT, TO SKIP PRINTING
4167 011346      PRINTF  #FMT26
      (7) 011346 012746 013205
      (6) 011352 012746 000001      MOV      #FMT26,-(SP)
      (3) 011356 010600      MOV      #1,-(SP)
      (4) 011360 104417      MOV      SP,R0
      (4) 011362 062706 000004      TRAP    (SPNTF)
4168 011366 000734      BR       7$           ;GO TO PRINT 'TEST NOT RUN'
4169
4170      ;SEE IF V.35 CAN BE RUN
4171 011370 032737 000200 002354 10$:     BIT      #I422,RAX15   ;SEE IF 422 IS SELECTED
4172 011376 001357      BNE      9$           ;BR IF YES, TO SKIP TEST
4173 011400 005737 002462      TST     TSTCON        ;SEE IF H3254 AND H3255 USED
4174 011404 001010      BNE      12$          ;BR IF H3254 AND H3255 NOT USED
4175 011406 012737 000004 002352 11$:     MOV      #MAINT2,WRIBYT ;SET MAINT2 FOR MANUFACTURING TEST CONN.
4176 011414 042737 000020 002422      BIC     #V35,MODINT   ;ASSERT V.35
4177 011422 000137 011704      JMP     42$           ;GO SET DTR AND RUN THE TEST
4178 011426 032737 000020 002354 12$:     BIT      #V35,RAX15   ;SEE IF V.35 IS SELECTED
4179 011434 001405      BEQ     13$          ;BR IF NOT
4180 011436 042737 000020 002422      BIC     #V35,MODINT   ;ASSERT V.35
4181 011444 000137 011700      JMP     27$          ;GO SET DTR AND RUN THE TEST
4182 011450 000137 011336 13$:     JMP     9$           ;WRONG INTRFCE, GO SKIP TEST
4183
4184
4185      ;SEE IF 422 CAN BE RUN
4186 011454 005737 002462 14$:     TST     TSTCON        ;SEE IF H3254 AND H3255 USED
4187 011460 001005      BNE      16$          ;BR IF NOT
4188 011462 042737 000200 002422 15$:     BIC     #I422,MODINT   ;ASSERT 422
4189 011470 000137 011700      JMP     27$          ;GO TO RUN TEST
4190 011474 032737 000200 002354 16$:     BIT      #I422,RAX15   ;SEE IF 422 IS SELECTED
4191 011502 001367      BNE      15$          ;IF YES, GO ASSERT 422 AND RUN TEST
4192 011504 000137 011336      JMP     9$           ;WRONG INTRFCE, GO SKIP TEST
4193
4194      ;SEE IF XYZ CAN BE RUN
4195 011510 032737 000200 002354 17$:     BIT      #I422,RAX15   ;SEE IF 422 IS SELECTED
4196 011516 001402      BEQ     18$          ;BR IF NOT
4197 011520 000137 011336      JMP     9$           ;WRONG INTRFCE, GO SKIP TEST
4198 011524 032737 000100 002354 18$:     BIT      #XYZ,RAX15   ;SEE IF XYZ IS SELECTED
4199 011532 001002      BNE      19$          ;BR IF YES
4200 011534 000137 011336      JMP     9$           ;WRONG INTRFCE, GO SKIP TEST
4201 011540 023727 002464 000004 19$:     CMP     BDRATE,#4     ;SEE IF BAUD RATE < OR = 56K
4202 011546 003402      BLE     20$          ;BR IF YES
4203 011550 000137 011230      JMP     6$           ;BAUD RATE TOO FAST FOR XYZ
4204 011554 042737 000100 002422 20$:     BIC     #XYZ,MODINT   ;ASSERT XYZ
4205 011562 005737 002462      TST     TSTCON        ;SEE IF H3254,5 BEING USED
4206 011566 001044      BNE      27$          ;BR IF NOT
4207 011570 012737 000004 002352      MOV     #MAINT2,WRIBYT
4208 011576 000137 011704      JMP     42$          ;GO SET MAINT2 FOR MANUFACTURING TEST CONN.
4209
4210      ;*** NO INTERFACE REQUESTED - FIND ONE TO TEST ***
4211
4212 011602 032737 000010 002354 21$:     BIT      #INTGRL,RAX15 ;SEE IF INTEGRAL MODEM SELECTED
4213 011610 001402      BEQ     22$          ;BR IF NOT
4214 011612 000137 011206      JMP     5$           ;SEE IF INTEGRAL MODEM CAN BE RUN

```

```

4215 011616 032737 000020 002354 22$: BIT #V35,RAX15 ;SEE IF V.35 SELECTED
4216 011624 001402 BEQ 23$ ;BR IF NOT
4217 011626 000137 011370 JMP 10$ ;GO SEE IF V.35 CAN BE RUN
4218 011632 032737 000200 002354 23$: BIT #I422,RAX15 ;SEE IF 422 SELECTED
4219 011640 001402 BEQ 24$ ;BR IF NOT
4220 011642 000137 011462 JMP 15$ ;GO ASSERT AND RUN 422
4221 011646 005737 002462 24$: TST TSTCON ;SEE IF H3254 AND H3255 USED
4222 011652 001002 BNE 25$ ;BR IF NOT
4223 011654 000137 011406 JMP 11$ ;GO ASSERT AND RUN V.35 BY DEFAULT
4224 011660 032737 000100 002354 25$: BIT #XYZ,RAX15 ;SEE IF XYZ SELECTED
4225 011666 001402 BEQ 26$ ;BR IF NOT
4226 011670 000137 011540 JMP 19$ ;GO SEE IF XYZ CAN BE RUN
4227 011674 000137 011336 26$: JMP 9$ ;GO SKIP TEST
4228
4229 ;*** SET MAINT1 OR MAINT2 IF NEEDED, AND SET DTR ***
4230
4231 ;SET MAINT1 IF LOCAL LOOPBACK DESIRED
4232 011700 005037 002352 27$: CLR WRIBYT ;INIT DATA TO BE WRITTEN
4233 011704 012737 000013 002364 42$: MOV #13,REGNUM ;SET REG NO. = 13
4234 011712 023727 002462 000002 CMP TSTCON,#2 ;SEE IF MODEM LOCAL LPBK DESIRED
4235 011720 001032 BNE 29$ ;BR IF NOT, TO CHK REMOTE LPBK
4236 011722 112737 000010 002352 MOVB #MAINT1,WRIBYT
4237 011730 004737 003450 JSR PC,WRITLU ;SET MAINT1 IN REG 13
4238 011734 012737 000017 002364 MOV #17,REGNUM ;SET REG NO. = 17
4239 011742 012702 000000 MOV #0,R2 ;INIT TIMER
4240 011746 004737 003372 28$: JSR PC,READLU ;READ REG 17
4241 011752 132737 000004 002350 BITB #TESTMD,REDBYT ;SEE IF TEST MODE BIT SET IN REG 17 YET
4242 011760 001050 BNE 31$ ;BR IF YES, TO SET DTR
4243 011762 005202 INC R2 ;INCREMENT TIMER
4244 011764 001370 BNE 28$ ;BR IF NO TIME-OUT YET
4245 011766 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
4246 ;REPORT TEST MODE NOT SET BY MAINT1
4247 ERRDF 52,EM52,ERR4
(4) 011772 104455 TRAP C$ERDF
(5) 011774 000064 .WORD 52
(5) 011776 014471 .WORD EM52
(5) 012000 016376 .WORD ERR4
4248 012002 000137 011260 JMP 7$ ;BR TO SKIP TEST
4249 ;SET MAINT2 IF REMOTE LOOPBACK DESIRED
4250 012006 023727 002462 000003 29$: CMP TSTCON,#3 ;SEE IF MODEM REMOTE LPBK DESIRED
4251 012014 001032 BNE 31$ ;BR IF NOT, TO SET DTR
4252 012016 112737 000004 002352 MOVB #MAINT2,WRIBYT
4253 012024 004737 003450 JSR PC,WRITLU ;SET MAINT2 IN REG 13
4254 012030 012737 000017 002364 MOV #17,REGNUM ;SET REG NO. = 17
4255 012036 012702 000000 MOV #0,R2 ;INIT TIMER
4256 012042 004737 003372 30$: JSR PC,READLU ;READ REG 17
4257 012046 132737 000004 002350 BITB #TESTMD,REDBYT ;SEE IF TEST MODE BIT SET IN REG 17 YET
4258 012054 001012 BNE 31$ ;BR IF YES, TO SET DTR
4259 012056 005202 INC R2 ;INCREMENT TIMER
4260 012060 001370 BNE 30$ ;BR IF NO TIME-OUT YET
4261 012062 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
4262 ;REPORT TEST MODE NOT SET BY MAINT2
4263 ERRDF 53,EM53,ERR4
(4) 012066 104455 TRAP C$ERDF
(5) 012070 000065 .WORD 53
(5) 012072 014525 .WORD EM53

```

```

(5) 012074 016376
4264 012076 000137 011260          JMP      7$          ;BR TO SKIP TEST          .WORD   FRR4
4265                                     ;SET DTR
4266 012102 012737 000013 002364 31$:  MOV     #13,REGNUM    ;SET REG NO. = 13
4267 012110 152737 000100 002352   BISB   #DTR,WRIBYT    ;SET DTR BIT TO BE WRITTEN
4268 012116 004737 003450           JSR    PC,WRITLU     ;LOAD REG 13
4269
4270                                     ;*** BRANCH HERE TO RUN TEST ***
4271 012122 062766 000002 000006 32$:  ADD     #2,6(SP)      ;INCREMENT RETURN ADRS
4272
4273                                     ;*** BRANCH HERE TO SKIP TEST ***
4274 012130 012602          33$:  MOV     (SP)+,R2      ;RESTORE R2
4275 012132 012637 002366          MOV     (SP)+,AXNUM    ;RESTORE AX BYTE NO.
4276 012136 012637 002364          MOV     (SP)+,REGNUM  ;RESTORE LU REG NO.
4277 012142 152777 000010 170270   BISB   #LULOOP,@BSEL1 ;SET LULOOP
4278 012150 005037 002336          CLR    SUBRPC        ;CLEAR SUBROUTINE CALL PC
4279 012154 000207          RTS     PC           ;RETURN
4280
4281
4282
4283
4284
4285                                     ;*****
4286                                     ;*  CHKABT - THIS SUBROUTINE READS AX0-16 AND CHECKS FOR RAB, REOM SET. IF
4287                                     ;*  EITHER IS NOT SET, A RETURN IS MADE TO THE TEST, AT THE ADDRESS
4288                                     ;*  CONTAINED IN RETADR.
4289                                     ;*****
4290 012156 013746 002366          CHKABT: MOV    AXNUM,-(SP) ;SAVE AX BYTE NO.
4291 012162 016637 000002 002336     MOV    2(SP),SUBRPC
4292 012170 162737 000004 002336     SUB    #4,SUBRPC      ;GET PC OF SUBROUTINE CALL
4293 012176 005037 002366          CLR    AXNUM         ;SET AX0 ADDRESS
4294 012202 004737 003624          JSR    PC,READAX     ;READ REG AX0
4295 012206 032737 000004 002356     BIT    #RABT,RAX16   ;CHK FOR RAB SET
4296 012214 001007          BNE    6$           ;BR IF RAB SET
4297 012216 004737 004226          JSR    PC,GETALL    ;GET REGS FOR PRINTOUT
4298                                     ;REPORT RAB NOT SET
4299                                     ERRDF  40,EM40,ERR6
(4) 012222 104455
(5) 012224 000050          TRAP   C$ERDF
(5) 012226 014044          .WORD  40
(5) 012230 017566          .WORD  EM40
4300 012232 000412          .WORD  ERR6
4301 012234 032737 000002 002356 6$:  BR     8$
4302 012242 001015          BIT    #REOM,RAX16  ;CHK FOR REOM SET
4303 012244 004737 004226          BNE    9$           ;BR IF REOM SET
4304                                     JSR    PC,GETALL    ;GET REGS FOR PRINTOUT
4305                                     ;REPORT REOM NOT SET
4306                                     ERRDF  31,EM31,ERR6
(4) 012250 104455          TRAP   C$ERDF
(5) 012252 000037          .WORD  31
(5) 012254 013667          .WORD  EM31
(5) 012256 017566          .WORD  ERR6
4306 012260 011637 002366          8$:  MOV     (SP),AXNUM   ;RESTORE AX BYTE NO.
4307 012264 013706 002332          MOV     PSTACK,SP   ;RESTORE STACK POINTER TO BASE LEVEL
4308 012270 013746 002346          MOV     RETADR,-(SP) ;FIX ERROR RETURN PC
4309 012274 000402          BR     16$
4310 012276 012637 002366          9$:  MOV     (SP)+,AXNUM ;RESTORE AX BYTE NO.

```

CZDMSC MB203 STATIC TESTS #2
CZDMSC.P11 13-MAR-80 12:31

MACY11 30A(1052) 13-MAR-80 12:35 D 7 PAGE 2-46
GLOBAL SUBROUTINES

SEQ 0081

4311 012302 000207 168: RTS PC ;RETURN
4312
4313
4314
4315
4316

4318
4319
4320
4321
4322
4323
4324
4325
4326
4327

.SBTTL GLOBAL ERROR REPORT SECTION

:/
:/ THE GLOBAL ERROR REPORT SECTION CONTAINS ERROR MESSAGES
:/ THAT ARE USED IN MORE THAN ONE TEST.
:/

012304	052045	047445	022466	FMT1:	.ASCIZ	/%T%06%N/	
012312	000116						
4328	012314	047045	040445	040506	FMT2:	.ASCIZ	/%N%AFAILING REG: /
	012322	046111	047111	020107			
	012330	042522	035107	000040			
4329	012336	040445	054105	042520	FMT3:	.ASCIZ	/%AEXPECTED: %03%S5%AACTUAL: %03%N/
	012344	052103	042105	020072			
	012352	047445	022463	032523			
	012360	040445	041501	052524			
	012366	046101	020072	047445			
	012374	022463	000116				
4330	012400	047045	052045	047045	FMT4:	.ASCIZ	/%N%T%N%T%N/
	012406	052045	047045	000			
4331	012413	045	031517	051445	FMT5:	.ASCIZ	/%03%S5%03%S5%03%S5%03%N/
	012420	022465	031517	051445			
	012426	022465	031517	051445			
	012434	022465	031517	047045			
	012442	000					
4332	012443	045	032123	047445	FMT6:	.ASCIZ	/%S4%03%S5%03%S5%03%S5%03%N/
	012450	022463	032523	047445			
	012456	022463	032523	047445			
	012464	022463	032523	047445			
	012472	022463	000116				
4333	012476	052045	047445	022462	FMT7:	.ASCIZ	/%T%02%N/
	012504	000116					
4334	012506	040445	054105	042524	FMT8:	.ASCIZ	/%AEXTENDED REG AX%01%A-%T%N/
	012514	042116	042105	051040			
	012522	043505	040440	022530			
	012530	030517	040445	022455			
	012536	022524	000116				
4335	012542	052045	047045	000	FMT9:	.ASCIZ	/%T%N/
4336	012547	045	050101	020103	FMT10:	.ASCIZ	/%APC OF SUBR CALL: %06%N/
	012554	043117	051440	041125			
	012562	020122	040503	046114			
	012570	020072	047445	022466			
	012576	000116					
4337	012600	040445	042522	020107	FMT11:	.ASCIZ	/%AREG %02%A LOADED WITH: %03%N/
	012606	047445	022462	020101			
	012614	047514	042101	042105			
	012622	053440	052111	035110			
	012630	022440	031517	047045			
	012636	000					
4338	012637	045	046501	031070	FMT12:	.ASCIZ	/%AM8203 SW PACK #1 (REG 11) = %03%N/
	012644	031460	051440	020127			
	012652	040520	045503	021440			
	012660	020061	051050	043505			
	012666	030440	024461	036440			

	012674	022440	031517	047045	
	012702	000			
4339	012703	045	046501	031070	FMT13: .ASCIZ /%AM8203 SW PACK #2 (REG 15) = %03%N/
	012710	031460	051440	020127	
	012716	040520	045503	021440	
	012724	020062	051050	043505	
	012732	030440	024465	036440	
	012740	022440	031517	047045	
	012746	000			
4340	012747	045	046501	031070	FMT14: .ASCIZ /%AM8203 SW PACK #3 (REG 16) = %03%N/
	012754	031460	051440	020127	
	012762	040520	045503	021440	
	012770	020063	051050	043505	
	012776	030440	024466	036440	
	013004	022440	031517	047045	
	013012	000			
4341	013013	045	046501	042117	FMT15: .ASCIZ /%AMODEM INTERFACE REG (AX3-15) = %03%N/
	013020	046505	044440	052116	
	013026	051105	040506	042503	
	013034	051040	043505	024040	
	013042	054101	026463	032461	
	013050	020051	020075	047445	
	013056	022463	000116		
4342	013062	047045	040445	047506	FMT18: .ASCIZ /%N%AFOR DEVICE AT ADRS %06%A ,%N/
	013070	020122	042504	044526	
	013076	042503	040440	020124	
	013104	042101	051522	020040	
	013112	047445	022466	020101	
	013120	022454	000116		
4343	013124	047045	040445	042524	FMT19: .ASCIZ /%N%ATEST %D2%A NOT RUN%N/
	013132	052123	022440	031104	
	013140	040445	047040	052117	
	013146	051040	047125	047045	
	013154	000			
4344	013155	045	022516	041101	FMT25: .ASCIZ /%N%ABAUD RATE INCORRECT/
	013162	052501	020104	040522	
	013170	042524	044440	041516	
	013176	051117	042522	052103	
	013204	000			
4345	013205	045	022516	046501	FMT26: .ASCIZ /%N%AMODEM INTERFACE NOT SELECTED/
	013212	042117	046505	044440	
	013220	052116	051105	040506	
	013226	042503	047040	052117	
	013234	051440	046105	041505	
	013242	042524	000104		
4346					
4347					
4348					
4349	013246	042522	020107	047516	EM2: .ASCIZ /REG NOT INITIALIZED BY MST CLR/
	013254	020124	047111	052111	
	013262	040511	044514	042532	
	013270	020104	054502	046440	
	013276	052123	041440	051114	
	013304	000			
4350	013305	122	043505	046440	EM3: .ASCIZ /REG MISCOMPARE/
	013312	051511	047503	050115	

4351	013320	051101	000105						
	013324	051117	054504	047040	EM7:	.ASCIZ	/ORDY NOT SET/		
	013332	052117	051440	052105					
	013340	000							
4352	013341	117	042122	020131	EM8:	.ASCIZ	/ORDY NOT CLEARED/		
	013346	047516	020124	046103					
	013354	040505	042522	000104					
4353	013362	041517	051117	047040	EM9:	.ASCIZ	/OCOR NOT SET/		
	013370	052117	051440	052105					
	013376	000							
4354	013377	117	047503	020122	EM10:	.ASCIZ	/OCOR NOT CLEARED/		
	013404	047516	020124	046103					
	013412	040505	042522	000104					
4355	013420	040517	052103	047040	EM11:	.ASCIZ	/OACT NOT SET/		
	013426	052117	051440	052105					
	013434	000							
4356	013435	117	041501	020124	EM12:	.ASCIZ	/OACT NOT CLEARED/		
	013442	047516	020124	046103					
	013450	040505	042522	000104					
4357	013456	051111	054504	047040	EM17:	.ASCIZ	/IRDY NOT SET/		
	013464	052117	051440	052105					
	013472	000							
4358	013473	111	042122	020131	EM18:	.ASCIZ	/IRDY NOT CLEARED/		
	013500	047516	020124	046103					
	013506	040505	042522	000104					
4359	013514	041511	051111	047040	EM19:	.ASCIZ	/ICIR NOT SET/		
	013522	052117	051440	052105					
	013530	000							
4360	013531	111	044503	020122	EM20:	.ASCIZ	/ICIR NOT CLEARED/		
	013536	047516	020124	046103					
	013544	040505	042522	000104					
4361	013552	040511	052103	047040	EM21:	.ASCIZ	/IACT NOT SET/		
	013560	052117	051440	052105					
	013566	000							
4362	013567	111	041501	020124	EM22:	.ASCIZ	/IACT NOT CLEARED/		
	013574	047516	020124	046103					
	013602	040505	042522	000104					
4363	013610	051522	046517	047040	EM28:	.ASCIZ	/RSOM NOT CLEARED/		
	013616	052117	041440	042514					
	013624	051101	042105	000					
4364	013631	122	047523	020115	EM29:	.ASCIZ	/RSOM NOT SET/		
	013636	047516	020124	042523					
	013644	000124							
4365	013646	042522	046517	047040	EM30:	.ASCIZ	/REOM NOT CLEARED/		
	013654	052117	041440	042514					
	013662	051101	042105	000					
4366	013667	122	047505	020115	EM31:	.ASCIZ	/REOM NOT SET/		
	013674	047516	020124	042523					
	013702	000124							
4367	013704	041522	023526	020104	EM34:	.ASCIZ	/RCV'D DATA MISCOMPARE/		
	013712	040504	040524	046440					
	013720	051511	047503	050115					
	013726	051101	000105						
4368	013732	041502	020103	047516	EM35:	.ASCIZ	/BCC NOT CLEARED/		
	013740	020124	046103	040505					
	013746	042522	000104						

4369	013752	041502	020103	047516	EM36:	.ASCIZ	/BCC NOT SET/
	013760	020124	042523	000124			
4370	013766	041105	045514	047040	EM37:	.ASCIZ	/EBLK NOT CLEARED/
	013774	052117	041440	042514			
	014002	051101	042105	000			
4371	014007	105	046102	020113	EM38:	.ASCIZ	/EBLK NOT SET/
	014014	047516	020124	042523			
	014022	000124					
4372	014024	040522	020102	047516	EM39:	.ASCIZ	/RAB NOT CLEARED/
	014032	020124	046103	040505			
	014040	042522	000104				
4373	014044	040522	020102	047516	EM40:	.ASCIZ	/RAB NOT SET/
	014052	020124	042523	000124			
4374	014060	053117	051122	047040	EM41:	.ASCIZ	/OVRR NOT CLEARED/
	014066	052117	041440	042514			
	014074	051101	042105	000			
4375	014101	117	051126	020122	EM42:	.ASCIZ	/OVRR NOT SET/
	014106	047516	020124	042523			
	014114	000124					
4376	014116	053523	050040	041501	EM43:	.ASCIZ	/SW PACK #1 INCORRECT/
	014124	020113	030443	044440			
	014132	041516	051117	042522			
	014140	052103	000				
4377	014143	123	020127	040520	EM44:	.ASCIZ	/SW PACK #2 INCORRECT/
	014150	045503	021440	020062			
	014156	047111	047503	051122			
	014164	041505	000124				
4378	014170	053523	050040	041501	EM45:	.ASCIZ	/SW PACK #3 INCORRECT/
	014176	020113	031443	044440			
	014204	041516	051117	042522			
	014212	052103	000				
4379	014215	122	053103	051440	EM46:	.ASCIZ	/RCV SILO NOT CLEARED BY IC/
	014222	046111	020117	047516			
	014230	020124	046103	040505			
	014236	042522	020104	054502			
	014244	044440	000103				
4380	014250	051501	042523	041115	EM47:	.ASCIZ	/ASSEMB BIT COUNT INCORRECT/
	014256	041040	052111	041440			
	014264	052517	052116	044440			
	014272	041516	051117	042522			
	014300	052103	000				
4381	014303	117	042104	053040	EM48:	.ASCIZ	/ODD VRC PARITY BIT NOT SET/
	014310	041522	050040	051101			
	014316	052111	020131	044502			
	014324	020124	047516	020124			
	014332	042523	000124				
4382	014336	042117	020104	051126	EM49:	.ASCIZ	/ODD VRC PARITY BIT NOT CLEARED/
	014344	020103	040520	044522			
	014352	054524	041040	052111			
	014360	047040	052117	041440			
	014366	042514	051101	042105			
	014374	000					
4383	014375	105	042526	020116	EM50:	.ASCIZ	/EVEN VRC PARITY BIT NOT SET/
	014402	051126	020103	040520			
	014410	044522	054524	041040			
	014416	052111	047040	052117			

4384	014424	051440	052105	000		
	014431	105	042526	020116	EM51:	.ASCIZ /EVEN VRC PARITY BIT NOT CLEARED/
	014436	051126	020103	040520		
	014444	044522	054524	041040		
	014452	052111	047040	052117		
	014460	041440	042514	051101		
	014466	042105	000			
4385	014471	124	051505	020124	EM52:	.ASCIZ /TEST MODE NOT SET BY MAINT1/
	014476	047515	042504	047040		
	014504	052117	051440	052105		
	014512	041040	020131	040515		
	014520	047111	030524	000		
4386	014525	124	051505	020124	EM53:	.ASCIZ /TEST MODE NOT SET BY MAINT2/
	014532	047515	042504	047040		
	014540	052117	051440	052105		
	014546	041040	020131	040515		
	014554	047111	031124	000		
4387	014561	124	020130	047125	EM54:	.ASCIZ /TX UNDERRUN ERROR/
	014566	042504	051122	047125		
	014574	042440	051122	051117		
	014602	000				
4388	014603	104	051124	047040	EM55:	.ASCIZ /DTR NOT SET/
	014610	052117	051440	052105		
	014616	000				
4389	014617	122	047111	020107	EM56:	.ASCIZ /RING NOT SET/
	014624	047516	020124	042523		
	014632	000124				
4390	014634	047515	051104	047040	EM57:	.ASCIZ /MODR NOT SET/
	014642	052117	051440	052105		
	014650	000				
4391	014651	110	054104	047040	EM58:	.ASCIZ /HDX NOT SET/
	014656	052117	051440	052105		
	014664	000				
4392	014665	123	041124	020131	EM59:	.ASCIZ /STBY NOT SET/
	014672	047516	020124	042523		
	014700	000124				
4393	014702	052122	020123	047516	EM60:	.ASCIZ /RTS NOT SET/
	014710	020124	042523	000124		
4394	014716	051503	047040	052117	EM61:	.ASCIZ /CS NOT SET/
	014724	051440	052105	000		
4395	014731	103	051101	020122	EM62:	.ASCIZ /CARR NOT SET/
	014736	047516	020124	042523		
	014744	000124				
4396	014746	044523	050507	047040	EM63:	.ASCIZ /SIGQ NOT SET/
	014754	052117	051440	052105		
	014762	000				
4397	014763	123	043511	020122	EM64:	.ASCIZ /SIGR NOT SET/
	014770	047516	020124	042523		
	014776	000124				
4398	015000	052122	020123	047516	EM65:	.ASCIZ /RTS NOT CLEARED/
	015006	020124	046103	040505		
	015014	042522	000104			
4399	015020	040503	051122	047040	EM66:	.ASCIZ /CARR NOT CLEARED/
	015026	052117	041440	042514		
	015034	051101	042105	000		
4400						

```

4401
4402
4403 015041 111 041116 051525 DH1: .ASCIZ &INBUS/OUTBUS REG &
      015046 047457 052125 052502
      015054 020123 042522 020107
      015062 000
4404 015063 114 047111 020105 DH2: .ASCIZ /LINE UNIT INBUS REGS :/
      015070 047125 052111 044440
      015076 041116 051525 051040
      015104 043505 020123 000072
4405 015112 042522 030507 020060 DH3: .ASCIZ /REG10 REG11 REG12 REG13/
      015120 020040 042522 030507
      015126 020061 020040 042522
      015134 030507 020062 020040
      015142 042522 030507 000063
4406 015150 020040 020040 042522 DH4: .ASCIZ / REG14 REG15 REG16 REG17/
      015156 030507 020064 020040
      015164 042522 030507 020065
      015172 020040 042522 030507
      015200 020066 020040 042522
      015206 030507 000067
4407 015212 032461 000 DH5: .ASCIZ /15/
4408 015215 061 000066 DH6: .ASCIZ /16/
4409 015220 044514 042516 052440 DH7: .ASCIZ /LINE UNIT EXTENDED REGS :/
      015226 044516 020124 054105
      015234 042524 042116 042105
      015242 051040 043505 020123
      015250 000072
4410 015252 054101 026460 032461 DH8: .ASCIZ /AX0-15 AX0-16 AX1-15 AX1-16/
      015260 020040 054101 026460
      015266 033061 020040 054101
      015274 026461 032461 020040
      015302 054101 026461 033061
      015310 000
4411 015311 040 020040 040440 DH9: .ASCIZ / AX2-15 AX2-16 AX3-15 AX3-16/
      015316 031130 030455 020065
      015324 040440 031130 030455
      015332 020066 040440 031530
      015340 030455 020065 040440
      015346 031530 030455 000066
  
```

```

4412
4413 .EVEN
4414
4415
4416
4417
4418
  
```

```

4419 015354 BGNMSG ERR1
      (3) 015354
4420 015354 PRINTB #FMT1,#ADDRES,MPCSR
      (9) 015354 013746 002436
      (8) 015360 012746 036506
      (7) 015364 012746 012304
      (6) 015370 012746 000003
      (3) 015374 010600
      (4) 015376 104414
  
```

