IDENTIFICATION

PRODUCT CODE: MAINDEC-11-6D6LA-A-C
PRODUCT NAME: DL11-E.C/O OFF LINE TEST
DATE RELEASED: 21 DECEMBER 1975
MAINTAINER: DIAGNOSTIC GROUP

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.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A
LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE
TERMS OF SUCH A LICENSE.

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

COPYRIGHT (C) 1975 DIGITAL EQUIPMENT CORPORATION
ABSTRACT

Two separate diagnostic programs are provided for the DLII-E asynchronous modem interface, MAINEC-II-DOLLA-A (DLII-E off line tests) and MAINEC-II-DOLLA-A (DLII-E on line tests). The off line tests test all DLII-E logic. The off line tests do not require the use of a modem, however a special jumper connector H315 is required. The on line tests are essentially data reliability tests requiring the use of modems and a suitable terminal device.

The DLII-C and DLII-D can also be tested with this off line test. These are both tested in maintenance mode and only those tests marked C or D in the test number are executed. In order to test C and D versions it is necessary to modify the table at location 1300 according to the instructions contained there.

Tests which are not executed for DLII-C+D can be performed by using the select switch option (SS). TEST 56 is a data test which can be used for cable testing DLII-D's. WARNING—a failure in this test may occur due to a split baud rate of ACVTR/TVVTR.

This document describes the off line tests.

The available tests are:

PRG0 INPUT/OUTPUT LOGIC TESTS
PRG1 TRANSMITTER SCOPE LOOP
PRG2 RECEIVER SCOPE LOOP
PRG3 SINGLE CHARACTER MAINT. MODE DATA TEST
PRG4 SPECIAL BINARY COUNT MAINTENANCE MODE DATA TEST

2. REQUIREMENTS

2.1 EQUIPMENT

A. PDP 11 SYSTEM

B. DLII-E OR DLII-C OR DLII-D

C. SPECIAL JUMPER CONNECTOR H315 (SEE DLII MAINTENANCE MANUAL FOR DETAILED DESCRIPTION) IF DLII-E.

2.2 STORAGE

This program uses all of core (4K) except that area reserved for the bootstrap and absolute loaders.

3. LOADING PROCEDURE

The absolute loader is used to load the program.
USE PROCEDURE

THIS PROGRAM HAS BEEN MODIFIED TO RUN WITH OR WITHOUT A CONSOLE PROCESSOR, AND ALSO WITH OR WITHOUT A TTY IF A CONSOLE MACHINE IS USED; THEN THE PROGRAM LOOKS AT THE HARDWARE SWITCH REGISTER. IF A CONSOLE-LESS MACHINE IS USED THEN THE PROGRAM AUTOMATICALLY LOOKS AT THE CONTENTS OF LOCATION SOFTSR (176) AS A SWITCH REGISTER.

IT'S THE RESPONSIBILITY OF THE OPERATOR TO SET UP THIS LOCATION PRIOR TO STARTING THE PROGRAM.

BEFORE STARTING ANY OF THE SELECTABLE PROGRAMS MAKE SURE THAT THE TTY IS IN REMOTE MODE (IF THERE IS ONE); AND THAT THE PROGRAM SELECTED IS A LEGAL PROGRAM, IE: SR 0-2=0-4, OTHERWISE AN ERROR MESSAGE WILL OCCUR. (IGNORE THIS PARAGRAPH IF THERE IS NO TTY)

A MAP OF DEVICES PRESENT WILL BE TYPED AT RUN TIME. THIS MAP WILL NOT BE TYPED OUT AGAIN UNLESS THE PROGRAM IS RESTARTED AT LOCATION 200. A RESTART FROM THIS LOCATION WILL CAUSE THE MAP OF DEVICES TO BE TYPED OUT AGAIN AND THEN A NORMAL START WILL OCCUR.

PAGD: INPUT/OUTPUT LOGIC TESTS

A. LOAD ADDRESS = 000200 (RESTART LOAD ADDR. = 000204)
LOAD SR 0-2 = 0, AND PRESS START SWITCH.
THE DIAGNOSTIC WILL IDENTIFY THE PROGRAM YOU SELECTED.
IF THERE IS A TTY AND WILL HALT AT LOCATION 6444.
NOTE: IF THE CABLE IS LEFT CONNECTED TO THE MODERN THE FOLLOWING TESTS WILL FAIL:
AT22,AT23,AT25,AT30,AT32,AT56

B. THE PROGRAM WILL TYPE OUT INSTRUCTIONS TO SET IN THE DESIRED SR OPTIONS. IF TTY IS AVAILABLE AND WILL HALT AT LOCATION 4724.
PRESS CONTINUE WHEN THE OPTIONS ARE IN THE SR.
THE AVAILABLE OPTIONS ARE:
SR 0-9 ROUTINE TO BE RUN (IF ENABLED BY SR9)
SR6 HALT ON END OF PASS.
SR7 DISABLE STALL MODE
SR9 LOOP SELECTED ROUTINE
SR10 HALT AT END OF CURRENT TEST
SR11 INHIBIT ITERATION
SR12 SELECT LINE NUMBER AND LOCK ON IT
SR13 INHIBIT PRINTOUT
SR14 SCOPE
SR15 HALT ON ERROR.

C. THE PROGRAM WILL NOW REQUEST THE LINE #:IF SR12=1) YOU WISH TO "EE", AND WILL HALT AT LOCATION 3776.
LOAD THE LINE # AS REQUESTED AND PRESS CONTINUE.  
LINE NUMBER REFERS TO THE ADDRESSES TO WHICH THE DL1-1-E RESPONDS.

LINE 00 77561X 
LINE 10 77571X 
LINE 20 77581X 
LINE 30 77591X 
LINE 01 77562X 
LINE 11 77572X 
LINE 21 77582X 
LINE 31 77592X 
LINE 02 77563X 
LINE 12 77573X 
LINE 22 77583X 
LINE 32 77593X 
LINE 03 77564X 
LINE 13 77574X 
LINE 23 77584X 
LINE 33 77594X 
LINE 04 77565X 
LINE 14 77575X 
LINE 24 77585X 
LINE 34 77595X 
LINE 05 77566X 
LINE 15 77576X 
LINE 25 77586X 
LINE 35 77596X 
LINE 06 77567X 
LINE 16 77577X 
LINE 26 77587X 
LINE 36 77597X 
LINE 07 77569X 
LINE 17 77579X 
LINE 27 77589X

D. THE PROGRAM WILL NOW BEGIN TESTING THE DL1-1-E OR C/D YOU SELECTED. 
ALL DL1-1'S WILL BE TESTED AUTOMATICALLY AND SEQUENTIALLY 
UNLESS SR12 IS SELECTED.
NOTE: ALL LOGIC TESTS WILL NOT BE RUN AUTOMATICALLY.
THERE ARE TWO TESTS WHICH REQUIRE MANUAL INTERVENTION 
WHICH ARE USED TO TEST THE SPEED SELECTION SWITCHES.
THOSE ARE TESTS T34, T40. TO EXECUTE THESE TESTS USE SR9 AND 
SR 0-6 TO SELECT THEM.

E. REFER TO SECTION 5.1.2 FOR ERROR DESCRIPTION

F. AFTER ONE COMPLETE PASS THE BELL WILL RING 
FOLLOWED BY "END PASS = " WITH THE NUMBER OF 
PASSES COMPLETED SINCE PROGRAM LAST STARTED AND 
THE DEVICE ADDRESS UNDER TEST AND ITS TRAP VECTOR.
ALSO, THERE WILL BE A 5 ON THE DISPLAY LIGHTS FOR A 
FEW SECONDS JUST BEFORE THE TIME OF TYPING OUT.
PROGRAM WILL STORE AWAY IN CORE THE NUMBER OF PASSES 
COMPLETED, THE DEVICE ADDRESS UNDER TEST AND ITS 
TRAP VECTOR STARTING AT LOCATION 17420. 
IF SR9 WAS UP PROGRAM WILL HALT AT LOCATION 
252. PRESS CONTINUE FOR ANOTHER PASS.

4.2 PPG1 - TRANSMITTER SCOPE LOOP

A. LOAD ADDRESS = 000200 (RESTART = 000204) 
LOAD SR 0-2 = I, AND PRESS START SWITCH. 
THE DIAGNOSTIC WILL IDENTIFY THE PROGRAM YOU SELECTED, AND 
REQUEST THE LINE # YOU WISH TO TEST, IF TTY IS 
AVAILABLE AND WILL HALT AT LOCATION 3776. 
LOAD THE LINE # AS REQUESTED AND PRESS CONTINUE.

B. THE PROGRAM WILL REQUEST A CHARACTER CODE, AND A DELAY 
TIME, AND WILL HALT AT LOCATION 14370. 
The CHARACTER CODE IS THE DATA THE DL1-1-E WILL TRANSMIT 
AND THE DELAY IS THE TIME ELAPSED BETWEEN SUCCESSIVE TRANSMISSIONS OF ONE CHARACTER. LOAD CHARACTER CODE IN 
SR5 - SR8; SET DELAY TIME IN SR7 - SR6. 
PRESS CONTINUE WHEN THIS DONE.

C. THE PROGRAM WILL RUN WITHOUT ERROR OR END TYPECUTS.

4.3 PPG2 - RECEIVER SCOPE LOOP
A. LOAD ADDRESS = 000200 (RESTART = 000204)
   LOAD SR 0-2 = 3, AND PRESS START.
   THE DIAGNOSTIC WILL IDENTIFY THE PROGRAM YOU SELECTED, AND
   REQUEST THE LINE # YOU WISH TO TEST, IF TTY IS AVAILABLE
   AND WILL HALT AT LOCATION 3776
   LOAD THE LINE # AS REQUESTED AND PRESS CONTINUE.

B. THE PROGRAM WILL REQUEST A TEST CHARACTER CODE, AND A DELAY
   TIME AND WILL HALT AT LOCATION 14430.
   THE CHARACTER CODE IS THE DATA THAT THE DL11-E WILL
   BE TRANSMITTING AND THE DELAY IS THE ELAPSED TIME BETWEEN
   SUCCESSIVE CHARACTERS. LOAD CHARACTER CODE IN SR15-SRB.
   SET DELAY TIME IN SR7-SR9.
   PRESS CONTINUE WHEN THIS DONE.

C. THE PROGRAM WILL NOW RUN WITHOUT ERROR OR ENCODER:"S.

4.4 PRG3 - SINGLE CHARACTER MAINT MODE DATA TEST

A. LOAD ADDRESS = 000200 (RESTART = 000204)
   LOAD SR 0-2 = 3, AND PRESS START.
   THE DIAGNOSTIC WILL IDENTIFY THE PROGRAM YOU SELECTED, AND
   REQUEST THE LINE # YOU WISH TO TEST, IF TTY IS
   AVAILABLE AND WILL HALT AT LOCATION 3776.
   LOAD THE LINE # AS REQUESTED AND PRESS CONTINUE.