```

ERR1::
      MOV MPCSR,-(SP)
      MOV #ADDRES,-(SP)
      MOV #FMT1,-(SP)
      MOV #3,-(SP)
      MOV SP,R0
      TRAP C$PNTB
  
```

(4)	015400	062706	000010						
4421	015404			ENDMSG				ADD	#10,SP
(3)	015404								
(3)	015404	104423						L10002:	TRAP C\$MSG
4422									
4423									
4424									
4425	015406			BGNMSG ERR2					
(3)	015406							ERR2::	
4426	015406			PRINTB #FMT1,#ADDRES,MPCSR					
(9)	015406	013746	002436					MOV	MPCSR,-(SP)
(8)	015412	012746	036506					MOV	#ADDRES,-(SP)
(7)	015416	012746	012304					MOV	#FMT1,-(SP)
(6)	015422	012746	000003					MOV	#3,-(SP)
(3)	015426	010600						MOV	SP,R0
(4)	015430	104414						TRAP	C\$PNTB
(4)	015432	062706	000010					ADD	#10,SP
4427	015436			PRINTB #FMT2					
(7)	015436	012746	012314					MOV	#FMT2,-(SP)
(6)	015442	012746	000001					MOV	#1,-(SP)
(3)	015446	010600						MOV	SP,R0
(4)	015450	104414						TRAP	C\$PNTB
(4)	015452	062706	000004					ADD	#4,SP
4428	015456			PRINTB #FMT7,#DH1,REGNUM					
(9)	015456	013746	002364					MOV	REGNUM,-(SP)
(8)	015462	012746	015041					MOV	#DH1,-(SP)
(7)	015466	012746	012476					MOV	#FMT7,-(SP)
(6)	015472	012746	000003					MOV	#3,-(SP)
(3)	015476	010600						MOV	SP,R0
(4)	015500	104414						TRAP	C\$PNTB
(4)	015502	062706	000010					ADD	#10,SP
4429	015506			PRINTB #FMT3,GOODAT,BADDAT					
(9)	015506	013746	002372					MOV	BADDAT,-(SP)
(8)	015512	013746	002370					MOV	GOODAT,-(SP)
(7)	015516	012746	012336					MOV	#FMT3,-(SP)
(6)	015522	012746	000003					MOV	#3,-(SP)
(3)	015526	010600						MOV	SP,R0
(4)	015530	104414						TRAP	C\$PNTB
(4)	015532	062706	000010					ADD	#10,SP
4430	015536			PRINTX #FMT4,#DH2,#DH3					
(9)	015536	012746	015112					MOV	#DH3,-(SP)
(8)	015542	012746	015063					MOV	#DH2,-(SP)
(7)	015546	012746	012400					MOV	#FMT4,-(SP)
(6)	015552	012746	000003					MOV	#3,-(SP)
(3)	015556	010600						MOV	SP,R0
(4)	015560	104415						TRAP	C\$PNTX
(4)	015562	062706	000010					ADD	#10,SP
4431	015566			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13					
(11)	015566	013746	002274					MOV	LUR13,-(SP)
(10)	015572	013746	002272					MOV	LUR12,-(SP)
(9)	015576	013746	002270					MOV	LUR11,-(SP)
(8)	015602	013746	002266					MOV	LUR10,-(SP)
(7)	015606	012746	012413					MOV	#FMT5,-(SP)
(6)	015612	012746	000005					MOV	#5,-(SP)
(3)	015616	010600						MOV	SP,R0
(4)	015620	104415						TRAP	C\$PNTX

4432	(4)	015622	062706	000014			ADD	#14,SP
		015626			PRINTX	#FMT9,#DH4		
	(8)	015626	012746	015150			MOV	#DH4,-(SP)
	(7)	015632	012746	012542			MOV	#FMT9,-(SP)
	(6)	015636	012746	000002			MOV	#2,-(SP)
	(3)	015642	010600				MOV	SP,R0
	(4)	015644	104415				TRAP	C\$PNTX
4433	(4)	015646	062706	000006			ADD	#6,SP
		015652			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17		
	(11)	015652	013746	002304			MOV	LUR17,-(SP)
	(10)	015656	013746	002302			MOV	LUR16,-(SP)
	(9)	015662	013746	002300			MOV	LUR15,-(SP)
	(8)	015666	013746	002276			MOV	LUR14,-(SP)
	(7)	015672	012746	012443			MOV	#FMT6,-(SP)
	(6)	015676	012746	000005			MOV	#5,-(SP)
	(3)	015702	010600				MOV	SP,R0
	(4)	015704	104415				TRAP	C\$PNTX
	(4)	015706	062706	000014			ADD	#14,SP
4434		015712			ENDMSG			
	(3)	015712						
	(3)	015712	104423				L10003:	TRAP C\$MSG
4435								
4436								
4437								
4438								
4439								
4440		015714			BGNMSG	ERR3		
	(3)	015714						
4441		015714			PRINTB	#FMT1,#ADDRES,MPCSR		ERR3::
	(9)	015714	013746	002436			MOV	MPCSR,-(SP)
	(8)	015720	012746	036506			MOV	#ADDRES,-(SP)
	(7)	015724	012746	012304			MOV	#FMT1,-(SP)
	(6)	015730	012746	000003			MOV	#3,-(SP)
	(3)	015734	010600				MOV	SP,R0
	(4)	015736	104414				TRAP	C\$PNTB
	(4)	015740	062706	000010			ADD	#10,SP
4442		015744			PRINTB	#FMT2		
	(7)	015744	012746	012314			MOV	#FMT2,-(SP)
	(6)	015750	012746	000001			MOV	#1,-(SP)
	(3)	015754	010600				MOV	SP,R0
	(4)	015756	104414				TRAP	C\$PNTB
	(4)	015760	062706	000004			ADD	#4,SP
4443		015764			PRINTB	#FMT8,TMP1,TMP0		
	(9)	015764	013746	002516			MOV	TMP0,-(SP)
	(8)	015770	013746	002520			MOV	TMP1,-(SP)
	(7)	015774	012746	012506			MOV	#FMT8,-(SP)
	(6)	016000	012746	000003			MOV	#3,-(SP)
	(3)	016004	010600				MOV	SP,R0
	(4)	016006	104414				TRAP	C\$PNTB
	(4)	016010	062706	000010			ADD	#10,SP
4444		016014			PRINTB	#FMT3,GOODAT,BADDAT		
	(9)	016014	013746	002372			MOV	BADDAT,-(SP)
	(8)	016020	013746	002370			MOV	GOODAT,-(SP)
	(7)	016024	012746	012336			MOV	#FMT3,-(SP)
	(6)	016030	012746	000003			MOV	#3,-(SP)
	(3)	016034	010600				MOV	SP,R0

(4)	016036	104414			TRAP	C\$PNTB
(4)	016040	062706	000010		ADD	#10,SP
4445	016044			PRINTX #FMT4,#DH2,#DH3		
(9)	016044	012746	015112		MOV	#DH3,-(SP)
(8)	016050	012746	015063		MOV	#DH2,-(SP)
(7)	016054	012746	012400		MOV	#FMT4,-(SP)
(6)	016060	012746	000003		MOV	#3,-(SP)
(3)	016064	010600			MOV	SP,R0
(4)	016066	104415			TRAP	C\$PNTX
(4)	016070	062706	000010		ADD	#10,SP
4446	016074			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13		
(11)	016074	013746	002274		MOV	LUR13,-(SP)
(10)	016100	013746	002272		MOV	LUR12,-(SP)
(9)	016104	013746	002270		MOV	LUR11,-(SP)
(8)	016110	013746	002266		MOV	LUR10,-(SP)
(7)	016114	012746	012413		MOV	#FMT5,-(SP)
(6)	016120	012746	000005		MOV	#5,-(SP)
(3)	016124	010600			MOV	SP,R0
(4)	016126	104415			TRAP	C\$PNTX
(4)	016130	062706	000014		ADD	#14,SP
4447	016134			PRINTX #FMT9,#DH4		
(8)	016134	012746	015150		MOV	#DH4,-(SP)
(7)	016140	012746	012542		MOV	#FMT9,-(SP)
(6)	016144	012746	000002		MOV	#2,-(SP)
(3)	016150	010600			MOV	SP,R0
(4)	016152	104415			TRAP	C\$PNTX
(4)	016154	062706	000006		ADD	#6,SP
4448	016160			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17		
(11)	016160	013746	002304		MOV	LUR17,-(SP)
(10)	016164	013746	002302		MOV	LUR16,-(SP)
(9)	016170	013746	002300		MOV	LUR15,-(SP)
(8)	016174	013746	002276		MOV	LUR14,-(SP)
(7)	016200	012746	012443		MOV	#FMT6,-(SP)
(6)	016204	012746	000005		MOV	#5,-(SP)
(3)	016210	010600			MOV	SP,R0
(4)	016212	104415			TRAP	C\$PNTX
(4)	016214	062706	000014		ADD	#14,SP
4449	016220			PRINTX #FMT4,#DH7,#DH8		
(9)	016220	012746	015252		MOV	#DH8,-(SP)
(8)	016224	012746	015220		MOV	#DH7,-(SP)
(7)	016230	012746	012400		MOV	#FMT4,-(SP)
(6)	016234	012746	000003		MOV	#3,-(SP)
(3)	016240	010600			MOV	SP,R0
(4)	016242	104415			TRAP	C\$PNTX
(4)	016244	062706	000010		ADD	#10,SP
4450	016250			PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16		
(11)	016250	013746	002314		MOV	AX1.16,-(SP)
(10)	016254	013746	002312		MOV	AX1.15,-(SP)
(9)	016260	013746	002310		MOV	AX0.16,-(SP)
(8)	016264	013746	002306		MOV	AX0.15,-(SP)
(7)	016270	012746	012413		MOV	#FMT5,-(SP)
(6)	016274	012746	000005		MOV	#5,-(SP)
(3)	016300	010600			MOV	SP,R0
(4)	016302	104415			TRAP	C\$PNTX
(4)	016304	062706	000014		ADD	#14,SP
4451	016310			PRINTX #FMT9,#DH9		

(8)	016310	012746	015311				MOV	#DH9,-(SP)
(7)	016314	012746	012542				MOV	#FMT9,-(SP)
(6)	016320	012746	000002				MOV	#2,-(SP)
(3)	016324	010600					MOV	SP,R0
(4)	016326	104415					TRAP	C\$PNTX
(4)	016330	062706	000006				ADD	#6,SP
4452	016334			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16			
(11)	016334	013746	002324				MOV	AX3.16,-(SP)
(10)	016340	013746	002322				MOV	AX3.15,-(SP)
(9)	016344	013746	002320				MOV	AX2.16,-(SP)
(8)	016350	013746	002316				MOV	AX2.15,-(SP)
(7)	016354	012746	012443				MOV	#FMT6,-(SP)
(6)	016360	012746	000005				MOV	#5,-(SP)
(3)	016364	010600					MOV	SP,R0
(4)	016366	104415					TRAP	C\$PNTX
(4)	016370	062706	000014				ADD	#14,SP
4453	016374			ENDMSG				
(3)	016374							
(3)	016374	104423				L10004:	TRAP	C\$MSG
4454								
4455								
4456								
4457								
4458								
4459	016376			BGNMSG	ERR4			
(3)	016376							
4460	016376			PRINTB	#FMT10,SUBRPC	ERR4::		
(8)	016376	013746	002336				MOV	SUBRPC,-(SP)
(7)	016402	012746	012547				MOV	#FMT10,-(SP)
(6)	016406	012746	000002				MOV	#2,-(SP)
(3)	016412	010600					MOV	SP,R0
(4)	016414	104414					TRAP	C\$PNTB
(4)	016416	062706	000006				ADD	#6,SP
4461	016422			PRINTB	#FMT1,#ADDRES,MPCSR			
(9)	016422	013746	002436				MOV	MPCSR,-(SP)
(8)	016426	012746	036506				MOV	#ADDRES,-(SP)
(7)	016432	012746	012304				MOV	#FMT1,-(SP)
(6)	016436	012746	000003				MOV	#3,-(SP)
(3)	016442	010600					MOV	SP,R0
(4)	016444	104414					TRAP	C\$PNTB
(4)	016446	062706	000010				ADD	#10,SP
4462	016452			PRINTB	#FMT2			
(7)	016452	012746	012314				MOV	#FMT2,-(SP)
(6)	016456	012746	000001				MOV	#1,-(SP)
(3)	016462	010600					MOV	SP,R0
(4)	016464	104414					TRAP	C\$PNTB
(4)	016466	062706	000004				ADD	#4,SP
4463	016472			PRINTB	#FMT7,#DH1,REGNUM			
(9)	016472	013746	002364				MOV	REGNUM,-(SP)
(8)	016476	012746	015041				MOV	#DH1,-(SP)
(7)	016502	012746	012476				MOV	#FMT7,-(SP)
(6)	016506	012746	000003				MOV	#3,-(SP)
(3)	016512	010600					MOV	SP,R0
(4)	016514	104414					TRAP	C\$PNTB
(4)	016516	062706	000010				ADD	#10,SP
4464	016522			PRINTX	#FMT4,#DH2,#DH3			

(9) 016522 012746 015112
(8) 016526 012746 015063
(7) 016532 012746 012400
(6) 016536 012746 000003
(3) 016542 010600
(4) 016544 104415
(4) 016546 062706 000010
4465 016552
(11) 016552 013746 002274
(10) 016556 013746 002272
(9) 016562 013746 002270
(8) 016566 013746 002266
(7) 016572 012746 012413
(6) 016576 012746 000005
(3) 016602 010600
(4) 016604 104415
(4) 016606 062706 000014
4466 016612
(8) 016612 012746 015150
(7) 016616 012746 012542
(6) 016622 012746 000002
(3) 016626 010600
(4) 016630 104415
(4) 016632 062706 000006
4467 016636
(11) 016636 013746 002304
(10) 016642 013746 002302
(9) 016646 013746 002300
(8) 016652 013746 002276
(7) 016656 012746 012443
(6) 016662 012746 000005
(3) 016666 010600
(4) 016670 104415
(4) 016672 062706 000014
4468 016676
(9) 016676 012746 015252
(8) 016702 012746 015220
(7) 016706 012746 012400
(6) 016712 012746 000003
(3) 016716 010600
(4) 016720 104415
(4) 016722 062706 000010
4469 016726
(11) 016726 013746 002314
(10) 016732 013746 002312
(9) 016736 013746 002310
(8) 016742 013746 002306
(7) 016746 012746 012413
(6) 016752 012746 000005
(3) 016756 010600
(4) 016760 104415
(4) 016762 062706 000014
4470 016766
(8) 016766 012746 015311
(7) 016772 012746 012542
(6) 016776 012746 000002

PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

PRINTX #FMT4,#DH7,#DH8

PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16

PRINTX #FMT9,#DH9

MOV #DH3,-(SP)
MOV #DH2,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV LUR13,-(SP)
MOV LUR12,-(SP)
MOV LUR11,-(SP)
MOV LUR10,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH4,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV LUR17,-(SP)
MOV LUR16,-(SP)
MOV LUR15,-(SP)
MOV LUR14,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH8,-(SP)
MOV #DH7,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV AX1.16,-(SP)
MOV AX1.15,-(SP)
MOV AX0.16,-(SP)
MOV AX0.15,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH9,-(SP)
MOV #FMT6,-(SP)
MOV #2,-(SP)

(3)	017002	010600			MOV	SP,R0
(4)	017004	104415			TRAP	C\$PNTX
(4)	017006	062706	000006		ADD	#6,SP
4471	017012			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
(11)	017012	013746	002324		MOV	AX3.16,-(SP)
(10)	017016	013746	002322		MOV	AX3.15,-(SP)
(9)	017022	013746	002320		MOV	AX2.16,-(SP)
(8)	017026	013746	002316		MOV	AX2.15,-(SP)
(7)	017032	012746	012443		MOV	#FMT6,-(SP)
(6)	017036	012746	000005		MOV	#5,-(SP)
(3)	017042	010600			MOV	SP,R0
(4)	017044	104415			TRAP	C\$PNTX
(4)	017046	062706	000014		ADD	#14,SP
4472	017052			ENDMSG		
(3)	017052					
(3)	017052	104423			L10005:	TRAP
4473						C\$MSG
4474						
4475						
4476						
4477						
4478	017054			BGNMSG	ERR5	
(3)	017054					
4479	017054			PRINTB	#FMT1,#ADDRES,MPCSR	ERR5::
(9)	017054	013746	002436		MOV	MPCSR,-(SP)
(8)	017060	012746	036506		MOV	#ADDRES,-(SP)
(7)	017064	012746	012304		MOV	#FMT1,-(SP)
(6)	017070	012746	000003		MOV	#3,-(SP)
(3)	017074	010600			MOV	SP,R0
(4)	017076	104414			TRAP	C\$PNTB
(4)	017100	062706	000010		ADD	#10,SP
4480	017104			PRINTB	#FMT11,REGNUM,LOADAT	
(9)	017104	013746	002374		MOV	LOADAT,-(SP)
(8)	017110	013746	002364		MOV	REGNUM,-(SP)
(7)	017114	012746	012600		MOV	#FMT11,-(SP)
(6)	017120	012746	000003		MOV	#3,-(SP)
(3)	017124	010600			MOV	SP,R0
(4)	017126	104414			TRAP	C\$PNTB
(4)	017130	062706	000010		ADD	#10,SP
4481	017134			PRINTB	#FMT2	
(7)	017134	012746	012314		MOV	#FMT2,-(SP)
(6)	017140	012746	000001		MOV	#1,-(SP)
(3)	017144	010600			MOV	SP,R0
(4)	017146	104414			TRAP	C\$PNTB
(4)	017150	062706	000004		ADD	#4,SP
4482	017154			PRINTB	#FMT8,TMP1,TMP0	
(9)	017154	013746	002516		MOV	TMP0,-(SP)
(8)	017160	013746	002520		MOV	TMP1,-(SP)
(7)	017164	012746	012506		MOV	#FMT8,-(SP)
(6)	017170	012746	000003		MOV	#3,-(SP)
(3)	017174	010600			MOV	SP,R0
(4)	017176	104414			TRAP	C\$PNTB
(4)	017200	062706	000010		ADD	#10,SP
4483	017204			PRINTB	#FMT3,GOODAT,BADDAT	
(9)	017204	013746	002372		MOV	BADDAT,-(SP)
(8)	017210	013746	002370		MOV	GOODAT,-(SP)

(7) 017214 012746 012336
(6) 017220 012746 000003
(3) 017224 010600
(4) 017226 104414
(4) 017230 062706 000010
4484 017234
(9) 017234 012746 015112
(8) 017240 012746 015063
(7) 017244 012746 012400
(6) 017250 012746 000003
(3) 017254 010600
(4) 017256 104415
(4) 017260 062706 000010
4485 017264
(11) 017264 013746 002274
(10) 017270 013746 002272
(9) 017274 013746 002270
(8) 017300 013746 002266
(7) 017304 012746 012413
(6) 017310 012746 000005
(3) 017314 010600
(4) 017316 104415
(4) 017320 062706 000014
4486 017324
(8) 017324 012746 015150
(7) 017330 012746 012542
(6) 017334 012746 000002
(3) 017340 010600
(4) 017342 104415
(4) 017344 062706 000006
4487 017350
(11) 017350 013746 002304
(10) 017354 013746 002302
(9) 017360 013746 002300
(8) 017364 013746 002276
(7) 017370 012746 012443
(6) 017374 012746 000005
(3) 017400 010600
(4) 017402 104415
(4) 017404 062706 000014
4488 017410
(9) 017410 012746 015252
(8) 017414 012746 015220
(7) 017420 012746 012400
(6) 017424 012746 000003
(3) 017430 010600
(4) 017432 104415
(4) 017434 062706 000010
4489 017440
(11) 017440 013746 002314
(10) 017444 013746 002312
(9) 017450 013746 002310
(8) 017454 013746 002306
(7) 017460 012746 012413
(6) 017464 012746 000005
(3) 017470 010600

PRINTX #FMT4,#DH2,#DH3

PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

PRINTX #FMT4,#DH7,#DH8

PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16

MOV #FMT3,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #DH3,-(SP)
MOV #DH2,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV LUR13,-(SP)
MOV LUR12,-(SP)
MOV LUR11,-(SP)
MOV LUR10,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH4,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV LUR17,-(SP)
MOV LUR16,-(SP)
MOV LUR15,-(SP)
MOV LUR14,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH8,-(SP)
MOV #DH7,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV AX1.16,-(SP)
MOV AX1.15,-(SP)
MOV AX0.16,-(SP)
MOV AX0.15,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0

(4)	017472	104415				TRAP	C\$PNTX
(4)	017474	062706	000014			ADD	#14,SP
4490	017500			PRINTX	#FMT9,#DH9		
(8)	017500	012746	015311			MOV	#DH9,-(SP)
(7)	017504	012746	012542			MOV	#FMT9,-(SP)
(6)	017510	012746	000002			MOV	#2,-(SP)
(3)	017514	010600				MOV	SP,R0
(4)	017516	104415				TRAP	C\$PNTX
(4)	017520	062706	000006			ADD	#6,SP
4491	017524			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16		
(11)	017524	013746	002324			MOV	AX3.16,-(SP)
(10)	017530	013746	002322			MOV	AX3.15,-(SP)
(9)	017534	013746	002320			MOV	AX2.16,-(SP)
(8)	017540	013746	002316			MOV	AX2.15,-(SP)
(7)	017544	012746	012443			MOV	#FMT6,-(SP)
(6)	017550	012746	000005			MOV	#5,-(SP)
(3)	017554	010600				MOV	SP,R0
(4)	017556	104415				TRAP	C\$PNTX
(4)	017560	062706	000014			ADD	#14,SP
4492	017564			ENDMSG			
(3)	017564						
(3)	017564	104423				L10006:	TRAP
4493							C\$MSG
4494							
4495							
4496							
4497							
4498	017566			BGNMSG	ERR6		
(3)	017566						
4499	017566			PRINTB	#FMT10,SUBRPC	ERR6::	
(8)	017566	013746	002336			MOV	SUBRPC,-(SP)
(7)	017572	012746	012547			MOV	#FMT10,-(SP)
(6)	017576	012746	000002			MOV	#2,-(SP)
(3)	017602	010600				MOV	SP,R0
(4)	017604	104414				TRAP	C\$PNTB
(4)	017606	062706	000006			ADD	#6,SP
4500	017612			PRINTB	#FMT1,#ADDRES,MPCSR		
(9)	017612	013746	002436			MOV	MPCSR,-(SP)
(8)	017616	012746	036506			MOV	#ADDRES,-(SP)
(7)	017622	012746	012304			MOV	#FMT1,-(SP)
(6)	017626	012746	000003			MOV	#3,-(SP)
(3)	017632	010600				MOV	SP,R0
(4)	017634	104414				TRAP	C\$PNTB
(4)	017636	062706	000010			ADD	#10,SP
4501	017642			PRINTB	#FMT2		
(7)	017642	012746	012314			MOV	#FMT2,-(SP)
(6)	017646	012746	000001			MOV	#1,-(SP)
(3)	017652	010600				MOV	SP,R0
(4)	017654	104414				TRAP	C\$PNTB
(4)	017656	062706	000004			ADD	#4,SP
4502	017662			PRINTB	#FMT8,TMP1,TMP0		
(9)	017662	013746	002516			MOV	TMP0,-(SP)
(8)	017666	013746	002520			MOV	TMP1,-(SP)
(7)	017672	012746	012506			MOV	#FMT8,-(SP)
(6)	017676	012746	000003			MOV	#3,-(SP)
(3)	017702	010600				MOV	SP,R0

(4)	017704	104414							
(4)	017706	062706	000010					TRAP	CSPNTB
4503	017712			PRINTX	#FMT4,#DH2,#DH3			ADD	#10,SP
(9)	017712	012746	015112					MOV	#DH3,-(SP)
(8)	017716	012746	015063					MOV	#DH2,-(SP)
(7)	017722	012746	012400					MOV	#FMT4,-(SP)
(6)	017726	012746	000003					MOV	#3,-(SP)
(3)	017732	010600						MOV	SP,R0
(4)	017734	104415						TRAP	CSPNTX
(4)	017736	062706	000010					ADD	#10,SP
4504	017742			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13				
(11)	017742	013746	002274					MOV	LUR13,-(SP)
(10)	017746	013746	002272					MOV	LUR12,-(SP)
(9)	017752	013746	002270					MOV	LUR11,-(SP)
(8)	017756	013746	002266					MOV	LUR10,-(SP)
(7)	017762	012746	012413					MOV	#FMT5,-(SP)
(6)	017766	012746	000005					MOV	#5,-(SP)
(3)	017772	010600						MOV	SP,R0
(4)	017774	104415						TRAP	CSPNTX
(4)	017776	062706	000014					ADD	#14,SP
4505	020002			PRINTX	#FMT9,#DH4				
(8)	020002	012746	015150					MOV	#DH4,-(SP)
(7)	020006	012746	012542					MOV	#FMT9,-(SP)
(6)	020012	012746	000002					MOV	#2,-(SP)
(3)	020016	010600						MOV	SP,R0
(4)	020020	104415						TRAP	CSPNTX
(4)	020022	062706	000006					ADD	#6,SP
4506	020026			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17				
(11)	020026	013746	002304					MOV	LUR17,-(SP)
(10)	020032	013746	002302					MOV	LUR16,-(SP)
(9)	020036	013746	002300					MOV	LUR15,-(SP)
(8)	020042	013746	002276					MOV	LUR14,-(SP)
(7)	020046	012746	012443					MOV	#FMT6,-(SP)
(6)	020052	012746	000005					MOV	#5,-(SP)
(3)	020056	010600						MOV	SP,R0
(4)	020060	104415						TRAP	CSPNTX
(4)	020062	062706	000014					ADD	#14,SP
4507	020066			PRINTX	#FMT4,#DH7,#DH8				
(9)	020066	012746	015252					MOV	#DH8,-(SP)
(8)	020072	012746	015220					MOV	#DH7,-(SP)
(7)	020076	012746	012400					MOV	#FMT4,-(SP)
(6)	020102	012746	000003					MOV	#3,-(SP)
(3)	020106	010600						MOV	SP,R0
(4)	020110	104415						TRAP	CSPNTX
(4)	020112	062706	000010					ADD	#10,SP
4508	020116			PRINTX	#FMT5,AX0.15,AX0.16,AX1.15,AX1.16				
(11)	020116	013746	002314					MOV	AX1.16,-(SP)
(10)	020122	013746	002312					MOV	AX1.15,-(SP)
(9)	020126	013746	002310					MOV	AX0.16,-(SP)
(8)	020132	013746	002306					MOV	AX0.15,-(SP)
(7)	020136	012746	012413					MOV	#FMT5,-(SP)
(6)	020142	012746	000005					MOV	#5,-(SP)
(3)	020146	010600						MOV	SP,R0
(4)	020150	104415						TRAP	CSPNTX
(4)	020152	062706	000014					ADD	#14,SP
4509	020156			PRINTX	#FMT9,#DH9				

(8)	020156	012746	015311		MOV	#DH9,-(SP)
(7)	020162	012746	012542		MOV	#FMT9,-(SP)
(6)	020166	012746	000002		MOV	#2,-(SP)
(3)	020172	010600			MOV	SP,R0
(4)	020174	104415			TRAP	C\$PNTX
(4)	020176	062706	000006		ADD	#6,SP
4510	020202			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
(11)	020202	013746	002324		MOV	AX3.16,-(SP)
(10)	020206	013746	002322		MOV	AX3.15,-(SP)
(9)	020212	013746	002320		MOV	AX2.16,-(SP)
(8)	020216	013746	002316		MOV	AX2.15,-(SP)
(7)	020222	012746	012443		MOV	#FMT6,-(SP)
(6)	020226	012746	000005		MOV	#5,-(SP)
(3)	020232	010600			MOV	SP,R0
(4)	020234	104415			TRAP	C\$PNTX
(4)	020236	062706	000014		ADD	#14,SP
4511	020242			ENDMSG		
(3)	020242					
(3)	020242	104423			L10007:	TRAP C\$MSG
4512						
4513						
4514						
4515						
4516						
4517	020244			BGNMSG	ERR7	
(3)	020244					
4518	020244			PRINTB	#FMT1,#ADDRES,MPCSR	ERR7::
(9)	020244	013746	002436		MOV	MPCSR,-(SP)
(8)	020250	012746	036506		MOV	#ADDRES,-(SP)
(7)	020254	012746	012304		MOV	#FMT1,-(SP)
(6)	020260	012746	000003		MOV	#3,-(SP)
(3)	020264	010600			MOV	SP,R0
(4)	020266	104414			TRAP	C\$PNTB
(4)	020270	062706	000010		ADD	#10,SP
4519	020274			PRINTB	#FMT2	
(7)	020274	012746	012314		MOV	#FMT2,-(SP)
(6)	020300	012746	000001		MOV	#1,-(SP)
(3)	020304	010600			MOV	SP,R0
(4)	020306	104414			TRAP	C\$PNTB
(4)	020310	062706	000004		ADD	#4,SP
4520	020314			PRINTB	#FMT7,#DH1,REGNUM	
(9)	020314	013746	002364		MOV	REGNUM,-(SP)
(8)	020320	012746	015041		MOV	#DH1,-(SP)
(7)	020324	012746	012476		MOV	#FMT7,-(SP)
(6)	020330	012746	000003		MOV	#3,-(SP)
(3)	020334	010600			MOV	SP,R0
(4)	020336	104414			TRAP	C\$PNTB
(4)	020340	062706	000010		ADD	#10,SP
4521	020344			PRINTX	#FMT4,#DH2,#DH3	
(9)	020344	012746	015112		MOV	#DH3,-(SP)
(8)	020350	012746	015063		MOV	#DH2,-(SP)
(7)	020354	012746	012400		MOV	#FMT4,-(SP)
(6)	020360	012746	000003		MOV	#3,-(SP)
(3)	020364	010600			MOV	SP,R0
(4)	020366	104415			TRAP	C\$PNTX
(4)	020370	062706	000010		ADD	#10,SP

4522	020374			PRINTX #FMT5,LUR10,LUR11,LUR12,LUR13	
(11)	020374	013746	002274		MOV LUR13,-(SP)
(10)	020400	013746	002272		MOV LUR12,-(SP)
(9)	020404	013746	002270		MOV LUR11,-(SP)
(8)	020410	013746	002266		MOV LUR10,-(SP)
(7)	020414	012746	012413		MOV #FMT5,-(SP)
(6)	020420	012746	000005		MOV #5,-(SP)
(3)	020424	010600			MOV SP,R0
(4)	020426	104415			TRAP C\$PNTX
(4)	020430	062706	000014		ADD #14,SP
4523	020434			PRINTX #FMT9,#DH4	
(8)	020434	012746	015150		MOV #DH4,-(SP)
(7)	020440	012746	012542		MOV #FMT9,-(SP)
(6)	020444	012746	000002		MOV #2,-(SP)
(3)	020450	010600			MOV SP,R0
(4)	020452	104415			TRAP C\$PNTX
(4)	020454	062706	000006		ADD #6,SP
4524	020460			PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17	
(11)	020460	013746	002304		MOV LUR17,-(SP)
(10)	020464	013746	002302		MOV LUR16,-(SP)
(9)	020470	013746	002300		MOV LUR15,-(SP)
(8)	020474	013746	002276		MOV LUR14,-(SP)
(7)	020500	012746	012443		MOV #FMT6,-(SP)
(6)	020504	012746	000005		MOV #5,-(SP)
(3)	020510	010600			MOV SP,R0
(4)	020512	104415			TRAP C\$PNTX
(4)	020514	062706	000014		ADD #14,SP
4525	020520			PRINTX #FMT4,#DH7,#DH8	
(9)	020520	012746	015252		MOV #DH8,-(SP)
(8)	020524	012746	015220		MOV #DH7,-(SP)
(7)	020530	012746	012400		MOV #FMT4,-(SP)
(6)	020534	012746	000003		MOV #3,-(SP)
(3)	020540	010600			MOV SP,R0
(4)	020542	104415			TRAP C\$PNTX
(4)	020544	062706	000010		ADD #10,SP
4526	020550			PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16	
(11)	020550	013746	002314		MOV AX1.16,-(SP)
(10)	020554	013746	002312		MOV AX1.15,-(SP)
(9)	020560	013746	002310		MOV AX0.16,-(SP)
(8)	020564	013746	002306		MOV AX0.15,-(SP)
(7)	020570	012746	012413		MOV #FMT5,-(SP)
(6)	020574	012746	000005		MOV #5,-(SP)
(3)	020600	010600			MOV SP,R0
(4)	020602	104415			TRAP C\$PNTX
(4)	020604	062706	000014		ADD #14,SP
4527	020610			PRINTX #FMT9,#DH9	
(8)	020610	012746	015311		MOV #DH9,-(SP)
(7)	020614	012746	012542		MOV #FMT9,-(SP)
(6)	020620	012746	000002		MOV #2,-(SP)
(3)	020624	010600			MOV SP,R0
(4)	020626	104415			TRAP C\$PNTX
(4)	020630	062706	000006		ADD #6,SP
4528	020634			PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16	
(11)	020634	013746	002324		MOV AX3.16,-(SP)
(10)	020640	013746	002322		MOV AX3.15,-(SP)
(9)	020644	013746	002320		MOV AX2.16,-(SP)

(8) 020650 013746 002316
(7) 020654 012746 012443
(6) 020660 012746 000005
(3) 020664 010600
(4) 020666 104415
(4) 020670 062706 000014
4529 020674
(3) 020674
(3) 020674 104423
4530
4531
4532
4533
4534
4535 020676
(3) 020676
4536 020676
(8) 020676 013746 002336
(7) 020702 012746 012547
(6) 020706 012746 000002
(3) 020712 010600
(4) 020714 104414
(4) 020716 062706 000006
4537 020722
(9) 020722 013746 002436
(8) 020726 012746 036506
(7) 020732 012746 012304
(6) 020736 012746 000003
(3) 020742 010600
(4) 020744 104414
(4) 020746 062706 000010
4538 020752
(7) 020752 012746 012314
(6) 020756 012746 000001
(3) 020762 010600
(4) 020764 104414
(4) 020766 062706 000004
4539 020772
(9) 020772 013746 002364
(8) 020776 012746 015041
(7) 021002 012746 012476
(6) 021006 012746 000003
(3) 021012 010600
(4) 021014 104414
(4) 021016 062706 000010
4540 021022
(9) 021022 013746 002372
(8) 021026 013746 002370
(7) 021032 012746 012336
(6) 021036 012746 000003
(3) 021042 010600
(4) 021044 104414
(4) 021046 062706 000010
4541 021052
(9) 021052 012746 015112
(8) 021056 012746 015063

ENDMSG

BGNMSG ERR8

PRINTB #FMT10,SUBRPC

PRINTB #FMT1,#ADDRES,MPCSR

PRINTB #FMT2

PRINTB #FMT7,#DH1,REGNUM

PRINTB #FMT3,GOODAT,BADDAT

PRINTX #FMT4,#DH2,#DH3

MOV AX2.15,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

L10010:

TRAP C\$MSG

ERR8::

MOV SUBRPC,-(SP)
MOV #FMT10,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #6,SP

MOV MPCSR,-(SP)
MOV #ADDRES,-(SP)
MOV #FMT1,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #FMT2,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #4,SP

MOV REGNUM,-(SP)
MOV #DH1,-(SP)
MOV #FMT7,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV BADDAT,-(SP)
MOV GOODAT,-(SP)
MOV #FMT3,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

MOV #DH3,-(SP)
MOV #DH2,-(SP)

(7)	021062	012746	012400						
(6)	021066	012746	000003					MOV	#FMT4,-(SP)
(3)	021072	010600						MOV	#3,-(SP)
(4)	021074	104415						MOV	SP,R0
(4)	021076	062706	000010					TRAP	C\$PNTX
4542	021102			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13			ADD	#10,SP
(11)	021102	013746	002274					MOV	LUR13,-(SP)
(10)	021106	013746	002272					MOV	LUR12,-(SP)
(9)	021112	013746	002270					MOV	LUR11,-(SP)
(8)	021116	013746	002266					MOV	LUR10,-(SP)
(7)	021122	012746	012413					MOV	#FMT5,-(SP)
(6)	021126	012746	000005					MOV	#5,-(SP)
(3)	021132	010600						MOV	SP,R0
(4)	021134	104415						TRAP	C\$PNTX
(4)	021136	062706	000014					ADD	#14,SP
4543	021142			PRINTX	#FMT9,#DH4				
(8)	021142	012746	015150					MOV	#DH4,-(SP)
(7)	021146	012746	012542					MOV	#FMT9,-(SP)
(6)	021152	012746	000002					MOV	#2,-(SP)
(3)	021156	010600						MOV	SP,R0
(4)	021160	104415						TRAP	C\$PNTX
(4)	021162	062706	000006					ADD	#6,SP
4544	021166			PRINTX	#FMT6,LUR14,LUR15,LUR16,LUR17				
(11)	021166	013746	002304					MOV	LUR17,-(SP)
(10)	021172	013746	002302					MOV	LUR16,-(SP)
(9)	021176	013746	002300					MOV	LUR15,-(SP)
(8)	021202	013746	002276					MOV	LUR14,-(SP)
(7)	021206	012746	012443					MOV	#FMT6,-(SP)
(6)	021212	012746	000005					MOV	#5,-(SP)
(3)	021216	010600						MOV	SP,R0
(4)	021220	104415						TRAP	C\$PNTX
(4)	021222	062706	000014					ADD	#14,SP
4545	021226			PRINTX	#FMT4,#DH7,#DH8				
(9)	021226	012746	015252					MOV	#DH8,-(SP)
(8)	021232	012746	015220					MOV	#DH7,-(SP)
(7)	021236	012746	012400					MOV	#FMT4,-(SP)
(6)	021242	012746	000003					MOV	#3,-(SP)
(3)	021246	010600						MOV	SP,R0
(4)	021250	104415						TRAP	C\$PNTX
(4)	021252	062706	000010					ADD	#10,SP
4546	021256			PRINTX	#FMT5,AX0.15,AX0.16,AX1.15,AX1.16				
(11)	021256	013746	002314					MOV	AX1.16,-(SP)
(10)	021262	013746	002312					MOV	AX1.15,-(SP)
(9)	021266	013746	002310					MOV	AX0.16,-(SP)
(8)	021272	013746	002306					MOV	AX0.15,-(SP)
(7)	021276	012746	012413					MOV	#FMT5,-(SP)
(6)	021302	012746	000005					MOV	#5,-(SP)
(3)	021306	010600						MOV	SP,R0
(4)	021310	104415						TRAP	C\$PNTX
(4)	021312	062706	000014					ADD	#14,SP
4547	021316			PRINTX	#FMT9,#DH9				
(8)	021316	012746	015311					MOV	#DH9,-(SP)
(7)	021322	012746	012542					MOV	#FMT9,-(SP)
(6)	021326	012746	000002					MOV	#2,-(SP)
(3)	021332	010600						MOV	SP,R0
(4)	021334	104415						TRAP	C\$PNTX

(4)	021336	062706	000006						
4548	021342			PRINTX	#FMT6,AX2.15,AX2.16,AX3.15,AX3.16		ADD	#6,SP	
(11)	021342	013746	002324				MOV	AX3.16,-(SP)	
(10)	021346	013746	002322				MOV	AX3.15,-(SP)	
(9)	021352	013746	002320				MOV	AX2.16,-(SP)	
(8)	021356	013746	002316				MOV	AX2.15,-(SP)	
(7)	021362	012746	012443				MOV	#FMT6,-(SP)	
(6)	021366	012746	000005				MOV	#5,-(SP)	
(3)	021372	010600					MOV	SP,R0	
(4)	021374	104415					TRAP	C\$PNTX	
(4)	021376	062706	000014				ADD	#14,SP	
4549	021402			ENDMSG					
(3)	021402								
(3)	021402	104423					L10011:	TRAP	C\$MSG
4550									
4551									
4552									
4553									
4554									
4555	021404			BGNMSG	ERR10				
(3)	021404								
4556	021404			PRINTB	#FMT1,#ADDRES,MPCSR				ERR10::
(9)	021404	013746	002436				MOV	MPCSR,-(SP)	
(8)	021410	012746	036506				MOV	#ADDRES,-(SP)	
(7)	021414	012746	012304				MOV	#FMT1,-(SP)	
(6)	021420	012746	000003				MOV	#3,-(SP)	
(3)	021424	010600					MOV	SP,R0	
(4)	021426	104414					TRAP	C\$PNTB	
(4)	021430	062706	000010				ADD	#10,SP	
4557	021434			PRINTB	#FMT2				
(7)	021434	012746	012314				MOV	#FMT2,-(SP)	
(6)	021440	012746	000001				MOV	#1,-(SP)	
(3)	021444	010600					MOV	SP,R0	
(4)	021446	104414					TRAP	C\$PNTB	
(4)	021450	062706	000004				ADD	#4,SP	
4558	021454			PRINTB	#FMT8,TMP1,TMP0				
(9)	021454	013746	002516				MOV	TMP0,-(SP)	
(8)	021460	013746	002520				MOV	TMP1,-(SP)	
(7)	021464	012746	012506				MOV	#FMT8,-(SP)	
(6)	021470	012746	000003				MOV	#3,-(SP)	
(3)	021474	010600					MOV	SP,R0	
(4)	021476	104414					TRAP	C\$PNTB	
(4)	021500	062706	000010				ADD	#10,SP	
4559	021504			PRINTX	#FMT4,#DH2,#DH3				
(9)	021504	012746	015112				MOV	#DH3,-(SP)	
(8)	021510	012746	015063				MOV	#DH2,-(SP)	
(7)	021514	012746	012400				MOV	#FMT4,-(SP)	
(6)	021520	012746	000003				MOV	#3,-(SP)	
(3)	021524	010600					MOV	SP,R0	
(4)	021526	104415					TRAP	C\$PNTX	
(4)	021530	062706	000010				ADD	#10,SP	
4560	021534			PRINTX	#FMT5,LUR10,LUR11,LUR12,LUR13				
(11)	021534	013746	002274				MOV	LUR13,-(SP)	
(10)	021540	013746	002272				MOV	LUR12,-(SP)	
(9)	021544	013746	002270				MOV	LUR11,-(SP)	
(8)	021550	013746	002266				MOV	LUR10,-(SP)	

(7) 021554 012746 012413
(6) 021560 012746 000005
(3) 021564 010600
(4) 021566 104415
(4) 021570 062706 000014
4561 021574
(8) 021574 012746 015150
(7) 021600 012746 012542
(6) 021604 012746 000002
(3) 021610 010600
(4) 021612 104415
(4) 021614 062706 000006
4562 021620
(11) 021620 013746 002304
(10) 021624 013746 002302
(9) 021630 013746 002300
(8) 021634 013746 002276
(7) 021640 012746 012443
(6) 021644 012746 000005
(3) 021650 010600
(4) 021652 104415
(4) 021654 062706 000014
4563 021660
(9) 021660 012746 015252
(8) 021664 012746 015220
(7) 021670 012746 012400
(6) 021674 012746 000003
(3) 021700 010600
(4) 021702 104415
(4) 021704 062706 000010
4564 021710
(11) 021710 013746 002314
(10) 021714 013746 002312
(9) 021720 013746 002310
(8) 021724 013746 002306
(7) 021730 012746 012413
(6) 021734 012746 000005
(3) 021740 010600
(4) 021742 104415
(4) 021744 062706 000014
4565 021750
(8) 021750 012746 015311
(7) 021754 012746 012542
(6) 021760 012746 000002
(3) 021764 010600
(4) 021766 104415
(4) 021770 062706 000006
4566 021774
(11) 021774 013746 002324
(10) 022000 013746 002322
(9) 022004 013746 002320
(8) 022010 013746 002316
(7) 022014 012746 012443
(6) 022020 012746 000005
(3) 022024 010600
(4) 022026 104415

PRINTX #FMT9,#DH4

PRINTX #FMT6,LUR14,LUR15,LUR16,LUR17

PRINTX #FMT4,#DH7,#DH8

PRINTX #FMT5,AX0.15,AX0.16,AX1.15,AX1.16

PRINTX #FMT9,#DH9

PRINTX #FMT6,AX2.15,AX2.16,AX3.15,AX3.16

MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH4,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV LUR17,-(SP)
MOV LUR16,-(SP)
MOV LUR15,-(SP)
MOV LUR14,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH8,-(SP)
MOV #DH7,-(SP)
MOV #FMT4,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #10,SP

MOV AX1.16,-(SP)
MOV AX1.15,-(SP)
MOV AX0.16,-(SP)
MOV AX0.15,-(SP)
MOV #FMT5,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #14,SP

MOV #DH9,-(SP)
MOV #FMT9,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTX
ADD #6,SP

MOV AX3.16,-(SP)
MOV AX3.15,-(SP)
MOV AX2.16,-(SP)
MOV AX2.15,-(SP)
MOV #FMT6,-(SP)
MOV #5,-(SP)
MOV SP,R0
TRAP C\$PNTX

(4) 022030 062706 000014

4567 022034

ENDMSG

ADD #14,SP

(3) 022034

L10012:

TRAP C\$MSG

(3) 022034 104423

4568

4569

4570

4571

4572

4573

4574

4575

4576

4577

4578

4579

4580

4581

4582

4583 022036

BGNRPT

LSRPT::

(3) 022036

4584

4585 022036

ENDRPT

L10013:

TRAP CSRPT

(3) 022036

(3) 022036 104425

4586

4587

4588

4589

4590

4591

4592

4593

4594

4595

4596

4597

4598

4599

4600 022040

BGNPROT

LSPROT::

(3) 022040

4601 022040 177777

.WORD -1 ;DON'T CHK CSR ADRS

4602 022042 177777

.WORD -1 ;DON'T CHK MASSBUS UNIT NO.

4603 022044 177777

.WORD -1 ;DON'T CHK DRIVE NO.

4604 022046

ENDPROT

4605

4606

4607

4608

4609

.SBTTL REPORT CODING SECTION

:/ THE REPORT CODING SECTION CONTAINS THE
:/ 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.

.SBTTL LOAD DEVICE PROTECTION TABLE

:/ THIS TABLE IDENTIFIES THE LOAD DEVICE TO THE SUPERVISOR, SO THAT IT CAN BE
:/ PROTECTED FROM TESTING, IF DESIRED.

```

4611
4612
4613
4614
4615
4616
4617
4618 022046
(3) 022046
4619
4620 022046 010637 002332
4621 022052 005037 002336
4622 022056 005037 002416
4623 022062 005037 002420
4624 022066 005037 002410
4625 022072 005037 002424
4626 022076 005737 002376
4627 022102 001007
4628 022104 013737 000004 002404
4629 022112 013737 000006 002406
4630 022120 000406
4631 022122 013737 002404 000004 6$:
4632 022130 013737 002406 000006
4633 022136 012737 000001 002376 9$:
4634
4635 022144
(3) 022144 012700 000040
(3) 022150 104447
4636 022152
(2) 022152 103415
4637
4638 022154
(3) 022154 012700 000037
(3) 022160 104447
4639 022162
(2) 022162 103411
4640
4641 022164
(3) 022164 012700 000035
(3) 022170 104447
4642 022172
(2) 022172 103411
4643
4644 022174
(3) 022174 012700 000036
(3) 022200 104447
4645 022202
(2) 022202 103556
4646 022204 000416
4647 022206
4648 022206 005037 002402
4649
4650 022212 005037 002426
4651 022216
4652 022216 012737 177777 002330
4653 022224 005237 002400

```

```

.SBTTL INITIALIZE SECTION
:////////////////////
:// THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
:// AT THE BEGINNING OF THE TEST SEQUENCE ON THE NEXT UNIT.
:////////////////////

BGNINIT
LSINIT::

MOV SP,PSTACK ;SAVE BASE-LEVEL STACK POINTER
CLR SUBRPC ;CLEAR SUBR CALL PC
CLR DISILO ;CLEAR CURRENT STATE OF DISSI
CLR CHPTYP ;CLEAR USYRT CHIP TYPE INDICATOR
CLR ERROR1 ;CLEAR ERROR FLAGS
CLR SAVLEN ;CLEAR CHAR LENGTH FROM SETUP
TST FRSTIM ;SEE IF FIRST TIME THROUGH AFTER LOAD
BNE 6$ ;BR IF NOT
MOV @#4,SAVE4 ;SAVE ERROR TRAP VECTOR
MOV @#6,SAVE6
BR 9$
6$: MOV SAVE4,@#4 ;RESTORE ERROR TRAP VECTOR
MOV SAVE6,@#6
9$: MOV #1,FRSTIM ;MARK FLAG FOR NEXT TIME THROUGH
;SEE IF PROGRAM JUST STARTED, BR IF YES
READEF #EF.START
MOV #EF.START,RO
TRAP CS$REFG
BCS STARST
;SEE IF PROGRAM JUST RESTARTED, BR IF YES
READEF #EF.RESTART
MOV #EF.RESTART,RO
TRAP CS$REFG
BCS STARST
;SEE IF THIS IS A NEW PASS, BR IF YES
READEF #EF.NEW
MOV #EF.NEW,RO
TRAP CS$REFG
BCS NEWST
;SEE IF PROGRAM WAS JUST CONTINUED
READEF #EF.CONTINUE
MOV #EF.CONTINUE,RO
TRAP CS$REFG
BCS ENDIT
BR GETPRM
STARST: CLR STARES ;CLEAR FLAG TO SHOW JUST HAD STA OR RES
;CLEAR DEVICE MAP
CLR DEVMAP
NEWST: MOV #-1,LOGDEV ;RESET LOGICAL DEVICE TO -1
INC FRSPAS ;INCREMENT NO. OF PASSES AFTER LOAD

```

```

4654 022230 005237 002402          INC      STARES          ;INCREMENT NO. OF PASSES SINCE STA OR RES
4655 022234 012737 000001 002430  MOV      #BIT0,DEVPTD    ;INIT DEVICE MAP BIT POINTER
4656                                     ; GET UNIBUS ADDRESS, VECTOR, PRIORITY LEVEL, SWITCH PACKS, TEST
4657                                     ; CONNECTOR INFORMATION FOR THIS LOGICAL DEVICE
4658 022242          GETPRM:
4659 022242 005237 002330          INC      LOGDEV          ;INCREMENT LOGICAL DEVICE NUMBER
4660 022246 023737 002330 002012  CMP      LOGDEV,LSUNIT   ;SEE IF MAXIMUM UNIT NO. EXCEEDED
4661 022254 002360          BGE      NEWST           ;BR IF YES
4662 022256          GPHARD LOGDEV,R1      ;GET P-TABLE POINTER INTO R1
(3) 022256 013700 002330          MOV      LOGDEV,R0
(3) 022262 104442          TRAP    C$GPHRD
(3) 022264 010001          MOV      R0,R1
4663 022266          BCOMPLETE 10$      ;BR IF DEVICE AVAILABLE
(2) 022266 103403          BCS     10$
4664 022270 006337 002430          ASL     DEVPTR          ;SHIFT DEVICE MAP BIT POINTER
4665 022274 000762          BR      GETPRM         ;SKIP THIS DEVICE
4666 022276 053737 002430 002426 10$:  BIS     DEVPTR,DEVMAP   ;SET BIT FOR THIS DEVICE IN DEVICE MAP
4667 022304 006337 002430          ASL     DEVPTR          ;SHIFT DEVICE MAP BIT POINTER
4668 022310 062701 000002          ADD     #2,R1           ;INCREMENT R1 PAST MICROPROCESSOR TYPE
4669 022314 011137 002436          MOV     (R1),MPCSR     ;STORE POINTER TO MICROPROCESSOR CSR'S
4670 022320 011137 002440          MOV     (R1),BSEL1
4671 022324 005237 002440          INC     BSEL1          ;GET POINTER TO BSEL1 (MAINTENANCE REGISTER)
4672 022330 011137 002442          MOV     (R1),SEL4
4673 022334 062737 000004 002442          ADD     #4,SEL4        ;GET POINTER TO SEL4
4674 022342 012137 002444          MOV     (R1)+,SEL6
4675 022346 062737 000006 002444          ADD     #6,SEL6        ;STORE POINTER TO SEL6
4676 022354 011137 002446          MOV     (R1),MPIVEC   ;GET MICROPROCESSOR INPUT INTRPT VECTOR
4677 022360 012137 002450          MOV     (R1)+,MPOVEC
4678 022364 062737 000004 002450          ADD     #4,MPOVEC      ;GET MICROPROCESSOR OUTPUT INTRPT VECTOR
4679 022372 012137 002452          MOV     (R1)+,MPRIOR  ;GET MICROPROCESSOR DEVICE PRIORITY
4680 022376 062701 000002          ADD     #2,R1           ;INCREMENT R1 PAST LU TYPE
4681 022402 012137 002454          MOV     (R1)+,LUSW11   ;GET LU SWITCH PACK #1
4682 022406 012137 002456          MOV     (R1)+,LUSW12   ;GET LU SWITCH PACK #2
4683 022412 012137 002460          MOV     (R1)+,LUSW13   ;GET LU SWITCH PACK #3
4684 022416 012137 002462          MOV     (R1)+,TSTCON   ;GET TEST CONNECTOR INDICATOR
4685 022422 011137 002464          MOV     (R1),BDRATE    ;GET BAUD RATE
4686                                     ;SEE IF MANUAL INTERVENTION DESIRED BETWEEN UNITS FOR INSTALLATION OR REMOVAL
4687                                     ; OF TEST CONNECTORS, BR IF NOT
4688 022426 005737 002256          TST     MIFLAG
4689 022432 001442          BEQ     22$
4690                                     ;SEE IF MANUAL INTERVENTION ALLOWED BY SUPERVISOR
4691 022434          MANUAL
(3) 022434 104450          TRAP    C$MANI
4692                                     ;BR IF ALLOWED
4693 022436          BCOMPLETE 18$
(2) 022436 103412          BCS     18$
4694                                     ;PRINT MSG THAT OPERATOR INTERVENTION IS NOT ALLOWED
4695 022440          PRINTF #FMT16
(7) 022440 012746 022542          MOV     #FMT16,-(SP)
(6) 022444 012746 000001          MOV     #1,-(SP)
(3) 022450 010600          MOV     SP,R0
(4) 022452 104417          TRAP    C$PNTF
(4) 022454 062706 000004          ADD     #4,SP
4696 022460          16$:  BREAK          ;HANG UNTIL ^C TYPED
(3) 022460 104422          TRAP    C$BRK
4697 022462 000776          BR      16$

```

4698	022464				18\$:				
4699					:TYPE	'INSTALL TEST CONNECTOR(S) ON UNIT AT ADRS XXXXXX'			
4700	022464				PRINTF	#FMT17,MPCSR			
(8)	022464	013746	002436				MOV	MPCSR,-(SP)	
(7)	022470	012746	022664				MOV	#FMT17,-(SP)	
(6)	022474	012746	000002				MOV	#2,-(SP)	
(3)	022500	010600					MOV	SP,R0	
(4)	022502	104417					TRAP	C\$PNTF	
(4)	022504	062706	000006				ADD	#6,SP	
4701	022510	005037	002502						
4702	022514				CLR	REG2			
4703					20\$:				
4704	022514				:ASK	OPERATOR TO 'TYPE <Y> <CR> WHEN READY TO PROCEED'			
(3)	022514	104443			GMANIL	TYPEY,REG2,1,NO			
(3)	022516	000404					TRAP	C\$GMAN	
(4)	022520	002502					BR	10000\$	
(5)	022522	000120					.WORD	REG2	
(5)	022524	022753					.WORD	T\$CODE	
(5)	022526	000001					.WORD	TYPEY	
(3)	022530						.WORD	1	
4705	022530	023727	002502	000001			10000\$:		
4706	022536	001366			CMP	REG2,#1			
4707	022540				BNE	20\$			
4708	022540				22\$:				
4709	022540				ENDIT:				
(3)	022540				ENDINIT				
(3)	022540	104411					L10015:		
4710							TRAP	C\$INIT	
4711	022542	047045	040445	040515	FMT16:	.ASCII /%N%AMANUAL INTERVENTION IS NOT ALLOWED!%N/			
	022550	052516	046101	044440					
	022556	052116	051105	042526					
	022564	052116	047511	020116					
	022572	051511	047040	052117					
	022600	040440	046114	053517					
	022606	042105	022441	116					
4712	022613	045	052101	050131		.ASCIZ /%ATYPE CONTROL-C (^C) <CR> TO PROCEED:%N/			
	022620	020105	047503	052116					
	022626	047522	026514	020103					
	022634	057050	024503	036040					
	022642	051103	020076	047524					
	022650	050040	047522	042503					
	022656	042105	022472	000116					
4713	022664	047045	040445	047111	FMT17:	.ASCIZ /%N%AINSTALL TEST CONNECTOR(S) ON UNIT AT ADRS : %06%N/			
	022672	052123	046101	020114					
	022700	042524	052123	041440					
	022706	047117	042516	052103					
	022714	051117	051450	020051					
	022722	047117	052440	044516					
	022730	020124	052101	040440					
	022736	051104	020123	020072					
	022744	022440	033117	047045					
	022752	000							
4714	022753	124	050131	020105	TYPEY:	.ASCIZ /TYPE <Y><CR> WHEN READY TO PROCEED /			
	022760	054474	036076	051103					
	022766	020076	044127	047105					
	022774	051040	040505	054504					

	023002	052040	020117	051120	
	023010	041517	042505	020104	
	023016	000			
4715		023020			.EVEN
4716					
4717					
4718					
4719					
4720					

4722
4723
4724
4725
4726
4727
4728
4729
(3)
4730
4731
(3)
(3)
4732
4733
4734
4735
4736
4737
4738
(3)
(3)
4739
4740
4741
(3)
(3)
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
(3)
4755
4756
4757
(3)
(3)
4758
4759
4760
4761
4762

023020
023020
023020 012700 000340
023024 104441
023026 012737 023050 000004
023034 012737 000340 000006
023042 005777 157370
023046 000405
023050 062706 000004
023054
023054 013700 002330
023060 104451
023062 013737 002404 000004
023070 013737 002406 000006
023076
023076 104461
023100
023100
023100
023100 104412

```
.SBTTL AUTO DROP UNIT SECTION
://////
:// THE AUTO DROP CODING DETERMINES WHETHER OR NOT THE DEVICE WHOSE P-TABLE
:// WAS JUST OBTAINED IS READY FOR TESTING, AND IT IS DROPPED IF NOT READY.
://////

      BGNAUTO
      L$AUTO::

:ESTABLISH PRIORITY = 7
      SETPRI #PRI07

      MOV #6$,@#4 ;SET UP NON-EXISTENT MEMORY ERROR TRAP VECTOR
      TRAP C$SPRI
      MOV #PRI07,@#6
      TST @MPCSR ;ADDRESS SELO
      BR 9$ ;TAKE THIS BRANCH IF DEVICE RESPONDS
:COME HERE IF DEVICE CSR IS NON-EXISTENT
6$: ADD #4,SP ;CLEAN UP THE STACK POINTER
      DODU LOGDEV ;DROP THIS UNIT FROM TESTING

      MOV LOGDEV,R0
      TRAP C$DODU

9$: MOV SAVE4,@#4 ;RESTORE ERROR TRAP VECTOR
      MOV SAVE6,@#6
      ENDAUTO

      L10016:
      TRAP C$AUTO
```

```
.SBTTL CLEANUP CODING SECTION
://////
:// THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
:// AT THE END OF THE TEST SEQUENCE ON A PARTICULAR UNIT.
://////

      BGNCLN
      L$CLEAN::

      ENDCLN

      L10017:
      TRAP C$CLEAN
```



```
4764 .SBTTL DROP UNIT SECTION
4765
4766 :////////////////////
4767 :// THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
4768 :// TO NO LONGER BE TESTED.
4769 :////////////////////
4770
4771 023102          BGNDU
4772 (3) 023102
4773 :ISSUE UNIBUS RESET TO CLEAN UP          L$DU::
4774 (3) 023102 104433          BRESET
4775 :PRINT 'UNIT XX DROPPED'          TRAP    C$RESET
4776 (8) 023104 013746 002330          PRINTF #FMT27,LOGDEV
4777 (7) 023110 012746 023132          MOV     LOGDEV,-(SP)
4778 (6) 023114 012746 000002          MOV     #FMT27,-(SP)
4779 (3) 023120 010600          MOV     #2,-(SP)
4780 (4) 023122 104417          MOV     SP,RO
4781 (4) 023124 062706 000006          TRAP   C$PNTF
4782 4776 023130          ENDDU          ADD     #6,SP
4783 (3) 023130
4784 (3) 023130 104453          L10020:
4785 4777          TRAP    C$DU
4786 4778 023132 047045 040445 047125 FMT27: .ASCII /%N%AUNIT %D2%A DROPPED%N/
4787 023140 052111 022440 031104
4788 023146 040445 042040 047522
4789 023154 050120 042105 047045
4790 023162 000
4791 4779 023164          .EVEN
4792
4793 .SBTTL ADD UNIT SECTION
4794
4795 :////////////////////
4796 :// THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
4797 :// TO BE (A) TESTED FOR THE FIRST TIME, OR (B) RESUMED IN TESTING. IF
4798 :// 'EF.AUNIT' IS SET, THE UNIT WILL BE TESTED AS A NEW UNIT.
4799 :////////////////////
4800
4801 4793 023164          B$NAU
4802 (3) 023164
4803 4794 023164          ENDAU          L$AU::
4804 (3) 023164
4805 (3) 023164 104452          L10021:
4806 4795          TRAP    C$AU
4807 4796
4808 4797
4809 4798
4810 4799
4811 4800
```

4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850

.SBTTL HARDWARE TESTS

```

*****
.SBTTL      TEST 1 - BIT STUFFING TEST
*
* THE DEVICE IS ENABLED FOR TRANSMIT AND RECEIVE, AND A MESSAGE IS
* INITIATED IN BIT MODE . TWO LEADING FLAGS ARE SENT,
* FOLLOWED BY ALL SIXTEEN CHARS IN DATA PATTERN S. THIS PATTERN
* CONSISTS OF CHARACTERS WHICH REQUIRE NO BIT STUFFING AND CHARACTERS
* WHICH REQUIRE BIT STUFFING INDIVIDUALLY AND IN COMBINATION WITH
* ADJACENT CHARACTERS. ALL 16 CHARACTERS ARE READ AND COMPARED
* BY THE RECEIVER.
* PATTERN S = 000,017,036,074,170,360,037,076,174,370,077,176,374,
*             177,376,377
*****
BGNTST

```

023166
(3) 023166
023166 012737 023310 002346
023174 004737 005240
023200 000000
023202 000300
023204 004737 010574
023210 002642
023212 000020
023214 012737 001000 002412
023222 004737 004674
023226 004737 004674
023232 004737 004754
023236 000300
023240 012701 002642
023244 112137 023254 6\$:
023250 004737 007314 8\$:
023254 000000
023256 000000
023260 020127 002661
023264 103767
023266 111137 023304
023272 052737 001000 023304
023300 004737 007314
023304 000000 12\$:
023306 000000
023310 004737 003276 24\$:
023314
(3) 023314
(3) 023314 104401

```

T1::
MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
000
CRC2!CRC1 ;BIT MODE, NO ERR DETECTION
JSR PC,LDBYTS ;LOAD PAT S INTO TX SILO
PATS
16.
MOV #TXEOM,TXWORD
JSR PC,LDTXSI ;LOAD 2 EOM'S INTO TX SILO
JSR PC,LDTXSI
JSR PC,STPLU ;CLK MORE THAN ENTIRE MSG
192.
MOV #PATS,R1 ;INIT PAT S POINTER
6$: MOVB (R1)+,8$
JSR PC,CKDATA ;CHK A RCV'D CHAR
8$: .WORD 0
0
CMP R1,#PATS+15. ;SEE IF 15 CHARS CHECKED YET
BLO 6$ ;BR IF NOT YET
MOVB (R1),12$
BIS #RXEBL,12$ ;GET SET TO CHK EBLK = 1
JSR PC,CKDATA ;CHK LAST CHAR AND EBLK = 1
12$: .WORD 0
0
24$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST
L10022: TRAP C$ETST

```

4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
4904
4905
(4)

023316
(3) 023316
023316 012737 023664 002346

023324 004737 005240
023330 000226
023332 000311
023334 004737 010746
023340 000001
023342 000100
023344 004737 007000
023350 000030
023352 004737 004754
023356 000730
023360 004737 010746
023364 000001
023366 000074
023370 004737 004754
023374 000100
023376 012701 000100
023402 004737 007314
023406 000001
023410 000000
023412 005301
023414 001372
023416 004737 007314
023422 004001
023424 000010
023426 004737 007314
023432 004001
023434 000010
023436 012737 000012 002364
023444 012737 000200 002352
023452 004737 003450
023456 004737 006720
023462 132737 000010 002415
023470 001407
023472 004737 004226
023476
023476 104455

```
*****  
:SBTTL TEST 2 - RCV OVERRUN ERROR SET AND CLEAR TEST  
:  
* IN THIS TEST, A RCV OVERRUN ERROR IS FORCED IN EACH OF 2 SUBTESTS.  
* IN THE FIRST, A MESSAGE IS INITIATED, 64 001 CHARS ARE SENT, AND THE  
* RECEIVER IS NOT SERVICED IN RESPONSE TO THE USYRT RCV FLAG, WHICH CAUSES RCV  
* OVERRUN TO SET. THEN, A CHECK IS MADE TO INSURE THAT OVRR IS NOT  
* CLEARED BY THE LINE UNIT READING THE USYRT STATUS.  
* THEN, IC IS SET TO CLEAR THE ERROR, AND THIS IS VERIFIED.  
:  
* IN THE SECOND SUBTEST, RCV OVRUN IS FORCED AGAIN, AND A MASTER CLEAR  
* IS ISSUED TO CLEAR THE ERROR, AND THIS IS VERIFIED.  
:*****  
BGNTST  
T2::  
MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS  
-----  
CAUSE OVRR, SET IC TO CLEAR IT  
-----  
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S  
SYNCH  
CRC2:CRC1!STRIP!DDCMP ;DDCMP, NO ERR DET  
JSR PC,LODSIL ;LOAD 64 001 CHARS INTO TX SILO  
001  
64.  
JSR PC,RCV1ST ;CLOCK UNTIL FIRST DATA CHAR RCV'D  
24.  
JSR PC,STPLU ;CLOCK UNTIL 59 MORE RCV'D  
472.  
JSR PC,LODSIL ;LOAD 60 MORE INTO TX SILO  
001  
60.  
JSR PC,STPLU ;CLK 8 MORE TIMES TO FORCE UNDERRUN  
64.  
MOV #64.,R1 ;READ AND CHK 64 CHARS FROM RCV SILO  
6$: JSR PC,CKDATA  
001  
0  
DEC R1  
BNE 6$  
JSR PC,CKDATA ;READ CHAR, CHK OVRR = 1  
4001  
8.  
JSR PC,CKDATA ;READ CHAR, CHK OVRR STILL = 1  
4001  
8.  
MOV #12,REGNUM ;SET REG NO. = 12  
MOV #IC,WRIBYT  
JSR PC,WRITLU ;SET IC TO CLEAR RCVR  
JSR PC,RDRXSI ;READ RCV SILO  
BITB #OVRR,RXWORD+1 ;CHK FOR OVRR CLEARED  
BEQ 8$ ;BR IF OVRR CLEARED  
JSR PC,GETALL ;GET REGS FOR PRINTOUT  
:REPORT OVRR NOT CLEARED  
ERRDF 41,EM41,ERR7
```

TRAP C\$ERDF

(5) 023500 000051
(5) 023502 014060
(5) 023504 020244
4906 023506 000466
4907 023510

.WORD 41
.WORD EM41
.WORD ERR7

BR 24\$
8\$:

: CAUSE OVRN, SET MST CLR TO CLEAR IT

4911 023510 004737 005240
4912 023514 000226
4913 023516 000311
4914 023520 004737 010746
4915 023524 000001
4916 023526 000100
4917 023530 004737 007000
4918 023534 000030
4919 023536 004737 004754
4920 023542 000730
4921 023544 004737 010746
4922 023550 000001
4923 023552 000074
4924 023554 004737 004754
4925 023560 000100
4926 023562 012701 000100
4927 023566 004737 007314
4928 023572 000001
4929 023574 000000
4930 023576 005301
4931 023600 001372
4932 023602 004737 007314
4933 023606 004001
4934 023610 000010
4935 023612 004737 007314
4936 023616 004001
4937 023620 000010
4938 023622 012737 000012 002364
4939 023630 004737 003276
4940 023634 004737 006720
4941 023640 132737 000010 002415
4942 023646 001406
4943 023650 004737 004226

JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
SYNCH
CRC2!CRC1!STRIP!DDCMP ;DDCMP, NO ERR DET
JSR PC,LODSIL ;LOAD 64 001 CHARS INTO TX SILO
001
64.
JSR PC,RCV1ST ;CLOCK UNTIL FIRST DATA CHAR RCV'D
24.
JSR PC,STPLU ;CLOCK UNTIL 59 MORE RCV'D
472.
JSR PC,LODSIL ;LOAD 60 MORE INTO TX SILO
001
60.
JSR PC,STPLU ;CLK 8 MORE TIMES TO FORCE UNDERRUN
64.
MOV #64,R1 ;READ AND CHK 64 CHARS FROM RCV SILO
9\$: JSR PC,CKDATA
001
0
DEC R1
BNE 9\$
JSR PC,CKDATA ;READ CHAR, CHK OVRN = 1
4001
8.
JSR PC,CKDATA ;READ CHAR, CHK OVRN STILL 1
4001
8.
MOV #12,REGNUM ;SET REG NO. = 12
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,RDRXSI ;READ RCV SILO
BITB #OVRN,RXWORD+1 ;CHK FOR OVRN CLEARED
BEQ 24\$;BR IF OVRN CLEARED
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT OVRN NOT CLEARED
ERRDF 41,EM41,ERR7

4945 023654
(4) 023654 104455
(5) 023656 000051
(5) 023660 014060
(5) 023662 020244
4946 023664 004737 003276
4947 023670
(3) 023670
(3) 023670 104401
4948
4949
4950
4951
4952

24\$: JSR PC,MSTCLR ;ISSUE CLEAN UP MST CLR
ENDTST

TRAP C\$ERDF
.WORD 41
.WORD EM41
.WORD ERR7

L10023:
TRAP C\$ETST

4953
 4954
 4955
 4956
 4957
 4958
 4959
 4960
 4961
 4962
 4963
 4964
 4965
 4966
 4967
 4968
 4969
 4970
 4971
 4972

```

:*****
:SBTTL      TEST 3 - ABORT SEQUENCE TEST
:*
:* SET BIT MODE, CRC, AND ENABLE THE DEVICE FOR
:* TRANSMIT AND RECEIVE. SEND 2 FLAGS AND 4 DATA CHARS (001).
:* AS THE FIRST DATA CHAR IS BEING TRANSMITTED,
:* SET THE ABORT BIT (REG 11).
:* ON THE RECEIVER SIDE, CHECK FOR RECEPTION OF THE FIRST DATA CHAR
:* AND THEN THE SETTING OF RAB AND REOM A CHAR TIME LATER.
:* ALSO, CHECK FOR IACT = 0. THEN, CHECK THAT RAB
:* IS CLEARED BY READING THE USYRT STATUS, TRANSMITTING A NEW MSG,
:* RECEIVING THE FIRST CHAR (003) AND CHECKING FOR RAB CLEARED.
:*
:* REPEAT THE ABOVE SEQUENCE, SET IC, AND CHECK THAT
:* THIS CLEARS RAB.
:*
:* REPEAT THE ABOVE SEQUENCE, ISSUE MASTER CLEAR, CHECK THAT THIS
:* CLEARS RAB.
:*****
  
```

4973 023672
 (3) 023672
 4974 023672 012737 024222 002346

```

BGNTST
:-----T3:-----
:MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
:-----
:CAUSE ABORT, START NEW MSG TO CLEAR IT
:-----
  
```

4975
 4976
 4977
 4978 023700 004737 005240
 4979 023704 000000
 4980 023706 000000
 4981 023710 004737 010516
 4982 023714 002732
 4983 023716 000014
 4984 023720 004737 007000
 4985 023724 000060
 4986 023726 004737 007314
 4987 023732 000001
 4988 023734 000010
 4989 023736 004737 007314
 4990 023742 003001
 4991 023744 000000
 4992 023746 004737 006260
 4993 023752 000000
 4994 023754 004737 007000
 4995 023760 000060
 4996 023762 004737 007314
 4997 023766 000003
 4998 023770 000000
 4999

```

:-----
:JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
:000
:000 ;BIT MODE, CRC
:JSR PC,LODMSG ;LOAD MSG INTO TX SILO
:MSG3
:12.
:JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
:48.
:JSR PC,CKDATA ;CHK CHR = 001, CLK ABORT CHAR
:001
:8.
:JSR PC,CKDATA ;CHK FOR RAB, EBLK, AND 001 CHAR
:RXABT!RXEBL!001
:0
:JSR PC,IACTIV ;CHK FOR IACT = 0
:0
:JSR PC,RCV1ST ;CLK AND RCV NEW MSG
:48.
:JSR PC,CKDATA ;CHK CHAR = 003
:003
:0
:-----
  
```

5000
 5001
 5002 023772 004737 005240
 5003 023776 000000
 5004 024000 000000
 5005 024002 004737 010516
 5006 024006 002732
 5007 024010 000014

```

:-----
:CAUSE ABORT, SET IC TO CLEAR IT
:-----
:JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
:000
:000 ;BIT MODE, CRC
:JSR PC,LODMSG ;LOAD MSG INTO TX SILO
:MSG3
:12.
  
```

```
5008 024012 004737 007000 JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
5009 024016 000060 48.
5010 024020 004737 007314 JSR PC,CKDATA ;CHK CHR = 001, CLK ABORT CHAR
5011 024024 000001 001
5012 024026 000010 8.
5013 024030 004737 007314 JSR PC,CKDATA ;CHK FOR RAB, EBLK, AND 001 CHAR
5014 024034 003001 RXABT!RXEBL!001
5015 024036 000000 0
5016 024040 012737 000012 002364 MOV #12,REGNUM ;SET REG NO. = 12
5017 024046 012737 000200 002352 MOV #IC,WRIBYT
5018 024054 004737 003450 JSR PC,WRITLU ;SET IC TO CLEAR RCVR
5019 024060 004737 006720 JSR PC,RDRXSI ;READ RCV SILO
5020 024064 132737 000004 002415 BITB #RAB,RXWORD+1 ;CHK FOR RAB CLEARED
5021 024072 001407 BEQ 8$ ;BR IF RAB CLEARED
5022 024074 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
5023 ;REPORT RAB NOT CLEARED
5024 024100 ERRDF 39,EM39,ERR7
(4) 024100 104455 TRAP C$ERDF
(5) 024102 000047 .WORD 39
(5) 024104 014024 .WORD EM39
(5) 024106 020244 .WORD ERR7
5025 024110 000444 BR 24$
5026 024112 8$:
5027 -----
5028 ;CAUSE ABORT, ISSUE MASTER CLEAR TO CLEAR IT
5029 -----
5030 024112 004737 005240 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
5031 024116 000000 000
5032 024120 000000 000 ;BIT MODE, CRC
5033 024122 004737 010516 JSR PC,LODMSG ;LOAD MSG INTO TX SILO
5034 024126 002732 MSG3
5035 024130 000014 12.
5036 024132 004737 007000 JSR PC,RCV1ST ;CLK AND RCV FIRST DATA CHAR
5037 024136 000060 48.
5038 024140 004737 007314 JSR PC,CKDATA ;CHK CHR = 001, CLK ABORT CHAR
5039 024144 000001 001
5040 024146 000010 8.
5041 024150 004737 007314 JSR PC,CKDATA ;CHK FOR RAB, EBLK, AND 001 CHAR
5042 024154 003001 RXABT!RXEBL!001
5043 024156 000000 0
5044 024160 012737 000012 002364 MOV #12,REGNUM ;SET REG NO. = 12
5045 024166 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
5046 024172 004737 006720 JSR PC,RDRXSI ;READ RCV SILO
5047 024176 132737 000004 002415 BITB #RAB,RXWORD+1 ;CLK FOR RAB CLEARED
5048 024204 001406 BEQ 24$ ;BR IF RAB CLEARED
5049 024206 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
5050 ;REPORT RAB NOT CLEARED
5051 ERRDF 39,EM39,ERR7
(4) 024212 104455 TRAP C$ERDF
(5) 024214 000047 .WORD 39
(5) 024216 014024 .WORD EM39
(5) 024220 020244 .WORD ERR7
5052 024222 004737 003276 24$: JSR PC,MSTCLR ;ISSUE MST CLR TO CLEAN UP
5053 024226 ENDTST
(3) 024226 L10024:
(3) 024226 104401 TRAP C$ETST
```

5054
 5055
 5056
 5057
 5058
 5059
 5060
 5061
 5062
 5063
 5064
 5065
 5066
 5067
 (3)
 5068
 5069
 5070
 5071
 5072
 5073
 5074
 5075
 5076
 5077
 5078
 5079
 5080
 5081
 5082
 5083
 5084
 (3)
 (3)
 5085
 5086
 5087
 5088
 5089
 5090
 5091
 5092
 5093
 5094
 5095
 5096
 5097
 5098
 5099
 (3)
 5100
 5101
 5102
 5103
 5104
 5105

024230
 024230
 024230 012737 024304 002346
 024236 004737 005240
 024242 000000
 024244 000040
 024246 004737 010516
 024252 002732
 024254 000005
 024256 004737 007000
 024262 000060
 024264 004737 007314
 024270 000001
 024272 000010
 024274 004737 007314
 024300 001001
 024302 000000
 024304 004737 003276
 024310
 024310
 024310 104401

```

:*****
:SBTTL      TEST 4 - ABORT AND IDLE FLAGS TEST
:*
:* TRANSMIT THE SAME ABORT SEQUENCE AS IN THE PREVIOUS TEST, BUT
:* WITH THE IDLE BIT SET. CHECK THAT FLAGS ARE SENT AND RECEIVED
:* (NOT ABORT CHARACTERS) BY VERIFYING THAT RAB DOES
:* NOT SET, AND THAT THE MESSAGE TERMINATES WITH EBLK = 1.
:*****
  
```

```

BGNTST
                                T4::
MOV      #24$,RETADR           ;SET TEST EXIT ADRS FOR ERRORS
JSR      PC,INITRN            ;MST CLR, LOAD 2 SOM'S
000
IDLE                                           ;BIT MODE, NO ERROR DET
JSR      PC,LODMSG            ;LOAD MSG INTO TX SILO
MSG3
5
JSR      PC,RCV1ST           ;CLK AND RCV FIRST DATA CHAR
48.
JSR      PC,CKDATA           ;CHK CHR = 001, CLK FLAG CHAR
001
8.
JSR      PC,CKDATA           ;CHK RAB = 0, EBLK = 1
RXEBL!001
0
24$: JSR      PC,MSTCLR       ;ISSUE MASTER CLEAR
ENDTST
                                L10025:
                                TRAP      C$ETST
  
```

```

:*****
:SBTTL      TEST 5 - TRANSMITTER UNDERRUN ERROR, IDLE ABORT CHARS, BIT MODE
:*
:* A MESSAGE IS INITIATED IN BIT MODE, 4 001 CHARS ARE SENT, AND THE TRANSMITTER
:* IS NOT SERVICED IN RESPONSE TO THE LAST TX FLAG, WHICH CAUSES TX
:* UNDERRUN ERROR TO SET. ON THE RECEIVER SIDE, CHECK THAT THE DATA
:* CHAR IS RECEIVED, AND THAT 8 CYCLES LATER THE RAB BIT SETS, AND
:* THE DEVICE IDLES ABORT CHARACTERS.
:*****
  
```

```

BGNTST
                                T5::
MOV      #24$,RETADR           ;SET TEST EXIT ADRS FOR ERRORS
JSR      PC,INITRN            ;MST CLR, LOAD 2 SOM'S
000
000
MOV      #TXEN,DISILO         ;SET TX ENB TO KEEP RTS HIGH
JSR      PC,LODSIL           ;LOAD 4 001 CHARS INTO TX SILO
  
```

5106	024342	000001		001		
5107	024344	000004		4		
5108	024346	004737	007000	JSR	PC,RCV1ST	;CLK AND RCV FIRST CHAR
5109	024352	000060		48.		
5110	024354	004737	007314	JSR	PC,CKDATA	;CHK DATA = 001, CLOCK ABORT CHAR
5111	024360	000001		001		
5112	024362	000011		9.		
5113	024364	004737	007314	JSR	PC,CKDATA	;CHK FOR RAB, EBLK, AND 001 CHAR
5114	024370	003001		RXABT!RXEBL!001		
5115	024372	000000		0		
5116	024374	004737	003276	24\$: JSR	PC,MSTCLR	;ISSUE MASTER CLEAR
5117	024400			ENDTST		
(3)	024400					L10026: TRAP C\$ETST
(3)	024400	104401				

```
*****  
:SBTTL TEST 6 - RECEIVER DISABLE TEST  
:  
:*  
:* TRANSMIT AND RECEIVE ARE ENABLED IN BIT MODE, AND 2 FLAGS  
:* ARE SENT, FOLLOWED BY 5 252 DATA CHARS. AFTER THE SECOND DATA CHAR HAS BEGUN  
:* TO BE RECEIVED, IC IS SET.  
:* THEN, THE PROGRAM CHECKS THAT A USYRT RCV FLAG IS NOT GENERATED, AND  
:* THE RECEIVER DATA PATH STOPS OPERATING IN THE MIDDLE OF THE CHAR.  
:*****  
BGNTST
```

5132	024402					
(3)	024402					
5133	024402	012737	024570	002346	MOV #24\$,RETADR	;SET TEST EXIT ADRS FOR ERRORS T6::
5134	024410	004737	005240		JSR PC,INITRN	;MST CLR, LOAD 2 SOM'S
5135	024414	000000			000	
5136	024416	000000			000	;BIT MODE, CRC
5137	024420	004737	010746		JSR PC,LODSIL	;LOAD 5 252 CHARS
5138	024424	000252			252	
5139	024426	000005			5	
5140	024430	004737	007000		JSR PC,RCV1ST	;CLK AND RCV FIRST DATA CHAR
5141	024434	000060			48.	
5142	024436	004737	004754		JSR PC,STPLU	;CLK TO MIDDLE OF 2ND CHAR
5143	024442	000004			4	
5144	024444	012737	000012	002364	MOV #12,REGNUM	;SET REG NO. = 12
5145	024452	012737	000200	002352	MOV #IC,WRIBYT	
5146	024460	004737	003450		JSR PC,WRITLU	;SET IC IN REG 12
5147	024464	004737	006260		JSR PC,IACTIV	;CHK IACT = 0
5148	024470	000000			0	
5149	024472	004737	005774		JSR PC,ISIRDY	;CHK ICIR = 1, IRDY = 0
5150	024476	000001			1	
5151	024500	005037	002370		CLR GOODAT	;SET EXPECTED DATA = 0
5152	024504	005037	002372		CLR BADDAT	
5153	024510	004737	006720		JSR PC,RDRXSI	;READ RCV SILO
5154	024514	105737	002414		TSTB RXWORD	;SEE IF SILO BITS 0-7 = 000
5155	024520	001404			BEQ 9\$;BR IF YES
5156	024522	012737	000010	002364	MOV #10,REGNUM	;SET REG NO. = 10
5157	024530	000406			BR 12\$	
5158	024532	105737	002415	9\$:	TSTB RXWORD+1	;SEE IF SILO BITS 8-11 = 000


```
5159 024536 001414          BEQ      24$          ;BR IF YES
5160 024540 012737 000012 002364    MOV      #12,REGNUM  ;SET REG NO. = 12
5161 024546 113737 002414 002372    12$:    MOVB     RXWORD,BADDAT ;GET ACTUAL DATA
5162 024554 004737 004226          JSR      PC,GETALL   ;GET REGS FOR PRINTOUT
5163                                     ;REPORT RCV SILO NOT CLEARED BY IC
5164 024560          ERRDF  46,EM46,ERR2
(4) 024560 104455                                     TRAP    C$ERDF
(5) 024562 000056                                     .WORD  46
(5) 024564 014215                                     .WORD  EM46
(5) 024566 015406                                     .WORD  ERR2
5165 024570 004737 003276    24$:    JSR      PC,MSTCLR   ;ISSUE MASTER CLEAR TO CLEAN UP
5166 024574          ENDTST
(3) 024574                                     L10027:
(3) 024574 104401                                     TRAP    C$ETST
5167
5168
5169
5170
5171
5172                                     ;*****
5173 .SBTTL      TEST 7 - ASSEMBLED BIT COUNT TEST
5174 ;*
5175 ;* THE FOLLOWING SEQUENCE IS PERFORMED 8 TIMES, EACH TIME USING A
5176 ;* DIFFERENT TX CHAR LENGTH (FROM 2 TO 8 BITS) AND A RCV CHAR LENGTH = 8
5177 ;* BITS :
5178 ;* A MESSAGE IS INITIATED IN BIT MODE, NO CRC.
5179 ;* 2 FLAGS ARE SENT, FOLLOWED BY 3 000 DATA CHARACTERS AND A
5180 ;* TERMINATING FLAG. AFTER THE RECEIVER HAS RECEIVED THE MESSAGE, AX0-16
5181 ;* IS READ TO RETRIEVE THE ASSEMBLED BIT COUNT. THIS COUNT IS CHECKED TO INSURE
5182 ;* THAT IT IS CORRECT FOR THE TX CHAR LENGTH USED IN THAT TRANSMISSION.
5183 ;* *****
5184 BGNTST
5185 024576
(3) 024576
5186 024576 012737 025334 002346    MOV      #24$,RETADR  ;SET TEST EXIT ADRS FOR ERRORS
5187 024604 004737 005240          JSR      PC,INITRN   ;FIND OUT WHICH USYRT CHIP
5188 024610 000000          0
5189 024612 000000          0
5190 024614 012701 000100          MOV      #TXLEN1,R1  ;SET INITIAL TX LENGTH TO 2 BITS
5191 024620 004737 003276    6$:    JSR      PC,MSTCLR   ;ISSUE MASTER CLEAR
5192 024624 004737 010324          JSR      PC,SETUP    ;PROGRAM THE USYRT
5193 024630 000000          000
5194 024632 000300          CRC2!CRC1
5195 024634 000000          000
5196 024636 000000          000
5197 024640 012737 000014 002364    MCV      #14,REGNUM  ;SET REG NO. = 14
5198 024646 012737 000140 002352    MOV      #TXEN!DISSI,WRIBYT
5199 024654 004737 003450          JSR      PC,WRITLU   ;SET TXEN AND DISSI IN REG 14 -
5200 024660 012737 000140 002416    MOV      #TXEN!DISSI,DISILO ;SET DISABLE SILO FLAG
5201 024666 012737 000012 002364    MOV      #12,REGNUM  ;SET LU REG NO. = 12
5202 024674 112737 000040 002352    MOVB     #LULP,WRIBYT
5203 024702 004737 003450          JSR      PC,WRITLU   ;SET LULP IN REG 12
5204 024706 012737 000002 002366    MOV      #2,AXNUM    ;SET AX BYTE NO. = 2
5205 024714 105037 002360          CLRB     WAX15
5206 024720 112737 000001 002362    MOVB     #TSOM,WAX16
5207 024726 004737 004012          JSR      PC,WRITAX   ;LOAD SOM CHAR
```

```

5208 024732 005004          CLR      R4          ;INIT COUNTER
5209 024734 012737 000011 002364  ---MOV--- #11,REGNUM  ;SET REG NO. = 11
5210 024742 004737 004754      JSR      PC,STPLU   ;CLOCK LU FOR A CYCLE
5211 024746 000001          1
5212 024750 004737 003372          JSR      PC,READLU  ;READ REG 11
5213 024754 132737 000100 002350  BITB    #OACT,REDBYT ;SEE IF OACT SET YET
5214 024762 001014          BNE     10$        ;BR IF OACT SET
5215 024764 005204          INC     R4         ;INCR COUNTER
5216 024766 020427 000004          CMP     R4,#4     ;SEE IF COUNT TOO BIG
5217 024772 002763          BLT    7$         ;BR IF NOT
5218 024774 004737 004226      JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
5219
5220 025000          ;REPORT OACT NOT SET
          ERRDF      11,EM11,ERR7
(4) 025000 104455
(5) 025002 000013
(5) 025004 013420
(5) 025006 020244
5221 025010 000137 025334          JMP     24$
5222 025014 004737 004012 10$:  JSR      PC,WRITAX  ;LOAD ANOTHER SOM CHAR
5223 025020 004737 004754      JSR      PC,STPLU   ;CLK FIRST FLAG
5224 025024 000010          8.
5225 025026 105037 002362  CLRB   WAX16
5226 025032 004737 004012      JSR      PC,WRITAX  ;LOAD FIRST 000 CHAR
5227 025036 004737 004754      JSR      PC,STPLU   ;CLK SECOND FLAG
5228 025042 000010          8.
5229 025044 004737 004012      JSR      PC,WRITAX  ;LOAD SECOND 000 CHAR
5230 025050 004737 004754      JSR      PC,STPLU   ;CLK FIRST 000 CHAR
5231 025054 000010          8.
5232 025056 012737 000006 002366  MOV     #6,AXNUM   ;SET AX BYTE NO. FOR AX 3
5233 025064 010137 002362      MOV     R1,WAX16   ;GET TX CHAR LENGTH
5234 025070 004737 004012      JSR      PC,WRITAX  ;SET TX CHAR LENGTH
5235 025074 012737 000002 002366  MOV     #2,AXNUM   ;SET AX BYTE NO. = 2
5236 025102 105037 002362  CLRB   WAX16
5237 025106 005737 002420      TST    CHPTYP     ;SEE IF SIG USYRT
5238 025112 001403          BEQ    5$         ;BR IF YES
5239 025114 112737 000002 002362  MOVB   #TEOM,WAX16 ;SET TEOM WITH LAST DATA CHAR
5240 025122 004737 004012 5$:  JSR      PC,WRITAX  ;LOAD 3RD 000 CHAR
5241 025126 004737 004754      JSR      PC,STPLU   ;CLK 2ND 000 CHAR
5242 025132 000010          8.
5243 025134 005737 002420      TST    CHPTYP     ;SEE IF SIG USYRT
5244 025140 001005          BNE   16$        ;BR IF NOT
5245 025142 112737 000002 002362  MOVB   #TEOM,WAX16
5246 025150 004737 004012      JSR      PC,WRITAX  ;LOAD AN EOM CHAR
5247 025154 012737 000001 002366  MOV     #1,AXNUM   ;SET AX BYTE NO. = 1
5248 025162 005003          CLR    R3
5249 025164 004737 003624 12$:  JSR      PC,READAX  ;READ AX0
5250 025170 132737 000002 002356  BITB   #REOM,RAX16 ;CHK FOR REOM = 1
5251 025176 001016          BNE   14$        ;BR IF YES
5252 025200 004737 004754      JSR      PC,STPLU   ;CLOCK LU FOR A CYCLE
5253 025204 000001          1
5254 025206 005203          INC    R3         ;INCR COUNT
5255 025210 020327 000023          CMP    R3,#19     ;SEE IF COUNT TOO BIG
5256 025214 002763          BLT   12$        ;BR IF NOT
5257 025216 004737 004226      JSR      PC,GETALL  ;GET REGS FOR PRINTOUT
5258
5259 025222          ;REPORT REOM NOT SET
          ERRDF      31,EM31,ERR10
    
```

```

TRAP  C$ERDF
.WORD 11
.WORD EM11
.WORD ERR7
    
```

```
(4) 025222 104455
(5) 025224 000037
(5) 025226 013667
(5) 025230 021404
5260 025232 000440
5261 025234 013702 002356
5262 025240 042702 000217
5263 025244 006102
5264 025246 120201
5265 025250 001421
5266 025252 010137 002370
5267 025256 006237 002370
5268 025262 152737 000002 002370
5269 025270 013737 002356 002372
5270 025276 004737 004226
5271
5272 025302
(4) 025302 104455
(5) 025304 000057
(5) 025306 014250
(5) 025310 015714
5273 025312 000410
5274 025314 005701
5275 025316 001406
5276 025320 062701 000040
5277 025324 042701 000400
5278 025330 000137 024620
5279 025334 005037 002416
5280 025340 004737 003276
5281 025344
(3) 025344
(3) 025344 104401
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
```

```
TRAP C$ERDF
.WORD 31
.WORD EM31
.WORD ERR10

14$: BR 24$
MOV RAX16,R2 ;GET AX0-16 CONTENTS
BIC #217,R2 ;MASK OFF ALL BUT ASSEMB BIT COUNT
ROL R2
CMPB R2,R1 ;CHK FOR CORRECT ASSEMB BIT COUNT
BEQ 9$ ;BR IF MATCH
MOV R1,GOODAT ;SET EXPECTED DATA
ASR GOODAT
BISB #REOM,GOODAT
MOV RAX16,BADDAT ;SET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT ASSEMB BIT COUNT INCORRECT
ERRDF 47,EM47,ERR3

TRAP C$ERDF
.WORD 47
.WORD EM47
.WORD ERR3

9$: BR 24$
TST R1 ;SEE IF ALL DONE YET
BEQ 24$ ;BR IF YES
ADD #TXLENO,R1 ;INCR TX LENGTH
BIC #400,R1 ;MASK OFF OVERFLOW IF 8 BITS
JMP 6$ ;PROCEED