B. THE PROGRAM WILL REQUEST A TEST CHARACTER, AND WILL HALT
   AT LOCATION 14514.
   LOAD THE TEST CHARACTER AND PRESS CONTINUE.

C. THE PROGRAM WILL NOW RUN CONTINUOUSLY REPORTING ANY DATA FAIL-

4.5 PRG4 - SPECIAL BINARY COUNT MAINT. MODE DATA TEST

A. LOAD ADDRESS = 000200
   LOAD SR 0-2 = 4, AND PRESS START.
   THE DIAGNOSTIC WILL IDENTIFY THE PROGRAM YOU SELECTED, AND
   REQUEST THE LINE # YOU WISH TO TEST, IF TTY IS AVAILABLE
   AND WILL HALT AT LOCATION 3776.
   LOAD THE LINE # AS REQUESTED AND PRESS CONTINUE.

B. THE PROGRAM WILL BEGIN TESTING THE LINE YOU SELECTED.
   AND REPORT ANY DATA ERRORS.

5. PROGRAM DESCRIPTIONS

5.1 PRGO - INPUT/OUTPUT LOGIC TESTS

THE INPUT/OUTPUT LOGIC TESTS CONSIST OF 57(8) ROUTINES WHICH
MAY BE RUN IN SEQUENTIAL ORDER OR INDIVIDUALLY LOOPED (SEE
SECT 4.1, C FOR SWITCH SETTINGS). THE JUMPER CONNECTOR "LS"
5.1.1 ROUTINE DESCRIPTIONS

<table>
<thead>
<tr>
<th>ROUTINE</th>
<th>TESTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT0-AT3</td>
<td>ADDRESSABILITY OF CSRS &amp; DBRS</td>
</tr>
<tr>
<td>AT4-AT12</td>
<td>DIDDLES ALL Bits IN THE CSRS AND CHECKS THAT THEY CAN BE READ/WRITTEN PROPERLY.</td>
</tr>
<tr>
<td>AT31-AT32</td>
<td>PROPER OPERATION OF RESET INSTRUCTION</td>
</tr>
<tr>
<td>AT33</td>
<td>PROPER OPERATION OF READY BIT</td>
</tr>
<tr>
<td>AT34</td>
<td>PROPER OPERATION OF TRANSMIT SPEED SELECTION</td>
</tr>
<tr>
<td>AT35-AT37</td>
<td>PROPER OPERATION OF DONE BIT</td>
</tr>
<tr>
<td>AT4C</td>
<td>PROPER OPERATION RECEIVER SPEED SELECT</td>
</tr>
<tr>
<td>AT41</td>
<td>PROPER OPERATION OF DATA OVERRUN</td>
</tr>
<tr>
<td>AT42-AT52</td>
<td>PROPER OPERATION OF INTERRUPTS</td>
</tr>
<tr>
<td>AT53</td>
<td>READING RXCSR DOES NOT CLEAR DONE</td>
</tr>
<tr>
<td>AT54</td>
<td>ERROR CAUSES INTERRUPT</td>
</tr>
<tr>
<td>AT55</td>
<td>DATA TEST MAINTENANCE MODE</td>
</tr>
<tr>
<td>AT56</td>
<td>DATA TEST WITH JUMPER</td>
</tr>
<tr>
<td>AT57</td>
<td>PROPER OPERATION OF BREAK BIT</td>
</tr>
</tbody>
</table>

5.1.2 ERROR DESCRIPTION

IF SR15 IS UP, PROGRAM WILL HALT AT LOCATION 5310 ON ANY ERROR.

IF A ROUTINE FAILS AND THE INHIBIT PRINTOUT SWITCH IS NOT ENABLED (SR13) A PRINTOUT RESULTS. THE PRINTOUT FORMAT IS:

T ROUTINE (PC OF ERROR CALL) RXCSR (ADDRESS OF DEVICE UNDER TEST) AND AN ADDITIONAL/MESSAGE (IF APPLICABLE)

TO05 PC=XXXX RXCSR=XXXX
T5E PC=XXXX RXCSR=XXXX DATA S/B:---WAS:--- INDICATING A DATA ERROR

THE ABOVE INFORMATION IS STORED IN CORE STARTING AT LOCATION 17400.

FOR EXAMPLE:
17400 WILL CONTAIN ROUTINE # THAT FAILED
17402 WILL CONTAIN ERROR PC
17404 WILL CONTAIN ADDRESS OF DEVICE UNDER TEST
17406 WILL CONTAIN DATA SHOULD BE (IN CASE OF DATA ERROR)
17410 WILL CONTAIN DATA WAS (IN CASE OF DATA ERROR)

TO RESUME TESTING PRESS CONTINUE.
IF THE VECTOR PROVIDED BY THE INTERRUPTING DLI-1-E IS INCORRECT
A TRAP TO THE WRONG LOCATION WILL OCCUR AND AN ERROR MESSAGE
WILL OCCUR.

5.3 JUMPER CONNECTOR

THE JUMPER CONNECTOR TESTS THOSE F/F'S, GATES (RING INDICATOR,
CARRIER TRANSITION, CLEAR TO SEND, AND SUPERVISORY RECEIVE
DATA) WHICH CANNOT BE TESTED UNLESS A DATA SET IS ACTUALLY
CONNECTED TO THE DLI-1-E. IN ADDITION TO TESTING DLI-1-E LOGIC
THE JUMPER ALSO TESTS CABLE WIRING TO/FROM THE DLI-1-E/DATA
SET. THE FOLLOWING TESTS WILL FAIL IF THE CABLE IS NOT
INARRNLED IN THE DLI-1-E:

AT22, AT23, AT25, AT30, AT32, AT56

5.2 PRG1-TRANSMITTER SCOPE LOOP

THE PURPOSE OF PRG1 IS TO ALLOW SCOPING OF TRANSMITTER
FUNCTIONS IN A RUN CONDITION USING USER SPECIFIED DLI-1-E
PARAMETERS AND DATA. NO ERROR PRINTOUTS ARE PROVIDED.

5.3 PRG2-RECEIVER SCOPE LOOP

THE PURPOSE OF PRG2 IS TO ALLOW SCOPING OF RECEIVER FUNCTIONS
IN A RUN CONDITION USING USER SPECIFIED DLI-1-E PARAMETERS
AND DATA. NO ERROR PRINTOUTS ARE PROVIDED.

5.4 PRG3-SINGLE CHARACTER MAINT MODE DATA TEST

PRG3 TRANSMITS, RECEIVES AND CHECKS RECEIVED DATA USING USER
SPECIFIED DLI-1-E PARAMETERS, AND DATA.

5.4.1 ERROR PRINTOUTS

SELF EXPLANATORY ERROR PRINTOUTS ARE PROVIDED.

5.5 PRG4-SPECIAL BINARY COUNT MAINT MODE DATA TEST

PRG4 IS THE SAME AS PRG ROUTINE 54 EXCEPT THAT
THE USER SPECIFIES DLI-1-E RUNNING PARAMETERS.

5.5.1 ERROR PRINTOUTS

SELF-EXPLANATORY PRINTOUTS ARE PROVIDED.

6.0 POWER FAIL

A POWER FAIL ROUTINE IS INCLUDED IN THE PROGRAM. WHEN THE POWER FAILS
THE PROGRAM WILL AUTOMATICALLY RESTART USING THE PRESENT SR OPTIONS AND THE LINE PREVIOUSLY SELECTED. NOTE: THE POWER MAY FAIL WHEN THE PROGRAM IS EXECUTING A 'RESET' INSTRUCTION, IN THIS CASE OPERATOR INTERVENTION IS NEEDED TO PRESS CONTINUE. AN ERROR TYPEOUT RESULTS AND WILL TYPE THE PROGRAM #, THE ROUTINE THAT WAS RUNNING AT THE TIME THE POWER FAILED (PROGRAM O ONLY), AND THE PC OF THE POWER FAIL ERROR CALL.

RECOVERED FROM POWER FAILURE:
P:PRG#: T:ROUTINE #: PC = (ADDRESS OF ERROR CALL)

1
.ENABLE ABS

--DIAGNOSTIC PROGRAM (OFF LINE TESTS)

PRG0- INPUT-OUTPUT LOGIC TESTS
PRG1- TRANSMITTER SCOPE LOOP
PRG2- RECEIVER SCOPE LOOP
PRG3- SINGLE CHARACTER MAINTENANCE MODE DATA TEST
PRG4- SPECIAL BINARY COUNT MAINTENANCE MODE DATA TEST

--SWITCH OPTIONS (SWITCH SET TO A)

SR15- HALT ON ERROR
SR14- SCOPE
SR13- INHIBIT PRINTOUT
SR12- SELECT LINE NUMBER AND LOCK ON IT
SR11- INHIBIT ITERATION
SR10- HALT AT END CURRENT TEST, TEST NO. IN DATA LIGHTS
SR9- SELECT ROUTINE
SR7- DISABLE STALL MODE AND RUN FULL SPEED
SR5- NUMBER OF ROUTINE TO BE SELECTED
SR6- HALT ON END OF PASS

--CHARACTER LENGTH 8

STOP CODE 2

ENDP

MACHINE:

0

;UNASSIGNED TRAP

40

;SP OVERFLOW, BUS ERROR TRAP

100

;reserved instruction trap

140

;TRACE TRAP

MAPVEC

;TRAP TO MAP VECTOR

PFAL

;POWER FAIL TRAP

EMTINT

;EMT TRAP

340

;TRAP TO TRAP REPORTER

HALT

;TRAP TO TRAP REPORTER

LOGIC

;TRAP TO TRAP REPORTER

4

;TRAP TO TRAP REPORTER

4

;TRAP TO TRAP REPORTER

4

;TRAP TO TRAP REPORTER

4

;TRAP TO TRAP REPORTER

4

;TRAP TO TRAP REPORTER

4

;TRAP TO TRAP REPORTER

4

;TRAP TO TRAP REPORTER

4
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
; TRAP TO TRAP REPORTER
EQUATE STATEMENTS
PSW=177776
SPBOT=1176
POP=240
OPEN=0
MANUAL=BIT15
BIT15=100000
BIT14=40000
BIT13=20000
BIT12=10000
BIT11=4000
BIT10=2000
BIT9=1000
BIT8=400
BIT7=200
BIT6=100
BIT5=40
BIT4=20
BIT3=10
BIT2=4
BIT1=2
BIT0=1
PPOPSP=5726
PPOPSP2=02626
PRTY7=300
PRTY6=300
PRTY5=240
PRTY4=240
PRTY3=140
PRTY2=100
PRTY1=140
PRTY0=0
TYPE-ENT+0
TYPES-ENT+1
STALL-ENT+2
ERROR-ENT+3
CATCH-ENT+4
HALT-ENT+5
STRXY-ENT+6
STRXX-ENT+7
EXHALT-ENT+10
SRESET-ENT+11
SCOPE-ENT+12
SREG-ENT+13
ASTREG-ENT+14
ERROR1-ENT+15
DELAY-ENT+16
TIMEX-ENT+17
TIME-ENT+20
AT-AST=-1
SG=100000
;FLAG FOR C/D TESTS
;...
HIGH BYTE OF SWITCH REGISTER

:GO TO START OF PROGRAM.