24$: CLR DISILO
JSR PC,MSTCLR ;ISSUE MASTER CLR TO CLEAN UP

ENDTST

L10030: TRAP C$ETST
```

```
*****
.SBTTL TEST 8 - SECONDARY STATION ADDRESS BIT TEST
:*
:* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
:* BIT MODE, AND THE SECA BIT (REG 17) IS SET.
:* 2 FLAGS ARE SENT, FOLLOWED BY 252, 000, AND A TERMINATING FLAG.
:* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
:* RECEIVED.
:*
:* NEXT, THE SECONDARY STATION ADDRESS BITS IN AX2-15 ARE LOADED
:* WITH THE FIRST WORD OF DATA PATTERN T. 2 FLAGS ARE SENT,
:* FOLLOWED BY THE FIRST WORD OF DATA PATTERN T, A 000 CHAR,
:* AND A TERMINATING FLAG.
:* THEN, THE RCV'D DATA IS CHECKED TO MAKE SURE THAT THE SEC STATION
:* ADDRESS IS RCV'D AS THE FIRST DATA CHAR, FOLLOWED BY U00.
:*
:* THEN, THE SUBTEST IS REPEATED FOR EACH OF THE REMAINING WORDS OF
:* DATA PATTERN T.
:* PATTERN T = 000,125,252,176,177
```

```
5306
5307 025346
(3) 025346
5308
5309
5310
5311 025346
(3) 025346
(3) 025346 104402
5312 025350 012737 025446 002346 MOV #3$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS TRAP C$BSUB
5313 025356 004737 005240 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
5314 025362 000000 000000 SECA ;SEC ADRS = 000
5315 025364 000020 010746 JSR PC,LOADSIL ;BIT MODE, CRC, SEC ADRS MODE
5316 025366 004737 010746 JSR PC,LOADSIL ;LOAD 252 INTO TX SILO
5317 025372 000252 000001 1
5318 025374 000001 010746 JSR PC,LOADSIL ;LOAD 000 DATA INTO TX SILO
5319 025376 004737 010746 JSR PC,LOADSIL ;LOAD 2 EOM'S INTO TX SILO
5320 025402 000000 000001 1
5321 025404 000001 010746 JSR PC,LOADSIL ;LOAD 2 EOM'S INTO TX SILO
5322 025406 004737 010746 TXEOM
5323 025412 001000 2
5324 025414 000002 004754 JSR PC,STPLU ;TRANSMIT THE MSG
5325 025416 004737 004754
5326
5327 025422 000060 48.
5328 025424 004737 006260 JSR PC,IACTIV ;CHK IACT = 0
5329 025430 000000 0
5330 025432 004737 004754 JSR PC,STPLU ;CLOCK 8 MORE CYCLES
5331 025436 000010 8.
5332 025440 004737 006260 JSR PC,IACTIV ;CHK IACT = 0
5333 025444 000000 0
5334 025446
5335 025446
(3) 025446
(3) 025446 104403
5336
5337
5338
5339 025450 012701 002662
5340 025454
5341 025454
(3) 025454
(3) 025454 104402
5342 025456 012737 025566 002346 MOV #24$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS TRAP C$BSUB
5343 025464 111137 025504 MOV (R1),5$ ;SET SEC ADRS
5344 025470 111137 025514 MOV (R1),6$ ;SET FIRST DATA CHAR
5345 025474 111137 025552 MOV (R1),9$ ;SET EXPECTED DATA CHAR
5346 025500 004737 005240 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
5347 025504 000000 0
5348 025506 000020 010746 JSR PC,LOADSIL ;BIT MODE, CRC, SEC ADRS MODE
5349 025510 004737 010746 JSR PC,LOADSIL ;LOAD 1ST DATA CHAR INTO TX SILO
5350 025514 000000 0
5351 025516 000001 1
5352 025520 004737 010746 JSR PC,LOADSIL ;LOAD A 000 CHAR INTO TX SILO
5353 025524 000000 0
5354 025526 000001 1
```

```
5355 025530 004737 010746 JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO
5356 025534 001000 TXEOM
5357 025536 000002 2
5358 025540 004737 007000 JSR PC,RCV1ST ;CLOCK AND RCV FIRST DATA CHAR
5359 025544 000060 48.
5360 025546 004737 007314 JSR PC,CKDATA ;CHK FOR CORRECT RCV'D SEC STA ADRS
5361 025552 000000 9$. WORD 0
5362 025554 000011 9.
5363 025556 004737 007314 JSR PC,CKDATA ;READ AND CHK 000 CHAR, EBLK=1, BCC=0
5364 025562 101000 CRCCHK!RXEBL!000
5365 025564 000000 0
5366 025566 24$:
5367 025566 ENDSUB
(3) 025566 L10033:
(3) 025566 104403 TRAP C$ESUB
5368 025570 005201 INC R1 ;INCR PATTERN POINTER
5369 025572 020127 002667 CMP R1,#ENDPAT ;SEE IF ALL DONE YET
5370 025576 103726 BLO A11 ;BR IF NO
5371 025600 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
5372 025604 004737 003276 ENDTST
(3) 025604 L10031:
(3) 025604 104401 TRAP C$ETST
5373
5374
5375
5376
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386
5387
5388
5389
5390
5391 025606
(3) 025606
5392
5393
5394
5395 025606
(3) 025606
(3) 025606 104402 T9.:
5396 025610 012737 025706 002346 MOV #3$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS TRAP C$BSUB
5397 025616 004737 005240 JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
5398 025622 000000 000 ;SEC ADRS = 000
5399 025624 000020 SECA ;BIT MODE, CRC, SEC ADRS MODE
5400 025626 004737 010746 JSR PC,LODSIL ;LOAD 377 INTO TX SILO
5401 025632 000377 377
5402 025634 000001 1
5403 025636 004737 010746 JSR PC,LODSIL ;LOAD 125 DATA INTO TX SILO
```

```
*****
.SBTTL TEST 9 - RDALL (ALL PARTIES ADDRESS) BIT TEST
*
* FIRST, A MASTER CLEAR IS ISSUED. THEN, THE LINE UNIT IS PLACED IN
* BIT MODE, AND THE SECA BIT IS SET.
* 2 FLAGS ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
* THEN, THE RECEIVER IS CHECKED TO MAKE SURE THAT NO DATA CHARS ARE
* RECEIVED.
* NEXT, THE RDALL BIT IN REG 17 IS SET TO 1. 2 FLAGS
* ARE SENT, FOLLOWED BY 377, 125, AND A TERMINATING FLAG.
* THEN, THE REC'D DATA IS CHECKED TO MAKE SURE THAT 377
* IS REC'D AS THE FIRST DATA CHAR, FOLLOWED BY 125.
*****
BGNTST
```

```
-----
: SET SEC ADR = 000, SEND ADR = 377, WITH RDALL = 0
-----
```

```
BGNSUB
T9.1: TRAP C$BSUB
```

```
5404 025642 000125          125
5405 025644 000001          1
5406 025646 004737 010746   JSR    PC,LODSIL      ;LOAD 2 EOM'S INTO TX SILO
5407 025652 001000          TXEOM
5408 025654 000002          2
5409 025656 004737 004754   JSR    PC,STPLU      ;TRANSMIT THE MSG
5410 025662 000060          48.
5411 025664 004737 006260   JSR    PC,IACTIV     ;CHK IACT = 0
5412 025670 000000          0
5413 025672 004737 004754   JSR    PC,STPLU      ;CLOCK 8 MORE CYCLES
5414 025676 000010          8.
5415 025700 004737 006260   JSR    PC,IACTIV     ;CHK IACT = 0
5416 025704 000000          0
5417 025706
5418 025706          3$:
(3) 025706
(3) 025706 104403          ENDSUB
5419
5420
5421
5422 025710          :-----
(3) 025710          ; SET SEC ADR = 000, SEND ADR = 377, WITH RDALL = 1
(3) 025710 104402          :-----
          BGNSUB
          T9.2:
5423 025712 012737 026006 002346   MOV    #24$,RETADR   ;SET SUBTEST EXIT ADRS FOR ERRORS
5424 025720 004737 005240          JSR    PC,INITRN     ;MST CLR, LOAD 2 SOM'S
5425 025724 000000          000                ;SEC ADRS = 000
5426 025726 000024          SECA!RDALL          ;BIT MODE, CRC, SEC ADRS MODE, RDALL
5427 025730 004737 010746   JSR    PC,LODSIL     ;LOAD 1ST DATA CHAR INTO TX SILO
5428 025734 000377          377
5429 025736 000001          1
5430 025740 004737 010746   JSR    PC,LODSIL     ;LOAD A 125 CHAR INTO TX SILO
5431 025744 000125          125
5432 025746 000001          1
5433 025750 004737 010746   JSR    PC,LODSIL     ;LOAD 2 EOM'S INTO TX SILO
5434 025754 001000          TXEOM
5435 025756 000002          2
5436 025760 004737 007000   JSR    PC,RCV1ST     ;CLOCK AND RCV FIRST DATA CHAR
5437 025764 000060          48.
5438 025766 004737 007314   JSR    PC,CKDATA     ;CHK FOR 377 CHAR RCV'D
5439 025772 000377          377
5440 025774 000010          8.
5441 025776 004737 007314   JSR    PC,CKDATA     ;READ AND CHK 125 CHAR, EBLK=1, BCC=0
5442 026002 101125          CRCCHK!RXEBL!125
5443 026004 000000          0
5444 026006
5445 026006          24$:
(3) 026006          ENDSUB
(3) 026006 104403          L10036:
5446 026010          ENDTST          TRAP    C$ESUB
(3) 026010          L10034:
(3) 026010 104401          TRAP    C$ETST
5447
5448
5449
5450
5451
```

5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
(3)
(3)
5498
5499
5500
5501
5502
5503
5504

026012
(3) 026012
026012 012737 026140 002346
026020 004737 005240
026024 000226
026026 000011
026030 004737 010516
026034 002722
026036 000004
026040 004737 004754
026044 100021
026046 004737 007212
026052 000011
026054 000001
026056 004737 004754
026062 000001
026064 004737 007212
026070 000011
026072 000002
026074 004737 004754
026100 000003
026102 004737 007212
026106 000011
026110 000004
026112 004737 007000
026116 000014
026120 004737 007314
026124 000032
026126 000010
026130 004737 007314
026134 000377
026136 000000
026140 004737 003276
026144
(3) 026144
(3) 026144 104401

```
*****  
:SBTTL TEST 10 - INSERT ERROR (IERR) BIT TEST - CHAR MODE, NO CRC  
:  
:* THE LINE UNIT IS PLACED IN DDCMP MODE WITH NO ERROR DETECTION, AND 2  
:* SYNCHS, A 000 CHAR, A 377 CHAR, AND 2 SYNCHS ARE LOADED INTO THE  
:* TRANSMITTER SILO. THEN, THE LU IS CLOCKED UNTIL THE 2ND BIT OF THE 000  
:* CHAR IS ABOUT TO BE SENT AND THE IERR BIT IS SET FOR A CLOCK TIME AND  
:* THEN CLEARED. IN THE SAME WAY, IERR IS SET PRIOR TO THE SENDING OF THE 4TH  
:* AND 5TH BITS OF THE 000 CHAR. IT IS ALSO SET FOR THE SENDING OF THE FIRST  
:* 4 BITS OF THE 377 CHAR. THE PROGRAM READS THE FIRST RCV'D CHAR FROM AX0  
:* AND CHECKS IT TO BE 032, AND READS THE 2ND CHAR AND CHECKS IT TO BE 377.  
:* THEN, A MASTER CLEAR IS DONE TO IDLE THE DEVICE.  
:*****  
BGNST
```

```
T10::  
MOV #15$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS  
JSR PC,INITRN ;LOAD 2 SOM'S, CLOCK THEM INTO USYRT  
SYNCH  
STRIP!DDCMP  
JSR PC,LODMSG ;LOAD MSG INTO TX SILO  
MSG2+4  
4  
JSR PC,STPLU ;CLOCK LU UNTIL 2ND BIT OF 000 CHAR  
CHPCHK!17.  
JSR PC,STPERR ;SET IERR 1 CYCLF  
STRIP!DDCMP  
1  
JSR PC,STPLU ;CLOCK LU UNTIL 4TH BIT OF 000 CHAR  
1  
JSR PC,STPERR ;SET IERR FOR 2 CYCLES  
STRIP!DDCMP  
2  
JSR PC,STPLU ;CLOCK LU UNTIL 1ST BIT OF 377 CHAR  
3  
JSR PC,STPERR ;SET IERR FOR 4 CYCLES  
STRIP!DDCMP  
4  
JSR PC,RCV1ST ;CLOCK AND RCV 1ST CHAR  
12.  
JSR PC,CKDATA ;READ AND COMPARE 1ST CHAR TO 032  
032  
8.  
JSR PC,CKDATA ;READ AND COMPARE 2ND CHAR TO 377  
377  
0  
JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
```

L10037: TRAP CSETST

```
*****  
:SBTTL TEST 11 - SWITCH PACK PRINTOUT AND TEST
```

5505
 5506
 5507
 5508
 5509
 5510
 5511
 5512
 5513
 5514
 5515
 5516
 5517
 5518
 5519
 5520
 5521
 5522
 5523
 5524 026146
 (3) 026146
 5525
 5526
 5527
 5528 026146
 (3) 026146
 (3) 026146 104402
 5529 026150 004737 003276
 5530 026154 012737 000011 002364
 5531 026162 004737 003372
 5532 026166 142737 000321 002350
 5533 026174 023727 002400 000001
 5534 026202 001403
 5535 026204 005737 002260
 5536 026210 001424
 5537 026212
 5538
 5539 026212
 (8) 026212 013746 002436
 (7) 026216 012746 013062
 (6) 026222 012746 000002
 (3) 026226 010600
 (4) 026230 104417
 (4) 026232 062706 000006
 5540
 5541 026236
 (8) 026236 013746 002350
 (7) 026242 012746 012637
 (6) 026246 012746 000002
 (3) 026252 010600
 (4) 026254 104417
 (4) 026256 062706 000006
 5542 026262 005737 002262
 5543 026266 001420
 5544 026270 123737 002350 002454
 5545 026276 001414

```

: *
: * - READ AND PRINT SWITCH PACK #1 :
: * THE PROGRAM READS REG 11 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
: * (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
: * THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
: * SWITCHES ARE IN BITS 1,2,3,5.
: *
: * - READ AND PRINT SWITCH PACK #2 :
: * THE PROGRAM READS REG 15 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
: * (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
: * THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
: * SWITCHES ARE IN BITS 0-7.
: *
: * - READ AND PRINT SWITCH PACK #3 :
: * THE PROGRAM READS REG 16 AND PRINTS THE CONTENTS. IF DESIRED BY THE OPERATOR,
: * (AS INDICATED IN THE SOFTWARE P-TABLE), THE PROGRAM WILL THEN COMPARE IT TO
: * THE EXPECTED VALUE (GIVEN IN THE HARDWARE P-TABLE). THE
: * SWITCHES ARE IN BITS 0-7.
: *****

```

BGNST

T11::

 : READ AND PRINT SWITCH PACK #1, IF DESIRED

BGNSUB

T11.1:

TRAP C\$BSUB

```

JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #11,REGNUM ;SET LU REG NO. = 11
JSR PC,READLU ;READ LU REG 11
BICB #321,REDBYT ;MASK OFF NON-SWITCH BITS
CMP FRSPAS,#1 ;SEE IF IN FIRST PASS AFTER LOAD
BEQ 3$ ;BR IF YES
TST PRNFLG ;SEE IF PRINTOUT IS ALLOWED ON ALL PASSES
BEQ 4$ ;BR IF NOT

```

3\$:

```

;PRINT DEVICE ADDRESS
PRINTF #FMT18,MPCSR

```

```

MOV MPCSR,-(SP)
MOV #FMT18,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #6,SP

```

```

;PRINT SWITCH PACK #1
PRINTF #FMT12,REDBYT

```

```

MOV REDBYT,-(SP)
MOV #FMT12,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #6,SP

```

4\$:

```

TST SWIFLG ;SEE IF TEST IS ALLOWED
BEQ 6$ ;BR IF NOT
CMPB REDBYT,LUSWI1 ;COMPARE SWITCHES TO EXPECTED
BEQ 6$ ;BR IF MATCH

```



```
5546 026300 013737 002454 002370      MOV      LUSWI1,GOODAT      ;SET EXPECTED DATA
5547 026306 013737 002350 002372      MOV      REDBYT,BADDAT     ;SET ACTUAL DATA
5548 026314 004737 004226                JSR      PC,GETALL          ;GET REGS FOR PRINTOUT
5549                                     ;REPORT SWITCH PACK #1 INCORRECT
5550 026320                ERRDF   43,EM43,ERR2
(4) 026320 104455
(5) 026322 000053                TRAP     C$ERDF
(5) 026324 014116                .WORD   43
(5) 026326 015406                .WORD   EM43
5551 026330                .WORD   ERR2
6$: 026330
5552 026330                ENDSUB
(3) 026330
(3) 026330 104403                L10041: TRAP     C$ESUB
5553 -----
5554 : READ AND PRINT SWITCH PACK #2, IF DESIRED
5555 -----
5556 026332                BGNSUB
(3) 026332
(3) 026332 104402                T11.2: TRAP     C$BSUB
5557 026334 004737 003276                JSR      PC,MSTCLR          ;ISSUE MASTER CLEAR
5558 026340 012737 000015 002364      MOV      #15,REGNUM         ;SET LU REG NO. = 15
5559 026346 004737 003372                JSR      PC,READLU          ;READ LU REG 15
5560 026352 023727 002400 000001      CMP      FRSPAS,#1          ;SEE IF IN FIRST PASS AFTER LOAD
5561 026360 001403                BEQ      3$                 ;BR IF YES
5562 026362 005737 002260                TST      PRNFLG             ;SEE IF PRINTOUT IS ALLOWED ON ALL PASSES
5563 026366 001412                BEQ      4$                 ;BR IF NOT
5564 026370
5565 026370 3$: PRINTF #FMT13,REDBYT
(8) 026370 013746 002350                MOV      REDBYT,-(SP)
(7) 026374 012746 012703                MOV      #FMT13,-(SP)
(6) 026400 012746 000002                MOV      #2,-(SP)
(3) 026404 010600                MOV      SP,R0
(4) 026406 104417                TRAP     C$PNTF
(4) 026410 062706 000006                ADD      #6,SP
5566 026414 005737 002262                4$: TST      SWIFLG          ;SEE IF TEST IS ALLOWED
5567 026420 001420                BEQ      6$                 ;BR IF NOT
5568 026422 123737 002350 002456      CMPB     REDBYT,LUSWI2      ;COMPARE SWITCHES TO EXPECTED
5569 026430 001414                BEQ      6$                 ;BR IF MATCH
5570 026432 013737 002456 002370      MOV      LUSWI2,GOODAT     ;SET EXPECTED DATA
5571 026440 013737 002350 002372      MOV      REDBYT,BADDAT     ;SET ACTUAL DATA
5572 026446 004737 004226                JSR      PC,GETALL          ;GET REGS FOR PRINTOUT
5573                                     ;REPORT SWITCH PACK #2 INCORRECT
5574 026452                ERRDF   44,EM44,ERR2
(4) 026452 104455
(5) 026454 000054                TRAP     C$ERDF
(5) 026456 014143                .WORD   44
(5) 026460 015406                .WORD   EM44
5575 026462                .WORD   ERR2
5576 026462                6$: ENDSUB
(3) 026462
(3) 026462 104403                L10042: TRAP     C$ESUB
5577 -----
5578 : READ AND PRINT SWITCH PACK #3, IF DESIRED
5579 -----
5580 026464                BGNSUB
(3) 026464                T11.3:
```

```

(3) 026464 104402
5581 026466 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR TRAP C$8SUB
5582 026472 012737 000016 002364 MOV #16,REGNUM ;SET LU REG NO. = 16
5583 026500 004737 003372 JSR PC,READLU ;READ LU REG 16
5584 026504 023727 002400 000001 CMP FRSPAS,#1 ;SEE IF IN FIRST PASS AFTER LOAD
5585 026512 001403 BEQ 3$ ;BR IF YES
5586 026514 005737 002260 TST PRNFLG ;SEE IF PRINTOUT IS ALLOWED ON ALL PASSES
5587 026520 001412 BEQ 4$ ;BR IF NOT
5588 026522
5589 026522 3$: PRINTF #FMT14,REDBYT
(8) 026522 013746 002350 MOV REDBYT,-(SP)
(7) 026526 012746 012747 MOV #FMT14,-(SP)
(6) 026532 012746 000002 MOV #2,-(SP)
(3) 026536 010600 MOV SP,R0
(4) 026540 104417 TRAP C$PNTF
(4) 026542 062706 000006 ADD #6,SP
5590 026546 005737 002262 4$: TST SWIFLG ;SEE IF TEST IS ALLOWED
5591 026552 001420 BEQ 6$ ;BR IF NOT
5592 026554 123737 002350 002460 CMPB REDBYT,LUSW13 ;COMPARE SWITCHES TO EXPECTED
5593 026562 001414 BEQ 6$ ;BR IF MATCH
5594 026564 013737 002460 002370 MOV LUSW13,GOODAT ;SET EXPECTED DATA
5595 026572 013737 002350 002372 MOV REDBYT,BADDAT ;SET ACTUAL DATA
5596 026600 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
5597 ;REPORT SWITCH PACK #3 INCORRECT
5598 026604 ERRDF 45,EM45,ERR2
(4) 026604 104455 TRAP C$ERRDF
(5) 026606 000055 .WORD 45
(5) 026610 014170 .WORD EM45
(5) 026612 015406 .WORD ERR2
5599 026614 6$:
5600 026614 ENDSUB
(3) 026614 L10043:
(3) 026614 104403 TRAP C$ESUB
5601 026616 ENDTST L10040:
(3) 026616 104401 TRAP C$ETST
5602
5603
5604
5605
5606
5607
5608 ;*****
5609 .SBTTL TEST 12 - REG AX3-15 PRINTOUT
5610 ;*
5611 ;* IN THIS TEST, REG AX3-15 IS READ AND THE CONTENTS PRINTED OUT IF DESIRED BY
5612 ;* THE OPERATOR, AS INDICATED IN THE SOFTWARE P-TABLE. THE DEFAULT IS TO NOT
5613 ;* PRINT THE REG.
5614 ;*****
5614 026620 BGNST
(3) 026620 T12::
5615 026620 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
5616 026624 142777 000010 153606 BICB #LULOOP,@BSEL1 ;CLEAR LULOOP
5617 026632 012737 000006 002366 MOV #6,AXNUM ;SET AX BYTE NO. FOR AX3-15
5618 026640 004737 003624 JSR PC,READAX ;READ AX3-15,AX3-16
5619 026644 023727 002400 000001 CMP FRSPAS,#1 ;SEE IF FIRST PASS AFTER LOAD
5620 026652 001403 BEQ 3$ ;BR IF YES
    
```

```

5621 026654 005737 002260          TST   PRNFLG          ;SEE IF PRINTOUT IS ALLOWED ON ALL PASSES
5622 026660 001424          BEQ   4$              ;BR IF NOT
5623 026662
5624
5625 026662          3$:
;PRINT DEVICE ADDRESS
      PRINTF #FMT18,MPCSR
      MOV   MPCSR,-(SP)
      MOV   #FMT18,-(SP)
      MOV   #2,-(SP)
      MOV   SP,R0
      TRAP  C$PNTF
      ADD   #6,SP
(8) 026662 013746 002436
(7) 026666 012746 013062
(6) 026672 012746 000002
(3) 026676 010600
(4) 026700 104417
(4) 026702 062706 000006
5626
5627 026706          ;PRINT AX3-15
      PRINTF #FMT15,RAX15
      MOV   RAX15,-(SP)
      MOV   #FMT15,-(SP)
      MOV   #2,-(SP)
      MOV   SP,R0
      TRAP  C$PNTF
      ADD   #6,SP
(8) 026706 013746 002354
(7) 026712 012746 013013
(6) 026716 012746 000002
(3) 026722 010600
(4) 026724 104417
(4) 026726 062706 000006
5628 026732          4$:
5629 026732          ENDTST
(3) 026732
(3) 026732 104401          L10044:
                                TRAP  C$ETST
5630
5631
5632
5633
5634
5635
5636
5637
5638
5639
5640
5641
5642
5643
5644
5645
5646
5647
5648
5649
5650
5651
5652 026734
(3) 026734
5653
5654
5655
5656 026734 012737 027264 002346
5657 026742 004737 003276
5658 026746 004737 010324
5659 026752 000226
5660 026754 000011
5661 026756 000000
  
```

```

:*****
:SBTTL      TEST 13 - CRC GENERATION TEST
:
:* - CRC-16, CHAR MODE:
:* THE FOLLOWING MESSAGE IS SENT IN DDCMP MODE WITH CRC-16 SELECTED -
:* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOP AND STEPLU
:* TO CLOCK THE DATA.  AT THE END OF THE MESSAGE THE
:* PROGRAM CHECKS FOR BCC = 1 (IN REG 12) INDICATING NO ERROR.
:
:* - CRC-CCITT - 1'S PRESET:
:* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT
:* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC = 0, INDICATING NO ERROR.
:
:* - CRC-CCITT - 0'S PRESET:
:* THE ABOVE SUBTEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT
:* THE END OF THE MESSAGE THE PROGRAM CHECKS FOR BCC = 0, INDICATING NO ERROR.
:*****
BGNTST
  
```

T13::

 : CRC 16, CHAR MODE

```

      MOV   #24$,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
      JSR   PC,MSTCLR        ;ISSUE MASTER CLEAR
      JSR   PC,SETUP         ;PROGRAM THE USYRT
      SYNCH
      STRIP!DDCMP
      000
  
```

5662	026760	000000		000		
5663	026762	004737	010516	JSR	PC,LODMSG	;LOAD MSG INTO TX SILO
5664	026766	002670		MSG1		
5665	026770	000011		9.		
5666	026772	004737	004754	JSR	PC,STPLU	;CLOCK THE MSG
5667	026776	000136		94.		
5668	027000	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000
5669	027004	000000		000		
5670	027006	000000		0		
5671	027010	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
5672	027014	000125		125		
5673	027016	000000		0		
5674	027020	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
5675	027024	000252		252		
5676	027026	000000		0		
5677	027030	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
5678	027034	000377		377		
5679	027036	000000		0		
5680	027040	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC = 1
5681	027044	100400		CRCCHK!400		
5682	027046	000000		0		

: CRC-CCITT-1'S PRESET, BIT MODE

5687	027050	004737	003276	JSR	PC,MSTCLR	;ISSUE MASTER CLEAR
5688	027054	004737	010324	JSR	PC,SETUP	;PROGRAM THE USYRT
5689	027060	000000		000		
5690	027062	000000		000		
5691	027064	000000		000		
5692	027066	000000		000		
5693	027070	004737	010516	JSR	PC,LODMSG	;LOAD MSG INTO TX SILO
5694	027074	002670		MSG1		
5695	027076	000011		9.		
5696	027100	004737	004754	JSR	PC,STPLU	;CLOCK THE MSG
5697	027104	000146		102.		
5698	027106	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000
5699	027112	000000		000		
5700	027114	000000		0		
5701	027116	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
5702	027122	000125		125		
5703	027124	000000		0		
5704	027126	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
5705	027132	000252		252		
5706	027134	000000		0		
5707	027136	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
5708	027142	000377		377		
5709	027144	000000		0		
5710	027146	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC = 0
5711	027152	101000		CRCCHK!1000		
5712	027154	000000		0		

: CRC-CCITT-0'S PRESET, BIT MODE

5713
5714
5715
5716
5717

5718	027156	004737	003276	JSR	PC,MSTCLR	;ISSUE MASTER CLEAR
5719	027162	004737	010324	JSR	PC,SETUP	;PROGRAM THE USYRT
5720	027166	000000		000		
5721	027170	000100		CRC1		
5722	027172	000000		000		
5723	027174	000000		000		
5724	027176	004737	010516	JSR	PC,LODMSG	;LOAD MSG INTO TX SILO
5725	027202	002670		MSG1		
5726	027204	000011		9.		
5727	027206	004737	004754	JSR	PC,STPLU	;CLOCK THE MSG
5728	027212	000146		102.		
5729	027214	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000
5730	027220	000000		000		
5731	027222	000000		0		
5732	027224	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
5733	027230	000125		125		
5734	027232	000000		0		
5735	027234	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
5736	027240	000252		252		
5737	027242	000000		0		
5738	027244	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
5739	027250	000377		377		
5740	027252	000000		0		
5741	027254	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC - 0
5742	027260	101000		CRCCHK!1000		
5743	027262	000000		0		
5744						
5745	027264	004737	003276	24\$: JSR	PC,MSTCLR	;ISSUE MASTER CLEAR TO CLEAN UP
5746	027270			ENDTST		
(3)	027270					L10045: TRAP C\$ETST
(3)	027270	104401				

5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771

027272

```
*****
:SBTTL      TEST 14 - CRC ERROR DETECTION TEST
:*
:* - CRC-16, CHAR MODE :
:* THE FOLLOWING MESSAGE IS SENT IN DDCMP MODE, WITH CRC-16 SELECTED -
:* 2 SYNCHS, 000, 125, 252, 377, 000, AND 2 SYNCHS, USING LULOOP AND STEPLU
:* TO CLOCK THE DATA. JUST BEFORE THE FIRST BIT OF THE LAST 000 CHAR IS SENT,
:* THE IERR BIT IS SET IN REG 17 TO CAUSE A 1 TO BE SENT, INTRODUCING A DATA
:* ERROR. AT THE END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 0, INDICATING
:* AN ERROR.
:*
:* - CRC-CCITT - 1'S PRESET :
:* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-1'S SELECTED. AT THE
:* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 1, INDICATING AN ERROR.
:*
:* - CRC-CCITT - 0'S PRESET :
:* THE ABOVE TEST IS PERFORMED IN BIT MODE WITH CRC-CCITT-0'S SELECTED. AT THE
:* END OF THE MESSAGE, THE PROGRAM CHECKS FOR BCC = 1, INDICATING AN ERROR.
:*****
BGNTST
```

T14::

(3) 027272

5772

5773

5774

5775 027272 012737 027674 002346

5776 027300 004737 005240

5777 027304 000226

5778 027306 000011

5779 027310 004737 005622

5780 027314 000000

5781 027316 100010

5782 027320 004737 010516

5783 027324 002676

5784 027326 000006

5785 027330 004737 004754

5786 027334 000010

5787 027336 004737 007212

5788 027342 000011

5789 027344 000001

5790 027346 004737 004754

5791 027352 000122

5792 027354 004737 007314

5793 027360 000001

5794 027362 000000

5795 027364 004737 007314

5796 027370 000125

5797 027372 000000

5798 027374 004737 007314

5799 027400 000252

5800 027402 000000

5801 027404 004737 007314

5802 027410 000377

5803 027412 000000

5804 027414 004737 007314

5805 027420 100000

5806 027422 000000

5807

5808

5809

5810

5811 027424 004737 005240

5812 027430 000000

5813 027432 000000

5814 027434 004737 005622

5815 027440 000000

5816 027442 100010

5817 027444 004737 010516

5818 027450 002676

5819 027452 000006

5820 027454 004737 004754

5821 027460 000010

5822 027462 004737 007212

5823 027466 000000

5824 027470 000001

5825 027472 004737 004754

5826 027476 000122

 : CRC 16, CHAR MODE

```

MOV #24$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;LOAD 2 SOM'S, CLOCK THEM INTO THE USYRT
SYNCH
STRIP!DDCMP
JSR PC,TXCHAR ;LOAD 000 CHAR, TX 1ST SYNCH
000
CHPCHK.8.
JSR PC,LODMSG ;LOAD MSG INTO TX SILO
MSG1+6
6
JSR PC,STPLU ;CLOCK LINE UNIT UNTIL 1ST BIT OF 000 CHAR
8.
JSR PC,STPERR ;MAKE 1ST BIT = 1 INSTEAD OF 0
STRIP!DDCMP
1
JSR PC,STPLU ;CLOCK REST OF MESSAGE
82.
JSR PC,CKDATA ;READ AND COMPARE CHAR TO 001 (INTENDED ERROR)
001
0
JSR PC,CKDATA ;READ AND COMPARE CHAR TO 125
125
0
JSR PC,CKDATA ;READ AND COMPARE CHAR TO 252
252
0
JSR PC,CKDATA ;READ AND COMPARE CHAR TO 377
377
0
JSR PC,CKDATA ;READ AND COMPARE CHAR TO 000, CHK BCC = 0
CRCCHK!000
0
  
```

 : CRC-CCITT-1'S PRESET, BIT MODE

```

JSR PC,INITRN ;LOAD 2 SOM'S, CLOCK THEM INTO THE USYRT
000
000
JSR PC,TXCHAR ;LOAD 000 CHAR, TX 1ST FLAG
000
CHPCHK!8.
JSR PC,LODMSG ;LOAD MSG INTO TX SILO
MSG1+6
6
JSR PC,STPLU ;CLOCK LINE UNIT UNTIL 1ST BIT OF 000 CHAR
8.
JSR PC,STPERR ;MAKE 1ST BIT = 1 INSTEAD OF 0
000
1
JSR PC,STPLU ;CLOCK REST OF MESSAGE
82.
  
```

5827	027500	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 001 (INTENDED ERROR)
5828	027504	000001		001		
5829	027506	000000		0		
5830	027510	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
5831	027514	000125		125		
5832	027516	000000		0		
5833	027520	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
5834	027524	000252		252		
5835	027526	000000		0		
5836	027530	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
5837	027534	000377		377		
5838	027536	000000		0		
5839	027540	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC = 1
5840	027544	101400		CRCCHK!1400		
5841	027546	000000		0		

: CRC-CCITT-0'S PRESET, BIT MODE

5846	027550	004737	005240	JSR	PC,INITRN	;LOAD 2 SOM'S, CLOCK THEM INTO THE USYRT
5847	027554	000000		000		
5848	027556	000100		CRC1		
5849	027560	004737	005622	JSR	PC,TXCHAR	;LOAD 000 CHAR, TX 1ST FLAG
5850	027564	000000		000		
5851	027566	100010		CHPCHK!8.		
5852	027570	004737	010516	JSR	PC,LODMSG	;LOAD MSG INTO TX SILO
5853	027574	002676		MSG1+6		
5854	027576	000006		6		
5855	027600	004737	004754	JSR	PC,STPLU	;CLOCK LINE UNIT UNTIL 1ST BIT OF 000 CHAR
5856	027604	000010		8.		
5857	027606	004737	007212	JSR	PC,STPERR	;MAKE 1ST BIT = 1 INSTEAD OF 0
5858	027612	000100		CRC1		
5859	027614	000001		1		
5860	027616	004737	004754	JSR	PC,STPLU	;CLOCK REST OF MESSAGE
5861	027622	000122		82.		
5862	027624	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 001 (INTENDED ERROR)
5863	027630	000001		001		
5864	027632	000000		0		
5865	027634	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 125
5866	027640	000125		125		
5867	027642	000000		0		
5868	027644	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 252
5869	027650	000252		252		
5870	027652	000000		0		
5871	027654	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 377
5872	027660	000377		377		
5873	027662	000000		0		
5874	027664	004737	007314	JSR	PC,CKDATA	;READ AND COMPARE CHAR TO 000, CHK BCC = 1
5875	027670	101400		CRCCHK!1400		
5876	027672	000000		0		
5877						
5878	027674	004737	003276	JSR	PC,MSTCLR	;ISSUE MASTER CLEAR TO CLEAN UP
5879	027700					
(3)	027700					
(3)	027700	104401				L10046: TRAP CSETST
5880						

5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
(3)
5905
5906
5907
5908
5909
5910
(3)
(3)
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933

027702
027702

012737 000006 002366
012737 000017 002364
027716
027716 104402
027720 012737 030122 002346
027726 004737 005240
027732 000026
027734 000111
027736 004737 010574
027742 002613
027744 000010
027746 004737 010746
027752 001000
027754 000002
027756 005037 002360
027762 012737 000347 002362
027770 004737 004012
027774 004737 004754
030000 000010
030002 004737 004754
030006 000010
030010 005001
030012 004737 004754
030016 000010
030020 004737 003372
030024 005201
030026 020127 000004

```
*****  
:SBTTL TEST 15 - VRC PARITY GENERATION TEST  
:  
:* SUBTEST 1 - TEST OF CORRECT ODD VRC PARITY GENERATION :  
:* THE LINE UNIT IS PLACED IN CHAR MODE, WITH ODD VRC AND 7-BIT CHARS SELECTED.  
:* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)  
:* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.  
:* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 1 AND FOR THE  
:* LAST 4 CHARS IT SHOULD = 0.  
:  
:* SUBTEST 2 - TEST OF CORRECT EVEN VRC PARITY GENERATION :  
:* THE LINE UNIT IS PLACED IN CHAR MODE, WITH EVEN VRC AND 7-BIT CHARS SELECTED.  
:* THE DATA CHARS IN PATTERN Q ARE TRANSMITTED, AND AS THE 8TH BIT (PARITY BIT)  
:* OF EACH DATA CHAR IS SENT THE PROGRAM CHECKS TXDATA FOR THE PROPER STATE.  
:* FOR THE FIRST 4 CHARS IN PATTERN Q THE PARITY BIT SHOULD = 0 AND FOR THE  
:* LAST 4 CHARS IT SHOULD = 1.  
:  
:* DATA PATTERN Q = 000,120,125,137,040,052,057,177  
:*****  
:BGNTST
```

T15::

: TEST ODD VRC GENERATION

```
MOV #6,AXNUM ;SET AX BYTE NO. FOR AX3  
MOV #17,REGNUM ;SET REG NO. = 17  
BGNSUB  
  
T15.1: TRAP CSBSUB  
MOV #8$,RETADR ;SET SUBTEST EXIT ADDRESS FOR ERRORS  
JSR PC,INTRN ;MST CLR, LOAD 2 SOM'S  
026  
CRC1!STRIP!DDCMP ;CHAR MODE, ODD VRC  
JSR PC,LDBYTS ;LOAD DATA INTO TX SILO  
PATQ  
8.  
JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO  
TXEOM  
2  
CLR WAX15  
MOV #TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO,WAX16  
JSR PC,WRITAX ;SET TX AND RCV LENGTHS = 7  
JSR PC,STPLU ;CLOCK FIRST SYNCH  
8.  
JSR PC,STPLU ;CLOCK 2ND SYNCH  
8.  
CLR R1 ;INIT CHAR COUNT  
JSR PC,STPLU ;CLOCK A CHAR  
2$.  
8.  
JSR PC,READLU ;READ REG 17  
INC R1 ;INCR CHAR COUNT  
CMP R1,#4 ;SEE IF 4 CHARS CLKD YET
```



```
5934 030032 003014          BGT      4$          ;BR IF YES
5935 030034 132737 000040 002350 BITB     #TXDATA,REDBYT ;SEE IF PARITY BIT IS SET
5936 030042 001024          BNE      6$          ;BR IF YES
5937 030044 004737 004226          JSR      PC,GETALL   ;GET REGS FOR PRINTOUT
5938                               ;REPORT ODD VRC PARITY BIT NOT SET
5939 030050          ERRDF  48,EM48,ERR7
(4) 030050 104455          TRAP     C$ERDF
(5) 030052 000060          .WORD   48
(5) 030054 014303          .WORD   EM48
(5) 030056 020244          .WORD   ERR7
5940 030060          ESCAPE  SUB
(3) 030060 104410          TRAP     C$ESCAPE
(3) 030062 000040          .WORD   L10050-
5941 030064 132737 000040 002350 4$:  BITB     #TXDATA,REDBYT ;SEE IF PARITY BIT IS CLEARED
5942 030072 001410          BEQ      6$          ;BR IF YES
5943 030074 004737 004226          JSR      PC,GETALL   ;GET REGS FOR PRINTOUT
5944                               ;REPORT ODD VRC PARITY BIT NOT CLEARED
5945          ERRDF  49,EM49,ERR7
(4) 030100 104455          TRAP     C$ERDF
(5) 030102 000061          .WORD   49
(5) 030104 014336          .WORD   EM49
(5) 030106 020244          .WORD   ERR7
5946 030110          ESCAPE  SUB
(3) 030110 104410          TRAP     C$ESCAPE
(3) 030112 000010          .WORD   L10050-
5947 030114 020127 000010          6$:  CMP      R1,#8.    ;SEE IF ALL CHARS TESTED YET
5948 030120 002734          BLT      2$          ;BR IF NOT
5949 030122          8$:
5950 030122          ENDSUB
(3) 030122          L10050:
(3) 030122 104403          TRAP     C$ESUB
5951
5952          :-----:
5953          : TEST EVEN VRC GENERATION
5954          :-----:
5954 030124 012737 000006 002366          MOV      #6,AXNUM    ;SET AX BYTE NO. FOR AX3
5955 030132 012737 000017 002364          MOV      #17,REGNUM  ;SET REG NO. = 17
5956 030140          BGNSUB
(3) 030140          T15.2:
(3) 030140 104402          TRAP     C$BSUB
5957 030142 012737 030344 002346          MOV      #18$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS
5958 030150 004737 005240          JSR      PC,INITRN   ;MST CLR, LOAD 2 SOM'S
5959 030154 000026          026
5960 030156 000211          CRC2!STRIP!DDCMP    ;CHAR MODE, EVEN VRC
5961 030160 004737 010574          JSR      PC,LDBYTS   ;LOAD DATA INTO TX SILO
5962 030164 002613          PATQ
5963 030166 000010          8.
5964 030170 004737 010746          JSR      PC,LODSIL   ;LOAD 2 EOM'S INTO TX SILO
5965 030174 001000          TXEOM
5966 030176 000002          2
5967 030200 005037 002360          CLR      WAX15
5968 030204 012737 000347 002362          MOV      #TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO,WAX16
5969 030212 004737 004012          JSR      PC,WRITAX   ;SET TX AND RCV LENGTHS = 7
5970 030216 004737 004754          JSR      PC,STPLU    ;CLOCK FIRST SYNCH
5971 030222 000010          8.
5972 030224 004737 004754          JSR      PC,STPLU    ;CLOCK 2ND SYNCH
5973 030230 000010          8.
```

```

5974 030232 005001          CLR    R1          ;INIT CHAR COUNT
5975 030234 004737 004754 12$: JSR    PC,STPLU    ;CLOCK A CHAR
5976 030240 000010          8.
5977 030242 004737 003372    JSR    PC,READLU   ;READ REG 17
5978 030246 005201          INC    R1          ;INCR CHAR COUNT
5979 030250 020127 000004    CMP    R1,#4       ;SEE IF 4 CHARS CLKD YET
5980 030254 003014          BGT    14$         ;BR IF YES
5981 030256 132737 000040 002350 BITB   #TXDATA,REDBYT ;SEE IF PARITY BIT IS CLEARED
5982 030264 001424          BEQ    16$         ;BR IF YES
5983 030266 004737 004226    JSR    PC,GETALL   ;GET REGS FOR PRINTOUT
5984                                     ;REPORT EVEN VRC PARITY BIT NOT CLEARED
5985 030272          ERRDF  51,EM51,ERR7
(4) 030272 104455
(5) 030274 000063          TRAP   C$ERDF
(5) 030276 014431          .WORD 51
(5) 030300 020244          .WORD EM51
5986 030302          ESCAPE SUB          .WORD  ERR7
(3) 030302 104410          TRAP   C$ESCAPE
(3) 030304 000040 002350 14$: BITB   #TXDATA,REDBYT ;SEE IF PARITY BIT IS SET
5987 030306 132737 000040 002350 BNE    16$         ;BR IF YES
5988 030314 001010          JSR    PC,GETALL   ;GET REGS FOR PRINTOUT
5989 030316 004737 004226    ;REPORT EVEN VRC PARITY BIT NOT SET
5990                                     ERRDF  50,EM50,ERR7
5991 030322          TRAP   C$ERDF
(4) 030322 104455          .WORD 50
(5) 030324 000062          .WORD EM50
(5) 030326 014375          .WORD  ERR7
(5) 030330 020244
5992 030332          ESCAPE SUB          TRAP   C$ESCAPE
(3) 030332 104410          .WORD  L10051-.
(3) 030334 000010
5993 030336 020127 000010 16$: CMP    R1,#8.       ;SEE IF ALL CHARS TESTED YET
5994 030342 002734          BLT    12$         ;BR IF NOT
5995 030344          18$:
5996 030344          ENDSUB
(3) 030344          L10051:
(3) 030344 104403          TRAP   C$ESUB
5997 030346 004737 003276    JSR    PC,MSTCLR   ;ISSUE MASTER CLEAR TO CLEAN UP
5998 030352          ENDTST          L10047:
(3) 030352          TRAP   C$ETST
(3) 030352 104401
5999
6000
6001
6002
6003
6004
6005
6006
6007
6008
6009
6010
6011
6012
6013

```

```

*****
.SBTTL      TEST 16 - VRC ERROR DETECTION TEST
*
* SUBTEST 1 - FORCING OF BCC USING ODD VRC
* THE LINE UNIT IS PLACED IN CHAR MODE WITH ODD VRC AND 7-BIT CHARS SELECTED.
* THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER
* 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM
* CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING
* NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE
* RECEIVED (INDICATING AN ERROR).

```

6014
6015
6016
6017
6018
6019
6020
6021
6022
6023
6024
6025
6026 030354
(3) 030354
6027
6028
6029
6030 030354 012737 000006 002366
6031 030362 012737 000012 002364
6032 030370
(3) 030370
(3) 030370 104402
6033 030372 012737 030610 002346
6034 030400 004737 005240
6035 030404 000026
6036 030406 000111
6037 030410 004737 010746
6038 030414 000400
6039 030416 000001
6040 030420 004737 010574
6041 030424 002623
6042 030426 000017
6043 030430 004737 010746
6044 030434 001000
6045 030436 000002
6046 030440 005037 002360
6047 030444 012737 000347 002362
6048 030452 013737 002362 002424
6049 030460 004737 004012
6050 030464 004737 004754
6051 030470 000130
6052 030472 012701 000007
6053 030476 004737 007212
6054 030502 000111
6055 030504 000001
6056 030506 004737 004754
6057 030512 000007
6058 030514 005301
6059 030516 001367
6060 030520 004737 004754
6061 030524 000020
6062 030526 012701 000010
6063 030532 012703 002623
6064 030536 112337 030546
6065 030542 004737 007314
6066 030546 100000

: * SUBTEST 2 - FORCING OF BCC USING EVEN VRC
: * THE LINE UNIT IS PLACED IN CHAR MODE WITH EVEN VRC AND 7-BIT CHARS SELECTED.
: * THE FIRST 8 DATA CHARS IN PATTERN R ARE TRANSMITTED NORMALLY, BUT THE OTHER
: * 7 CHARS ARE TRANSMITTED WITH BIT 0 STUCK AT 1 (USING IERR BIT). THE PROGRAM
: * CHECKS FOR BCC = 0 AFTER EACH OF THE FIRST 8 CHARS ARE RECEIVED (INDICATING
: * NO ERROR) AND CHECKS FOR BCC = 1 AFTER EACH OF THE REMAINING 7 CHARS ARE
: * RECEIVED (INDICATING AN ERROR).
: *
: * DATA PATTERN R = 000,100,120,124,164,172,176,177,000,100,120,124,164,
: * 172,176.
: *
: *****

BGNTST

T16::

: TEST ODD VRC ERROR DETECTION

MOV #6,AXNUM ;SET AX BYTE NO. FOR AX3
MOV #12,REGNUM ;SET REG NO.
BGNSUB

T16.1:
TRAP C\$BSUB
MOV #10\$,RETADR ;SET SUBTEST EXIT ADRS FOR ERRORS
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
026
CRC1!STRIP!DDCMP ;CHAR MODE, ODD VRC
JSR PC,LODSIL ;LOAD A THIRD SOM INTO TX SILO
TXSOM
1
JSR PC,LDBYTS ;LOAD DATA INTO TX BUFFER
PATR
15.
JSR PC,LODSIL ;LOAD 2 EOM'S INTO TX SILO
TXEOM
2
CLR WAX15
MOV #TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO,WAX16
MOV WAX16,SAVLEN ;STORE LENGTH 7
JSR PC,WRITAX ;SET TX AND RCV LENGTHS = 7
JSR PC,STPLU ;CLOCK 1ST 8 CHARS, WITH NO ERRORS
88.
MOV #7,R1 ;INIT COUNTER FOR LAST 7 CHARS
JSR PC,STPERR ;ASSERT IERR BIT FOR 1 TIME
CRC1!STRIP!DDCMP
1
JSR PC,STPLU ;CLOCK REST OF CHAR
7
DEC R1 ;DECR COUNTER
BNE 3\$;BR IF NOT DONE TRANSMITTING YET
JSR PC,STPLU ;CLOCK 2 TERMINATING SYNCHS
16.
MOV #8,R1 ;INIT COUNTER FOR ERROR-FREE CHARS
MOV #PATR,R3 ;INIT DATA PATTERN POINTER
5\$: MOVB (R3)+,6\$;GET AN EXPECTED DATA CHAR
JSR PC,CKDATA ;GO CHECK CHAR, CHK BCC=0
6\$: BCCCHK!000

6067	030550	000000			0		
6068	030552	005301			DEC R1	;DECR COUNTER	
6069	030554	001370			BNE 5\$;BR IF NOT DONE YET	
6070	030556	012701	000007		MOV #7,R1	;INIT COUNTER FOR ERROR CHARS	
6071	030562	112337	030600		MOV (R3)+,9\$;GET EXPECTED DATA CHAR	
6072	030566	052737	000001	030600	BIS #BIT0,9\$;EXPECT ERROR BIT 0 SET	
6073	030574	004737	007314		JSR PC,CKDATA	;CHECK DATA, CHK BCC=1	
6074	030600	100400			9\$: BCCCHK!RXBCC!000		
6075	030602	000000			0		
6076	030604	005301			DEC R1	;DECR COUNTER	
6077	030606	001365			BNE 8\$;BR IF NOT DONE YET	
6078	030610				10\$:		
6079	030610				ENDSUB		
(3)	030610						
(3)	030610	104403					L10053: TRAP C\$ESUB
6080					-----		
6081					: TEST EVEN VRC ERROR DETECTION		
6082					-----		
6083	030612	012737	000006	002366	MOV #6,AXNUM	;SET AX BYTE NO. FOR AX3	
6084	030620	012737	000012	002364	MOV #12,REGNUM	;SET REG NO.	
6085	030626				BGNSUB		
(3)	030626						
(3)	030626	104402					T16.2: TRAP C\$BSUB
6086	030630	012737	031046	002346	MOV #30\$,RETADR	;SET SUBTEST EXIT ADRS FOR ERRORS	
6087	030636	004737	005240		JSR PC,INITRN	;MST CLR, LOAD 2 SOM'S	
6088	030642	000026			026		
6089	030644	000211			CRC2!STRIP!DDCMP	;CHAR MODE, EVEN VRC	
6090	030646	004737	010746		JSR PC,LODSIL	;LOAD A THIRD SOM INTO TX SILO	
6091	030652	000400			TXSOM		
6092	030654	000001			1		
6093	030656	004737	010574		JSR PC,LDBYTS	;LOAD DATA INTO TX BUFFER	
6094	030662	002623			PATR		
6095	030664	000017			15.		
6096	030666	004737	010746		JSR PC,LODSIL	;LOAD 2 EOM'S INTO TX SILO	
6097	030672	001000			TXEOM		
6098	030674	000002			2		
6099	030676	005037	002360		CLR WAX15		
6100	030702	012737	000347	002362	MOV #TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO,WAX16		
6101	030710	013737	002362	002424	MOV WAX16,SAVLEN	;STORE LENGTH 7	
6102	030716	004737	004012		JSR PC,WRITAX	;SET TX AND RCV LENGTHS = 7	
6103	030722	004737	004754		JSR PC,STPLU	;CLOCK 1ST 8 CHARS, WITH NO ERRORS	
6104	030726	000130			88.		
6105	030730	012701	000007		MOV #7,R1	;INIT COUNTER FOR LAST 7 CHARS	
6106	030734	004737	007212		23\$: JSR PC,STPERR	;ASSERT IERR BIT FOR 1 TIME	
6107	030740	000211			CRC2.STRIP!DDCMP		
6108	030742	000001			1		
6109	030744	004737	004754		JSR PC,STPLU	;CLOCK REST OF CHAR	
6110	030750	000007			7		
6111	030752	005301			DEC R1	;DECR COUNTER	
6112	030754	001367			BNE 23\$;BR IF NOT DONE TRANSMITTING YET	
6113	030756	004737	004754		JSR PC,STPLU	;CLOCK 2 TERMINATING SYNCHS	
6114	030762	000020			16.		
6115	030764	012701	000010		MOV #8,R1	;INIT COUNTER FOR ERROR-FREE CHARS	
6116	030770	012703	002623		MOV #PATR,R3	;INIT DATA PATTERN POINTER	
6117	030774	112337	031004		25\$: MOV (R3)+,26\$;GET EXPECTED DATA CHAR	
6118	031000	004737	007314		JSR PC,CKDATA	;CHK DATA, CHECK BCC=0	

```

6119 031004 100000          26$: BCCCHK!000
6120 031006 000000          0
6121 031010 005301          DEC R1          ;DECR COUNTER
6122 031012 001370          BNE 25$        ;BR IF NOT DONE YET
6123 031014 012701 000007  MOV #7,R1      ;INIT COUNTER FOR ERROR CHARS
6124 031020 112337 031036  28$: MOVB (R3)+,29$ ;GET EXPECTED DATA CHAR
6125 031024 052737 000001 031036  BIS #BIT0,29$ ;SET EXPECTED ERROR BIT 0
6126 031032 004737 007314  JSR PC,CKDATA ;CHK DATA, CHK BCC=1
6127 031036 100400          29$: BCCCHK!RXBCC.000
6128 031040 000000          0
6129 031042 005301          DEC R1          ;DECR COUNTER
6130 031044 001365          BNE 28$        ;BR IF NOT DONE YET
6131 031046          30$:
6132 031046          ENDSUB
(3) 031046
(3) 031046 104403          L10054: TRAP C$ESUB
6133 031050 004737 003276  JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
6134 031054          ENDTST
(3) 031054
(3) 031054 104401          L10052: TRAP C$ETST
6135
6136
6137
6138
6139
6140
6141
6142
6143
6144
6145
6146
6147
6148
6149
6150 031056
(3) 031056
6151 031056 012737 000021 002434  MOV #17,,TSTNUM ;SET TEST NO.
6152 031064 012737 031336 002346  MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
6153 031072 004737 003276          JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6154 031076 004737 011016          JSR PC,CKLPBK ;CHECK LOOPBACK -
6155 031102 000010          INTGRL          ; SEE IF TEST SHOULD BE RUN
6156 031104 012737 000323 031122  MOV #1422.XYZ!V35!OP.TEST,6$ ;SET UP TO SELECT INTEGRAL MODEM
6157 031112 004737 010324          JSR PC,SETUP ;PROGRAM THE USYRT
6158 031116 000226
6159 031120 000011
6160 031122 000000          6$: STRIP!DDCMP
6161 031124 000000          .WORD 0
6162 031126 004737 010656          000
6163 031132 142777 000010 151300  JSR PC,LDMSG1 ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
6164 031140 012737 000012 002364  BICB #LULOOP,@SEL1 ;CLEAR LULOOP, CLOCK MSG
6165 031146 012703 002762          MOV #12,REGNUM ;SET LU REG NO. = 12
6166 031152 013702 002264          MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
6167 031156 004737 003372          9$: MOV TCOUNT,R2 ;INIT TIMER
6168 031162 132737 000020 002350  10$: JSR PC,READLU ;READ REG 12
6169 031170 001011          BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
          BNE 12$ ;BR IF YES
  
```

```

*****
:SBTTL TEST 17 - INTEGRAL MODEM INTERFACE TEST - CHAR MODE, CRC
:*
:* THE INTEGRAL MODEM IS SELECTED BY THE PROGRAM IN AX3-15, AND A
:* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
:* ON THE LINE UNIT OR AT THE CABLE. THE MESSAGE CONSISTS OF
:* 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
:* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
:* SKIPPED FOR THAT UNIT.
*****
BGNTST
  
```

T17::

```
6170 031172 005202          INC      R2          ;INCREMENT TIMER
6171 031174 001370          BNE     10$         ;BR IF NO TIME-OUT YET
6172 031176 004737 004226   JSR     PC,GETALL   ;GET REGS FOR PRINTOUT
6173          ;REPORT IRDY NOT SET
6174 031202          ERRDF  17,EM17,ERR7
(4) 031202 104455          TRAP   C$ERDF
(5) 031204 000021          .WORD 17
(5) 031206 013456          .WORD EM17
(5) 031210 020244          .WORD ERR7
6175 031212 000451          BR      24$         ;ESCAPE TO END OF TEST
6176 031214 012337 031224   12$:  MOV   (R3)+,16$   ;
6177 031220 004737 007314   JSR   PC,CKDATA     ;COMPARE RCV'D DATA CHAR TO EXPECTED
6178 031224 000000          16$:  0
6179 031226 000000          0
6180 031230 020327 003000   CMP   R3,#RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET
6181 031234 103746          BLO   9$            ;BR IF NOT YET
6182 031236 004737 004646   JSR   PC,WAIT50     ;STALL FOR 50 MICRO-SEC
6183
6184 031242 004737 005052   JSR   PC,DACTIV     ;CHECK DACT = 0
6185 031246 000000          0
6186 031250 004737 006260   JSR   PC,IACTIV     ;CHECK IACT STILL = 1
6187 031254 000001          1
6188 031256 012737 000013 002364   MOV   #13,REGNUM    ;SET REG NO. = 13
6189 031264 004737 003372   JSR   PC,READLU     ;READ REG 13
6190 031270 042737 000232 002350   BIC   #RING!HDX!MODR!STBY,REDBYT ;CLR UNUSED BITS
6191 031276 023727 002350 000000   CMP   REDBYT,#0     ;CHECK REG 13 FOR 000 (MODEM SIGNALS SHOULD BE CLEARED)
6192 031304 001414          BEQ   24$           ;BR IF CLEARED
6193 031306 012737 000000 002370   MOV   #0,GOODAT     ;SET EXPECTED DATA = 0
6194 031314 013737 002350 002372   MOV   REDBYT,BADDAT ;SET ACTUAL DATA
6195 031322 004737 004226   JSR   PC,GETALL     ;GET REGS FOR PRINTOUT
6196          ;REPORT REG MISCMPARE
6197 031326          ERRDF  3,EM3,ERR2
(4) 031326 104455          TRAP   C$ERDF
(5) 031330 000003          .WORD 3
(5) 031332 013305          .WORD EM3
(5) 031334 015406          .WORD ERR2
6198 031336          24$:
6199 031336          ENDTST
(3) 031336
(3) 031336 104401          L10055: TRAP   C$ETST
6200
6201
6202
6203
6204
6205
6206          ;*****
6207          ;SBTTL      TEST 18 - V.35 MODEM INTERFACE TEST - CHAR MODE, CRC
6208          ;*
6209          ;* THE V.35 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A
6210          ;* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
6211          ;* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,
6212          ;* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF
6213          ;* 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
6214          ;* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
6215          ;* SKIPPED FOR THAT UNIT.
6215          ;*****
```

```

6216 031340          BGNSTST
(3) 031340
6217 031340 012737 000022 002434      MOV    #18,,TSTNUM      ;SET TEST NO.
6218 031346 012737 031610 002346      MOV    #24$,RETADR     ;SET TEST EXIT ADDRESS FOR ERRORS
6219 031354 004737 003276          JSR    PC,MSTCLR       ;ISSUE MASTER CLEAR
6220 031360 004737 011016          JSR    PC,CKLPBK      ;CHECK LOOPBACK -
6221 031364 000020          V35                   ;SEE IF TEST SHOULD BE RUN
6222 031366 012737 000313 031404      MOV    #I422!XYZ!INTGRL!OP!TEST,6$ ;SET UP TO SELECT V35
6223 031374 004737 010324          JSR    PC,SETUP       ;PROGRAM THE USYRT
6224 031400 000226          SYNCH
6225 031402 000011          STRIP!DDCMP
6226 031404 000000          6$: .WORD 0
6227 031406 000000          000
6228 031410 142777 000010 151022      BICB  #LULOOP,@BSEL1  ;CLEAR LULOOP
6229 031416 012737 000013 002364      MOV    #13,REGNUM     ;SET LU REG NO. = 13
6230 031424 004737 003372          JSR    PC,READLU     ;READ REG 13
6231 031430 132737 000001 002350      BITB  #CARR,REDBYT   ;CHECK FOR CARRIER FALSELY SET
6232 031436 001415          BEQ    8$            ;BR IF NOT SET
6233 031440 012737 000000 002370      MOV    #000,GOODAT   ;SET EXPECTED DATA
6234 031446 013737 002350 002372      MOV    REDBYT,BADDAT ;SET ACTUAL DATA
6235 031454 004737 004226          JSR    PC,GETALL     ;GET REGS FOR PRINTOUT
6236
6237 031460          ;REPORT CARRIER NOT CLEARED
(4) 031460 104455          ERRDF 66,EM66,ERR7
(5) 031462 000102          TRAP  C$ERDF
(5) 031464 015020          .WORD 66
(5) 031466 020244          .WORD EM66
6238 031470 000447          .WORD ERR7
6239 031472 152777 000010 150740      8$: BR    24$
6240 031500 004737 010656          BISB  #LULOOP,@BSEL1 ;SET LULOOP AGAIN
6241 031504 142777 000010 150726          JSR    PC,LDMSG1     ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
6242 031512 012737 000012 002364          BICB  #LULOOP,@BSEL1 ;CLEAR LULOOP, CLOCK MSG
6243 031520 012703 002762          MOV    #12,REGNUM    ;SET LU REG NO. = 12
6244 031524 013702 002264          MOV    #RCVBUF,R3    ;GET POINTER TO RCV MSG BUF
6245 031530 004737 003372          9$: MOV    TCOUNT,R2 ;INIT TIMER
6246 031534 132737 000020 002350      10$: JSR    PC,READLU  ;READ REG 12
6247 031542 001011          BITB  #IRDY,REDBYT  ;SEE IF IRDY IS SET YET
6248 031544 005202          BNE   12$           ;BR IF YES
6249 031546 001370          INC   R2            ;INCREMENT TIMER
6250 031550 004737 004226          BNE   10$           ;BR IF NO TIME-OUT YET
6251          JSR    PC,GETALL  ;GET REGS FOR PRINTOUT
6252          ;REPORT IRDY NOT SET
(4) 031554 104455          ERRDF 17,EM17,ERR7
(5) 031556 000021          TRAP  C$ERDF
(5) 031560 013456          .WORD 17
(5) 031562 020244          .WORD EM17
6253 031564 000411          .WORD ERR7
6254 031566 012337 031576          12$: BR    24$
6255 031572 004737 007314          MOV    (R3)+,16$     ;ESCAPE TO END OF TEST
6256 031576 000000          16$: JSR    PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
6257 031600 000000          0
6258 031602 020327 003000          0
6259 031606 103746          CMP   R3,#RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET
6260 031610          BLO   9$            ;BR IF NOT YET
6261 031610          24$: ENDTST
(3) 031610

```

L10056:

(3) 031610 104401

TRAP CSETST

6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
(3)
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
(4)
(5)
(5)
(5)
6303
6304
6305
6306
6307
6308
6309
6310
6311

031612
031612 012737 000023 002434
031620 012737 031772 002346
031626 004737 003276
031632 004737 011016
031636 000100
031640 012737 000233 031656
031646 004737 010324
031652 000226
031654 000011
031656 000000
031660 000000
031662 004737 010656
031666 142777 000010 150544
031674 012737 000012 002364
031702 012703 002762
031706 013702 002264
031712 004737 003372
031716 132737 000020 002350
031724 001011
031726 005202
031730 001370
031732 004737 004226
031736
031740 000021
031742 013456
031744 020244
031746 000411
031750 012337 031760
031754 004737 007314
031760 000000
031762 000000
031764 020327 003000
031770 103746
031772
031772

```
*****
:SBTTL TEST 19 - RS 232C AND RS 423 MODEM INTERFACE TEST - CHAR MODE, CRC
:*
:* THE RS232C & RS423 (XYZ) MODLM INTERFACE IS SELECTED BY THE PROGRAM IN
:* AX3-15, AND A MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURN-
:* AROUND CONNECTOR ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,
:* OR A MODEM TEST MODE. THE MESSAGE CONSISTS
:* OF 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE
:* P-TABLE FOR THE CURRENT UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS
:* PROVIDED, THE TEST WILL BE SKIPPED FOR THAT UNIT.
*****
BGNTST
T19::
MOV #19.,TSTNUM ;SET TEST NO.
MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,CKLPBK ;CHECK LOOPBACK -
XYZ ; SEE IF TEST SHOULD BE RUN
MOV #1422!V35!INTGRL!OP!TEST,6$ ;SET UP TO SELECT XYZ
JSR PC,SETUP ;PROGRAM THE USYRT
SYNCH
STRIP!DDCMP
6$: .WORD 0
000
JSR PC,LDMSG1 ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
BICB #LULOOP,@BSEL1 ;CLEAR LULOOP, CLOCK MSG
MOV #12,REGNUM ;SET LU REG NO. = 12
MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
9$: MOV TCOUNT,R2 ;INIT TIMER
10$: JSR PC,READLU ;READ REG 12
BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
BNE 12$ ;BR IF YES
INC R2 ;INCREMENT TIMER
BNE 10$ ;BR IF NO TIME-OUT YET
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT IRDY NOT SET
ERRDF 17,EM17,ERR7
TRAP C$ERDF
:WORD 17
:WORD EM17
:WORD ERR7
BR 24$ ;ESCAPE TO END OF TEST
12$: MOV (R3)+,16$
JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
16$: 0
0
CMP R3,#RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET
BLO 9$ ;BR IF NOT YET
24$:
ENDTST
```


L10057: TRAP C\$ETST

(3) 031772
(3) 031772 104401
6312
6313
6314
6315
6316
6317
6318
6319
6320
6321
6322
6323
6324
6325
6326
6327
6328
6329 031774
(3) 031774
6330 031774 012737 000024 002434
6331 032002 012737 032154 002346
6332 032010 004737 003276
6333 032014 004737 011016
6334 032020 000200
6335 032022 012737 000133 032040
6336 032030 004737 010324
6337 032034 000226
6338 032036 000011
6339 032040 000000
6340 032042 000000
6341 032044 004737 010656
6342 032050 142777 000010 150362
6343 032056 012737 000012 002364
6344 032064 012703 002762
6345 032070 013702 002264
6346 032074 004737 003372
6347 032100 132737 000020 002350
6348 032106 001011
6349 032110 005202
6350 032112 001370
6351 032114 004737 004226
6352
6353 032120
(4) 032120 104455
(5) 032122 000021
(5) 032124 013456
(5) 032126 020244
6354 032130 000411
6355 032132 012337 032142
6356 032136 004737 007314
6357 032142 000000
6358 032144 000000
6359 032146 020327 003000
6360 032152 103746

:SBTTL TEST 20 - RS 422 MODEM INTERFACE TEST - CHAR MODE, CRC
:*****
:* THE RS 422 MODEM INTERFACE IS SELECTED BY THE PROGRAM IN AX3-15, AND A
:* MESSAGE IS TRANSMITTED, RECEIVED, AND CHECKED USING A TURNAROUND CONNECTOR
:* ON THE LINE UNIT OR AT THE MODEM SIDE OF THE CABLE,
:* OR A MODEM TEST MODE. THE MESSAGE CONSISTS OF
:* 5 SYNCHS, 000,125,252,377,000, AND 1 SYNCH. IF THE P-TABLE FOR THE CURRENT
:* UNIT INDICATES THAT NO EXTERNAL TURNAROUND IS PROVIDED, THE TEST WILL BE
:* SKIPPED FOR THAT UNIT.
:*****
:BGNTST

T20::
MOV #20,,TSTNUM ;SET TEST NO.
MOV #24\$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,CKLPBK ;CHECK LOOPBACK -
I422 ; SEE IF TEST SHOULD BE RUN
MOV #XYZ!V35!INTGRL!OP!TEST,6\$;SET UP TO SELECT 422
JSR PC,SETUP ;PROGRAM THE USYRT
SYNCH
STRIP!DDCMP
6\$: .WORD 0
000
JSR PC,LDMSG1 ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
BICB #LULOOP,@BSEL1 ;CLEAR LULOOP, CLOCK MSG
MOV #12,REGNUM ;SET LU REG NO. = 12
MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
9\$: MOV TCOUNT,R2 ;INIT TIMER
10\$: JSR PC,READLU ;READ REG 12
BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
BNE 12\$;BR IF YES
INC R2 ;INCREMENT TIMER
BNE 10\$;BR IF NO TIME-OUT YET
JSR PC,GETALL ;GET REGS FOR PRINTOUT
:REPORT IRDY NOT SET
ERRDF 17,EM17,ERR7
TRAP C\$ERDF
.WORD 17
.WORD EM17
.WORD ERR7
BR 24\$;ESCAPE TO END OF TEST
12\$: MOV (R3)+,16\$
JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
16\$: 0
0
CMP R3,#RCVBUF+14. ;SEE IF ALL CHARS CHECKED YET
BLO 9\$;BR IF NOT YET

6361 032154
6362 032154
(3) 032154
(3) 032154 104401

24\$:
ENDTST

L10060:
TRAP C\$ETST

6363
6364
6365
6366
6367
6368
6369
6370
6371
6372
6373
6374
6375
6376
6377
6378
6379

:SBTTL TEST 21 - HALF-DUPLEX BIT (HALF DUPX) TEST
:
:* THIS TEST VERIFIES THAT SETTING HALF-DUPLEX BIT IN REG 13 DOES NOT INHIBIT
:* LOADING OF THE USYRT TRANSMITTER FROM THE TRANSMITTER SILO.
:* A MASTER CLEAR IS ISSUED, DDCMP MODE IS ENTERED, AND THE HALF DUPX
:* BIT IN REG 13 IS SET. A MESSAGE IS LOADED INTO THE TX SILO
:* CONSISTING OF 2 SYNCHS, 000,125,252,377,000, AND 2 MORE SYNCHS.
:* THE LINE UNIT IS THEN CLOCKED EXTENSIVELY, AND THE TX SILO IS CHECKED TO
:* BE UNLOADED (ALL CHARS SHOULD HAVE BEEN REMOVED) AND THE RECEIVER
:* IS MONITORED TO INSURE THAT NO RCV FLAGS ARE GENERATED.
:*****

6380 032156
(3) 032156
6381 032156 012737 032246 002346
6382 032164 012737 000013 002364
6383 032172 004737 005240
6384 032176 000226
6385 032200 000011
6386 032202 112737 000020 002352
6387 032210 004737 003450
6388 032214 004737 010516
6389 032220 002674
6390 032222 000007
6391 032224 004737 004754
6392 032230 000136
6393 032232 004737 004362
6394 032236 000001
6395 032240 004737 005774
6396 032244 000001
6397 032246 004737 003276
6398 032252
(3) 032252
(3) 032252 104401

BGNTST

T21::
MOV #24\$,RETADR ;SET TEST EXIT ADRS FOR ERRORS
MOV #13,REGNUM ;SET REG NO. = 13
JSR PC,INITRN ;MST CLR, LOAD 2 SOM'S
SYNCH
STRIP!DDCMP
MOVB #HDX,WRIBYT
JSR PC,WRITLU ;SET HDX BIT IN REG 13
JSR PC,LODMSG ;LOAD MSG INTO TX SILO
MSG1+4
7
JSR PC,STPLU ;CLK MORE THAN ENTIRE MSG
94.
JSR PC,OSIRDY ;CHK ORDY = 1, OCOR = 0
1
JSR PC,ISIRDY ;CHK ICIR = 1, IRDY = 0
1
JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP

L10061:
TRAP C\$ETST

6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409
6410
6411

:SBTTL TEST 22 - HALF-DUPLEX RCV DISABLED TEST WITH SILOS DISABLED
:
:* THIS TEST SENDS A MESSAGE IN HDX, CHAR MODE, WITH NO ERROR DETECTION, AND
:* THE SILOS DISABLED. THE MSG CONSISTS OF 2 SYNCHS AND 2 000 CHARS.
:* THE DATA IS SENT WITH LULOOP SET FOR TTL DATA LOOPBACK. THE PROGRAM CHECKS
:* THAT THE RECEIVER NEVER BECOMES ACTIVE, BECAUSE THE RCV CLOCK IS INHIBITED
:* WHEN THE HDX BIT IS SET.

```
6412
6413 032254
(3) 032254
6414 032254 012737 032452 002346
6415 032262 004737 003276
6416 032266 004737 010324
6417 032272 000226
6418 032274 000301
6419 032276 000000
6420 032300 000000
6421 032302 012737 000013 002364
6422 032310 112737 000020 002352
6423 032316 004737 003450
6424 032322 012737 000014 002364
6425 032330 112737 000140 002352
6426 032336 004737 003450
6427 032342 012737 000140 002416
6428 032350 012737 000002 002366
6429 032356 112737 000000 002360
6430 032364 112737 000001 002362
6431 032372 004737 004012
6432 032376 004737 004754
6433 032402 000013
6434 032404 112737 000000 002360
6435 032412 112737 000000 002362
6436 032420 004737 004012
6437 032424 004737 004754
6438 032430 000010
6439 032432 004737 004012
6440 032436 004737 004754
6441 032442 000013
6442 032444 004737 006260
6443 032450 000000
6444 032452 005037 002416
6445 032456 004737 003276
6446 032462
(3) 032462
(3) 032462 104401
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463
6464
```

:BGNTST
T22::
MOV #24\$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,SETUP ;PROGRAM USYRT FOR CHAR MODE, NO CRC
SYNCH
CRC2!CRC1!DDCMP
000
000
MOV #13,REGNUM ;SET LU REG NO. = 13
MOVB #HDX,WRIBYT ;SET HDX BIT IN REG 13
JSR PC,WRITLU
MOV #14,REGNUM ;SET LU REG NO. = 14
MOVB #TXEN!DISSI,WRIBYT ;DISABLE SILOS
JSR PC,WRITLU
MOV #TXEN!DISSI,DISILO
MOV #2,AXNUM ;SET AX BYTE NO FOR AX1
MOVB #000,WAX15 ;SET TSOM IN USYRT
MOVB #TSOM,WAX16
JSR PC,WRITAX
JSR PC,STPLU ;CLOCK FIRST SYNCH OUT
11.
MOVB #000,WAX15 ;LOAD FIRST 000 DATA CHAR INTO USYRT
MOVB #000,WAX16
JSR PC,WRITAX
JSR PC,STPLU ;CLOCK SECOND SYNCH
8.
JSR PC,WRITAX ;LOAD SECOND 000 CHAR
JSR PC,STPLU ;CLOCK FIRST 000 CHAR OUT
11.
JSR PC,IACTIV ;CHK FOR IACT = 0 (RECEIVER NOT ACTIVE)
0
24\$: CLR DISILO ;CLEAR DISABLE SILO FLAG
JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST
L10062: TRAP C\$ETST

```
*****  
:SBTTL TEST 23 - INTERACTION OF MODEM CONTROL BITS  
:  
:* THIS TEST WILL BE RUN ONLY IF THE P-TABLE FOR THIS UNIT INDICATES THAT THE  
:* H3254 AND H3255 TEST CONNECTORS ARE INSTALLED. OTHERWISE, THE TEST WILL BE  
:* SKIPPED FOR THE UNIT.  
:* THE FOLLOWING SUBTESTS ARE PERFORMED:  
:* - A MASTER CLEAR IS DONE AND REG 13 IS READ AND CHECKED FOR INITIALIZED  
:* STATE, WITH LULOOK SET TO 1. THEN, LULOOK IS CLEARED AND REG 13 IS READ  
:* AND CHECKED FOR THE PROPER STATE, WITH LULOOK CLEARED.  
:* REG 13 IS THEN LOADED WITH 0'S, AND READ AND CHECKED FOR THE INITIALIZED  
:* STATE.  
:* REG 17 IS THEN READ AND CHECKED FOR INITIALIZED STATE.
```

6465
 6466
 6467
 6468
 6469
 6470
 6471
 6472
 6473
 6474
 6475
 6476
 6477
 (3)
 6478
 6479
 6480
 6481
 6482
 6483
 6484
 6485
 6486
 (3)
 (3)
 6487
 6488
 6489
 6490
 6491
 6492
 6493
 6494
 6495
 6496
 (4)
 (5)
 (5)
 (5)
 6497
 (3)
 (3)
 6498
 6499
 6500
 6501
 6502
 6503
 6504
 6505
 6506
 (4)
 (5)
 (5)
 (5)
 6507

032464
 032464
 032464 012737 000027 002434
 032472 012737 033764 002346
 032500 004737 011016
 032504 100000

 032506
 032506
 032506 104402
 032510 004737 003276
 032514 012737 000013 002364
 032522 004737 003372
 032526 023727 002350 000210
 032534 001416
 032536 012737 000210 002370
 032544 013737 002350 002372
 032552 004737 004226

 032556
 032556 104455
 032560 000003
 032562 013305
 032564 015406

 032566
 032566 104410
 032570 000170
 032572 142777 000010 147640
 032600 004737 003372
 032604 023727 002350 000000
 032612 001416
 032614 012737 000000 002370
 032622 013737 002350 002372
 032630 004737 004226

 032634
 032634 104455
 032636 000002
 032640 013246
 032642 015406
 032644

```

:* - RUN IS SET IN BSEL1, AND REG 13 IS READ AND CHECKED FOR RING SET.
:* - POLL IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR SIGQ SET.
:* - BPOLL IS SET IN REG 12, ONLY TO LIGHT THE LED FOR THIS SIGNAL.
:* - DTR IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR DTR AND MODR SET.
:* - SELFR IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR SGR SET.
:* - HDX IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR HDX SET.
:* - MAINT1 IS SET IN REG 13, AND REG 17 IS READ AND CHECKED FOR TEST MODE SET.
:* - SELSBY IS SET IN REG 13, AND REG 13 IS READ AND CHECKED FOR STBY SET.
:* - A MASTER CLEAR IS DONE, 2 TSOM'S ARE LOADED INTO THE TX SILO, THE LINE
  UNIT IS CLOCKED UNTIL THE TRANSMITTER IS ACTIVE, AND REG 13 IS READ AND
  CHECKED FOR RTS, CS, CARR SET.
  
```

```

*****
BGNTST
T23::
MOV #23,,TSTNUM ;SET TEST NO.
MOV #A12,RETADR ;SET TEST EXIT ADRS FOR ERRORS
JSR PC,CKLPBK ;SEE IF H3254,5 INSTALLED - SKIP TEST IF NOT
TCCHEK
  
```

```

-----
: DO MASTER CLEAR, CHK REGS 13,17 FOR INITIALIZED STATES
-----
  
```

```

BGNSUB
T23.1: TRAP CSBSUB
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
MOV #13,REGNUM ;SET REG NO. = 13
JSR PC,READLU ;READ REG 13
CMP REDBYT,#RING!MODR ;CHECK REG 13 FOR INIT'D STATE
BEQ 6$ ;BR IF REG 13 INIT'D
MOV #RING!MODR,GOODAT ;SET EXPECTED DATA
MOV REDBYT,BADDAT ;SET ACTUAL DATA
3$: JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT REG MISCMPARE
ERRDF 3,EM3,ERR2
TRAP CSERDF
.WORD 3
.WORD EM3
.WORD ERR2
ESCAPE SUB
TRAP C$ESCAPE
.WORD L10064-.
6$: BICB #LULOOP,@BSEL1 ;CLEAR LULOOP
JSR PC,READLU ;READ REG 13
CMP REDBYT,#0 ;CHECK FOR INITIALIZED STATE
BEQ 8$ ;BR IF OK
MOV #0,GOODAT ;GET EXPECTED DATA
MOV REDBYT,BADDAT ;GET ACTUAL DATA
JSR PC,GETALL ;GET REGS FOR PRINTOUT
;REPORT REG NOT INITIALIZED BY MASTER CLEAR
ERRDF 2,EM2,ERR2
TRAP CSERDF
.WORD 2
.WORD EM2
.WORD ERR2
ESCAPE SUB
  
```