:THIS IS AN END OF PASS HALT; NOT AN ERROR HALT.
:THIS HAPPENS ONLY IF SW6 IS UP. PRESS CONTINUE
:TO GET ANOTHER PASS.

DEVICE ADDRESS LIST
:SB BITO IS SET TO A 1 BY MAPPER IF DEVICE NOT FOUND
:TO TEST THAT LINE NOT FOUND CLEAR BITO IN THAT DEVICE ADDRESS
:IN THIS TABLE AFTER MAPPING DONE

***************
RXRD: 175610 :LINE 0 DEVICE ADDRESS (RXCSR)
RXRD: 175620 :LINE 1 DEVICE ADDRESS (RXCSR)
RXRD: 175630 :LINE 2 DEVICE ADDRESS (RXCSR)
RXRD: 175640 :LINE 3 DEVICE ADDRESS (RXCSR)
RXRD: 175650 :LINE 4 DEVICE ADDRESS (RXCSR)
RXRD: 175660 :LINE 5 DEVICE ADDRESS (RXCSR)
RXRD: 175670 :LINE 6 DEVICE ADDRESS (RXCSR)
RXRD: 175680 :LINE 7 DEVICE ADDRESS (RXCSR)
RXRD: 175690 :LINE 8 DEVICE ADDRESS (RXCSR)
RXRD: 1756A0 :LINE 9 DEVICE ADDRESS (RXCSR)
RXRD: 1756B0 :LINE 10 DEVICE ADDRESS (RXCSR)
RXRD: 1756C0 :LINE 11 DEVICE ADDRESS (RXCSR)
RXRD: 1756D0 :LINE 12 DEVICE ADDRESS (RXCSR)
RXRD: 1756E0 :LINE 13 DEVICE ADDRESS (RXCSR)
RXRD: 1756F0 :LINE 14 DEVICE ADDRESS (RXCSR)
RXRD: 176000 :LINE 15 DEVICE ADDRESS (RXCSR)
RXRD: 176010 :LINE 16 DEVICE ADDRESS (RXCSR)
RXRD: 176020 :LINE 17 DEVICE ADDRESS (RXCSR)
RXRD: 176030 :LINE 18 DEVICE ADDRESS (RXCSR)
RXRD: 176040 :LINE 19 DEVICE ADDRESS (RXCSR)
RXRD: 176050 :LINE 20 DEVICE ADDRESS (RXCSR)
RXRD: 176060 :LINE 21 DEVICE ADDRESS (RXCSR)
RXRD: 176070 :LINE 22 DEVICE ADDRESS (RXCSR)
RXRD: 176080 :LINE 23 DEVICE ADDRESS (RXCSR)
RXRD: 176090 :LINE 24 DEVICE ADDRESS (RXCSR)
RXRD: 1760A0 :LINE 25 DEVICE ADDRESS (RXCSR)
RXRD: 1760B0 :LINE 26 DEVICE ADDRESS (RXCSR)
RXRD: 1760C0 :LINE 27 DEVICE ADDRESS (RXCSR)
RXRD: 1760D0 :LINE 28 DEVICE ADDRESS (RXCSR)
RXRD: 1760E0 :LINE 29 DEVICE ADDRESS (RXCSR)
RXRD: 1760F0 :LINE 30 DEVICE ADDRESS (RXCSR)
RXRD: 176100 :LINE 31 DEVICE ADDRESS (RXCSR)
RXRD: 176110 :LINE 32 DEVICE ADDRESS (RXCSR)
RXRD: 176120 :LINE 33 DEVICE ADDRESS (RXCSR)
RXRD: 176130 :LINE 34 DEVICE ADDRESS (RXCSR)
RXRD: 176140 :LINE 35 DEVICE ADDRESS (RXCSR)
RXRD: 176150 :LINE 36 DEVICE ADDRESS (RXCSR)
RXRD: 176160 :LINE 37 DEVICE ADDRESS (RXCSR)
RXRD: 176170 :LINE 38 DEVICE ADDRESS (RXCSR)
RXRD: 176177 :LINE 39 SPECIAL ADDRESS FOR XCR
RXRD: 177777 :LINE 40 SPECIAL ADDRESS FOR XCR

CHARACTER LENGTH, PRIORITY, CYD MASK
INITIALLY SET FOR DL11-E, PRIORITY=4, CHARACTER LENGTH=8
BIT IS SET TO A 1 THAT LINE HAS DL11-C OR DL11-D
EX: 140377 = DL11-C OR DL11-D, PRIORITY = 4, CHARACTER LENGTH = 8
BITS 12-14 = PRIORITY LEVEL THAT LINE
BITS 0-7 = CHARACTER MASK EX. 377=8, 177=7, 77=6, 37=5

***************

únasm: 0
ímask: 0
ślmask: 177740

rxcsr: 0
rxbuf: 0
txcsr: 0
txbuf: 0
rxtvr: 0
rxlvs: 0

ys: 17756c
ys: 177554
ys: 177554

; MASK FOR DEVICE UT
; MASK FOR CHARE LENGTH FOR DEVICE UT
; MASK FOR MAX RANDOM STALL

; RECEIVER UNDER TEST
; RECEIVER BUFFER UNDER TEST
; TRANSMITTER CSR UNDER TEST
; TRANSMITTER BUFFER UNDER TEST
; RECEIVER VECTOR UNDER TEST
; RECEIVER PRIORITY LEVEL UT
; TRANSMITTER VECTOR UNDER TEST
; TRANSMITTER PRIORITY LEVEL UT

; LSP CSR
; LSP BUFFER
; LSP CSR
MAIN: MAC11 27.732  10-SEP-76 09:54 PAGE 18

COOLAB.FIL

LINE: 0
TEMP: OPEN
TIEL: CO
FONE: OD
TOPC: DO
FROMP: OD
PASCNT: 0
START: MOV #SPBOT,46 :SAVE CURRENT VECTOR
            MOV 6,SP
            MOV -1,(SP)
            MOV $1,4
            MOV $5,SRPTR
            BR 25
            MOV #SOFTSR,SRPTR
            BRN #SRPTR
            INC SRPTR
            CLR #TITLE
            MOV @#177061
            TST @#177061
            MOV @#177061
            XOR @#177061
            MOV @#177061
            TYPE MESSI
            JMP @START
MESSI: .ASCII (15),12,12,'YOU ARE ON AN XOR TESTER'

XORFLG: .WORD 0
XORA: CMP (%6)+,%6+
       MOV (%6)+,%6+
       MOV @1,%XORADD
       CLR #XORFLG
       JMP @START

START: MOV #SPBOT,46 :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
            MOV #MAPT,24
            CLR FMAP
            STEP %Z,CLPCC
            JUMP %Z,OVERLAY
            FIEL $START
            :TITLE PRINTED AND MAP MADE
            :YES, SKIP OVER THIS
            :OVERLAY TRAP AREA
            :CLEAR DEVICE UT PARAMETERS
            :SET BOTTOM OF SP STACK.
TYPE
FTITLE
INC FNONE
MOV #MAPNE,MACHER
MOV #RXCRD,%4
MAP: CMP (X4),#RXEND
BEQ MAPEND
BIC #BITO,(4)
CLR PSW
TST 0(4)
BR MAPOK
BIS POPSP2
BR MAP
MAP: MOV (4)+TEMP1
ISR TEMP1
MDEVAD 6
TYPE MDEVAD
INC FNONE
BR MAP
MAPENC: MOV #ERIP,MACHER
TST FNONE
BEQ MAPERR
START1: MOV #RXCRD,%1
START2: BIT #BITO,(1)
BNE START3
MOV #LINFO
SUB #RXCRD,LINFO
ASR LINFO
MOV (1),%1
ISR %7,FORMAD
BR START4
START3: CMP %1,#RXEND
BNE START2
END OF TABLE
START4: MOV #1,PASCNT
CLR PSW
CLR RTNNO
MOV #5RPT,#0
BIC #177770,#0
MOV #3,PRGNUM
MOV #4B,PRGTAB:4
: GO TO SELECTED PROGRAM.
MOV #START.NXTST, M: ADDR OF 1ST ROUTINE TO NXTST

MOV #RTMP RACHER, M: RESET RACHER TRAP.

CLR M: ;SET BOTTOM OF STACK.

MOVS PSW: ;ISSUE RESET.

JR #7, FORWD: ;ROLL FORWARD TO "NEXT" ROUTINE.

BNE GTRDYC: ;CHECK SELECT ROUTINE SWITCH.

TST UMASK: ;C/D DEVICE

BPL GTHDAI: ;NO CONTINUE

TST RTNNO: ;THIS A C/D TEST.

BPL GTRDYN: ;NO, DO NEXT TEST

TST: ;GO RUN CURRENT ROUTINE.

BR CHNB: ;NO GO. MANUAL RTN BYPASSED.

GTRDYB: ;(ISR) TO RD

CMP RTHNO, #0: ;COMPARE RTNNO TO (RD).

BNE GTRDYN: ;BRANCH IF ROUTINE NOT FOUND YET.

TST ACURST: ;GO RUN ROUTINE.

GTRDYC: ;NO, CHECK FOR LAST ROUTINE.

BNE GTRDYN: ;BRANCH IF NOT LAST ROUTINE.

ISR #7, INCR INTN: ;YES, INCORRECT ROUTINE SELECTED.

BR GETRDY: ;START OVER.

BIT #BIT4, #5RPRPT: ;CHECK FOR SCOPE OPTION.

BEQ CHNA: ;BRANCH IF SCOPE SW NOT SET.

MOV #SCOPTR, #5: ;SET UP TO RETURN TO ROUTINE.

RTI: ;RETURN TO ROUTINE.

TST #XORFLG: ;TEST FOR XOR

BPL 1$: ;

MOV #4, -(#6): ;

MOV #XOR, #4: ;

TST #17706: ;

MOV #(X)+, #4: ;

BIT #BIT11, #5RPRPT: ;TEST INHIBIT ITERATION SWITCH

BNE CHNAA: ;BRANCH IF INHIBIT ITERATION SW SET.

DEC ICTA: ;DECREMENT ITERATION COUNT.

BNE CHNAB: ;BRANCH IF COUNT NOT 0.

POP SP2: ;POP STACK TWICE

BIT #BIT15, #0: ;

BEQ CHNAB: ;

MOV RTNNO, #0: ;

BIC #BIT15, #0: ;

HALT: ;

BIT #BIT9, #5RPRPT: ;CHECK SELECT ROUTINE SWITCH

BNE GTRDY: ;BRANCH IF SELECT RTN SW SET.

CMP #1, NXTST: ;LAST TEST?