```
(3) 032644 104410
(3) 032646 000112
6508 032650 005037 002352      8$: CLR WRIBYT ;SET DATA = 0 TO BE WRITTEN
6509 032654 004737 003450      JSR PC,WRITLU ;LOAD 0'S INTO REG 13
6510 032660 004737 003372      JSR PC,READLU ;READ REG 13
6511 032664 023727 002350 000000  CMP REDBYT,#000 ;CHECK FOR REG 13 CLEARED
6512 032672 001407          BEQ 9$ ;BR IF CLEARED
6513 032674 012737 000000 002370  MOV #000,GOODAT ;SET EXPECTED DATA
6514 032702 013737 002350 002372  MOV REDBYT,BADDAT ;SET ACTUAL DATA
6515 032710 000720          BR 3$ ;GO PRINT ERROR
6516 032712 012737 000017 002364  9$: MOV #17,REGNUM ;SET REG NO. = 17
6517 032720 004737 003372      JSR PC,READLU ;READ REG 17
6518 032724 042737 000002 002350  BIC #MCLK,REDBYT ;IGNORE MCLK BIT
6519 032732 123727 002350 000051  CMPB REDBYT,#TXDATA!ICIR!DDCMP ;CHK REG 17 FOR INIT'D STATE
6520 032740 001407          BEQ 10$ ;BR IF REG 17 INITIALIZED
6521 032742 012737 000051 002370  MOV #TXDATA!ICIR!DDCMP,GOODAT ;SET EXPECTED DATA
6522 032750 013737 002350 002372  MOV REDBYT,BADDAT ;SET ACTUAL DATA
6523 032756 000675          BR 3$ ;GO REPORT ERROR
6524 032760
6525 032760      10$: ENDSUB
(3) 032760
(3) 032760 104403
6526
6527
6528 :-----
6529 : SET RUN IN BSEL1, CHECK FOR RING SET IN REG 13
6530 :-----
6530 032762      BGNSUB
(3) 032762
(3) 032762 104402
6531 032764 004737 003276      JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6532 032770 105077 147444      CLRB @BSEL1 ;CLEAR LULOOP
6533 032774 112777 000200 147436  MOVB #RUN,@BSEL1 ;SET RUN BIT IN BSEL1
6534 033002 112777 000010 147430  MOVB #LULOOP,@BSEL1 ;CLEAR RUN, SET LULOOP
6535 033010 012737 000013 002364  MOV #13,REGNUM ;SET REG NO. = 13
6536 033016 004737 003372      JSR PC,READLU ;READ REG 13
6537 033022 132737 000200 002350  BITB #RING,REDBYT ;SEE IF RING = 1
6538 033030 001010          BNE 9$ ;BR IF RING = 1
6539 033032 004737 004226      JSR PC,GETALL ;GET REGS FOR PRINTOUT
6540 ;REPORT RING NOT SET
6541 033036      ERRDF 56,EM56,ERR7
(4) 033036 104455
(5) 033040 000070
(5) 033042 014617
(5) 033044 020244
6542 033046      ESCAPE SUB
(3) 033046 104410
(3) 033050 000002
6543 033052      9$: ENDSUB
6544 033052
(3) 033052
(3) 033052 104403
6545
6546
6547 :-----
6548 : SET POLL IN REG 13, CHK FOR SIGO SET IN REG 17
6549 :-----
6549 033054      BGNSUB
```

TRAP C\$ESCAPE
.WORD L10064-

L10064: TRAP C\$ESUB

T23.2: TRAP C\$BSUB

TRAP C\$ERDF
.WORD 56
.WORD EM56
.WORD ERR7

TRAP C\$ESCAPE
.WORD L10065-

L10065: TRAP C\$ESUB

```
(3) 033054
(3) 033054 104402
6550 033056 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6551 033062 112737 000200 002352 MOVB #POLL,WRIBYT ;SET REG NO. = 13
6552 033070 012737 000013 002364 MOV #13,REGNUM ;SET POLL IN REG 13
6553 033076 004737 003450 JSR PC,WRITLU ;SET REG NO. = 17
6554 033102 012737 000017 002364 MOV #17,REGNUM ;READ REG 17
6555 033110 004737 003372 JSR PC,READLU ;SEE IF SIGQ = 1
6556 033114 132737 000100 002350 BITB #SIGQ,REDBYT ;BR IF SIGQ = 1
6557 033122 001006 BNE 6$ ;GET REGS FOR PRINTOUT
6558 033124 004737 004226 JSR PC,GETALL
6559 ;REPORT SIGQ NOT SET
6560 033130 ERRDF 63,EM63,ERR7
(4) 033130 104455 TRAP C$ERDF
(5) 033132 000077 .WORD 63
(5) 033134 014746 .WORD EM63
(5) 033136 020244 .WORD ERR7
6561 033140 6$:
6562 033140 ENDSUB
(3) 033140
(3) 033140 104403 L10066: TRAP C$ESUB
6563
6564 -----
6565 ; SET BPOLL IN REG 12, TO LIGHT LED ONLY
6566 -----
6567 033142 BGNSUB
(3) 033142
(3) 033142 104402 T23.4: TRAP C$BSUB
6568 033144 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6569 033150 012737 000012 002364 MOV #12,REGNUM ;SET LU REG NO. = 12
6570 033156 112737 000100 002352 MOVB #BPOLL,WRIBYT ;SET BPOLL IN LU REG 12
6571 033164 004737 003450 JSR PC,WRITLU
6572 033170 ENDSUB
(3) 033170
(3) 033170 104403 L10067: TRAP C$ESUB
6573
6574 -----
6575 ; SET DTR IN REG 13, CHECK FOR DTR AND MODR SET IN REG 13
6576 -----
6577 033172 BGNSUB
(3) 033172
(3) 033172 104402 T23.5: TRAP C$BSUB
6578 033174 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6579 033200 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
6580 033206 112737 000104 002352 MOVB #DTR!MAINT2,WRIBYT
6581 033214 004737 003450 JSR PC,WRITLU ;SET DTR IN REG 13
6582 ; (ALSO SET MAINT2 FOR MANUFACT. TEST CONN.)
6583 033220 142777 000010 147212 BICB #LULOOP,@BSEL1 ;CLEAR LULOOP
6584 033226 004737 003372 JSR PC,READLU ;READ REG 13
6585 033232 132737 000100 002350 BITB #DTR,REDBYT ;SEE IF DTR = 1
6586 033240 001010 BNE 6$ ;BR IF DTR = 1
6587 033242 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6588 ;REPORT DTR NOT SET
6589 033246 ERRDF 55,EM55,ERR7
(4) 033246 104455 TRAP C$ERDF
(5) 033250 000067 .WORD 55
```

```
(5) 033252 014603 .WORD EM55
(5) 033254 020244 .WORD ERR7
6590 033256 ESCAPE SUB
(3) 033256 104410 TRAP C$ESCAPE
(3) 033260 000026 .WORD L10070-
6591 033262 132737 000010 002350 6$: BITB #MODR,REDBYT ;SEE IF MODR = 1
6592 033270 001006 BNE 12$ ;BR IF MODR = 1
6593 033272 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6594 ;REPORT MODR NOT SET
6595 ERRDF 57,EM57,ERR7
(4) 033276 104455 TRAP C$ERDF
(5) 033300 000071 .WORD 57
(5) 033302 014634 .WORD EM57
(5) 033304 020244 .WORD ERR7
6596 033306 12$: ENDSUB
6597 033306 L10070: TRAP C$ESUB
(3) 033306 104403
6598
6599
6600 ;-----
6601 ; SET SELFR IN REG 13, CHK FOR SIGR SET IN REG 17
6602 ;-----
6602 033310 BGNSUB
(3) 033310 T23.6: TRAP C$BSUB
(3) 033310 104402
6603 033312 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6604 033316 112737 000040 002352 MOVB #SEFR,WRIBYT
6605 033324 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
6606 033332 004737 003450 JSR PC,WRITLU ;SET SELFR IN REG 13
6607 033336 012737 000017 002364 MOV #17,REGNUM ;SET REG NO. = 17
6608 033344 004737 003372 JSR PC,READLU ;READ REG 17
6609 033350 132737 000200 002350 BITB #SIGR,REDBYT ;SEE IF SIGR = 1
6610 033356 001006 BNE 6$ ;BR IF SIGR = 1
6611 033360 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6612 ;REPORT SIGR NOT SET
6613 ERRDF 64,EM64,ERR7
(4) 033364 104455 TRAP C$ERDF
(5) 033366 000100 .WORD 64
(5) 033370 014763 .WORD EM64
(5) 033372 020244 .WORD ERR7
6614 033374 6$: ENDSUB
6615 033374 L10071: TRAP C$ESUB
(3) 033374 104403
6616
6617 ;-----
6618 ; SET HDX IN REG 13, CHK FOR HDX SET IN REG 13
6619 ;-----
6620 033376 BGNSUB
(3) 033376 T23.7: TRAP C$BSUB
(3) 033376 104402
6621 033400 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6622 033404 112737 000020 002352 MOVB #HDX,WRIBYT
6623 033412 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
6624 033420 004737 003450 JSR PC,WRITLU ;SET HDX IN REG 13
6625 033424 004737 003372 JSR PC,READLU ;READ REG 13
```

```
6626 033430 132737 000020 002350 BITB #HDX,REDBYT ;SEE IF HDX = 1
6627 033436 001006 BNE 6$ ;BR IF HDX = 1
6628 033440 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6629 ;REPORT HDX NOT SET
6630 033444 ERRDF 58,EM58,ERR7
(4) 033444 104455 TRAP C$ERDF
(5) 033446 000072 .WORD 58
(5) 033450 014651 .WORD EM58
(5) 033452 020244 .WORD ERR7
6631 033454 6$:
6632 033454 ENDSUB
(3) 033454 L10072:
(3) 033454 104403 TRAP C$ESUB
6633
6634 -----
6635 ; SET MAINT1 IN REG 13, CHK FOR TEST MODE SET IN REG 17
6636 -----
6637 033456 BGNSUB
(3) 033456 T23.8:
(3) 033456 104402 TRAP C$BSUB
6638 033460 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6639 033464 112737 000010 002352 MOVB #MAINT1,WRIBYT
6640 033472 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
6641 033500 004737 003450 JSR PC,WRITLU ;SET MAINT1 IN REG 13
6642 033504 012737 000017 002364 MOV #17,REGNUM ;SET REG NO. = 17
6643 033512 142777 000010 146720 BICB #LULOOP,@BSEL1 ;CLEAR LULOOP
6644 033520 004737 003372 JSR PC,READLU ;READ REG 17
6645 033524 132737 000004 002350 BITB #TESTMD,REDBYT ;SEE IF TESTMD = 1
6646 033532 001006 BNE 6$ ;BR IF TESTMD = 1
6647 033534 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6648 ;REPORT TEST MODE NOT SET BY MAINT1
6649 033540 ERRDF 52,EM52,ERP7
(4) 033540 104455 TRAP C$ERDF
(5) 033542 000064 .WORD 52
(5) 033544 014471 .WORD EM52
(5) 033546 020244 .WORD ERR7
6650 033550 6$:
6651 033550 ENDSUB
(3) 033550 L10073:
(3) 033550 104403 TRAP C$ESUB
6652
6653 -----
6654 ; SET SELSBY IN REG 13, CHK FOR STBY SET IN REG 13
6655 -----
6656 033552 BGNSUB
(3) 033552 T23.9:
(3) 033552 104402 TRAP C$BSUB
6657 033554 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6658 033560 112737 000002 002352 MOVB #SELSBY,WRIBYT
6659 033566 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
6660 033574 004737 003450 JSR PC,WRITLU ;SET SELSBY IN REG 13
6661 033600 004737 003372 JSR PC,READLU ;READ REG 13
6662 033604 132737 000002 002350 BITB #STBY,REDBYT ;SEE IF STBY = 1
6663 033612 001006 BNE 6$ ;BR IF STBY = 1
6664 033614 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6665 ;REPORT STBY NOT SET
```



```
6666 033620 ERRDF 59,EM59,ERR7
(4) 033620 104455
(5) 033622 000073 TRAP C$ERDF
(5) 033624 014665 .WORD 59
(5) 033626 020244 .WORD EM59
6667 033630 6$: ENDSUB .WORD ERR7
6668 033630
(3) 033630
(3) 033630 104403 L10074: TRAP C$ESUB
6669
6670
6671
6672
6673 033632
(3) 033632
(3) 033632 104402 T23.10: TRAP C$BSUB
6674 033634 004737 005240 JSR PC,INITRN ;MST CLR, LOAD SOM'S, CLK TRANSMITTER
6675 033640 000000 000
6676 033642 000000 000
6677 033644 012737 000013 002364 MOV #13,REGNUM ;SET REG NO. = 13
6678 033652 004737 003372 JSR PC,READLU ;READ REG 13
6679 033656 132737 000040 002350 BITB #RTS,REDBYT ;SEE IF RTS = 1
6680 033664 001010 BNE 6$ ;BR IF RTS = 1
6681 033666 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6682 ;REPORT RTS NOT SET
6683 033672 ERRDF 60,EM60,ERR7
(4) 033672 104455 TRAP C$ERDF
(5) 033674 000074 .WORD 60
(5) 033676 014702 .WORD EM60
(5) 033700 020244 .WORD ERR7
6684 033702 ESCAPE SUB TRAP C$ESCAPE
(3) 033702 104410 .WORD L10075-.
(3) 033704 000056
6685 033706 132737 000004 002350 6$: BITB #CS,REDBYT ;SEE IF CS = 1
6686 033714 001010 BNE 9$ ;BR IF CS = 1
6687 033716 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6688 ;REPORT CS NOT SET
6689 033722 ERRDF 61,EM61,ERR7
(4) 033722 104455 TRAP C$ERDF
(5) 033724 000075 .WORD 61
(5) 033726 014716 .WORD EM61
(5) 033730 020244 .WORD ERR7
6690 033732 ESCAPE SUB TRAP C$ESCAPE
(3) 033732 104410 .WORD L10075-.
(3) 033734 000026
6691 033736 132737 000001 002350 9$: BITB #CARR,REDBYT ;SEE IF CARR = 1
6692 033744 001006 BNE 12$ ;BR IF CARR = 1
6693 033746 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6694 ;REPORT CARR NOT SET
6695 033752 ERRDF 62,EM62,ERR7
(4) 033752 104455 TRAP C$ERDF
(5) 033754 000076 .WORD 62
(5) 033756 014731 .WORD EM62
(5) 033760 020244 .WORD ERR7
6696 033762
6697 033762 12$: ENDSUB
```

(3) 033762 L10075: TRAP C\$ESUB
(3) 033762 104403
6698
6699 033764
6700 033764 004737 003276 A12: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
6701 033770
(3) 033770 L10063: TRAP C\$ETST
(3) 033770 104401
6702
6703
6704
6705
6706
6707

SBTTL TEST 24 - DATA TEST - BIT MODE, NO ERR DET

*
* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH ERROR DETECTION
* INHIBITED. THE MESSAGE CONSISTS OF 5 FLAGS, PAT A REPEATED 2 TIMES,
* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
* TEST WILL NOT BE RUN.
* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
* 375,373,367,357,337,277,177
* 8-BIT CHARACTERS ARE USED.

BGNTST

6721 033772
(3) 033772 T24::
6722 033772 012737 000030 002434 MOV #24,,TSTNUM ;SET TEST NO.
6723 034000 012737 034160 002346 MOV #24\$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
6724 034006 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6725 034012 004737 011016 JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
6726 034016 000000 0
6727 034020 013737 002422 034036 MOV MODINT,6\$;SET MODEM SELECTION
6728 034026 004737 010324 JSR PC,SETUP ;PROGRAM THE USYRT
6729 034032 000000 000
6730 034034 000300 CRC2!CRC1 ;BIT MODE, NO ERR DET
6731 034036 000000 6\$: .WORD 0 ;MODEM SELECTION GOES HERE
6732 034040 000000 000
6733 034042 004737 010204 JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
6734 034046 012737 001177 003100 MOV #RXEBL!177,RCVBUF+78. ; SET LAST DATA CHAR IN BUFFER
6735 034054 142777 000010 146356 BICB #LULOOP,2\$SEL1 ;CLEAR LULOOP, CLOCK MSG
6736 034062 012737 000012 002364 MOV #12,REGNUM ;SET LU REG NO. = 12
6737 034070 012703 002762 MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
6738 034074 013702 002264 9\$: MOV TCOUNT,R2 ;INIT TIMER
6739 034100 004737 003372 10\$: JSR PC,READLU ;READ REG 12
6740 034104 132737 000020 002350 BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
6741 034112 001011 BNE 12\$;BR IF YES
6742 034114 005202 INC R2 ;INCREMENT TIMER
6743 034116 001370 BNE 10\$;BR IF NO TIME-OUT YET
6744 034120 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6745 ;REPORT IRDY NOT SET
6746 034124 ERRDF 17,EM17,ERR7
(4) 034124 104455 TRAP C\$ERRDF
(5) 034126 000021 .WORD 17

```
(5) 034130 013456  
(5) 034132 020244  
6747 034134 000411  
6748 034136 012337 034146  
6749 034142 004737 007314  
6750 034146 000000  
6751 034150 000000  
6752 034152 020327 003102  
6753 034156 103746  
6754 034160  
6755 034160  
(3) 034160  
(3) 034160 104401
```

```
BR 24$ ;ESCAPE TO END OF TEST  
MOV (R3)+,16$  
JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED  
12$: 0  
16$: 0  
CMP R3,#RCVBUF+80. ;SEE IF ALL CHARS CHECKED YET  
BLO 9$ ;BR IF NOT YET  
24$:  
ENDTST
```

L10076: TRAP CSETST

6756
6757
6758
6759
6760
6761
6762
6763
6764
6765
6766
6767
6768
6769
6770
6771
6772
6773
6774

```
::*****  
:SBTTL TEST 25 - DATA TEST - CHAR MODE, NO ERR DET  
:*  
:* A MESSAGE IS INITIATED IN CHAR MODE, WITH ERROR DETECTION  
:* INHIBITED. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,  
:* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
:* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.  
:* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
:* TEST WILL NOT BE RUN.  
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
:* 375,373,367,357,337,277,177  
:* 8-BIT CHARACTERS ARE USED.  
:*****
```

```
BGNTST  
T25::  
MOV #25.,TSTNUM ;SET TEST NO.  
MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS  
JSR PC,MSTCLR ;ISSUE MASTER CLEAR  
JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION  
0  
MOV MODINT,6$ ;SET MODEM SELECTION  
JSR PC,SETUP ;PROGRAM THE USYRT  
SYNCH  
CRC2!CRC1!STRIP!DDCMP ;CHAR MODE, NO ERR DET  
6$: .WORD 0 ;MODEM SELECTION GOES HERE  
000  
JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF  
MOV #177,RCVBUF+78. ;SET LAST DATA CHAR IN BUFFER  
BICB #LULOOP,@BSEL1 ;CLEAR LULOOP, CLOCK MSG  
MOV #12,REGNUM ;SET LU REG NO. = 12  
MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF  
9$: MOV TCOUNT,R2 ;INIT TIMER  
10$: JSR PC,READLU ;READ REG 12  
BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET  
BNE 12$ ;BR IF YES  
INC R2 ;INCREMENT TIMER  
BNE 10$ ;BR IF NO TIME-OUT YET
```

```
6775 034162  
(3) 034162  
6776 034162 012737 000031 002434  
6777 034170 012737 034350 002346  
6778 034176 004737 003276  
6779 034202 004737 011016  
6780 034206 000000  
6781 034210 013737 002422 034226  
6782 034216 004737 010324  
6783 034222 000226  
6784 034224 000311  
6785 034226 000000  
6786 034230 000000  
6787 034232 004737 010204  
6788 034236 012737 000177 003100  
6789 034244 142777 000010 146166  
6790 034252 012737 000012 002364  
6791 034260 012703 002762  
6792 034264 013702 002264  
6793 034270 004737 003372  
6794 034274 132737 000020 002350  
6795 034302 001011  
6796 034304 005202  
6797 034306 001370
```

6798 034310 004737 004226
6799
6800 034314
(4) 034314 104455
(5) 034316 000021
(5) 034320 013456
(5) 034322 020244
6801 034324 000411
6802 034326 012337 034336
6803 034332 004737 007314
6804 034336 000000
6805 034340 000000
6806 034342 020327 003102
6807 034346 103746
6808 034350
6809 034350
(3) 034350
(3) 034350 104401

```

;REPORT JSR PC,GETALL ;GET REGS FOR PRINTOUT
ERRDF 17,EM17,ERR7
TRAP C$ERDF
.WORD 17
.WORD EM17
.WORD ERR7

12$: BR 24$ ;ESCAPE TO END OF TEST
MOV (R3)+,16$
JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED

16$: 0
0
CMP R3,#RCVBUF+80. ;SEE IF ALL CHARS CHECKED YET
BLO 9$ ;BR IF NOT YET

24$:
ENDTST

L10077:
TRAP C$SETST

```

6810
6811
6812
6813
6814
6815
6816
6817
6818
6819
6820
6821
6822
6823
6824
6825
6826
6827
6828

```

:*****
:SBTTL TEST 26 - DATA TEST - BIT MODE, CRC-CCITT-1
:*
:* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH CRC-CCITT-1 ERROR
:* DETECTION. THE MESSAGE CONSISTS OF 5 FLAGS, PAT A REPEATED 2 TIMES,
:* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
:* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
:* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
:* TEST WILL NOT BE RUN.
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
:* 375,373,367,357,337,277,177
:* 8-BIT CHARACTERS ARE USED.
:*****
BGNTST

```

6829 034352
(3) 034352
6830 034352 012737 000032 002434
6831 034360 012737 034540 002346
6832 034366 004737 003276
6833 034372 004737 011016
6834 034376 000000
6835 034400 013737 002422 034416
6836 034406 004737 010324
6837 034412 000000
6838 034414 000000
6839 034416 000000
6840 034420 000000
6841 034422 004737 010204
6842 034426 012737 101177 003100
6843 034434 142777 000010 145776
6844 034442 012737 000012 002364
6845 034450 012703 002762
6846 034454 013702 002264

```

726::
MOV #26,,TSTNUM ;SET TEST NO.
MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC,MSTCLR ;ISSUE MASTER CLEAR
JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
0
MOV MODINT,6$ ;SET MODEM SELECTION
JSR PC,SETUP ;PROGRAM THE USYRT
000
000 ;BIT MODE CRC-CCITT-1
6$: .WORD 0 ;MODEM SELECTION GOES HERE
000
JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
MOV #CRCCHK!RXEBL!177,RCVBUF+78. ;SET LAST DATA CHAR IN BUFFER
BICB #LULOOP,@BSEL1 ;CLEAR LULOOP, CLOCK MSG
MOV #12,REGNUM ;SET LU REG NO. = 12
MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
9$: MOV TCOUNT,R2 ;INIT TIMER

```

```
6847 034460 004737 003372 10$: JSR PC,READLU ;READ REG 12
6848 034464 132737 000020 002350 BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
6849 034472 001011 BNE 12$ ;BR IF YES
6850 034474 005202 INC R2 ;INCREMENT TIMER
6851 034476 001370 BNE 10$ ;BR IF NO TIME-OUT YET
6852 034500 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
6853 :REPORT IRDY NOT SET
6854 034504 ERRDF 17,EM17,ERR7
(4) 034504 104455 TRAP C$ERDF
(5) 034506 000021 .WORD 17
(5) 034510 013456 .WORD EM17
(5) 034512 020244 .WORD ERR7
6855 034514 000411 BR 24$ ;ESCAPE TO END OF TEST
6856 034516 012337 034526 12$: MOV (R3)+,16$
6857 034522 004737 007314 JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
6858 034526 000000 16$: 0
6859 034530 000000 0
6860 034532 020327 003102 CMP R3,#RCVBUF+80. ;SEE IF ALL CHARS CHECKED YET
6861 034536 103746 BLO 9$ ;BR IF NOT YET
6862 034540
6863 034540 24$:
(3) 034540 ENDTST
(3) 034540 104401 L10100: TRAP C$ETST
```

6864
6865
6866
6867
6868
6869
6870
6871
6872
6873
6874
6875
6876
6877
6878
6879
6880
6881
6882

```
::*****  
:SBTTL TEST 27 - DATA TEST - BIT MODE, CRC-CCITT-0  
:*  
:* A MESSAGE IS INITIATED IN BIT-STUFF MODE, WITH CRC-CCITT-0 ERROR  
:* DETECTION. THE MESSAGE CONSISTS OF 5 FLAGS, PAT A REPEATED 2 TIMES,  
:* AND 2 FLAGS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,  
:* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.  
:* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE  
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE  
:* TEST WILL NOT BE RUN.  
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,  
:* 375,373,367,357,337,277,177  
:* 8-BIT CHARACTERS ARE USED.  
:*****
```

```
6883 034542 BGNTST  
(3) 034542 T27::  
6884 034542 012737 000033 002434 MOV #27.,TSTNUM ;SET TEST NO.  
6885 034550 012737 034730 002346 MOV #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS  
6886 034556 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR  
6887 034562 004737 011016 JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION  
6888 034566 000000 0  
6889 034570 013737 002422 034606 MOV MODINT,6$ ;SET MODEM SELECTION  
6890 034576 004737 010324 JSR PC,SETUP ;PROGRAM THE USYRT  
6891 034602 000000 000  
6892 034604 000100 CRC1 ;BIT MODE, CRC-CCITT-0  
6893 034606 000000 6$: .WORD 0 ;MODEM SELECTION GOES HERE  
6894 034610 000000 000  
6895 034612 004737 010204 JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
```

```

6896 034616 012737 101177 003100      MOV      #CRCCHK!RXEBL!177,RCVBUF+78. ;SET LAST DATA CHAR IN BUFFER
6897 034624 142777 000010 145606      BICB     #LULOOP,@BSL1 ;CLEAR LULOOP, CLOCK MSG
6898 034632 012737 000012 002364      MOV      #12,REGNUM ;SET LU REG NO. = 12
6899 034640 012703 002762      MOV      #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
6900 034644 013702 002264      9$:     MOV      TCOUNT,R2 ;INIT TIMER
6901 034650 004737 003372      10$:    JSR      PC,READLU ;READ REG 12
6902 034654 132737 000020 002350      BITB     #IRDY,REDBYT ;SEE IF IRDY IS SET YET
6903 034662 001011      BNE      12$ ;BR IF YES
6904 034664 005202      INC      R2 ;INCREMENT TIMER
6905 034666 001370      BNE      10$ ;BR IF NO TIME-OUT YET
6906 034670 004737 004226      JSR      PC,GETALL ;GET REGS FOR PRINTOUT
6907      :REPORT IRDY NOT SET
6908 034674      ERRDF   17,EM17,ERR7
(4) 034674 104455      TRAP     C$ERDF
(5) 034676 000021      .WORD   17
(5) 034700 013456      .WORD   EM17
(5) 034702 020244      .WORD   ERR7
6909 034704 000411      BR       24$ ;ESCAPE TO END OF TEST
6910 034706 012337 034716 12$:     MOV      (R3)+,16$
6911 034712 004737 007314      JSR      PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
6912 034716 000000      16$:     0
6913 034720 000000      0
6914 034722 020327 003102      CMP      R3,#RCVBUF+80. ;SEE IF ALL CHARS CHECKED YET
6915 034726 103746      BLO      9$ ;BR IF NOT YET
6916 034730      24$:
6917 034730      ENDTST
(3) 034730      L10101:
(3) 034730 104401      TRAP     C$ETST
6918
6919
6920
6921
6922
6923
6924
6925
6926
6927
6928
6929
6930
6931
6932
6933
6934
6935
6936
6937 034732
(3) 034732
6938 034732 012737 000034 002434      MOV      #28.,TSTNUM ;SET TEST NO.
6939 034740 012737 035120 002346      MOV      #24$,RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
6940 034746 004737 003276      JSR      PC,MSTCLR ;ISSUE MASTER CLEAR
6941 034752 004737 011016      JSR      PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
6942 034756 000000      0
6943 034760 013737 002422 034776      MOV      MODINT,6$ ;SET MODEM SELECTION
6944 034766 004737 010324      JSR      PC,SETUP ;PROGRAM THE USYRT
  
```

```

*****
:SBTTL      TEST 28 - DATA TEST - CHAR MODE, CRC-16
:*
:* A MESSAGE IS INITIATED IN CHAR MODE, WITH CRC-16 ERROR
:* DETECTION. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,
:* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
:* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
:* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
:* TEST WILL NOT BE RUN.
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
:*             375,373,367,357,337,277,177
:* 8-BIT CHARACTERS ARE USED.
*****
BGNTST
  
```

T28::

```

6945 034772 000226          SYNCH
6946 034774 000011          STRIP!DDCMP
6947 034776 000000          6$: .WORD 0 ;MODEM SELECTION GOES HERE
6948 035000 000000          000
6949 035002 004737 010204    JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
6950 035006 012737 100577 003100  MOV #CRCCHK!RXBCC!177,RCVBUF+78. ;SET LAST DATA CHAR IN BUFFER
6951 035014 142777 000010 145416  BICB #LULOOP, @BSEL1 ;CLEAR LULOOP, CLOCK MSG
6952 035022 012737 000012 002364  MOV #12,REGNUM ;SET LU REG NO. = 12
6953 035030 012703 002762          MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
6954 035034 013702 002264          9$: MOV TCOUNT,R2 ;INIT TIMER
6955 035040 004737 003372          10$: JSR PC,READLU ;READ REG 12
6956 035044 132737 000020 002350  BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
6957 035052 001011          BNE 12$ ;BR IF YES
6958 035054 005202          INC R2 ;INCREMENT TIMER
6959 035056 001370          BNE 10$ ;BR IF NO TIME-OUT YET
6960 035060 004737 004226    JSR PC,GETALL ;GET REGS FOR PRINTOUT
6961          ;REPORT IRDY NOT SET
6962 035064          ERRDF 17,EM17,ERR7
(4) 035064 104455          TRAP C$ERDF
(5) 035066 000021          .WORD 17
(5) 035070 013456          .WORD EM17
(5) 035072 020244          .WORD ERR7
6963 035074 000411          BR 24$ ;ESCAPE TO END OF TEST
6964 035076 012337 035106    12$: MOV (R3)+,16$
6965 035102 004737 007314    JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
6966 035106 000000          16$: 0
6967 035110 000000          0
6968 035112 020327 003102    CMP R3,#RCVBUF+80. ;SEE IF ALL CHARS CHECKED YET
6969 035116 103746          BLO 9$ ;BR IF NOT YET
6970 035120          24$:
6971 035120          ENDTST
(3) 035120          L10102:
(3) 035120 104401          TRAP C$ETST
6972
6973
6974
6975
6976
6977
6978          ;*****
6979          .SBTTL TEST 29 - DATA TEST - CHAR MODE, ODD VRC
6980          ;*
6981          ;* A MESSAGE IS INITIATED IN CHAR MODE, WITH ODD VRC ERROR DETECTION
6982          ;* SELECTED. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,
6983          ;* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
6984          ;* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
6985          ;* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
6986          ;* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
6987          ;* TEST WILL NOT BE RUN.
6988          ;* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
6989          ;* 375,373,367,357,337,277,177
6990          ;* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).
6991          ;*****
6991 035122          BGNST
(3) 035122          T29::
6992 035122 012737 000035 002434    MOV #29,.TSTNUM ;SET TEST NO.
6993 035130 012737 035304 002346    MOV #24$.RETADR ;SET TEST EXIT ADDRESS FOR ERRORS

```

```

6994 035136 004737 003276 JSR PC,MSTCLR ;ISSUE MASTER CLEAR
6995 035142 004737 011016 JSR PC,CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
6996 035146 000000 0
6997 035150 013737 002422 035166 MOV MODINT,6$ ;SET MODEM SELECTION
6998 035156 004737 010324 JSR PC,SETUP ;PROGRAM THE USYRT
6999 035162 000026 026
7000 035164 000111 CRC1!STRIP!DDCMP
7001 035166 000000 6$: .WORD 0 ;MODEM SELECTION GOES HERE
7002 035170 000347 TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO
7003 035172 004737 010204 JSR PC,LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
7004 035176 142777 000010 145234 BICB #LULOOP,28SEL1 ;CLEAR LULOOP, CLOCK MSG
7005 035204 012737 000012 002364 MOV #12,REGNUM ;SET LU REG NO. = 12
7006 035212 012703 002762 MOV #RCVBUF,R3 ;GET POINTER TO RCV MSG BUF
7007 035216 013702 002264 9$: MOV TCOUNT,R2 ;INIT TIMER
7008 035222 004737 003372 10$: JSR PC,READLU ;READ REG 12
7009 035226 132737 000020 002350 BITB #IRDY,REDBYT ;SEE IF IRDY IS SET YET
7010 035234 001011 BNE 12$ ;BR IF YES
7011 035236 005202 INC R2 ;INCREMENT TIMER
7012 035240 001370 BNE 10$ ;BR IF NO TIME-OUT YET
7013 035242 004737 004226 JSR PC,GETALL ;GET REGS FOR PRINTOUT
7014 :REPORT IRDY NOT SET
7015 035246 ERRDF 17,EM17,ERR7
(4) 035246 104455 TRAP C$ERDF
(5) 035250 000021 .WORD 17
(5) 035252 013456 .WORD EM17
(5) 035254 020244 .WORD ERR7
7016 035256 000412 BR 24$ ;ESCAPE TO END OF TEST
7017 035260 112337 035272 12$: MOVB (R3)+,16$ ;GET AN EXPECTED DATA BYTE
7018 035264 005203 INC R3 ;INCREMENT POINTER
7019 035266 004737 007314 JSR PC,CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
7020 035272 100000 16$: BCCCHK
7021 035274 000000 0
7022 035276 020327 003102 CMP R3,#RCVBUF+80. ;SEE IF ALL CHARS CHECKED YET
7023 035302 103745 BLO 9$ ;BR IF NOT YET
7024 035304 24$:
7025 035304 ENDTST
(3) 035304 L10103: TRAP C$ETST
(3) 035304 104401
7026
7027
7028
7029
7030
7031
7032
7033
7034
7035
7036
7037
7038
7039
7040
7041
7042
7043
    
```

```

*****
:SBTTL TEST 30 - DATA TEST - CHAR MODE, EVEN VRC
:*
:* A MESSAGE IS INITIATED IN CHAR MODE, WITH EVEN VRC ERROR DETECTION
:* SELECTED. THE MESSAGE CONSISTS OF 5 SYNCHS, PAT A REPEATED 2 TIMES,
:* AND 2 SYNCHS. IF THE H3254 AND H3255 TEST CONNECTORS ARE INSTALLED,
:* THE TEST WILL BE RUN WITH THE V.35 INTERFACE SELECTED.
:* IF EXTERNAL TURNAROUND IS PROVIDED ON A PARTICULAR INTERFACE, THE
:* TEST WILL BE RUN ON THAT INTERFACE. IF THERE IS NO EXTERNAL TURNAROUND, THE
:* TEST WILL NOT BE RUN.
:* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
:* 375,373,367,357,337,277,177
:* 7-BIT CHARACTERS ARE USED. (HI BIT OF A PATTERN CHAR IS NOT USED).
    
```



```
7044
7045 035306
(3) 035306
7046 035306 012737 000036 002434
7047 035314 012737 035470 002346
7048 035322 004737 003276
7049 035326 004737 011016
7050 035332 000000
7051 035334 013737 002422 035352
7052 035342 004737 010324
7053 035346 000026
7054 035350 000211
7055 035352 000000
7056 035354 000347
7057 035356 004737 010204
7058 035362 142777 000010 145050
7059 035370 012737 000012 002364
7060 035376 012703 002762
7061 035402 013702 002264
7062 035406 004737 003372
7063 035412 132737 000020 002350
7064 035420 001011
7065 035422 005202
7066 035424 001370
7067 035426 004737 004226
7068
7069 035432
(4) 035432 104455
(5) 035434 000021
(5) 035436 013456
(5) 035440 020244
7070 035442 000412
7071 035444 112337 035456
7072 035450 005203
7073 035452 004737 007314
7074 035456 100000
7075 035460 000000
7076 035462 020327 003102
7077 035466 103745
7078 035470
7079 035470
(3) 035470
(3) 035470 104401
7080
7081
7082
7083
7084
7085
7086
7087
7088
7089
7090
7091
7092
```

```
*****
BGNTST
T30::
MOV #30, TSTNUM ;SET TEST NO.
MOV #24$, RETADR ;SET TEST EXIT ADDRESS FOR ERRORS
JSR PC, MSTCLR ;ISSUE MASTER CLEAR
JSR PC, CKLPBK ;CHECK LOOPBACK, GET MODEM SELECTION
0
MOV MODINT, 6$ ;SET MODEM SELECTION
JSR PC, SETUP ;PROGRAM THE USYRT
026
CRC2!STRIP!DDCMP
6$: .WORD 0 ;MODEM SELECTION GOES HERE
TXLEN2!TXLEN1!TXLENO!RXLEN2!RXLEN1!RXLENO
JSR PC, LODATA ;LOAD MSG INTO TX SILO AND RCV'D DATA BUF
BICB #LULOOP, @BSEL1 ;CLEAR LULOOP, CLOCK MSG
MOV #12, REGNUM ;SET LU REG NO. = 12
MOV #RCVBUF, R3 ;GET POINTER TO RCV MSG BUF
9$: MOV TCOUNT, R2 ;INIT TIMER
10$: JSR PC, READLU ;READ REG 12
BITB #IRDY, REDBYT ;SEE IF IRDY IS SET YET
BNE 12$ ;BR IF YES
INC R2 ;INCREMENT TIMER
BNE 10$ ;BR IF NO TIME-OUT YET
JSR PC, GETALL ;GET REGS FOR PRINTOUT
:REPORT IRDY NOT SET
ERRDF 17, EM17, ERR7
TRAP C$ERDF
.WORD 17
.WORD EM17
.WORD ERR7
BR 24$ ;ESCAPE TO END OF TEST
12$: MOVB (R3)+, 16$ ;GET AN EXPECTED DATA CHAR
INC R3 ;INCREMENT POINTER
JSR PC, CKDATA ;COMPARE RCV'D DATA CHAR TO EXPECTED
16$: BCCCHK
0
CMP R3, #RCVBUF+80. ;SEE IF ALL CHARS CHECKED YET
BLO 9$ ;BR IF NOT YET
24$:
ENDTST
L10104: TRAP C$ETST
```

```
*****
SBTTL TEST 31 - CONTIGUOUS ONES IN SEC. STA. ADRS. MODE, BIT MODE
*
* IN THIS TEST, A MESSAGE CONSISTING OF 5 ONES CHARS (377 OCT)
* IS SENT IN SECONDARY STATION ADDRESS MODE, WITH THE STATION ADRS
* FOR THIS LINE = 377. THE PROGRAM CHECKS FOR CORRECT RECEPTION OF
* THE FIRST CHARACTER (STATION ADDRESS) AND THE REMAINING 4
* ONES CHARACTERS (DATA). THIS TEST EXERCISES THE SECONDARY STATION
```

```
7093          ;* ADDRESS LOGIC, AND CHECKS THAT THE SEC. STA. ADRS. CAN BE BIT-STUFFED
7094          ;* AND TRANSMITTED AND RECEIVED CORRECTLY.
7095          ;*****
7096 035472      BGNTST
(3) 035472
7097 035472 012737 035576 002346      MOV #24$,RETADR      ;SET TEST EXIT ADDRESS FOR ERRORS
7098 035500 004737 005240      JSR PC,INITRN      ;MST CLR, LOAD 2 SOM'S
7099 035504 000377      377
7100 035506 000320      CRC2!CRC1!SECA      ;BIT MODE, NO ERROR DET, SEC ADR MODE
7101 035510 004737 010746      JSR PC,LODSIL      ;LOAD 5 377-CHARS INTO TX SILO
7102 035514 000377      377
7103 035516 000005      5
7104 035520 004737 010746      JSR PC,LODSIL      ;LOAD 2 EOM'S INTO TX SILO
7105 035524 001000      TXEOM
7106 035526 000002      2
7107 035530 004737 004754      JSR PC,STPLU      ;CLOCK MORE THAN ENTIRE MSG
7108 035534 000240      160.
7109 035536 004737 007314      JSR PC,CKDATA      ;RCV SEC ADRS = 377
7110 035542 000377      377
7111 035544 000000      0
7112 035546 012701 000003      MOV #3,R1          ;RCV 3 MORE 377 CHARS
7113 035552 004737 007314      JSR PC,CKDATA
7114 035556 000377      377
7115 035560 000000      0
7116 035562 005301      DEC R1
7117 035564 001372      BNE 6$
7118 035566 004737 007314      JSR PC,CKDATA      ;RCV LAST 377 CHAR, CHK EBLK = 1
7119 035572 001377      1377
7120 035574 000000      0
7121 035576 004737 003276      JSR PC,MSTCLR      ;ISSUE MASTER CLEAR TO CLEAN UP
7122 035602      24$: JSR PC,MSTCLR
(3) 035602      ENDTST
(3) 035602 104401      L10105: TRAP C$E!ST
```

```
7123
7124
7125
7126
7127
7128          ;*****
7129          ;SBTTL TEST 32 - DDCMP MESSAGE TEST - CHAR MODE
7130          ;*
7131          ;* IN THIS TEST, THREE USYRT MESSAGES ARE SENT TO SIMULATE A DDCMP HEADER,
7132          ;* DDCMP DATA MESSAGE, AND THE START OF A NEW DDCMP HEADER.
7133          ;* FIRST, THE DATA IN PATTERN A IS TRANSMITTED AND RECEIVED
7134          ;* AND THEN CRC (CRC-16) IS SENT, FOLLOWED BY THE DATA IN PATTERN A
7135          ;* AGAIN AND THE CRC ON THAT DATA, AND FINALLY THE DATA IN 'MSG1' IS
7136          ;* SENT WITH ITS CORRESPONDING CRC.
7137          ;* PATTERN A = 125,252,000,377,001,002,004,010,020,040,100,200,376,
7138          ;* 375,373,367,357,337,277,177
7139          ;* MSG1 = SYNCH,SYNCH,SYNCH,SYNCH,000,125,252,377,000,SYNCH,SYNCH
7140          ;*****
7141 035604      BGNTST
(3) 035604
7142 035604 012737 036362 002346      MOV #24$,RETADR      ;SET TEST EXIT ADRS FOR ERRORS
7143          ;-----
7144          ; TRANSMIT AND RCV ENTIRE MSG
```