BNE GTRDYN: ;BRANCH IF NOT LAST TEST.

ADD E, #0: ;PROGRAM END.

CMP #E, #E: ;

MOV #E, #E: ;
I02

BR CHNAB

:INIT FOR C/D - WITHOUT JUMPER RESET STARTS ASSEMBLING CHARACTER SETTING DONE
:SET MAINT, DELAY, CLEAR RX DONE

CINIT: TST UMASK ;C/D DEVICE
BPL CDIX ;NO, EXIT
BIS #BIT2, ATXCSR ;SET MAINT BIT
DELAY ;WAIT 1.5 SEC
150D.
TST ARXBUF ;CLEAR RX DONE

CINIT: RTS %7

:FROMCWD: MOV NTXST, %5 ;ADDR OF NEXT ROUTINE TO RS.
MOV (5)+, RTNM ;GET NEXT ROUTINE NUMBER.
MOV (5)+, NXTST ;GET ADDR OF NEXT "NEXT" ROUTINE.
MOV (5)+, ICTR ;GET ITERATION COUNT.
MOV (5)+, SCOPOTR ;GET SCOPE LOOP ENTRY POINTER.
MOV %5, CURST ;ADDR OF NOW CURRENT TEST "CURST".
RTS %7 ;EXIT FORWARD SUBROUTINE.

:FROMINT: MOV A$6, -(B) ;GET SAVED PC.
SUB $2, A$6 ;DECREMENT PC BY 2.
EINT: MOV A$6, $%6
ASL A$6
BIS #110001, $%6 ;EMT ARG X 2.
ADD #EINTAB, A$6 ;FORM EMT RTN ADDR.
JMP A$6

;SAVE REGS 0 TO 4 SUBROUTINE.

:SAVRG: MOV (6)+, SVRPG ;SAVE PC AND PSW.
MOV (6)+, SVRPSW
MOV %4, -(B) ;SAVE REGS 0 - 4
MOV %3, -(B) ;IN STACK.
MOV %2, -(B) ;
MOV %1, -(B) ;
MOV %0, -(B) ;
MOV SVRPSW, -(B) ;RESTORE PC AND PSW.
MOV SVRPG, -(B) ;EXIT.

SVRPG: OPEN
SVRPSW: OPEN

;RESTORE REGS 0 TO 4 SUBROUTINE.

:STRG: MOV (6)+, RESTPC ;SAVE PC AND PSW.
MOV (6)+, RESTPSW ;
MOV (6)+, %0 ;RESTORE REGS 0 - 4 ;FROM STACK.
MOV (6)+, %1
MOV (6)+, %2
MOV (6)+, %3
MOV (6)+, %4
MOV RESTPSW, -(B) ;RESTORE PC AND PSW.
MOV PE, PC, -(B)
RTI

;EXIT

RTI

ASITPC: OPEN
ASITSC: OPEN

;ROUTINE TO SET RECEIVER INTERRUPT VECTOR AND PRIORITY
STLSRV: JSR #7,TSTVEC
MOV @6,STPRA+2 ;MOVE VECTOR ADDR TO STPRA+2
ADD #2,3,%E ;SET UP EXIT
MOV RXVTR,%1
STPRA: MOV #OPEN,(1)+ ;SET VECTOR ADDRESS
MOV RXVL,(1)+ ;SET PRIORITY
RTI ;EXIT

;ROUTINE TO SET TRANSMITTER INTERRUPT VECTOR AND PRIORITY.
STLSPV: JSR #7,TSTVEC
MOV @6,STPPA+2 ;MOVE VECTOR ADDR TO STPPA+2
ADD #2,3,%E ;SET UP EXIT
MOV TXVTR,%1
STPPA: MOV #OPEN,(1)+ ;SET VECTOR ADDRESS.
MOV TXVL,(1)+ ;SET PRIORITY
RTI ;EXIT.

;ROUTINE TO ISSUE RESET.
SASSERT: MOV $2525,%D ;DATA TO RD.
COM %D ;COMPLEMENT (RD).
MOV %D,$2525+2 ;(RD) TO SASSET+2.
RESET ;(RD) IS DISPLAYED. EXIT.
RTI ;EXIT.

;RANDOM NUMBER GENERATOR. ROUTINE EXITS WITH NUMBER IN REGISTER 0.
RANGEN: MOV RP1,%D
ROL %D
ROL %D
ADD RP2,%D
MOV %D,RP1
ROL %D
ROL %D
ADD RP2,%D
ROL %D
ROL %D
MOV %D,RP2
MOV RP1,%D
RTS #7 ;EXIT. NUMBEI IN RO

RP1: 1233
RP2: 7622

;L AçD - CLEAR CURRENT DEVICE PARAMETERS
CLÅCD: CLR TBUF
CLR TXCSR
CLR RXCSR
CLR RXBUF
CLR RXVTR
CLR TXVTR
CLR TLVL
CLR TXVL
P=%7
; SUBROUTINE TO OUTPUT ASCII MESSAGE ON TELETYPewriter.
MOV #26, XC ; GET ADDRESS THAT CONTAINS MESSAGE ADDRESS.
ADD #2, XC   ; SET UP EXIT.
MOV #0, XD   ; ADDRESS OF MESSAGE TO RO.
MOVB (D), YPDT ; GET CHARACTER
CMPB #100, YPDT ; CHECK FOR ""CHARACTER
BNE YPC ; BRANCH IF NOT "".
RTI
; PC:
CMPB #45, YPDT ; CHECK FOR "".
BEQ YPF ; BRANCH IF "".
BEQ YPR ; NOT "". CHECK FOR "".
JSR #7, YPF ; BRANCH IF "".
JSR #7, YPC ; TYPE CHAR IN YPDT
BR YPD
; TYPD:
MOV B YPDT, YTPB ; OUTPUT CHARACTER TO PRINTER
TSB YTPS ; WAIT FOR DONE FLAG.
BPL -4
; RTS:
RTS #7 ; EXIT
; TYPF:
MOVB #15, YPDT ; MOVE CARRIAGE RETURN CODE TO YPDT.
JSR #7, YPF ; GO TYPE CHAR.
; TYPG:
MOVB #12, YPDT ; MOVE LF CODE TO YPDT.
JSR #7, YPC ; GO TYPE CHAR.
BR YPD
; TYPD:
OPEN
; IMPD:
MOV #26, XD ; GET ADDRESS THAT CONTAINS MESSAGE ADDRESS
ADD #2, XD   ; UPDATE TO NEXT MESSAGE ADDRESS
MOV #00, XTSB ; ADDRESS OF MESSAGE TO XTSB
CMP #1, XTSB ; CHECK FOR TERMINATOR
BNE XTPS ; BRANCH IF NOT TERMINATOR.
RTI
; TYPSA:
TYPE ; CALL ON TYP SUB TO TYPE MESSAGE
; TYPB:
OPEN ; ADDRESS OF MESSAGE GOES HERE
BR XSYS ; GO PROCESS NEXT MESSAGE
; OVERLAY VECTOR AREA
MOV #300, X1 ; GET DLII-E VECTOR BASE ADDRESS
MOV #302, X2
MOV #4, X3
MOV #2, X1+ ; LOAD VECTOR WITH IOT ERROR TRAP
MOV #3, X1+ ; ALL VECTORS BEEN LOADED
ADD #4, X2
BEQ OVRLYA
BR OVRLYA
; OVRLYA:
RTS 7 ; EXIT
; SUBROUTINE TO DELAY A SPECIFIED NUMBER OF MILLISECONDS
MOV #26, DLCNT ; GET DELAY COUNT ADDRESS.
ADDC #2, DLCNT ; SET UP EXIT ADDRESS
MOV #DLCNT+(-6) ; DELAY COUNT TO STACK
BEQ DLYC
CLR PSW ; SET PRIORITY 0
DLYA:
MOV #226, -X6 ; 1 MSEC COUNT TO STACK
DLYB:    DEC     8:6  ; DECREMENT I MSEC COUNT
BNZ     DLYB  ; BRANCH IF NOT 0
POPS    ; ZERO, UNCOVER MSECs. COUNT.
DEC     8:6  ; DECREMENT IT
BNE     DLYA  ; BRK IF NOT DONE DELAYING
DLYC:    POPS    ; DONE
RTI     ; EXIT
DLCNT:   OPEN    ; CONTAINS MILLISECONDS COUNT ADDRESS.

: SUBROUTINE TO STALL A RANDOM NUMBER OF MILLISECONDS. MAXIMUM STALL
: DETERMINED BY CONTENTS OF LOC STILMSK.
STAL:    JSR     %7, RNGEN  ; GO GET RANDOM NUMBER
           BIC     SILMSK,%D  ; IN AQ, APPLY STALL MASK.
BEQ     STALB  ; BRANCH IF RESULT IS 0.
MOV     %D,STAL
DELAY   ; DELAY
STALA:   OPEN    ; DELAY COUNT
STALB:   RTI     ; DONE. EXIT.

: SUBROUTINE TO GENERATE RANDOM CHARACTER COUNT
GRCNT:   JSR     %7, RNGEN  ; GET RANDOM NUMBER
           BIC     RRCNS,%D  ; APPLY MASK
BEQ     GRCNT  ; TRY AGAIN IF RESULT 0
MOV     %D, RNCNT
RTS     %7  ; EXIT.

RRCNS:   OPEN    ; RANDOM CHARACTER MASK.
RNCNT:   OPEN    ; RANDOM CHARACTER COUNT.

: SUBROUTINE TO SKIP ON FLAG AND TIME OUT IF SKIP FAILS
TMX:     MOV     RXCSR, SIOT  ; SET UP RXCSR ADDRESS
BR     TIME1
TMXT:    MOV     TXCSR, SIOT  ; SET UP TXCSR ADDRESS
TIME1:   CLR     TIMER
TIME2:   INC     TIMER
BEQ     TIME2  ; BRANCH IF COUNTER OVERFLOW
TSTB    #SIOT
BPL     TIME2
ADC     #2,%5  ; SET UP EXIT RETURN

TIME2:   RTI
TIMEX:   RTI
TIMEx:   0
SIOT:    0

: SUBROUTINE TO SELECT LINE
LINS:    BIT     #BIT2,ASRPTR
BNE     LINS|L  ; BRANCH IF SET
CLR     Found
RTS     5
LINS|L:   JSR     %7, OVLAY
JSR     %7, CLRC0
TYPE    LNLIN
HALT
MOV     ASRPTR TEMP
BIC     #177740, TEMP
MOV TEMP, LENO
ASL TEMP, LENO : SAVE FOR TYPING
MOV  TEMP, %1
MOV  RXCSR(I), %1  ; GET RXCSR DEVICE ADDRESS
BIT  #BIT, %1      ; YES DEVICE THERE
BEQ  LINA  ; NO, REPORT

LINA:  TYPE
      MNOLIN
      BR  LINSX

-LINB:  JSR  %7, FORMAD
        CLR  PSW
        BIS  #BIT, FMAP  ; SET MAPPING FLAG
        BIC  #BIT, TXCSR
        NOP
        NOP
        TST  RXVTR
        BEQ  LINA
        BIC  #BIT, TXCSR
        MOV  #PTY, PSW
        JSR  5, OACHV  ; TYPE LINE #
        LINENO
        SELINE
        2
        TYPE
        ALINE
        RTS  5

: SUBROUTINE TO INITIALIZE BINARY COUNT PATTERNS

INBIN:  MOV  #1, RIND  ; SET ALL VARIABLES
        JSR  %5, BMOVE  ; TO MINUS 1.
        RIND
        RIND+1
        11
        RTS  %7  ; EXIT

RIND:  OPEN
PTD:  OPEN
PTL:  OPEN
PTO:  OPEN
PTOP:  OPEN
PTIP:  OPEN

: SPECIAL BINARY COUNT PATTERN SUBROUTINE. EXITS WITH BIN CHAR IN RO

GTBIN:  MOV  PTO, PTI  ; PREVIOUS BIN CHAR TO PTI
        COM  PTI
        COM  RIND
        BNE  +6
        INC  PTI
        BIC  #17, PTI  ; MASK TO 8 BITS
        MOV  PTI, PTO  ; SAVE BIN CHAR IN PTO
        MOV  PTI, %0  ; BIN CHAR TO RO.
        RTS  %7  ; EXIT.