```
7145  
7146 035612 004737 005240      :-----  
7147 035616 000226      JSR    PC,INITRN      ;MST CLR, LOAD 2 SOM'S  
7148 035620 000011      SYNCH  
7149 035622 004737 010574      STRIP!DDCMP  
7150 035626 002557      JSR    PC,LDBYTS      ;LOAD 20 WORDS OF PAT A INTO TX SILO  
7151 035630 000024      PATA  
7152 035632 004737 010746      20.  
7153 035636 001000      JSR    PC,LODSIL      ;LOAD AN EOM INTO TX SILO  
7154 035640 000001      TXEOM  
7155 035642 004737 010574      1  
7156 035646 002557      JSR    PC,LDBYTS      ;LOAD 20 WORDS OF PAT A INTO TX SILO  
7157 035650 000024      PATA  
7158 035652 004737 010746      20.  
7159 035656 001000      JSR    PC,LODSIL      ;LOAD 1 EOM INTO TX SILO  
7160 035660 000001      TXEOM  
7161 035662 004737 010746      1  
7162 035666 000400      JSR    PC,LODSIL      ;LOAD 3 SOM'S INTO TX SILO  
7163 035670 000003      TXSOM  
7164 035672 004737 010516      3  
7165 035676 002670      JSR    PC,LODMSG      ;LOAD MSG1 INTO TX SILO  
7166 035700 000013      MSG1  
7167 035702 004737 004754      11.  
7168 035706 000300      JSR    PC,STPLU      ;CLOCK HDR MSG AND CRC CHARS  
7169 035710 012737 000013 002364      192.  
7170 035716 004737 003372      MOV    #13,REGNUM      ;SET REG. NO. = 13  
7171 035722 032737 000040 002350      JSR    PC,READLU      ;READ REG 13  
7172 035730 001010      BIT    #RTS,REDBYT      ;SEE IF RTS SET  
7173 035732 004737 004226      BNE    2$              ;BR IF RTS SET  
7174      JSR    PC,GETALL      ;GET REGS FOR PRINTOUT  
7175      ;REPORT RTS NOT SET  
7175 035736      ERRDF 60,EM60,ERR7  
7176 035736 104455      TRAP  C$ERDF  
7177 035740 000074      .WORD 60  
7178 035742 014702      .WORD EM60  
7179 035744 020244      .WORD ERR7  
7176 035746 000137 036362      JMP    24$              ;EXIT TEST  
7177 035752 004737 004754      2$: JSR    PC,STPLU      ;CLK DATA MSG AND FIRST CRC CHAR  
7178 035756 000250      168.  
7179 035760 012703 000040      MOV    #32,R3          ;SET COUNTER FOR CHECKING RTS  
7180 035764 004737 004754      4$: JSR    PC,STPLU      ;CLOCK LINE UNIT FOR 1 CYCLE  
7181 035770 000001      1  
7182 035772 004737 003372      JSR    PC,READLU      ;READ REG 13  
7183 035776 032737 000040 002350      BIT    #RTS,REDBYT      ;CHK FOR RTS SET  
7184      BNE    5$              ;BR IF RTS SET  
7185 036004 001007      JSR    PC,GETALL      ;GET REGS FOR PRINTOUT  
7186 036006 004737 004226      ;REPORT RTS NOT SET  
7187      ERRDF 60,EM60,ERR7  
7188 036012      TRAP  C$ERDF  
7189 036012 104455      .WORD 60  
7190 036014 000074      .WORD EM60  
7191 036016 014702      .WORD ERR7  
7192 036020 020244  
7189 036022 000557      BR    24$              ;DECR COUNTER  
7190 036024 005303      5$: DEC    R3  
7191 036026 001356      BNE    4$              ;BR IF NOT DONE YET  
7192
```

```

7193      : READ AND CHK HEADER AND CRC
7194
7195 036030 012701 002557      :
7196 036034 112137 036044      :
7197 036040 004737 007314      :
7198 036044 000000      :
7199 036046 000000      :
7200 036050 020127 002601      :
7201 036054 103767      :
7202 036056 004737 007314      :
7203 036062 100277      :
7204 036064 000000      :
7205 036066 004737 007314      :
7206 036072 100577      :
7207 036074 000000      :
7208 036076 004737 007314      :
7209 036102 000156      :
7210 036104 000000      :
7211 036106 004737 007314      :
7212 036112 000236      :
7213 036114 000000      :
7214
7215
7216
7217 036116 012701 002557      :
7218 036122 112137 036132      :
7219 036126 004737 007314      :
7220 036132 000000      :
7221 036134 000000      :
7222 036136 020127 002601      :
7223 036142 103767      :
7224 036144 004737 007314      :
7225 036150 100277      :
7226 036152 000000      :
7227 036154 004737 007314      :
7228 036160 100577      :
7229 036162 000000      :
7230 036164 004737 007314      :
7231 036170 000156      :
7232 036172 000000      :
7233 036174 004737 007314      :
7234 036200 000236      :
7235 036202 000000      :
7236
7237
7238
7239 036204 012737 000012 002364      :
7240 036212 112737 000200 002352      :
7241 036220 004737 003450      :
7242 036224 012737 000013 002364      :
7243 036232 004737 004754      :
7244 036236 000150      :
7245 036240 004737 003372      :
7246 036244 032737 000040 002350      :
7247 036252 001407      :
7248 036254 004737 004226      :
  
```

: READ AND CHK HEADER AND CRC

```

-----
7$:      MOV      #PATA,R1      :INIT PATTERN A POINTER
        MOVB     (R1)+,8$      :GET AN EXPECTED CHAR
        JSR      PC,CKDATA     :READ AND CHK A CHAR
8$:      .WORD    0
        0
        CMP      R1,#PATB-2    :SEE IF CHKING NEXT-TO-LAST CHAR YET
        BLO     7$            :BR IF NOT YET
        JSR      PC,CKDATA     :READ AND CHK CHAR, BCC=0
        CRCCHK.277
        0
        JSR      PC,CKDATA     :READ AND CHK LAST CHAR, BCC=1
        CRCCHK!RXBCC.177
        0
        JSR      PC,CKDATA     :READ AND CHK HI CRC BYTE
        156
        0
        JSR      PC,CKDATA     :READ AND CHK LO CRC BYTE
        236
        0
  
```

: READ AND CHK DATA MSG AND CRC

```

-----
9$:      MOV      #PATA,R1      :INIT PATTERN A POINTER
        MOVB     (R1)+,12$     :GET AN EXPECTED CHAR
        JSR      PC,CKDATA     :READ AND CHK A CHAR
12$:     .WORD    0
        0
        CMP      R1,#PATB-2    :SEE IF CHKING NEXT-TO-LAST CHAR YET
        BLO     9$            :BR IF NOT YET
        JSR      PC,CKDATA     :READ AND CHK CHAR, BCC=0
        CRCCHK!277
        0
        JSR      PC,CKDATA     :READ AND CHK LAST CHAR, BCC=1
        CRCCHK!RXBCC!177
        0
        JSR      PC,CKDATA     :READ AND CHK HI CRC BYTE
        156
        0
        JSR      PC,CKDATA     :READ AND CHK LO CRC BYTE
        236
        0
  
```

: CLOCK 3RD MESSAGE ('MSG1' DATA)

```

-----
        MOV      #12,REGNUM    :SET REG NO. = 12
        MOVB     #IC,WRIBYT    :SET IC TO CLEAR RECEIVER FOR NEW MSG
        JSR      PC,WRITLU
        MOV      #13,REGNUM    :RESTORE REG NO. TO 13
        JSR      PC,STPLU      :CLOCK THE REST OF MSG
        104
        JSR      PC,READLU     :READ REG 13
        BIT      #RTS,REDBYT   :SEE IF RTS IS CLEARED
        BEQ     14$           :BR IF RTS CLEARED
        JSR      PC,GETALL     :GET REGS FOR PRINTOUT
  
```

7249
7250 J36260
 (4) 036260 104455
 (5) 036262 000101
 (5) 036264 015000
 (5) 036266 020244
7251 036270 000434
7252
7253
7254
7255 036272 004737 007314
7256 036276 000000
7257 036300 000000
7258 036302 004737 007314
7259 036306 000125
7260 036310 000000
7261 036312 004737 007314
7262 036316 000252
7263 036320 000000
7264 036322 004737 007314
7265 036326 100377
7266 036330 000000
7267 036332 004737 007314
7268 036336 100400
7269 036340 000000
7270 036342 004737 007314
7271 036346 000160
7272 036350 000000
7273 036352 004737 007314
7274 036356 000034
7275 036360 000000
7276 036362 004737 003276
7277 036366
 (3) 036366
 (3) 036366 104401
7278
7279
7280
7281
7282

;REPORT RTS NOT CLEARED
ERRDF 65,EM65,ERR7

TRAP C\$ERDF
.WORD 65
.WORD EM65
.WORD ERR7

BR 24\$

;READ AND CHECK 3RD MESSAGE AND CRC

14\$: JSR PC,CKDATA ;READ AND CHECK 000 DATA CHAR
 000
 0
 JSR PC,CKDATA ;READ AND CHECK 125 DATA CHAR
 125
 0
 JSR PC,CKDATA ;READ AND CHECK 252 DATA CHAR
 252
 0
 JSR PC,CKDATA ;READ AND CHECK 377 DATA CHAR, AND BCC=0
 CRCCHK!377
 0
 JSR PC,CKDATA ;READ AND CHECK 000 DATA CHAR, AND BCC-1
 CRCCHK!RXBCC.000
 0
 JSR PC,CKDATA ;READ AND CHK HI CRC BYTE
 160
 0
 JSR PC,CKDATA ;READ AND CHK LO CRC BYTE
 034
 0
24\$: JSR PC,MSTCLR ;ISSUE MASTER CLEAR TO CLEAN UP
ENDTST

L10106: TRAP C\$ETST

.SBTTL HARDWARE PARAMETER CODING SECTION

////////////////////////////////////
:/ 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.
////////////////////////////////////

7284									
7285									
7286									
7287									
7288									
7289									
7290									
7291									
7292									
7293									
7294									
7295									
7296									
7297	036370		BGNHRD						
(3)	036370	000046						.WORD	L10107-L\$HARD/2
(3)	036372							L\$HARD::	
7298									
7299	036372		GPRMA	ADDRES,2,0,160000,177776,YES					
(4)	036372	001031						.WORD	T\$CODE
(4)	036374	036506						.WORD	ADDRES
(4)	036376	160000						.WORD	T\$LOLIM
(4)	036400	177776						.WORD	T\$HILIM
7300	036402		GPRMA	VECTOR,4,0,0,674,YES					
(4)	036402	002031						.WORD	T\$CODE
(4)	036404	036534						.WORD	VECTOR
(4)	036406	000000						.WORD	T\$LOLIM
(4)	036410	000674						.WORD	T\$HILIM
7301	036412		GPRMD	PRIPTY,6,0,7000,4,7,YES					
(4)	036412	003032						.WORD	T\$CODE
(4)	036414	036565						.WORD	PRIPTY
(4)	036416	007000						.WORD	7000
(4)	036420	000004						.WORD	T\$LOLIM
(4)	036422	000007						.WORD	T\$HILIM
7302	036424		GPRMD	SWPAC1,12,0,377,0,056,YES					
(4)	036424	005032						.WORD	T\$CODE
(4)	036426	036616						.WORD	SWPAC1
(4)	036430	000377						.WORD	377
(4)	036432	000000						.WORD	T\$LOLIM
(4)	036434	000056						.WORD	T\$HILIM
7303	036436		GPRMD	SWPAC2,14,0,377,0,377,YES					
(4)	036436	006032						.WORD	T\$CODE
(4)	036440	036657						.WORD	SWPAC2
(4)	036442	000377						.WORD	377
(4)	036444	000000						.WORD	T\$LOLIM
(4)	036446	000377						.WORD	T\$HILIM
7304	036450		GPRMD	SWPAC3,16,0,377,0,377,YES					
(4)	036450	007032						.WORD	T\$CODE
(4)	036452	036720						.WORD	SWPAC3
(4)	036454	000377						.WORD	377
(4)	036456	000000						.WORD	T\$LOLIM
(4)	036460	000377						.WORD	T\$HILIM
7305	036462		GPRMD	LOOPBK,20,0,7,0,4,YES					
(4)	036462	010032						.WORD	T\$CODE
(4)	036464	036761						.WORD	LOOPBK
(4)	036466	000007						.WORD	7
(4)	036470	000000						.WORD	T\$LOLIM

	037064	047075	047117	024505	
	037072	035040	000040		
7318	037076	046120	040505	042523	BAUDRT: .ASCII /PLEASE SELECT BAUD RATE; TYPE '0' FOR 2.4K; '1' FOR 4.8K; /<15><12>
	037104	051440	046105	041505	
	037112	020124	040502	042125	
	037120	051040	052101	035505	
	037126	052040	050131	020105	
	037134	030047	020047	047506	
	037142	020122	027062	045464	
	037150	020073	030447	020047	
	037156	047506	020122	027064	
	037164	045470	006473	012	
7319	037171	047	023462	043040	.ASCII /'2' FOR 9.6K; '3' FOR 19.2K; '4' FOR 56K; '5' FOR 250K; /<15><12>
	037176	051117	034440	033056	
	037204	035513	023440	023463	
	037212	043040	051117	030440	
	037220	027071	045462	020073	
	037226	032047	020047	047506	
	037234	020122	033065	035513	
	037242	023440	023465	043040	
	037250	051117	031040	030065	
	037256	035513	005015		
7320	037262	033047	020047	047506	.ASCIZ /'6' FOR 500K; OR '7' FOR 1 MEG BAUD : /
	037270	020122	030065	045460	
	037276	020073	051117	023440	
	037304	023467	043040	051117	
	037312	030440	046440	043505	
	037320	041040	052501	020104	
	037326	020072	000		
7321		037332			.EVEN
7322					
7323					
7324					
7325					
7326					
7327					
7328					

.SBTTL SOFTWARE PARAMETER CODING SECTION

:/ THE SOFTWARE 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.
:////

```
7330
7331
7332
7333
7334
7335
7336
7337
7338
7339
7340
7341
7342 037332          BGNSFT
      (3) 037332 000016
      (3) 037334
7343
7344 037334          GPRMI  ISMANI,0,1,YES
      (4) 037334 000130
      (4) 037336 037370
      (4) 037340 000001
7345 037342          GPRML  ISPRNT,2,1,YES
      (4) 037342 001130
      (4) 037344 037456
      (4) 037346 000001
7346 037350          GPRML  ISWPAK,4,1,YES
      (4) 037350 002130
      (4) 037352 037541
      (4) 037354 000001
7347 037356          GPRMD  TIMCNT,6,0,177777,0,177777,YES
      (4) 037356 003032
      (4) 037360 037606
      (4) 037362 177777
      (4) 037364 000000
      (4) 037366 177777
7348
7349 037370          ENDSFT
      (2)
      (3) 037370
7350
7351 037370 051511 046440 047101 ISMANI: .ASCIZ /IS MAN. INTERVEN. DESIRED TO MOUNT TEST CONNECTOR(S) /
      037376 020056 047111 042524
      037404 053122 047105 020056
      037412 042504 044523 042522
      037420 020104 047524 046440
      037426 052517 052116 052040
      037434 051505 020124 047503
      037442 047116 041505 047524
      037450 024122 024523 000040
7352 037456 044123 052517 042114 ISPRNT: .ASCIZ /SHOULD SWITCH PACK AND AX3-15 PRINTOUT BE ALLOWED /
      037464 051440 044527 041524
      037472 020110 040520 045503
      037500 040440 042116 040440
      037506 031530 030455 020065
      037514 051120 047111 047524
      037522 052125 041040 020105
      037530 046101 047514 042527
```