GTBINP:  MOV  PTOP, PTIP  ; PREVIOUS BIN CHAR TO PTIP
        COM  PTIP
        COM  FIND
        BNE  +6
        INC  PTIP
        BIC  #17, PTIP  ; MASK TO 8 BITS.
MOV PTIP,PTOP ;SAVE BIN CHAR IN PTOP.
MOV PTIP,R1 ;BIN CHAR TO R1.
RTS %7 ;EXIT.

;TOTAL ASCII CONVERT ROUTINE
;ANCH: SAVREG
MOV @5+,OACNVX ;GET OCTAL VALUE.
MOV @5+,%1 ;GET DESTINATION ADDR.
MOV @5+,%2 ;GET CONVERT COUNT.
ADD %1,%2 ;DEVELOP ADDR TO STORE 1ST CHAR.

OACNVX: MOV OACNVX,%3
BIT #177770,%3 ;ISOLATE LEAST SIGNIFICANT DIGIT.
ADD %60,%3 ;CONVERT DIGIT TO ASCII.
MOV %3,-(1) ;STORE ASCII CHARACTER.
BIC #7,OACNVX
ROR OACNVX
ROR OACNVX
ROR OACNVX
DEC %2 ;DONE ALL DIGITS?
BNE OACNVX ;BRANCH IF NOT DONE.

;SUBROUTINE TO MOVE A VARIABLE NUMBER OF BYTES.
;BMOV: SAVREG
MOV (5)+,%1 ;SAVE REGS.
MOV (5)+,%2 ;GET "FROM" ADDRESS
MOV (5)+,%3 ;GET "TO" ADDRESS

BMOV: MOVB (1)+,%3 ;MOVE BYTE
DEC %3 ;DECREMENT COUNT.
BNE BMOV ;BRANCH IF NOT DONE.

;iRTS %5 ;DONE EXIT

;BINARY TO DECIMAL ASCII CONVERT SUBROUTINE.
;BDCNV: SAVREG
MOV #DECVL,%D ;SET UP ADDR TO STORE DECIMAL ASCII IN RC.
MOV @5+,%D ;BINARY VALUE TO R1.
MOV @5+,%BDCNV ;GET DEST ADDR.
MOV @5+,%BDCNV ;GET CHAR COUNT.
MOV #10TENP,%2 ;ADDR OF TEN POWER STRING TO R2.
MOV %5,CNVCTR ;SET UP FOR 5 POWER CONVERSIONS.

SUB %5,CNVCTR ;BEGIN CONVERSION.
DEC CNVCTR ;SUB CNVCTR ;DONE 5 CONVERSIONS?
BNE BDCNV ;BRANCH IF NOT YET 5.
MOV %5,BDCNV ;DONE 5 CONVERSIONS.
ISAS %5,BMOV

;iRTS %5 ;YES EXIT.

;SUBTR: SUB DIGIT ;CLEAR DIGIT
;iRTS %5 ;YES EXIT.
SUBTRACT TEN POWER FROM BINARY VALUE.

BRANCH IF UNSUCCESSFUL SUBTRACTION.

RESTORE SUBTRACTED VALUE.

CONVERT (DIGIT) TO ASCII.

MOVE ASCII CHAR TO DECIMAL FIELD.

EXIT.

OPEN

OPEN

OPEN TO TEMP:

10000.

100.

1.

DECIMAL: BYTE 040,040,040,040,040,040

CLEAR DATA TERM. READY

SET MAINTENANCE BIT

GET CHARACTER COUNT

WAIT FOR

READY FLAG

GET CHARACTER

MOVE CHARACTER

MASK OFF NON TRANSMITTED BITS

TRANSMIT CHARACTER

WAIT FOR

DME

GET RECEIVED CHARACTER

CHK DATA

DECREMENT CHARACTER COUNT

POP STACK

TYPE SELECT OPTION MESSAGE.

COMMON HALT.

EXIT.

EXIT.

COMMON HALT.

EXIT.

INC

START

STORE AWAY PASS COUNT

INC

LINENO ERRST+22

INC

RXCSR,ERRST+24

INC

RXVTR,ERRST+26

INC

#60,TMP6

RESET

DEC

TEMP6
BNE .16, .SRPTR
BIT #BIT12, .SRPTR
BEQ .28
ISR PC, EOPHLT

BB: BIT #BIT13, .SRPTR
BNE PRGEXT
ISR %5, BDCHV

PS: PASCNT
APCMT
ISR %5, 0ACNV
; CONVERT LINE NUMBER
LINENO
ACPML
ISR %5, 0ACNV
; CONVERT RXCSR
RXCSR
APRXC
ISR %5, 0ACNV
; CONVERT VECTOR
RXVTR
APVEC
TYPE
APGEND
JMP .46

TEMP5: .WORD 0

PRGEXT: BIT #BIT12, .SRPTR
BEQ PRGXTI
BNE PRGEXT
INC PASCNT
BR PRGXTL

PRGXTI: MOV LINENO, TEMP
ASL TEMP
; GET LINENO
PRGEXT: ADD %5, TEMP
ADD %5, TEMP
; UPDATE LINE NUMBER
PRGCHA: MOV TEMP %1
MOV RXCSR(1), %1
; GET RXCSR DEVICE ADDRESS
CMP #177777, %1
; LAST ONE
BNE PRGEB
INC PASCNT
CLR LINENO
CLR TEMP
PRGXTL: MOV #42, %5
BEQ CONT
RESQ
RESQ
LOGIC: JSR 7

CONT: BIT #BIT12, .SRPTR
BEQ PRGCHA
BIT #BIT12, .SRPTR
BNE PRGEB
MOV TEMP, LINENO

ISR .T.FORMAD
RTS 7

:CONDITIONAL ERROR HALT ROUTINE.
EHL: BCT $SRPTE ;CHECK FOR HALT ON ERROR.
BPL EHLTA ;BRANCH IF NO HALT DESIRED.
HALT HALT ;HALT.
EHLTA: RTI ;IN DATA LIGHTS.

:MASK - MASK DATA ACCORDING TO LINE NUMBER
MASK: MOV UMASK,RMASK ;SET MASK
BIC $17000,RMASK ;REMOVE C/D FLAG+PRIORITY
COM RMASK
BIC RMASK,CRBUFA ;MASK DESIRED BITS
RTS 7

:DATA CHECK ROUTINE, TEST ERROR BITS
CTCK: MOV CRBUFB,CRBUFB ;DID ANY ERROR BITS SET
BIT $17000,CRBUFB
BNE DTCKX ;YES, TYPE ERROR
CMP CRBUFB,CRBUFA ;COMPARE EXPECTED AND RECEIVED
BEC DTCKA ;CHARS. BRANCH IF SAME.
DTCKA: MOV CRBUFA,ERRST+6
MOV CRBUFB,ERRST+10
JSR %5,OACNV
CRBUFB %5,OACNV
GO TO OCTAL TO ASCII CONVERT.
CRBUFA %5,OACNV
SOURCE ADDR.
CRBUFA 3
DESTINATION ADDR.
3
#OF DIGITS TO CONVERT.
3
CRBUFB %5,OACNV
GO TO OCTAL TO ASCII CONVERT.
CRBUFB %5,OACNV
SOURCE ADDR.
RTA
DESTINATION ADDR.
ERROR!

CTCKA: RTI

:ERROR HANDLER
EPP: MOV 8-1,ERRB ;SET UP ONE MESSAGE CALL.
MOV #240,ERRB+2
CLR ERRB
BR ERRB ERRB

EPRI: MOV #6,ERRB ;DEVELOP ADDT'L MESSAGE ADDR.
MOV #ERRB,ERRB
MOV #1,ERRB+2
MOV #2,ERRB

EPPA: BIT $813,SRPTE ;INHIBIT ERROR PRINT?
BNE ERRC
MOV #6,ERRB ;DEVELOP CALLING ADDR.
SUB #2,ERRB
MOV RTNO,TNNO
BIC #813,TNMC
JSR %5,OACNV ;GO TO OCTAL TO ASCII CONVERT.
ERCC  : SOURCE ADDR.
APC   : DESTINATION ADDR.
65    : NOF DIGITS TO CONVERT.
ISR   : GO TO OCTAL TO ASCII CONVERT.
RXCSR : SOURCE ADDR.
MAXIMUM: DESTINATION ADDR.
65    : NOF DIGITS TO CONVERT.
ISR   : GO TO OCTAL TO ASCII CONVERT.
RTNNO : SOURCE ADDR.
RTNUMB : DESTINATION ADDR.
3     : NOF DIGITS TO CONVERT.
TYPES : TYPE.
END   : ERROR HEADER.
ERRB  : ADDT'L ERROR MESSAGE IF ANY.

ERRB:  OPEN
-1
MOV   RTNNO, ERRST
MOV   ERRD, ERRST+2
MOV   RXCSR, ERRST+4

ERRC:  ENHALT
ADD   ERR, B#6
RTI

ERRD:  OPEN

ERRD:  OPEN

ERROR TRAP HANDLER - TYPE TC AND FROM WHERE ERROR TRAP OCCURRED

ERTP:  MOV   PSW, OLDP5
MOV   @PR1+7, PS
ASR   OLDP5
ASR   OLDP5
ASR   OLDP5
BIC   #1777740, OLDP5
MOV   OLDP5, TOPC
MOV   B#6, FROMPC

ERTPA: ISR   B#5, 0ACNV
TOPC  MTG
6     %5, 0ACNV
FROMPC

ERROR - MAP VECTOR OR REPORT ERROR DEPENDING ON FMAP FLAG

MAPVEC: MOV   B#6, TOPC
POPSP2
MOV   B#6, FROMPC
SUB   B#4, TOPC
TST   FMAP
BEC   ERTPA
MOV   TOPC, TYVTR
MOV   B#4, TOPC
MOV   TOPC, R/YTR
MOV   TOPC, R/A/TR

: NOT MAPPING, REPORT ERROR
: STORE VECTOR
```
; Format Form Device at Addresses
; Format: MOV %1, RXCSR
; ADD %2, %1
; MOV %2, RXBUF
; ADD %2, %1
; MOV %1, TXCSR
; ADD %2, %1
; MOV %1, TXBUF
; MOV LINENO, TEMP
            ; GET PRIORITY
; ASL TEMP
; MOV @MSAO, TEMP
; MOV @TEMP, TEMP1
; MOV TEMP1, TEMP1
; SWAP TEMP
; ASL TEMP1
; BIC @177437, TEMP1
; MOV TEMP1, RLVL
; MOV TEMP1, TXLVL
; RTS %7

; DoThis - Selectable Test Decision Maker
; DoThis: BIT #BIT3.SRPTR
            ; Is SELECT TEST SWITCH SET
; BNE GORBACK
            ; RETURN TO TEST IF SW SET
; JMP GRTDXY
            ; Go to NEXT TEST
; GORBACK: RTS %7
; PFAIL: MOV #PWRUP, 24
; HALT: MOV #PFAIL, 24
; PWRUP: MOV #RESET, 16
; TYPE: MOV #SPBOT, 16
; ERROR: JMP RESTART

; Decide if Vector to be Mapped and Map
; S_VEC: CMP #0, FOUNDV
            ; Need Vector Mapping
; BNE TSTIVEX
            ; No, EXIT
; JSR %7, OVRLAY
; CLR RXVTR
; CLR PSW
; BIT #BIT0, FMAP
            ; Set Mapping Flag
; BIT #BIT6, 2TXCSR
            ; Cause Interrupt
; NOP
; NOP
; TEST RXVTR
            ; Did Trap Occur?
; BNE #BIT13, ISPRT
; BNE #BIT13, ISPRT
; TYPE S_VEC
            ; No, ERROR
```
```
; ERROR
IMM  ; SVE
"STVA: BIC #BITB, @XCSR
MOV #PARTY, PSW  ; RAISE PRIORITY, RETURN
"STVE: RTS :?

; RESTART ROUTINE
RESAP: MOV PRGNUM, #0
ASL \\nJMP #RESTART, 0   ; GO RESTART SELECTED PROGRAM

; RESTART: PRGA
:PRGA: PROGRAM 0 RESTART ADDRESS
PRGA1: PROGRAM 1 RESTART ADDRESS
PRGA2: PROGRAM 5 RESTART ADDRESS
PRGA3: PROGRAM 3 RESTART ADDRESS
PRGA4: PROGRAM 4 RESTART ADDRESS
INCRGA
INCRGA
INCRGA

; RAD - INPUT-OUTPUT LOGIC TESTS
PRGA:  MOV #ATD.KSTART
ST:    #ATC.MSTART
SNE #PRG0B
:PRG0B: :MONITOR LOAD
:PRG0A: :TEST START TEST
:PRG0C: :TYPE TITLE AND INSTRUCTIONS

PRGA0: JSR #SETSR
PRGA1: JSR #SETINSEL
JMP GETADY  ; GET STARTED.

***************
ATD: 100000  ; TEST NUMBER
ATC: 101000  ; ADDRESS OF NEXT TEST
10000  ; ITERATION COUNT
00000  ; SCOPE ENTRY POINT
X X 1

***************
; TEST ABILITY TO REFERENCE RECEIVER CSR WITHOUT TRAPPING
AA: MOV #MAE.MACHER  ; SET UP MACHINE ERROR TRAP.
CAR  #RXCSR  ; REFERENCE RXCSR
ABB  #SOPSP2  ; OK IF NO TRAP. SCOPE
ABB  #SOPSP2  ; TRAPPED WHEN REFERENCING RXCSR.

***************
A1: 100001  ; TEST NUMBER
A2: 101000  ; ADDRESS OF NEXT TEST
10000  ; ITERATION COUNT
00000  ; SCOPE ENTRY POINT
X X 1
```

TEST ABILITY TO REFERENCE RECEIVER BUFFER WITHOUT TRAPPING

ACB: MOV @MACH, @XORF.3
BMI ABB
TST @XBUF
:REFERENCE RXBUF
JE AB
:CHECK IF NO TRAP SCOPE

ACB: POPSP2
ERROR
:TRAPPED WHEN REFERENCING RXBUF
BR ABB

ACB: !00003

ACB: MOV @MACH, @XORF.3
BMI ABB
TST @XCSR
:REFERENCE TXCSR

ACB: POPSP2
ERROR
:TRAPPED WHEN REFERENCING TXCSR
BR ABB

ACB: !00004

ACB: MOV @MACH, @XORF.3
BMI ABB
TST @XBUF
:REFERENCE TX BUF.

ACB: POPSP2
ERROR
:TRAPPED WHEN REFERENCING TXBUF
BR ABB

ACB: !00005

TEST THAT TXCSR BIT 0 (BREAK) CAN BE SET AND CLEARED
AND THAT RESET CLEARS IT

AEB: BIT #BITO.XCSR
:SEE IF BIT IS CLEAR
BMI AEB
:BR IF CLEAR
ERROR
:RESET DID NOT CLEAR IT

AEB: !BITO.XCSR
:SET TXCSR BIT 0
BIT #BITO.XCSR
:DO IT SET
BNE AEC
:YES, GO ON
ERROR
:TXCSR BIT 0 FAILED TO SET

AEB: !BITO.XCSR
:CLEAR TXCSR BIT 0
BIT *BIT0,DTXCSR : DID IT CLEAR
BEQ AED
ERROR
AED:
BIS *BIT0,DTXCSR
SRESET
SCOPE
 : ISSUE RESET TO CLEAR

************

;TEST NUMER
AT6: 100005
;ADDRESS OF NEXT TEST
A6: 100006
;ITERATION COUNT
10:
AGA: X=X+1
;SCOPE ENTRY POINT

;TEST THAT TSCSR BIT2 CAN BE SET, CLEARED, AND THAT RESET Clears IT.
AGA:
BIT *BIT2,DTXCSR
;SEE IF TSCSR BIT2 IS CLEAR.
BEQ AGB
ERROR
BR AGD
AGB:
BIS *BIT2,DTXCSR
;SET TSCSR BIT2.
BIT *BIT2,DTXCSR
;SEE IF BIT IS SET.
BNE AGC
ERROR
BR AGD
AGC:
BIC *BIT2,DTXCSR
;CLEAR TSCSR BIT2.
BIT *BIT2,DTXCSR
;SEE IF BIT IS CLEAR.
BEQ AGD
ERROR
BR AGD
AGD:
BIS *BIT2,DTXCSR
;SET TSCSR BIT2.
SRESET
SCOPE
; ISSUE RESET TO CLEAR BIT.

************

;TEST NUMER
AT7: 100006
;ADDRESS OF NEXT TEST
A7:
AGA: X=X+1
;SCOPE ENTRY POINT

;TEST THAT TSCSR BIT6 CAN BE SET, CLEARED, AND THAT RESET Clears IT.
AGA:
MOV *PRNT,P5W
;SET PRIORITY 7.
BIT *BIT6,DTXCSR
;SEE IF TSCSR BIT6 Is CLEAR.
BEQ AJB
ERROR
BR AJD
AJB:
BIS *BIT6,DTXCSR
;SET TSCSR BIT6.
BIT *BIT6,DTXCSR
;SEE IF BIT IS SET.
BNE AJC
ERROR
BR AJD
AJC:
BIC *BIT6,DTXCSR
;CLEAR TSCSR BIT6.
BIT *BIT6,DTXCSR
;SEE IF BIT IS CLEAR.
BEQ AJD
ERROR
BR AJD
AJD:
BIS *BIT6,DTXCSR
;SET TSCSR BIT6.
SRESET
SCOPE
; ISSUE RESET TO CLEAR BIT.
A12: TEST NUMBER
AT13: ADDRESS OF NEXT TEST
10: ITERATION COUNT
AQA: SCOPE ENTRY POINT
x=x+1

; TEST THAT RXCSR BIT3 CAN BE SET, CLEARED, AND THAT RESET clears IT.
AQA: BIT #BIT3, RXCSR ; set if RXCSR BIT3 IS CLEAR.
BEQ AQB ; branch if bit is clear.
ERROR AQA ; reset did not clear RXCSR BIT3
BR AQA

AQB: BIS #BIT3, RXCSR ; set RXCSR BIT3.
BIT #BIT3, RXCSR ; see if bit is set.
BNE AQC ; branch if bit is set.
ERROR AQC ; RXCSR BIT3 failed to set.
BR AQC

AQC: BIT #BIT3, RXCSR ; clear RXCSR BIT3.
BEQ AQB ; see if bit is clear.
ERROR AQB ; RXCSR BIT3 failed to clear.
BR AQB

AQB: BIS #BIT3, RXCSR ; set RXCSR BIT3.
SRESET SCOPE ; issue reset to clear bit.

A13: TEST NUMBER
AT14: ADDRESS OF NEXT TEST
10: ITERATION COUNT
AQA: SCOPE ENTRY POINT
x=x+1

; TEST THAT RXCSR BIT5 CAN BE SET, CLEARED, AND THAT RESET clears IT.
ARA: MOV #PR7, PSW ; PR7 to inhibit any INT
BIT #BITS, RXCSR ; see if RXCSR BITS is CLEAR.
BEQ ARB ; branch if bit is clear.
ERROR ARB ; reset did not clear RXCSR BITS
BR ARD

ARB: BIS #BITS, RXCSR ; set RXCSR BITS.
BIT #BITS, RXCSR ; see if bit is set.
BNE ARC ; branch if bit is set.
ERROR ARC ; RXCSR BITS failed to set.
BR ARC

ARC: BIT #BITS, RXCSR ; clear RXCSR BITS.
BEQ ARD ; see if bit is clear.
ERROR ARD ; RXCSR BIT4 failed to clear.
BR ARD

ARD: BIS #BITS, RXCSR ; set RXCSR BITS.
SRESET SCOPE ; issue reset to clear bit.

A14: TEST NUMBER
AT15: ADDRESS OF NEXT TEST
10: ITERATION COUNT
AQA: SCOPE ENTRY POINT
x=x+1
TEST THAT RXCSR BIT 6 CAN BE SET, CLEARED, AND THAT RESET CLEARS IT.

ASH: MOV #PRY7, PSW
      BIT #BIT6, RXCSR
      BRZ ASB
      BEQ ASB
      ERROR
      BR ASD

ASB: BIT #BIT6, RXCSR
      BR ASD
      BNE ASC
      ERROR
      RXCSR BIT 6 FAILED TO SET.