```

7353 037536 020104 000
      037541 123 047510 046125 !SWPAK: .ASCIZ /SHOULD SWITCH PACK TESTS BE ALLOWED /
      037546 020104 053523 052111
      037554 044103 050040 041501
      037562 020113 042524 052123
      037570 020123 042502 040440
      037576 046114 053517 042105
      037604 000040
7354 037606 051515 020107 044524 TIMCNT: .ASCIZ /MSG TIMER VALUE (0-177777), 0 = LONGEST TIME-OUT : /
      037614 042515 020122 040526
      037622 052514 020105 030050
      037630 030455 033467 033467
      037636 024467 020054 020060
      037644 020075 047514 043516
      037652 051505 020124 044524
      037660 042515 047455 052125
      037666 035040 000040
7355 .EVEN
7356
7357
7358
7359
7360
7361
7362
7363
7364
7365 037672
7366 037772 037772
7367 037772 000240
7368 037774 000240
7369 037776 000240
7370
7371
7372
7373
7374 040000 ENDMOD
7375
7376 040000 LASTAD
(2)
(4) 040000 000000 .EVEN
(4) 040002 000000 .WORD 0
(3) 040004 L$LAST:: .WORD 0
7377
7378 000001 .END

```

ABORT = 000004	2400#													
ADDRS = 036506	4420	4426	4441	4461	4479	4500	4518	4537	4556	7299	7310#			
ADR = 000020 G	2359#													
ANBITS = 002547	2801#													
APA = 000200	2578#													
ASBC0 = 000020	2535#													
ASBC1 = 000040	2534#													
ASBC2 = 000100	2533#													
ASSEMB = 000010	2211													
AXNUM = 002366	2721#	3142	3180	3194	3219	3221	3227*	3233*	3234	3236*	3238	3430	3438*	
	3481*	3645	3651*	3680*	3686*	3963	3965*	3973*	3984*	4111	4125*	4275*	4290	
	4293*	4306*	4310*	5204*	5232*	5235*	5247*	5617*	5908*	5954*	6030*	6083*	6428*	
AX0.15 = 002306	2648#	3226	4450	4469	4489	4508	4526	4546	4564					
AX0.16 = 002310	2649#	4450	4469	4489	4508	4526	4546	4564						
AX1 = 000001	2431#	2503#												
AX1.15 = 002312	2650#	4450	4469	4489	4508	4526	4546	4564						
AX1.16 = 002314	2651#	4450	4469	4489	4508	4526	4546	4564						
AX2 = 000002	2430#	2502#												
AX2.15 = 002316	2652#	4452	4471	4491	4510	4528	4548	4566						
AX2.16 = 002320	2653#	4452	4471	4491	4510	4528	4548	4566						
AX3.15 = 002322	2654#	4452	4471	4491	4510	4528	4548	4566						
AX3.16 = 002324	2655#	4452	4471	4491	4510	4528	4548	4566						
AX315U = 000372	2598#													
A11 = 025454	5340#	5370												
A12 = 033764	6479	6699#												
BADDAT = 002372	2723#	3817*	3818*	3839*	3840*	4429	4444	4483	4540	5152*	5161*	5269*	5547*	
	5571*	5595*	6194*	6234*	6493*	6503*	6514*	6522*						
BAUDRT = 037076	7306	7318#												
BCC = 000001	2479#	3843	3845	3851										
BCCCHK = 100000	2663#	3841	6066	6074	6119	6127	7020	7074						
BDRATE = 002464	2755#	4145	4201	4685*										
BIT0 = 000001 G	2359#	2381	2393	2402	2431	2443	2455	2467	2479	2491	2503	2515	2527	
	2539	2551	2560	2572	2585	2597	2608	2681	3221	3262	3392	3572	3613	
	3653	4655	6072	6125										
BIT00 = 000001 G	2359#													
BIT01 = 000002 G	2359#													
BIT02 = 000004 G	2359#													
BIT03 = 000010 G	2359#													
BIT04 = 000020 G	2359#													
BIT05 = 000040 G	2359#													
BIT06 = 000100 G	2359#													
BIT07 = 000200 G	2359#													
BIT08 = 000400 G	2359#													
BIT09 = 001000 G	2359#													
BIT1 = 000002 G	2359#	2380	2392	2401	2420	2430	2442	2454	2466	2478	2490	2502	2514	
	2526	2538	2550	2559	2571	2584	2596	2607	2682	3278	3556	3667		
BIT10 = 002000 G	2359#	2618	2630											
BIT11 = 004000 G	2359#	2617	2629											
BIT12 = 010000 G	2359#													
BIT13 = 020000 G	2359#													
BIT14 = 040000 G	2359#													
BIT15 = 100000 G	2359#	2661	2663	2664	2666	3360	3514							
BIT2 = 000004 G	2359#	2379	2391	2400	2419	2429	2441	2453	2465	2477	2489	2501	2513	
	2525	2537	2549	2558	2570	2583	2595	2606						
BIT3 = 000010 G	2359#	2378	2390	2399	2418	2428	2440	2452	2464	2476	2488	2500	2512	
	2524	2536	2548	2557	2569	2582	2594							

CZDMSC M8203 STATIC TESTS #2		MACY11 30A(1052)		13-MAR-80 12:35		PAGE 3-1		M 13		CROSS REFERENCE TABLE -- USER SYMBOLS		SEQ 0168	
CZDMSC.P11 13-MAR-80 12:31													
BIT4 = 000020 G	2359#	2377	2389	2417	2427	2439	2451	2463	2475	2487	2499	2511	2523
	2535	2547	2568	2581	2593								
BIT5 = 000040 G	2359#	2388	2409	2416	2426	2438	2450	2462	2474	2486	2498	2510	2522
	2534	2546	2567	2580	2592	2605							
BIT6 = 000100 G	2359#	2376	2387	2408	2415	2425	2437	2449	2461	2473	2485	2497	2509
	2521	2533	2545	2566	2579	2591	2604						
BIT7 = 000200 G	2359#	2375	2386	2398	2407	2414	2436	2448	2460	2472	2484	2496	2508
	2520	2532	2544	2556	2565	2578	2590	2603	3811	3812			
BIT8 = 000400 G	2359#	2620	2632										
BIT9 = 001000 G	2359#	2619	2631										
BOE = 000400 G	2359#												
BPOLL = 000100	2408#	6570											
BSEL1 = 002440	2744#	3006*	3008*	3009*	3023*	3024*	3029*	3121*	3123*	3225*	3364*	3365*	3367*
	3457*	3458*	3463*	3476*	3979*	3981*	4124*	4277*	4670*	4671*	5616*	6163*	6228*
	6239*	6241*	6291*	6342*	6498*	6532*	6533*	6534*	6583*	6643*	6735*	6789*	6843*
	6897*	6951*	7004*	7058*									
BSEL4 = 002442	2745#	3059	3080*										
CARR = 000001	2491#	6231	6691										
CHKABT = 012156	4290#												
CHPCHK = 100000	2661#	5474	5781	5816	5851								
CHPTYP = 002420	2734#	3361	3471*	3475*	3515	4623*	5237	5243					
CKDATA = 007314	3802#	4835	4842	4886	4891	4894	4927	4932	4935	4986	4989	4996	5010
	5017	5038	5041	5077	5080	5110	5113	5360	5363	5438	5441	5490	5493
	5668	5671	5674	5677	5680	5698	5701	5704	5707	5710	5729	5732	5735
	5738	5741	5792	5795	5798	5801	5804	5827	5830	5833	5836	5839	5862
	5865	5868	5871	5874	6065	6073	6118	6126	6177	6255	6305	6356	6749
	6803	6857	6911	6965	7019	7073	7109	7113	7118	7197	7202	7205	7208
	7211	7219	7224	7227	7230	7233	7255	7258	7261	7264	7267	7270	7273
CKLPBK = 011016	4110#	6154	6220	6282	6333	6480	6725	6779	6833	6887	6941	6995	7049
CRCCHK = 100000	2664#	3945	4060	5364	5442	5681	5711	5742	5805	5840	5875	6842	6896
	6950	7203	7206	7225	7228	7265	7268						
CRCTY0 = 000001	2585#												
CRCTY1 = 000002	2584#												
CRCTY2 = 000004	2583#												
CRC1 = 000100	2437#	4824	4872	4913	5194	5721	5848	5858	5914	6036	6054	6418	6730
	6784	6892	7000	7100									
CRC2 = 000200	2436#	4824	4872	4913	5194	5960	6089	6107	6418	6730	6784	7054	7100
CS = 000004	2489#	6685											
C\$AU = 000052	2211#	4794											
C\$AUTO = 000061	2211#	4741											
C\$BRK = 000022	2211#	4696											
C\$BSEG = 000004	2211#												
C\$BSUB = 000002	2211#	5311	5341	5395	5422	5528	5556	5580	5910	5956	6032	6085	6486
	6530	6549	6567	6577	6602	6620	6637	6656	6673				
C\$CEFG = 000045	2211#												
C\$CLCK = 000062	2211#												
C\$CLEA = 000012	2211#	4757											
C\$CLOS = 000035	2211#												
C\$CLP1 = 000006	2211#												
C\$CVEC = 000036	2211#												
C\$DCLN = 000044	2211#												
C\$DODU = 000051	2211#	4738											
C\$DRPT = 000024	2211#												
C\$DU = 000053	2211#	4776											
C\$EDIT = 000003	2211#	2251											
C\$ERDF = 000055	2211#	3268	3274	3284	3290	3398	3404	3562	3568	3578	3584	3619	3625

D 14

CZDMSC M8203 STATIC TESTS #2 MACY11 30A(1052) 13-MAR-80 12:35 PAGE 3-5
 CZDMSC.P11 13-MAR-80 12:31 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0172

FMT5	012413	4331#	4431	4446	4450	4465	4469	4485	4489	4504	4508	4522	4526	4542
		4546	4560	4564										
FMT6	012443	4332#	4433	4448	4452	4467	4471	4487	4491	4506	4510	4524	4528	4544
		4548	4562	4566										
FMT7	012476	4333#	4428	4463	4520	4539								
FMT8	012506	4334#	4443	4482	4502	4558								
FMT9	012542	4335#	4432	4447	4451	4466	4470	4486	4490	4505	4509	4523	4527	4543
		4547	4561	4565										
FRSPAS	002400	2726#	4653*	5533	5560	5584	5619							
FRSTIM	002376	2725#	4626	4633*										
FSAU =	000015	2211#	4793	4794										
FSAUTO=	000020	2211#	4729	4741										
F\$BGN =	000040	2211#	2217	4419	4425	4440	4459	4478	4498	4517	4535	4555	4583	4600
		4618	4729	4754	4771	4793	4820	4846	4865	4947	4973	5053	5067	5084
		5099	5117	5132	5166	5185	5281	5307	5311	5335	5341	5367	5372	5391
		5395	5418	5422	5445	5446	5465	5497	5524	5528	5552	5556	5576	5580
		5600	5601	5614	5629	5652	5746	5771	5879	5904	5910	5940	5946	5950
		5956	5986	5992	5996	5998	6026	6032	6079	6085	6132	6134	6150	6199
		6216	6261	6278	6311	6329	6362	6380	6398	6413	6446	6477	6486	6497
		6507	6525	6530	6542	6544	6549	6562	6567	6572	6577	6590	6597	6602
		6615	6620	6632	6637	6651	6656	6668	6673	6684	6690	6697	6701	6721
		6755	6775	6809	6829	6863	6883	6917	6937	6971	6991	7025	7045	7079
		7096	7122	7141	7277	7297	7342	7374						
F\$CLEA=	000007	2211#	4754	4757										
F\$DU =	000016	2211#	4771	4776										
F\$END =	000041	2211#	2217	4421	4434	4453	4472	4492	4511	4529	4549	4567	4585	4709
		4741	4757	4776	4794	4820	4846	4865	4947	4973	5053	5067	5084	5099
		5117	5132	5166	5185	5281	5307	5311	5335	5341	5367	5372	5391	5395
		5418	5422	5445	5446	5465	5497	5524	5528	5552	5556	5576	5580	5600
		5601	5614	5629	5652	5746	5771	5879	5904	5910	5940	5946	5950	5956
		5986	5992	5996	5998	6026	6032	6079	6085	6132	6134	6150	6199	6216
		6261	6278	6311	6329	6362	6380	6398	6413	6446	6477	6486	6497	6507
		6525	6530	6542	6544	6549	6562	6567	6572	6577	6590	6597	6602	6615
		6620	6632	6637	6651	6656	6668	6673	6684	6690	6697	6701	6721	6755
		6775	6809	6829	6863	6883	6917	6937	6971	6991	7025	7045	7079	7096
		7122	7141	7277	7308	7349	7374							
F\$HARD=	000004	2211#	7297	7308										
F\$HW =	000013	2211#	2296	2310										
F\$INIT=	000006	2211#	4618	4709										
F\$JMP =	000050	2211#												
F\$MOD =	000000	2211#	2217	7374										
F\$MSG =	000011	2211#	4419	4421	4425	4434	4440	4453	4459	4472	4478	4492	4498	4511
		4517	4529	4535	4549	4555	4567							
		2211#	4600	4604										
F\$PROT=	000021	2211#												
F\$PWR =	000017	2211#												
F\$RPT =	000012	2211#	4583	4585										
F\$SEG =	000003	2211#												
F\$SOFT=	000005	2211#	7342	7349										
F\$SRV =	000010	2211#												
F\$SUB =	000002	2211#	5311	5335	5341	5367	5395	5418	5422	5445	5528	5552	5556	5576
		5580	5600	5910	5950	5956	5996	6032	6079	6085	6132	6486	6525	6530
		6544	6549	6562	6567	6572	6577	6597	6602	6615	6620	6632	6637	6651
		6656	6668	6673	6697									
F\$SW =	000014	2211#	2324	2332										
F\$TEST=	000001	2211#	4820	4846	4865	4947	4973	5053	5067	5084	5099	5117	5132	5166
		5185	5281	5307	5372	5391	5446	5465	5497	5524	5601	5614	5629	5652

	5746	5771	5879	5904	5998	6026	6134	6150	6199	6216	6261	6278	6311
	6329	6362	6380	6398	6413	6446	6477	6701	6721	6755	6775	6809	6829
	6863	6883	6917	6937	6971	6991	7025	7045	7079	7096	7122	7141	7277
GETALL 004226	3218#	3266	3272	3282	3288	3396	3402	3560	3566	3576	3582	3617	3623
	3657	3663	3671	3677	3823	3828	3847	3853	3862	3868	3877	3883	3892
	3898	4245	4261	4297	4303	4903	4943	5022	5049	5162	5218	5257	5270
	5548	5572	5596	5937	5943	5983	5989	6172	6195	6235	6250	6300	6351
	6494	6504	6539	6558	6587	6593	6611	6628	6647	6664	6681	6687	6693
	6744	6798	6852	6906	6960	7013	7067	7173	7186	7248			
GETPRM 022242	4646	4658#	4665										
GETREG 003516	3094#	3224											
GOAH = 000010	2399#												
GOODAT 002370	2722#	3815*	3816*	3837*	3838*	4429	4444	4483	4540	5151*	5266*	5267*	5268*
	5546*	5570*	5594*	6193*	6233*	6492*	6502*	6513*	6521*				
GSCNTO= 000200	2211#												
GSDLM= 000372	2211#												
GSDISP= 000003	2211#												
GSEXCP= 000400	2211#												
GSHILI= 000002	2211#												
GSLOLI= 000001	2211#												
GSNO = 000000	2211#	4704											
GSOFFS= 000400	2211#	4704	7299	7300	7301	7302	7303	7304	7305	7306	7344	7345	7346
	7347												
GSOFSI= 000376	2211#	4704	7299	7300	7301	7302	7303	7304	7305	7306	7344	7345	7346
	7347												
GSPRMA= 000001	2211#	7299	7300										
GSPRMD= 000002	2211#	7301	7302	7303	7304	7305	7306	7347					
GSPRML= 000000	2211#	4704	7344	7345	7346								
GSRADA= 000140	2211#												
GSRADB= 000000	2211#												
GSRADD= 000040	2211#												
GSRADL= 000120	2211#	4704	7344	7345	7346								
GSRADO= 000020	2211#	7299	7300	7301	7302	7303	7304	7305	7306	7347			
GSXFER= 000004	2211#												
GYES = 000010	2211#	7299	7300	7301	7302	7303	7304	7305	7306	7344	7345	7346	7347
HDX = 000020	2417#	2487#	6190	6386	6422	6622	6626						
HELP = 000001	2197#	2243	2253	2276	2979								
HOE = 100000 G	2359#												
IACT = 000100	2473#	3615	3621										
IACTIV 006260	3605#	3732	3753	4992	5147	5328	5332	5411	5415	6186	6442		
IBE = 010000 G	2359#												
IC = 000200	2407#	2472#	4898	5017	5145	7240							
ICIR = 000010	2512#	3574	3580	6519	6521								
IDL = 000010	2582#												
IDLE = 000040	2438#	5071											
IDU = 000040 G	2359#												
IER = 020000 G	2359#												
IERR = 000002	2442#	3777	3783										
INITRN 005240	3428#	4822	4870	4911	4978	5002	5030	5069	5101	5134	5187	5313	5346
	5397	5424	5467	5776	5811	5846	5912	5958	6034	6087	6383	6674	7098
	7146												
INTFLG 002340	2709#												
INTGRL= 000010	2594#	2598	4115	4130	4147	4161	4212	6155	6222	6284	6335		
IRDY = 000020	2475#	3558	3564	3743	6168	6246	6296	6347	6740	6794	6848	6902	6956
	7009	7063											
ISIRDY 005774	3548#	3734	3747	3751	3755	5149	6395						

L\$NAME	002000	G	2251#		
L\$PRIO	002042	G	2251#		
L\$PROT	022040	G	2251	4600#	
L\$PRT	002112	G	2251#		
L\$REPP	002062	G	2251#		
L\$REV	002010	G	2251#		
L\$RPT	022036	G	4583#		
L\$SOFT	037334	G	2251	7342#	
L\$SPC	002056	G	2251#		
L\$SPCP	002020	G	2251#		
L\$SPTP	002024	G	2251#		
L\$STA	002030	G	2251#		
L\$SW	002256	G	2251	2324#	
L\$TEST	002114	G	2251#		
L\$TIML	002014	G	2251#		
L\$UNIT	002012	G	2251#	4660	
L10000	002254		2296	2310#	
L10001	002266		2324	2332#	
L10002	015404		4421#		
L10003	015712		4434#		
L10004	016374		4453#		
L10005	017052		4472#		
L10006	017564		4492#		
L10007	020242		4511#		
L10010	020674		4529#		
L10011	021402		4549#		
L10012	022034		4567#		
L10013	022036		4585#		
L10015	022540		4709#		
L10016	023076		4741#		
L10017	023100		4757#		
L10020	023130		4776#		
L10021	023164		4794#		
L10022	023314		4846#		
L10023	023670		4947#		
L10024	024226		5053#		
L10025	024310		5084#		
L10026	024400		5117#		
L10027	024574		5166#		
L10030	025344		5281#		
L10031	025604		5372#		
L10032	025446		5335#		
L10033	025566		5367#		
L10034	026010		5446#		
L10035	025706		5418#		
L10036	026006		5445#		
L10037	026144		5497#		
L10040	026616		5601#		
L10041	026330		5552#		
L10042	026462		5576#		
L10043	026614		5600#		
L10044	026732		5629#		
L10045	027270		5746#		
L10046	027700		5879#		
L10047	030352		5998#		
L10050	030122		5940	5946	5950#

RX5 = 000040	2450#	2522#													
RX6 = 000100	2449#	2521#													
RX7 = 000200	2448#	2520#													
R14NRW 002546	2798#														
SAVE4 002404	2728#	4628*	4631	4739											
SAVE6 002406	2729#	4629*	4632	4740											
SAVLEN 002424	2736#	3035*	3809	3978*	4625*	6048*	6101*								
SCRACH 002326	2704#														
SEC = 000020	2581#														
SECA = 000020	2439#	5315	5348	5399	5426	7100									
SEFR = 000040	2416#	6604													
SELSBY= 000002	2420#	6658													
SEL4 002442	2746#	4672*	4673*												
SEL6 002444	2747#	3007*	3122*	4674*	4675*										
SETUP 010324	3963#	5192	5658	5688	5719	6157	6223	6285	6336	6416	6728	6782	6836		
	6890	6944	6998	7052											
SFPTBL 002256 G	2324#														
SIGQ = 000100	2509#	6556													
SIGR = 000200	2508#	6609													
SOM = 000001	2402#														
STALL 004664	3321#	3366	3368	3459	3464	3477									
STARES 002402	2727#	4151	4156	4165	4648*	4654*									
STARST 022206	4636	4639	4647#												
STBY = 000002	2490#	6190	6662												
STEPLU= 000020	2377#	3365	3367	3458	3463	3476									
STEPMP= 000001	2381#	3008	3009												
STPCLK 003240	3005#	3057	3081												
STPERR 007212	3774#	5475	5480	5485	5787	5822	5857	6053	6106						
STPLU 004754	3356#	3524	3738	3781	3905	4831	4878	4883	4919	4924	5142	5210	5223		
	5227	5230	5241	5252	5325	5330	5409	5413	5473	5478	5483	5666	5696		
	5727	5785	5790	5820	5825	5855	5860	5924	5926	5929	5970	5972	5975		
	6050	6056	6060	6103	6109	6113	6391	6432	6437	6440	7107	7167	7177		
	7180	7243													
STR = 000040	2580#														
STRIP = 000010.	2440#	4872	4913	5469	5476	5481	5486	5660	5778	5788	5914	5960	6036		
	6054	6089	6107	6159	6225	6287	6338	6385	6784	6946	7000	7054	7148		
SUBRPC 002336	2708#	3255	3256	3258*	3259*	3296*	3385	3386	3388*	3389*	3410*	3431*	3432*		
	3484*	3504*	3505*	3531*	3549	3550	3552*	3553*	3590*	3606	3607	3609*	3610*		
	3631*	3646	3647	3649*	3650*	3685*	3724*	3725*	3761*	3804*	3805*	3915*	3985*		
	4113*	4114*	4278*	4291*	4292*	4460	4499	4536	4621*						
SVCGBL- 000000	2211#	2217	2225#	2251	2274	2296	2324	2967	2972	4419	4425	4440	4459		
	4478	4498	4517	4535	4555	4583	4600	4618	4729	4754	4771	4793	7297		
	7342	7376#													
SVCINS 000001	2211#	2222#	2251	2274	2296	2324	2967	2972	3268	3274	3284	3290	3398		
	3404	3562	3568	3578	3584	3619	3625	3659	3665	3673	3679	3825	3830		
	3849	3855	3864	3870	3879	3885	3894	3900	4153	4158	4167	4247	4263		
	4299	4305	4420	4421	4426	4427	4428	4429	4430	4431	4432	4433	4434		
	4441	4442	4443	4444	4445	4446	4447	4448	4449	4450	4451	4452	4453		
	4460	4461	4462	4463	4464	4465	4466	4467	4468	4469	4470	4471	4472		
	4479	4480	4481	4482	4483	4484	4485	4486	4487	4488	4489	4490	4491		
	4492	4499	4500	4501	4502	4503	4504	4505	4506	4507	4508	4509	4510		
	4511	4518	4519	4520	4521	4522	4523	4524	4525	4526	4527	4528	4529		
	4536	4537	4538	4539	4540	4541	4542	4543	4544	4545	4546	4547	4548		
	4549	4556	4557	4558	4559	4560	4561	4562	4563	4564	4565	4566	4567		
	4585	4635	4636	4638	4639	4641	4642	4644	4645	4662	4663	4691	4693		
	4695	4696	4700	4704	4709	4731	4738	4741	4757	4773	4775	4776	4794		

	4846	4905	4945	4947	5024	5051	5053	5084	5117	5164	5166	5220	5259
	5272	5281	5311	5335	5341	5367	5372	5395	5418	5422	5445	5446	5497
	5528	5539	5541	5550	5552	5556	5565	5574	5576	5580	5589	5598	5600
	5601	5625	5627	5629	5746	5879	5910	5939	5940	5945	5946	5950	5956
	5985	5986	5991	5992	5996	5998	6032	6079	6085	6132	6134	6174	6197
	6199	6237	6252	6261	6302	6311	6353	6362	6398	6446	6486	6496	6497
	6506	6507	6525	6530	6541	6542	6544	6549	6560	6562	6567	6572	6577
	6589	6590	6595	6597	6602	6613	6615	6620	6630	6632	6637	6649	6651
	6656	6666	6668	6673	6683	6684	6689	6690	6695	6697	6701	6746	6755
	6800	6809	6854	6863	6908	6917	6962	6971	7015	7025	7069	7079	7122
	7175	7188	7250	7277	7297	7299	7300	7301	7302	7303	7304	7305	7306
	7308	7342	7344	7345	7346	7347	7349	7376					
SVCSUB= 000001	2211#	2224#	5311	5341	5395	5422	5528	5556	5580	5910	5956	6032	6085
SVCTAG= 000001	6486	6530	6549	6567	6577	6602	6620	6637	6656	6673			
	2211#	2226#	2310	2332	4421	4434	4453	4472	4492	4511	4529	4549	4567
	4585	4704	4709	4741	4757	4776	4794	4846	4947	5053	5084	5117	5166
	5281	5335	5367	5372	5418	5445	5446	5497	5552	5576	5600	5601	5629
	5746	5879	5950	5996	5998	6079	6132	6134	6199	6261	6311	6362	6398
	6446	6525	6544	6562	6572	6597	6615	6632	6651	6668	6697	6701	6755
	6809	6863	6917	6971	7025	7079	7122	7277	7308	7349			
SVCTST= 000001	2211#	2223#	4820	4865	4973	5067	5099	5132	5185	5307	5391	5465	5524
	5614	5652	5771	5904	6026	6150	6216	6278	6329	6380	6413	6477	6721
	6775	6829	6883	6937	6991	7045	7096	7141					
SWIFLG 002262	2329#	5542	5566	5590									
SWPAC1 036616	7302	7313#											
SWPAC2 036657	7303	7314#											
SWPAC3 036720	7304	7315#											
SW0 = 000002	2466#												
SW1 = 000004	2465#												
SW2 = 000010	2464#												
SW3 = 000040	2462#												
SYNCH = 000226	2573#	4871	4912	5468	5659	5777	6158	6224	6286	6337	6384	6417	6783
	6945	7147											
SYN0 = 000001	2572#												
SYN1 = 000002	2571#												
SYN2 = 000004	2570#												
SYN3 = 000010	2569#												
SYN4 = 000020	2568#												
SYN5 = 000040	2567#												
SYN6 = 000100	2566#												
SYN7 = 000200	2565#												
SSLSYM= 010000	2211#	2310#	2332#	4421#	4434#	4453#	4472#	4492#	4511#	4529#	4549#	4567#	4585#
	4704#	4709#	4741#	4757#	4776#	4794#	4846#	4947#	5053#	5084#	5117#	5166#	5281#
	5335#	5367#	5372#	5418#	5445#	5446#	5497#	5552#	5576#	5600#	5601#	5629#	5746#
	5879#	5950#	5996#	5998#	6079#	6132#	6134#	6199#	6261#	6311#	6362#	6398#	6446#
	6525#	6544#	6562#	6572#	6597#	6615#	6632#	6651#	6668#	6697#	6701#	6755#	6809#
	6863#	6917#	6971#	7025#	7079#	7122#	7277#	7308#	7349#				
TCCHEK= 100000	2666#	4116	6481										
TCOUNT 002264	2330#	6166	6244	6294	6345	6738	6792	6846	6900	6954	7007	7061	
TEOM = 000002	2559#	5239	5245										
TERR = 000200	2556#												
TEST = 000001	2597#	4115	6156	6222	6284	6335							
TESTMD= 000004	2513#	4241	4257	6645									
TIMCNT 037606	7347	7354#											
TIMFLG 002344	2712#												
TMPO 002516	2778#	3220*	3223*	4443	4482	4502	4558						

T\$LOLI= 000000
T\$LSYM= 010000

7299#	7300#	7301#	7302#	7303#	7304#	7305#	7306#	7347#	4529	4549	4567	4585
2211#	2310	2332	4421	4434	4453	4472	4492	4511	5117	5166	5281	5335
4709	4741	4757	4776	4794	4846	4947	5053	5084	5601	5629	5746	5879
5367	5372	5418	5445	5446	5497	5552	5576	5600	6362	6398	6446	6525
5950	5996	5998	6079	6132	6134	6199	6261	6311	6701	6755	6809	6863
6544	6562	6572	6597	6615	6632	6651	6668	6697				
6917	6971	7025	7079	7122	7277	7308	7349					

T\$LTNO= 000040
T\$NEST= 177777

7376#	2217#	2296#	2310#	2324#	2332#	4419#	4421#	4425#	4434#	4440#	4453#	4459#
2211#	2217#	2296#	2310#	2324#	2332#	4419#	4421#	4425#	4434#	4440#	4453#	4459#
4472#	4478#	4492#	4498#	4511#	4517#	4529#	4535#	4549#	4555#	4567#	4583#	4585#
4600#	4604#	4618#	4709#	4729#	4741#	4754#	4757#	4771#	4776#	4793#	4794#	4820#
4846#	4865#	4947#	4973#	5053#	5067#	5084#	5099#	5117#	5132#	5166#	5185#	5281#
5307#	5311#	5335#	5341#	5367#	5372#	5391#	5395#	5418#	5422#	5445#	5446#	5465#
5497#	5524#	5528#	5552#	5556#	5576#	5580#	5600#	5601#	5614#	5629#	5652#	5746#
5771#	5879#	5904#	5910#	5950#	5956#	5996#	5998#	6026#	6032#	6079#	6085#	6132#
6134#	6150#	6199#	6216#	6261#	6278#	6311#	6329#	6362#	6380#	6398#	6413#	6446#
6477#	6486#	6525#	6530#	6544#	6549#	6562#	6567#	6572#	6577#	6597#	6602#	6615#
6620#	6632#	6637#	6651#	6656#	6668#	6673#	6697#	6701#	6721#	6755#	6775#	6809#
6829#	6863#	6883#	6917#	6937#	6971#	6991#	7025#	7045#	7079#	7096#	7122#	7141#
7277#	7297#	7308#	7342#	7349#	7374#							

T\$NSO = 000000
T\$NS1 = 000005

2217#	7374	2296#	2310	2324#	2332	4419#	4421	4425#	4434	4440#	4453	4459#	4472	4478#
4492	4498#	4511	4517#	4529	4535#	4549	4555#	4567	4583#	4585	4600#	4604		
4618#	4709	4729#	4741	4754#	4757	4771#	4776	4793#	4794	4820#	4846	4865#		
4947	4973#	5053	5067#	5084	5099#	5117	5132#	5166	5185#	5281	5307#	5372		
5391#	5446	5465#	5497	5524#	5601	5614#	5629	5652#	5746	5771#	5879	5904#		
5998	6026#	6134	6150#	6199	6216#	6261	6278#	6311	6329#	6362	6380#	6398		
6413#	6446	6477#	6701	6721#	6755	6775#	6809	6829#	6863	6883#	6917	6937#		
6971	6991#	7025	7045#	7079	7096#	7122	7141#	7277	7297#	7308	7342#	7349		
5311#	5335	5341#	5367	5395#	5418	5422#	5445	5528#	5552	5556#	5576	5580#		
5600	5910#	5950	5956#	5996	6032#	6079	6085#	6132	6486#	6525	6530#	6544		
6549#	6562	6567#	6572	6577#	6597	6602#	6615	6620#	6632	6637#	6651	6656#		
6668	6673#	6697												

T\$NS2 - 000002

2211#	4820#	4865#	4973#	5067#	5099#	5132#	5185#	5307#	5311#	5341#	5391#	5395#
5422#	5465#	5524#	5528#	5556#	5580#	5614#	5652#	5771#	5904#	5910#	5956#	6026#
6032#	6085#	6150#	6216#	6278#	6329#	6380#	6413#	6477#	6486#	6530#	6549#	6567#
6577#	6602#	6620#	6637#	6656#	6673#	6721#	6775#	6829#	6883#	6937#	6991#	7045#
7096#	7141#											

T\$TAGL= 177777
T\$TAGN= 010111

2211#	2296#	2324#	4419#	4425#	4440#	4459#	4478#	4498#	4517#	4535#	4555#	4583#
4600#	4618#	4729#	4754#	4771#	4793#	4820#	4865#	4973#	5067#	5099#	5132#	5185#
5307#	5311#	5341#	5391#	5395#	5422#	5465#	5524#	5528#	5556#	5580#	5614#	5652#
5771#	5904#	5910#	5956#	6026#	6032#	6085#	6150#	6216#	6278#	6329#	6380#	6413#
6477#	6486#	6530#	6549#	6567#	6577#	6602#	6620#	6637#	6656#	6673#	6721#	6775#

T\$TEMP= 000000

6829#	6883#	6937#	6991#	7045#	7096#	7141#	7297#	7342#	4511#	4529#	4549#	4567#	4585#
2274#	2310#	2332#	4421#	4434#	4453#	4472#	4492#	4511#	4529#	4549#	4567#	4585#	
4604#	4704#	4709#	4741#	4757#	4776#	4794#	4846#	4947#	5053#	5084#	5117#	5166#	
5281#	5335#	5367#	5372#	5418#	5445#	5446#	5497#	5552#	5576#	5600#	5601#	5629#	
5746#	5879#	5940#	5946#	5950#	5986#	5992#	5996#	5998#	6079#	6132#	6134#	6199#	
6261#	6311#	6362#	6398#	6446#	6497#	6507#	6525#	6542#	6544#	6562#	6572#	6590#	
6597#	6615#	6632#	6651#	6668#	6684#	6690#	6697#	6701#	6755#	6809#	6863#	6917#	
6971#	7025#	7079#	7122#	7277#	7299#	7300#	7301#	7302#	7303#	7304#	7305#	7306#	
7308#	7344#	7345#	7346#	7347#	7349#	7374#							

T11.1	026146		5528#																
T11.2	026332		5556#																
T11.3	026464		5580#																
T12	026620	G	2274	5614#															
T13	026734	G	2274	5652#															
T14	027272	G	2274	5771#															
T15	027702	G	2274	5904#															
T15.1	027716		5910#																
T15.2	030140		5956#																
T16	030354	G	2274	6026#															
T16.1	030370		6032#																
T16.2	030626		6085#																
T17	031056	G	2274	6150#															
T18	031340	G	2274	6216#															
T19	031612	G	2274	6278#															
T2	023316	G	2274	4865#															
T20	031774	G	2274	6329#															
T21	032156	G	2274	6380#															
T22	032254	G	2274	6413#															
T23	032464	G	2274	6477#															
T23.1	032506		6486#																
T23.10	033632		6673#																
T23.2	032762		6530#																
T23.3	033054		6549#																
T23.4	033142		6567#																
T23.5	033172		6577#																
T23.6	033310		6602#																
T23.7	033376		6620#																
T23.8	033456		6637#																
T23.9	033552		6656#																
T24	033772	G	2274	6721#															
T25	034162	G	2274	6775#															
T26	034352	G	2274	6829#															
T27	034542	G	2274	6883#															
T28	034732	G	2274	6937#															
T29	035122	G	2274	6991#															
T3	023672	G	2274	4973#															
T30	035306	G	2274	7045#															
T31	035472	G	2274	7096#															
T32	035604	G	2274	7141#															
T4	024230	G	2274	5067#															
T5	024312	G	2274	5099#															
T6	024402	G	2274	5132#															
T7	024576	G	2274	5185#															
T8	025346	G	2274	5307#															
T8.1	025346		5311#																
T8.2	025454		5341#																
T9	025606	G	2274	5391#															
T9.1	025606		5395#																
T9.2	025710		5422#																
UAM	= 000200	G	2359#																
UNIT	002432		2739#																
UNRR	= 000001		2467#	3821															
UPBITS	002536		2788#																
VECTOR	036534		7300	7311#															
V35	= 000020		2593#	2598	4115	4132	4176	4178	4180	4215	6156	6221	6284	6335					

CZDMSC M8203 STATIC TESTS #2		MACY11 30A(1052)		13-MAR-80 12:35		E 15		PAGE 3-19		SEQ 0186					
CZDMSC.P11 13-MAR-80 12:31		CROSS REFERENCE TABLE -- USER SYMBOLS													
WAIT50	004646	3307#	3450	3508	3740	4007	4033	4083	6182						
WAX	= 000010	2428#	2500#	3196											
WAX15	002360	2718#	3186*	3187	3439*	3441	3966*	3975*	5205*	5921*	5967*	6046*	6099*	6429*	
		6434*													
WAX16	002362	2719#	3190*	3191	3442*	3967*	3977*	3978	5206*	5225*	5233*	5236*	5239*	5245*	
		5922*	5968*	6047*	6048	6100*	6101	6430*	6435*						
WRDYTO=	000002	2682#	3178	3205											
WRIBYT	002352	2715#	3031*	3079*	3080	3142*	3143*	3144*	3145*	3180*	3181*	3182*	3187*	3191*	
		3194*	3195*	3196*	3197*	3338*	3341*	3446*	3776*	3777*	3783*	3971*	4175*	4207*	
		4232*	4236*	4252*	4267*	4898*	5017*	5145*	5198*	5202*	6386*	6422*	6425*	6508*	
		6551*	6570*	6580*	6604*	6622*	6639*	6658*	7240*						
WRITAX	004012	3176#	3443	3968	3980	5207	5222	5226	5229	5234	5240	5246	5923	5969	
		6049	6102	6431	6436	6439									
WRITLU	003450	3032	3073#	3146	3184	3188	3192	3198	3339	3342	3447	3778	3784	3972	
		4237	4253	4268	4899	5018	5146	5199	5203	6387	6423	6426	6509	6553	
		6571	6581	6606	6624	6641	6660	7241							
XYZ	= 000100	2591#	2598	4115	4137	4198	4204	74	6156	6222	6283	6335			
X\$ALWA=	000000	2211#													
X\$FALS=	000040	2211#													
X\$OFFS=	000400	2211#													
X\$TRUE=	000020	2211#													
\$LSTIN=	000001	2220#													
\$LSTTA=	000001	2221#													
.	- 040004	2203#	2699#	2899#	2944#	2972#	4715#	4779#	5940	5946	5986	5992	6497	6507	
		6542	6590	6684	6690	7321#	7366#								

ENDSFT	568#	2211#	7349												
ENDSRV	580#	2211#													
ENDSUB	596#	2211#	5335	5367	5418	5445	5552	5576	5600	5950	5996	6079	6132	6525	6544
	6562	6572	6597	6615	6632	6651	6668	6697							
ENDSW	614#	2211#	2332												
ENDTST	624#	2211#	4846	4947	5053	5084	5117	5166	5281	5372	5446	5497	5601	5629	5746
	5879	5998	6134	6199	6261	6311	6362	6398	6446	6701	6755	6809	6863	6917	6971
	7025	7079	7122	7277											
EQUALS	642#	2211#	2359												
ERRDF	714#	2211#	3268	3274	3284	3290	3398	3404	3562	3568	3578	3584	3619	3625	3659
	3665	3673	3679	3825	3830	3849	3855	3864	3870	3879	3885	3894	3900	4247	4263
	4299	4305	4905	4945	5024	5051	5164	5220	5259	5272	5550	5574	5598	5939	5945
	5985	5991	6174	6197	6237	6252	6302	6353	6496	6506	6541	6560	6589	6595	6613
	6630	6649	6666	6683	6689	6695	6746	6800	6854	6908	6962	7015	7069	7175	7188
	7250														
ERRHRD	718#	2211#													
ERROR	722#	2211#													
ERRSF	726#	2211#													
ERRSOF	730#	2211#													
ERRTBL	734#	2211#													
ESCAPE	744#	2211#	5940	5945	5986	5992	6497	6507	6542	6590	6684	6690			
EXIT	771#	2211#													
FEQUAL	810#	2211#													
GETBYT	824#	2211#													
GETPRI	834#	2211#													
GETWOR	829#	2211#													
GMANIA	839#	2211#													
GMANID	848#	2211#													
GMANIL	859#	2211#	4704												
GPHARD	868#	2211#	4662												
GPRMA	874#	2211#	7299	7300											
GPRMD	903#	2211#	7301	7302	7303	7304	7305	7306	7347						
GPRML	934#	2211#	4704#	7344	7345	7346									
HEADER	954#	2211#	2251												
INLOOP	962#	2211#													
IOSETU	966#	2211#													
IOSTAR	974#	2211#													
KT11	982#	2211#													
LASTAD	1147#	2211#	7376												
MANUAL	1162#	2211#	4691												
MEMORY	1166#	2211#													
MSBYTE	2000#	2211#	2251#												
MSCHEC	2118#	2211#													
MSCNTO	2182#	2211#	4704#	7299#	7300#	7301#	7302#	7303#	7304#	7305#	7306#	7344#	7345#	7346#	7347#
MSCOUN	2066#	2211#	4153#	4158#	4167#	4420#	4426#	4427#	4428#	4429#	4430#	4431#	4432#	4433#	4441#
	4442#	4443#	4444#	4445#	4446#	4447#	4448#	4449#	4450#	4451#	4452#	4460#	4461#	4462#	4463#
	4464#	4465#	4466#	4467#	4468#	4469#	4470#	4471#	4479#	4480#	4481#	4482#	4483#	4484#	4485#
	4486#	4487#	4488#	4489#	4490#	4491#	4499#	4500#	4501#	4502#	4503#	4504#	4505#	4506#	4507#
	4508#	4509#	4510#	4518#	4519#	4520#	4521#	4522#	4523#	4524#	4525#	4526#	4527#	4528#	4536#
	4537#	4538#	4539#	4540#	4541#	4542#	4543#	4544#	4545#	4546#	4547#	4548#	4556#	4557#	4558#
	4559#	4560#	4561#	4562#	4563#	4564#	4565#	4566#	4695#	4700#	4775#	5539#	5541#	5565#	5589#
	5625#	5627#													
MSDATA	1867#	2211#	2251#	2967#	2972#										
MSDECR	2029#	2211#	2310#	2332#	4421#	4434#	4453#	4472#	4492#	4511#	4529#	4549#	4567#	4585#	4604#
	4709#	4741#	4757#	4776#	4794#	4846#	4947#	5053#	5084#	5117#	5166#	5281#	5335#	5367#	5372#
	5418#	5445#	5446#	5497#	5552#	5576#	5600#	5601#	5629#	5746#	5879#	5950#	5996#	5998#	6079#

	6132#	6134#	6199#	6261#	6311#	6362#	6398#	6446#	6525#	6544#	6562#	6572#	6597#	6615#	6632#
	6651#	6668#	6697#	6701#	6755#	6809#	6863#	6917#	6971#	7025#	7079#	7122#	7277#	7308#	7349#
	7374#														
MSDEFA	2170#	2211#	4704#	7299#	7300#	7301#	7302#	7303#	7304#	7305#	7306#	7344#	7345#	7346#	7347#
MSSENDE	2074#	2211#	2310#	2332#	4421#	4434#	4453#	4472#	4492#	4511#	4529#	4549#	4567#	4585#	4709#
	4741#	4757#	4776#	4794#	4846#	4947#	5053#	5084#	5117#	5166#	5281#	5335#	5367#	5372#	5418#
	5445#	5446#	5497#	5552#	5576#	5600#	5601#	5629#	5746#	5879#	5950#	5996#	5998#	6079#	6132#
	6134#	6199#	6261#	6311#	6362#	6398#	6446#	6525#	6544#	6562#	6572#	6597#	6615#	6632#	6651#
	6668#	6697#	6701#	6755#	6809#	6863#	6917#	6971#	7025#	7079#	7122#	7277#	7308#	7349#	7374#
MSERRI	1649#	2211#	3268#	3274#	3284#	3290#	3398#	3404#	3562#	3568#	3578#	3584#	3619#	3625#	3659#
	3665#	3673#	3679#	3825#	3830#	3849#	3855#	3864#	3870#	3879#	3885#	3894#	3900#	4247#	4263#
	4299#	4305#	4905#	4945#	5024#	5051#	5164#	5220#	5259#	5272#	5550#	5574#	5598#	5939#	5945#
	5985#	5991#	6174#	6197#	6237#	6252#	6302#	6353#	6496#	6506#	6541#	6560#	6589#	6595#	6613#
	6630#	6649#	6666#	6683#	6689#	6695#	6746#	6800#	6854#	6908#	6962#	7015#	7069#	7175#	7188#
	7250#														
MSESCA	2006#	2211#	5940#	5946#	5986#	5992#	6497#	6507#	6542#	6590#	6684#	6690#			
MSDESCS	2010#	2211#	5940#	5946#	5986#	5992#	6497#	6507#	6542#	6590#	6684#	6690#			
MSEXCP	2101#	2211#	7299#	7300#	7301#	7302#	7303#	7304#	7305#	7306#	7347#				
MSEXIT	2014#	2211#													
MSSEXSE	2022#	2211#													
MSEXTJ	2018#	2211#													
MSGEN	2038#	2211#	2217#	2251#	2274#	2296#	2310#	2324#	2332#	2967#	2972#	4419#	4421#	4425#	4434#
	4440#	4453#	4459#	4472#	4478#	4492#	4498#	4511#	4517#	4529#	4535#	4549#	4555#	4567#	4583#
	4585#	4600#	4618#	4704#	4709#	4729#	4741#	4754#	4757#	4771#	4776#	4793#	4794#	4820#	4846#
	4865#	4947#	4973#	5053#	5067#	5084#	5099#	5117#	5132#	5166#	5185#	5281#	5307#	5311#	5335#
	5341#	5367#	5372#	5391#	5395#	5418#	5422#	5445#	5446#	5465#	5497#	5524#	5528#	5552#	5556#
	5576#	5580#	5600#	5601#	5614#	5629#	5652#	5746#	5771#	5879#	5904#	5910#	5950#	5956#	5996#
	5998#	6026#	6032#	6079#	6085#	6132#	6134#	6150#	6199#	6216#	6261#	6278#	6311#	6329#	6362#
	6380#	6398#	6413#	6446#	6477#	6486#	6525#	6530#	6544#	6549#	6562#	6567#	6572#	6577#	6597#
	6602#	6615#	6620#	6632#	6637#	6651#	6656#	6668#	6673#	6697#	6701#	6721#	6755#	6775#	6809#
	6829#	6863#	6883#	6917#	6937#	6971#	6991#	7025#	7045#	7079#	7096#	7122#	7141#	7277#	7297#
	7308#	7342#	7349#	7376#											
MSGENB	1938#	2211#	4704#												
MSGETS	2035#	2211#	2310#	2332#	4421#	4434#	4453#	4472#	4492#	4511#	4529#	4549#	4567#	4585#	4604#
	4709#	4741#	4757#	4776#	4794#	4846#	4947#	5053#	5084#	5117#	5166#	5281#	5335#	5367#	5372#
	5418#	5445#	5446#	5497#	5552#	5576#	5600#	5601#	5629#	5746#	5879#	5950#	5996#	5998#	6079#
	6132#	6134#	6199#	6261#	6311#	6362#	6398#	6446#	6525#	6544#	6562#	6572#	6597#	6615#	6632#
	6651#	6668#	6697#	6701#	6755#	6809#	6863#	6917#	6971#	7025#	7079#	7122#	7277#	7308#	7349#
	7374#														
MSGETT	1877#	2211#	5940#	5946#	5986#	5992#	6497#	6507#	6542#	6590#	6684#	6690#			
MSGNGB	1902#	2211#	2217#	2251#	2274#	2296#	2324#	2967#	2972#	4419#	4425#	4440#	4459#	4478#	4498#
	4517#	4535#	4555#	4583#	4600#	4618#	4729#	4754#	4771#	4793#	7297#	7342#	7376#		
MSGNIN	2049#	2211#	2251#	2274#	2296#	2324#	2967#	2972#	3268#	3274#	3284#	3290#	3398#	3404#	3562#
	3568#	3578#	3584#	3619#	3625#	3659#	3665#	3673#	3679#	3825#	3830#	3849#	3855#	3864#	3870#
	3879#	3885#	3894#	3900#	4153#	4158#	4167#	4247#	4263#	4299#	4305#	4420#	4421#	4426#	4427#
	4428#	4429#	4430#	4431#	4432#	4433#	4434#	4441#	4442#	4443#	4444#	4445#	4446#	4447#	4448#
	4449#	4450#	4451#	4452#	4453#	4460#	4461#	4462#	4463#	4464#	4465#	4466#	4467#	4468#	4469#
	4470#	4471#	4472#	4479#	4480#	4481#	4482#	4483#	4484#	4485#	4486#	4487#	4488#	4489#	4490#
	4491#	4492#	4499#	4500#	4501#	4502#	4503#	4504#	4505#	4506#	4507#	4508#	4509#	4510#	4511#
	4518#	4519#	4520#	4521#	4522#	4523#	4524#	4525#	4526#	4527#	4528#	4529#	4536#	4537#	4538#
	4539#	4540#	4541#	4542#	4543#	4544#	4545#	4546#	4547#	4548#	4549#	4556#	4557#	4558#	4559#
	4560#	4561#	4562#	4563#	4564#	4565#	4566#	4567#	4585#	4635#	4636#	4638#	4639#	4641#	4642#
	4644#	4645#	4662#	4663#	4691#	4693#	4695#	4696#	4700#	4704#	4709#	4731#	4738#	4741#	4757#
	4773#	4775#	4776#	4794#	4846#	4905#	4945#	4947#	5024#	5051#	5053#	5084#	5117#	5164#	5166#
	5220#	5259#	5272#	5281#	5311#	5335#	5341#	5367#	5372#	5395#	5418#	5422#	5445#	5446#	5497#
	5528#	5539#	5541#	5550#	5552#	5556#	5565#	5574#	5576#	5580#	5589#	5598#	5600#	5601#	5625#

MSPRIN	1636#	2211#	4153#	4158#	4167#	4420#	4426#	4427#	4428#	4429#	4430#	4431#	4432#	4433#	4441#
	4442#	4443#	4444#	4445#	4446#	4447#	4448#	4449#	4450#	4451#	4452#	4460#	4461#	4462#	4463#
	4464#	4465#	4466#	4467#	4468#	4469#	4470#	4471#	4479#	4480#	4481#	4482#	4483#	4484#	4485#
	4486#	4487#	4488#	4489#	4490#	4491#	4499#	4500#	4501#	4502#	4503#	4504#	4505#	4506#	4507#
	4508#	4509#	4510#	4518#	4519#	4520#	4521#	4522#	4523#	4524#	4525#	4526#	4527#	4528#	4536#
	4537#	4538#	4539#	4540#	4541#	4542#	4543#	4544#	4545#	4546#	4547#	4548#	4556#	4557#	4558#
	4559#	4560#	4561#	4562#	4563#	4564#	4565#	4566#	4695#	4700#	4775#	5539#	5541#	5565#	5589#
	5625#	5627#													
MSPUSH	1631#	2211#	2217#	2296#	2324#	4419#	4425#	4440#	4459#	4478#	4498#	4517#	4535#	4555#	4583#
	4600#	4618#	4729#	4754#	4771#	4793#	4820#	4865#	4973#	5067#	5099#	5132#	5185#	5307#	5311#
	5341#	5391#	5395#	5422#	5465#	5524#	5528#	5556#	5580#	5614#	5652#	5771#	5904#	5910#	5956#
	6026#	6032#	6085#	6150#	6216#	6278#	6329#	6380#	6413#	6477#	6486#	6530#	6549#	6567#	6577#
	6602#	6620#	6637#	6656#	6673#	6721#	6775#	6829#	6883#	6937#	6991#	7045#	7096#	7141#	7297#
	7342#														
MSPUT	1972#	2211#	4153#	4158#	4167#	4420#	4426#	4427#	4428#	4429#	4430#	4431#	4432#	4433#	4441#
	4442#	4443#	4444#	4445#	4446#	4447#	4448#	4449#	4450#	4451#	4452#	4460#	4461#	4462#	4463#
	4464#	4465#	4466#	4467#	4468#	4469#	4470#	4471#	4479#	4480#	4481#	4482#	4483#	4484#	4485#
	4486#	4487#	4488#	4489#	4490#	4491#	4499#	4500#	4501#	4502#	4503#	4504#	4505#	4506#	4507#
	4508#	4509#	4510#	4518#	4519#	4520#	4521#	4522#	4523#	4524#	4525#	4526#	4527#	4528#	4536#
	4537#	4538#	4539#	4540#	4541#	4542#	4543#	4544#	4545#	4546#	4547#	4548#	4556#	4557#	4558#
	4559#	4560#	4561#	4562#	4563#	4564#	4565#	4566#	4695#	4700#	4775#	5539#	5541#	5565#	5589#
	5625#	5627#													
MSPUT1	1981#	2211#	4153#	4158#	4167#	4420#	4426#	4427#	4428#	4429#	4430#	4431#	4432#	4433#	4441#
	4442#	4443#	4444#	4445#	4446#	4447#	4448#	4449#	4450#	4451#	4452#	4460#	4461#	4462#	4463#
	4464#	4465#	4466#	4467#	4468#	4469#	4470#	4471#	4479#	4480#	4481#	4482#	4483#	4484#	4485#
	4486#	4487#	4488#	4489#	4490#	4491#	4499#	4500#	4501#	4502#	4503#	4504#	4505#	4506#	4507#
	4508#	4509#	4510#	4518#	4519#	4520#	4521#	4522#	4523#	4524#	4525#	4526#	4527#	4528#	4536#
	4537#	4538#	4539#	4540#	4541#	4542#	4543#	4544#	4545#	4546#	4547#	4548#	4556#	4557#	4558#
	4559#	4560#	4561#	4562#	4563#	4564#	4565#	4566#	4695#	4700#	4775#	5539#	5541#	5565#	5589#
	5625#	5627#													
MSRADI	2077#	2211#	4704#	7299#	7300#	7301#	7302#	7303#	7304#	7305#	7306#	7344#	7345#	7346#	7347#
MSRBRO	1952#	2211#													
MSRNRO	1962#	2211#	4662#												
MSSETS	2032#	2211#	2217#	2296#	2324#	4419#	4425#	4440#	4459#	4478#	4498#	4517#	4535#	4555#	4583#
	4600#	4618#	4729#	4754#	4771#	4793#	4820#	4865#	4973#	5067#	5099#	5132#	5185#	5307#	5311#
	5341#	5391#	5395#	5422#	5465#	5524#	5528#	5556#	5580#	5614#	5652#	5771#	5904#	5910#	5956#
	6026#	6032#	6085#	6150#	6216#	6278#	6329#	6380#	6413#	6477#	6486#	6530#	6549#	6567#	6577#
	6602#	6620#	6637#	6656#	6673#	6721#	6775#	6829#	6883#	6937#	6991#	7045#	7096#	7141#	7297#
	7342#														
MSSTAR	1733#	2211#													
MS SVC	1933#	2211#	3268	3274	3284	3290	3398	3404	3562	3568	3578	3584	3619	3625	3659
	3665	3673	3679	3825	3830	3849	3855	3864	3870	3879	3885	3894	3900	4153#	4158#
	4167#	4247	4263	4299	4305	4420#	4421#	4426#	4427#	4428#	4429#	4430#	4431#	4432#	4433#
	4434#	4441#	4442#	4443#	4444#	4445#	4446#	4447#	4448#	4449#	4450#	4451#	4452#	4453#	4460#
	4461#	4462#	4463#	4464#	4465#	4466#	4467#	4468#	4469#	4470#	4471#	4472#	4479#	4480#	4481#
	4482#	4483#	4484#	4485#	4486#	4487#	4488#	4489#	4490#	4491#	4492#	4499#	4500#	4501#	4502#
	4503#	4504#	4505#	4506#	4507#	4508#	4509#	4510#	4511#	4518#	4519#	4520#	4521#	4522#	4523#
	4524#	4525#	4526#	4527#	4528#	4529#	4536#	4537#	4538#	4539#	4540#	4541#	4542#	4543#	4544#
	4545#	4546#	4547#	4548#	4549#	4556#	4557#	4558#	4559#	4560#	4561#	4562#	4563#	4564#	4565#
	4566#	4567#	4585#	4635#	4638#	4641#	4644#	4662#	4691#	4695#	4696#	4700#	4704#	4709#	4731#
	4738#	4741#	4757#	4773#	4775#	4776#	4794#	4846#	4905	4945	4947#	5024	5051	5053#	5084#
	5117#	5164	5166#	5220	5259	5272	5281#	5311#	5335#	5341#	5367#	5372#	5395#	5418#	5422#
	5445#	5446#	5497#	5528#	5539#	5541#	5550	5552#	5556#	5565#	5574	5576#	5580#	5589#	5598
	5600#	5601#	5625#	5627#	5629#	5746#	5879#	5910#	5939	5940#	5945	5946#	5950#	5956#	5985
	5986#	5991	5992#	5996#	5998#	6032#	6079#	6085#	6132#	6134#	6174	6197	6199#	6237	6252
	6261#	6302	6311#	6353	6362#	6398#	6446#	6486#	6496	6497#	6506	6507#	6525#	6530#	6541

	6542#	6544#	6549#	6560	6562#	6567#	6572#	6577#	6589	6590#	6595	6597#	6602#	6613	6615#
	6620#	6630	6632#	6637#	6649	6651#	6656#	6666	6668#	6673#	6683	6684#	6689	6690#	6695
	6697#	6701#	6746	6755#	6800	6809#	6854	6863#	6908	6917#	6962	6971#	7015	7025#	7069
	7079#	7122#	7175	7188	7250	7277#									
MSLAB	1929#	2211#	3268#	3274#	3284#	3290#	3398#	3404#	3562#	3568#	3578#	3584#	3619#	3625#	3659#
	3665#	3673#	3679#	3825#	3830#	3849#	3855#	3864#	3870#	3879#	3885#	3894#	3900#	4153#	4158#
	4167#	4247#	4263#	4299#	4305#	4420#	4421#	4426#	4427#	4428#	4429#	4430#	4431#	4432#	4433#
	4434#	4441#	4442#	4443#	4444#	4445#	4446#	4447#	4448#	4449#	4450#	4451#	4452#	4453#	4460#
	4461#	4462#	4463#	4464#	4465#	4466#	4467#	4468#	4469#	4470#	4471#	4472#	4479#	4480#	4481#
	4482#	4483#	4484#	4485#	4486#	4487#	4488#	4489#	4490#	4491#	4492#	4499#	4500#	4501#	4502#
	4503#	4504#	4505#	4506#	4507#	4508#	4509#	4510#	4511#	4518#	4519#	4520#	4521#	4522#	4523#
	4524#	4525#	4526#	4527#	4528#	4529#	4536#	4537#	4538#	4539#	4540#	4541#	4542#	4543#	4544#
	4545#	4546#	4547#	4548#	4549#	4556#	4557#	4558#	4559#	4560#	4561#	4562#	4563#	4564#	4565#
	4566#	4567#	4585#	4635#	4638#	4641#	4644#	4662#	4691#	4695#	4696#	4700#	4704#	4709#	4731#
	4738#	4741#	4757#	4773#	4775#	4776#	4794#	4846#	4905#	4945#	4947#	5024#	5051#	5053#	5084#
	5117#	5164#	5166#	5220#	5259#	5272#	5281#	5311#	5335#	5341#	5367#	5372#	5395#	5418#	5422#
	5445#	5446#	5497#	5528#	5539#	5541#	5550#	5552#	5556#	5565#	5574#	5576#	5580#	5589#	5598#
	5600#	5601#	5625#	5627#	5629#	5746#	5879#	5910#	5939#	5940#	5945#	5946#	5950#	5956#	5985#
	5986#	5991#	5992#	5996#	5998#	6032#	6079#	6085#	6132#	6134#	6174#	6197#	6199#	6237#	6252#
	6261#	6302#	6311#	6353#	6362#	6398#	6446#	6486#	6496#	6497#	6506#	6507#	6525#	6530#	6541#
	6542#	6544#	6549#	6560#	6562#	6567#	6572#	6577#	6589#	6590#	6595#	6597#	6602#	6613#	6615#
	6620#	6630#	6632#	6637#	6649#	6651#	6656#	6666#	6668#	6673#	6683#	6684#	6689#	6690#	6695#
	6697#	6701#	6746#	6755#	6800#	6809#	6854#	6863#	6908#	6917#	6962#	6971#	7015#	7025#	7069#
	7079#	7122#	7175#	7188#	7250#	7277#									
MSSTL	1921#	2211#	3268#	3274#	3284#	3290#	3398#	3404#	3562#	3568#	3578#	3584#	3619#	3625#	3659#
	3665#	3673#	3679#	3825#	3830#	3849#	3855#	3864#	3870#	3879#	3885#	3894#	3900#	4153#	4158#
	4167#	4247#	4263#	4299#	4305#	4420#	4421#	4426#	4427#	4428#	4429#	4430#	4431#	4432#	4433#
	4434#	4441#	4442#	4443#	4444#	4445#	4446#	4447#	4448#	4449#	4450#	4451#	4452#	4453#	4460#
	4461#	4462#	4463#	4464#	4465#	4466#	4467#	4468#	4469#	4470#	4471#	4472#	4479#	4480#	4481#
	4482#	4483#	4484#	4485#	4486#	4487#	4488#	4489#	4490#	4491#	4492#	4499#	4500#	4501#	4502#
	4503#	4504#	4505#	4506#	4507#	4508#	4509#	4510#	4511#	4518#	4519#	4520#	4521#	4522#	4523#
	4524#	4525#	4526#	4527#	4528#	4529#	4536#	4537#	4538#	4539#	4540#	4541#	4542#	4543#	4544#
	4545#	4546#	4547#	4548#	4549#	4556#	4557#	4558#	4559#	4560#	4561#	4562#	4563#	4564#	4565#
	4566#	4567#	4585#	4635#	4638#	4641#	4644#	4662#	4691#	4695#	4696#	4700#	4704#	4709#	4731#
	4738#	4741#	4757#	4773#	4775#	4776#	4794#	4846#	4905#	4945#	4947#	5024#	5051#	5053#	5084#
	5117#	5164#	5166#	5220#	5259#	5272#	5281#	5311#	5335#	5341#	5367#	5372#	5395#	5418#	5422#
	5445#	5446#	5497#	5528#	5539#	5541#	5550#	5552#	5556#	5565#	5574#	5576#	5580#	5589#	5598#
	5600#	5601#	5625#	5627#	5629#	5746#	5879#	5910#	5939#	5940#	5945#	5946#	5950#	5956#	5985#
	5986#	5991#	5992#	5996#	5998#	6032#	6079#	6085#	6132#	6134#	6174#	6197#	6199#	6237#	6252#
	6261#	6302#	6311#	6353#	6362#	6398#	6446#	6486#	6496#	6497#	6506#	6507#	6525#	6530#	6541#
	6542#	6544#	6549#	6560#	6562#	6567#	6572#	6577#	6589#	6590#	6595#	6597#	6602#	6613#	6615#
	6620#	6630#	6632#	6637#	6649#	6651#	6656#	6666#	6668#	6673#	6683#	6684#	6689#	6690#	6695#
	6697#	6701#	6746#	6755#	6800#	6809#	6854#	6863#	6908#	6917#	6962#	6971#	7015#	7025#	7069#
	7079#	7122#	7175#	7188#	7250#	7277#									
MSWORD	1994#	2211#	2251#	2274#	3268#	3274#	3284#	3290#	3398#	3404#	3562#	3568#	3578#	3584#	3619#
	3625#	3659#	3665#	3673#	3679#	3825#	3830#	3849#	3855#	3864#	3870#	3879#	3885#	3894#	3900#
	4247#	4263#	4299#	4305#	4704#	4905#	4945#	5024#	5051#	5164#	5220#	5259#	5272#	5550#	5574#
	5598#	5939#	5945#	5985#	5991#	6174#	6197#	6237#	6252#	6302#	6353#	6496#	6506#	6541#	6560#
	6589#	6595#	6613#	6630#	6649#	6666#	6683#	6689#	6695#	6746#	6800#	6854#	6908#	6962#	7015#
	7069#	7175#	7188#	7250#	7299#	7300#	7301#	7302#	7303#	7304#	7305#	7306#	7344#	7345#	7346#
	7347#	7376													
MSXFER	1682#	2211#													
OPEN	1171#	2211#													
POINTE	1176#	2211#	2241												
PRINTB	1239#	2211#	4420	4426	4427	4428	4429	4441	4442	4443	4444	4460	4461	4462	4463
	4479	4480	4481	4482	4483	4499	4500	4501	4502	4518	4519	4520	4536	4537	4538

	4539	4540	4556	4557	4558										
PRINTF	1279#	2211#	4153	4158	4167	4695	4700	4775	5539	5541	5565	5589	5625	5627	
PRINTS	1319#	2211#													
PRINTX	1359#	2211#	4430	4431	4432	4433	4445	4446	4447	4448	4449	4450	4451	4452	4464
	4465	4466	4467	4468	4469	4470	4471	4484	4485	4486	4487	4488	4489	4490	4491
	4503	4504	4505	4506	4507	4508	4509	4510	4521	4522	4523	4524	4525	4526	4527
	4528	4541	4542	4543	4544	4545	4546	4547	4548	4559	4560	4561	4562	4563	4564
	4565	4566													
READBU	1399#	2211#													
REDEF	1403#	2211#	4635	4638	4641	4644									
RFLAGS	1408#	2211#													
SETPRI	1413#	2211#	4731												
SETVEC	1418#	2211#													
SLASH	1424#	2211#													
STARS	1438#	2211#													
SVC	1452#	2210#	2211												
XFER	1612#	2211#													
XFERF	1616#	2211#													
XFERT	1620#	2211#													

. ABS. 040004 000

ERRORS DETECTED: 0

CZDMSC.BIN,CZDMSC.LST/CRF/NL:TOC=SVC34R.MLB,CZDMSC.P11
 RUN-TIME: 133 160 14 SECONDS
 RUN-TIME RATIO: 770/309=2.4
 CORE USED: 18K (35 PAGES)