ASC: BIT #BIT6, RXCSR
      BRZ ASC
      BEQ ASC
      ERROR
      RXCSR BIT 6 FAILED TO CLEAR.

ASD: BIT #BIT6, RXCSR
      SRESET
      SCOPE

;*******************************************************************************

AT5:  15
      TEST NUMBER
      AT16
      ADDR OF NEXT TEST
      100.
      ITERATION COUNT
      X=X+1
      SCOPE ENTRY POINT

;*******************************************************************************

;TEST THAT RXCSR BIT 7 IS CLEAR AND CAN BE READ RELIABLY.

ATA:  BIT #BIT7, RXCSR
      BR ATB
      BEQ ATB
      ERROR
      RXCSR BIT 7 IS NOT CLEAR.

ATB:  SCOPE

;*******************************************************************************

AT16: 16
      TEST NUMBER
      AT17
      ADDR OF NEXT TEST
      100.
      ITERATION COUNT
      X=X+1
      SCOPE ENTRY POINT

;*******************************************************************************

;TEST THAT RXCSR BIT 10 IS CLEAR AND CAN BE READ RELIABLY.

AXA:  BIT #BIT10, RXCSR
      BR AXB
      BEQ AXB
      ERROR
      RXCSR BIT 10 IS NOT CLEAR.

AXB:  SCOPE

;*******************************************************************************

AT17: 100017
      TEST NUMBER
      AT20
      ADDR OF NEXT TEST
      100.
      ITERATION COUNT
      X=X+1
      SCOPE ENTRY POINT

;*******************************************************************************

;TEST THAT RXCSR BIT 11 IS CLEAR AND CAN BE READ RELIABLY.

ATA:  BIT #BIT11, RXCSR
      BRAY
      BEQ AYB
      ERROR
      RXCSR BIT 11 IS NOT CLEAR.

AYA:  SCOPE

;*******************************************************************************
AYB:  SCOPE

AT20:  100020
  AT21
  100.
  AZO
  X=X+1

TEST THAT RXCSR BIT14 IS CLEAR AND CAN BE READ RELIABLY.
AZA:  BIT  #BIT14,WRXCSR  SEE IF RXCSR BIT14 IS CLEAR.
      AZD  BEQ    AZB  BRANCH IF BIT IS CLEAR.
      ERR  RXCSR BIT14 IS NOT CLEAR.
      SRES  RESIZE BIT IF ERROR
AZB:  SCOPE

AT21:  100021
  AT22
  100.
  AAAA
  X=X+1

TEST THAT RXCSR BIT15 IS CLEAR AND CAN BE READ RELIABLY.
AAAA:  BIT  #BIT15,WRXCSR  SEE IF RXCSR BIT15 IS CLEAR.
      AAA  BEQ    AAA  BRANCH IF BIT IS CLEAR.
      ERR  RXCSR BIT15 IS NOT CLEAR.
      SRES  RESIZE BIT IF ERROR
AAAA:  SCOPE

ALL PREVIOUS TESTS MUST HAVE BEEN RUN SUCCESSFULLY PRIOR
TO RUNNING THE FOLLOWING TESTS. ALSO, THE JUMPER CONNECTOR
MUST BE INSERTED IN THE DIII-E CABLE IN PLACE OF THE MODEM. COMMENTS
REFER TO OPERATION WITH JUMPER INSERTED.

AT22:  22
  AT23
  100.
  AFBA
  X=X+1

TEST THAT CARRIER DETECT SETS AND CLEARS WHEN DATA TERMINAL
READY SETS AND CLEARS.
AFBA:  BIS  #BIT1,WRXCSR  SET DATA TERMINAL READY
       BIT  #BIT2,WRXCSR  TEST CARRIER DETECT
       BNE  AFBB  SHOULD BE SET
       ERR  AFBB  WASN'T
AFBB:  BR  AFBC  CLEAR DATA TERMINAL READY
       BIT  #BIT1,WRXCSR  TEST CARRIER DETECT
       BEQ  AFBC  WAS SET, ERROR
AFBC:  SCOPE

AT23:  23
  AT24
  100.
**SCOPE ENTRY POINT**

**TEST THAT MODEM INTERRUPT (BIT 15) SETS WHEN CARRIER DETECT CHANGES STATE, AND IS CLEARED WHEN RXCSR IS READ.**

**AJBA:**

1. MOV RXCSR, RXCSR
2. BIT #BIT15, RXCSR
3. TEST MODEM INTERRUPT
4. BR AGBB
5. AGBB
6. ERROR
7. BR AGBE
8. AGBE
9. GO TO SCOPE

**AJBB:**

1. MOV RXCSR, RXCSR
2. BIT #BIT15, RXCSR
3. SETTING DATA TERMINAL READY
4. ERROR
5. BR AGBE
6. AGBE
7. GO TO SCOPE
8. AGBC
9.应该设置 GO TO AGBC
10. AGBC
11. BIT #BIT15, RXCSR
12. MODEM INTERRUPT BIT SHOULD HAVE BEEN CLEARED
13. DEO AGBD
14. IT WAS GO TO AGBD
15. AGBD
16. BIT #BIT15, RXCSR
17. CLEARING DATA TERMINAL READY
18. ERROR
19. BR AGBE
20. AGBE
21. GO TO SCOPE
22. AGBD
23. BIT #BIT15, RXCSR
24. MODEM INTERRUPT WILL SET
25. MOV RXCSR, RXCSR
26. BIT #BIT15, RXCSR
27. TEST MODEM INTERRUPT
28. BNE AGBE
29. IT WASN'T
30. AGBE
31. BR AGBE
32. AGBE
33. IT WASN'T
34. SCOPE

**A24:**

1. TEST NUMBER
2. AT25
3. ADDRESS OF NEXT TEST
4. IOO
5. ITERATION COUNT
6. AJBA
7. SCOPE ENTRY POINT
8. X=X+1

**TEST THAT CLEAR TO SEND (BIT 13) SETS/CLEARS A IN DATA TERMINAL READY SET/CLEAR**

**AJBA:**

1. MOV RXCSR, RXCSR
2. BIT #BIT13, RXCSR
3. CLEAR DATA TERMINAL READY
4. ERROR
5. BR AJBB
6. AJBB
7. CLEAR TO SEND SHOULD BE CLEAR
8. BR AGBP
9. AGBP
10. BIT #BIT13, RXCSR
11. SET DATA TERMINAL READY
12. ERROR
13. BR AJBD
14. AJBD
15. CLEAR TO SEND SHOULD BE SET
16. BR AGBD
17. AGBD
18. BIT #BIT13, RXCSR
19. CLEAR DATA TERMINAL READY
20. ERROR
21. BR AJBC
22. AJBC
23. CLEAR TO SEND SHOULD BE CLEAR
A26: SCOPE

********************************************************************

A26: 26
A26: ADDRESS OF NEXT TEST
A26: Iteration count
A26: SCOPE ENTRY POINT

********************************************************************

TEST THAT RING (BIT 14 RXCSR) SETS WHEN REQUEST TO
RESET SETS AND CLEARS AND RESET CLEARS RING

A2B: BIC #BIT2, RXCSR : CLEAR REQUEST TO SEND
    BIT #BIT15, RXCSR : TEST MODEM INTERRUPT BIT
    BNE A2BB : SHOULD BE CLEAR
    BEQ A2BB : Should be clear

A2BB: BIC #BIT2, RXCSR : CLEAR REQUEST TO SEND
    BIT #BIT15, RXCSR : TEST MODEM INTERRUPT BIT

********************************************************************

A27: 27
A27: ADDRESS OF NEXT TEST
A27: Iteration count
A27: SCOPE ENTRY POINT

********************************************************************

TEST THAT SUPERVISORY RECEIVE DATA (BIT 10 RXCSR) SETS/CLEARS

A3B: BIC #BIT3, RXCSR : CLEAR SUPERVISOR XMIT DATA
    BIT #BIT10, RXCSR : TEST SUPERVISORY RECEIVE DATA.
    BEQ A2BB : SHOULD HAVE BEEN CLEAR
BR ALBD
ALBE: BIT #BIT3.0RXCSR : SET SUPERVISORY XMIT DATA
BIT #BIT10.0RXCSR : TEST SUPERVISORY RECEIVE DATA
BNE ALBC ; SHOULD HAVE BEEN SET
ERROR BR ALBD
ALPC: BIT #BIT3.0RXCSR : CLEAR SUPERVISORY XMIT DATA
BIT #BIT10.0RXCSR : TEST SUPERVISORY RECEIVE DATA
BEQ ALBD ; SHOULD HAVE BEEN CLEAR
ALBC: SCOPE : SCOPE

A-32: 30 ; TEST NUMBER
AT3: ADDRESS OF NEXT TEST
100: ITERATION COUNT
AMBC : SCOPE ENTRY POINT
X=X+1

*****************************************************************************

; TEST THAT SUP REC DATA TRANSITIONS SET MODEM INTERRUPT
AMBA: BIC #BIT3.0RXCSR : CLEAR SUP REC
BIT #BIT3.0RXCSR : SET SUP REC
BIT #BIT15.0RXCSR : TEST MODEM INTERRUPT
BNE AMBB ; MODEM INTERRUPT SHOULD BE SET
ERROR BR AMBE
AMAB: BIT #BIT15.0RXCSR : MODEM INTERRUPT SHOULD BE
BEQ AMBC ; CLEARED BY PREVIOUS READ
ERROR BR AMBE
AMBC: BIC #BIT3.0RXCSR : 1-0 TRANS OF SUP REC DATA
BIT #BIT15.0RXCSR : TEST MODEM INTERRUPT
BNE AMBD ; SHOULD BE SET
ERROR BR AMBE
AMBD: BIS #BIT3.0RXCSR : 0-1 TRANS OF SUP REC DATA
BIT #BIT15.0RXCSR : TEST MODEM INTERRUPT
BNE AMBE ; SHOULD BE SET
AMBE: SCOPE : SCOPE

*****************************************************************************

A-31: 100031 ; TEST NUMBER
AT32 : ADDRESS OF NEXT TEST
10 : ITERATION COUNT
ABAR : SCOPE ENTRY POINT
X=X+1

*****************************************************************************

; TEST THAT RESET CLEARS ALL TXCSR BITS, AND SETS BIT 7 (READY)
ABAR: MOVS #PR1T,PSW ; SET PRIORITY
MOVS #1,0TXCSR ; SET ALL POSSIBLE BITS IN TXCSR
SRESET ; ISSUE RESET TO CLEAR BITS
CMP #BIT7,0TXCSR ; SEE IF ONLY BIT 7 IS SET
BEQ ABAB ; BRANCH IF ONLY BIT 7 IS SET
MOVS 0TXCSR.0TXCSR ; SAVE CONTENTS OF TXCSR
MIV #BIT7,TEMP ; MOVE EXPECTED TXCSR TO TEMP
JSP %S,0AHV ; GO TO OCTAL TO ASCII CONVERT.
EMP %S,0AHV ; SOURCE ADDR.
ATXSB
6
ISR %5,0ACNV
TXCSR
ATXHAS
6
ERROR1
HITXSR
SCOPE:
:DESTINATION ADDR.
:GO TO OCTAL TO ASCII CONVERT.
:SOURCE ADDR.
:DESTINATION ADDR.
:DESTINATION ADDR.
:DESTINATION ADDR.
:DESTINATION ADDR.
:RESET FAILED TO CLEAR ALL BITS EXCEPT BIT 7 - SEE PRINTOUT
:SCOPE

*32: TEST NUMBER
*33: ADDRESS OF NEXT TEST
10: ITERATION COUNT
ACAA: SCOPE ENTRY POINT
x=x+1

:TEST THAT RESET CLEARS ALL RXCSR BITS EXCEPT DATA TERMINAL READY, RING

CLEAR TO SEND, CARRIER DET
ACAA: MOV #PR17 P5W
BIC #BIT1, ARXCSR
MOV #1, ARXCSR
RXCSR
CLR ATXBUF
TMEKX
ERROR
MOV #1, ATXBUF
TMEKX
ERROR
RESET
MOV ARXCSR,R SRT
CMP #30002, RXCSR
BEQ ACAA
MOV #30002 TEMP
ISR %5,0ACNV
GO TO OCTAL TO ASCII CONVERT.
TEMP
ARXSB
DESTINATION ADDR.
6
ISR %5,0ACNV
GO TO OCTAL TO ASCII CONVERT.
RXCSR
ARXHAS
DESTINATION ADDR.
5
ERROR1
RXCSR
RESET FAILED TO CLEAR ALL BITS EXCEPT BIT 0. SEE ERROR PRINTOUT.
ACAA: BIC #BIT1, ARXCSR
SCOPE

*33: TEST NUMBER
*34: ADDRESS OF NEXT TEST
10: ITERATION COUNT
ACAA: SCOPE ENTRY POINT
x=x+1

:TEST THAT LOADING TXBUF (TRANSMITTER BUFFER) CLEARS TXCSR BIT 7. READY
ACAA: CLR ATXBUF
TMEKX
TIMEKX: TIME OUT TX DONE
ERROR| : LOAD TX BUF
: ERROR, DONE NOT SETTING
: TEST TXCSR BIT 7 (READY BIT)
: BRANCH IF BIT NOT SET
: ERROR, LOADING TXBUF FAILED TO CLEAR READY.

ERROR| BR | ADAC
: TIMEX
: ERROR
: BIT
: BIT7, TXCSR
: SBE
: .4
: ERROR
: READY DID NOT SET

RESET| SCOPE
: SCOPE

A-34| 100034
: TEST NUMBER
: ADDRESS OF NEXT TEST
: ITERATION COUNT
: SCOPE ENTRY POINT
: X=X+1

"TEST" THAT TRANSMIT SPEEDS ARE ARRANGED IN ASCENDING ORDER BY CHECKING THAT "TIME TO READY BIT (TXCSR BIT 7) DECREASES AS A HIGHER SPEED IS SELECTED."

AIAA| JSR | %7, DO THIS
: TEST IF THIS TEST SELECTED

TYPES
: MSETX
: MSETC
: MS0
: HALT
JSR | %7, AIAS
: TYPE
: TYPE
: MS1
JSR | %7, AIAS
: TYPE
: TYPE
: MS2
JSR | %7, AIAS
: TYPE
: TYPE
: MS3
JSR | %7, AIAS
: TYPE
: TYPE
: MS4
JSR | %7, AIAS
: TYPE
: TYPE
: MS5
JSR | %7, AIAS
: TYPE
: TYPE
: CTRO
MOV | AIAST, CTRO
: MOVE ELAPSED TIME TO CTRO.
MOV | AIAST, CTR1
: MOVE ELAPSED TIME TO CTR1.
MOV | AIAST, CTR2
: MOVE ELAPSED TIME TO CTR2.
MOV | AIAST, CTR3
: MOVE ELAPSED TIME TO CTR3.
MOV | AIAST, CTR4
: MOVE ELAPSED TIME TO CTR4
MOV | AIAST, CTR5
: MOVE ELAPSED TIME TO CTR5.
TYPE
MSEG
HALT
JSR %7, AIAS
MOV AIAS, CTR6
:OUTPUT CHAR AND TIME
MOV AIAS, CTR6
:MOVE ELAPSED TIME TO CTR6
MSEG
HALT
JSR %7, AIAS
MOV AIAS, CTR7
:OUTPUT CHAR AND TIME
JSR %7, CMPT
:CHECK THAT CTR0 THROUGH CTR7 CONTAIN
BR AIAR
:DESCENDING VALUES
ERROR!
:TRANSMIT SPEEDS NOT ARRANGED IN
EXIT
:ASCENDING ORDER.
AIAR:
SCOPE
:SCOPE
AIAS:
CLR AIAS
:TSET AIAS
CLR CTR6
BPL -4
CLR CTR6
BPL -4
CLR CTR6
BPL AIAS
:CLEAR ELAPSED TIME COUNTER.
RTS %7
:WAIT FOR TX READY.
BPL -4
CLR TXBSF
BPL -4
CLR TXBSF
:LOAD TXBSF.
BPL AIAS
:WAIT 75 US
INC AIAS
:INCREMENT ELAPSED TIME COUNTER.
TSTB @TXCSR
:READY SET?
BPL AIAS
:BRANCH IF READY NOT SET.
RTS %7
:EXIT.
TIME:
MOV #$15, %C
DEC %C
BNE TIME
RTS %7
AIAS:
OPEN

*****************************************************************************
A-35:
TEST NUMBER
AT36
ADDRESS OF NEXT TEST
10.
ITERATION COUNT
AIAR
SCOPE ENTRY POINT
X=X+1
*****************************************************************************
:TEST THAT OUTPUTTING A CHARACTER WITH THE MAINTENANCE BIT SET (TXCSR BIT 2)
:RESULTS IN DONE BIT SETTING (RXCSR BIT 7) NO LATER THAN 500 MSEC, AND
:THAT RESET INSTRUCTION CLEARS THE DONE BIT

AIAR:
BIS #BIT2, @TXCSR
:SET MAINTENANCE BIT
CLR @TXCSR
:LOAD TXBUF
DELAY
:WAIT 500 MSEC.
SOD:
TSTB @RXCSR
:SEE IF DONE BIT IS SET
BMI ATR
:BRANCH IF DONE BIT IS SET
ERROR
:DONE BIT FAILED TO SET
BR ATR
ALAS:
RESET
:ISSUE RESET TO CLEAR DONE BIT
SOD:
TSTB @RXCSR
:SEE IF DONE BIT IS CLEARED
BPL ATR
:BRANCH IF DONE BIT IS CLEARED
ERROR: RESET FAILED TO CLEAR DONE BIT

-------------------------------
L 359: 100036
A 357: ADDRESS OF NEXT TEST
I 356: ITERATION COUNT
A 358: : SCOPE ENTRY POINT
X=X+1

-------------------------------
L 360: TEST THAT DONE BIT (RXCSR BIT 7) IS CLEARED BY READING RXBUF.
L 361: DONE BIT SET BY OUTPUTTING CHARACTER WITH MAINTENANCE BIT SET (TXCSR BIT 2).
A 362: BIS #BIT2, @TXCSR
A 363: SET MAINTENANCE BIT (TXCSR BIT 2)
A 364: CLR @TXBUF
A 365: LOAD TXBUF
A 366: TIMERX
A 367: WAIT FOR DONE BIT TO SET.
A 368: ERROR
A 369: TST @RXBUF
A 370: READ RXBUF TO CLEAR DONE BIT
A 371: TST @RXCSR
A 372: SEE IF DONE BIT IS CLEAR
A 373: BPL A4AC
A 374: BRANCH IF DONE BIT IS CLEAR
A 375: ERROR
A 376: RXBUF FAILED TO CLEAR DONE BIT
A 377: SCOPE
A 378: SCOPE

-------------------------------
L 379: 100037
A 380: ADDRESS OF NEXT TEST
I 381: ITERATION COUNT
A 382: SCOPE ENTRY POINT
X=X+1

-------------------------------
L 383: TEST THAT RECEIVER ACTIVE SETS CLEAR WHEN Receiver DONE SETS.
L 384: CLR WHEN Receiver DONE SETS
A 385: JSR %7, CDINIT
A 386: INIT IF C-D DEVICE
A 387: BIS #BIT2, @TXCSR
A 388: SET MAINT
A 389: CLR @TXBUF
A 390: TRANSMIT CHAR
A 391: CLR TEMP
A 392: CLEAR BUSY INDICATOR
A 393: BIT #BIT11, @RXCSR
A 394: IS RECEIVER ACTIVE SET
A 395: BEQ A4AB1
A 396: BRANCH IF CLEAR
A 397: INC TEMP
A 398: YES, REMEMBER THAT
A 399: A4AB1: TSTB @RXCSR
A 400: SEE IF DONE SET
A 401: BPL A4AC
A 402: BRANCH IF DONE
A 403: CMP TEMP, #0
A 404: DID RECEIVER ACTIVE SET
A 405: BNE A4AC
A 406: RECEIVER ACTIVE NEVER SET
A 407: BR A4AC
A 408: DID DONE CLEAR ACTIVE
A 409: BEQ A4AC
A 410: ERROR
A 411: RXBUF FAILED TO CLEAR RX DONE
A 412: SCOPE
A 413: SCOPE

-------------------------------
L 414: 100038
A 415: ADDRESS OF NEXT TEST
I 416: ITERATION COUNT
A 417: : SCOPE ENTRY POINT
X=X+1

-------------------------------
L 418: TEST THAT RECEIVE SPEEDS ARE ARRANGED IN ASCENDING ORDER BY CHECKING THAT TIME
L 419: ELAPSED TO DONE BIT SETTING (RXCSR BIT 7), DECREASES AS A HIGHER SPEED
; THIS IS NOT DONE IN MAINTENANCE MODE TX AND RX
; POTS MUST BE STEPPED TOGETHER
; IS SELECTED.
ARIA:  JSR   %7,DOTHIS ; CHECK IF THIS TEST TO BE DONE
     TYPE
     MSE TX
     MSE TC
     MS C
     - 1
     HALT
     JSR   %7,AGAS
     MOV   AGAST,CTR0 ; MOVE ELAPSED TIME TO CTR0
     TYPE
     MS I
     HALT
     JSR   %7,AGAS
     MOV   AGAST,CTR1 ; MOVE ELAPSED TIME TO CTR1
     TYPE
     MS 2
     HALT
     JSR   %7,AGAS
     MOV   AGAST,CTR2 ; MOVE ELAPSED TIME TO CTR2.
MS3
HALT
ISR %7, AQAS
MOV AQAST,CTR3
;OUTPUT CHARACTER AND TIME DONED BIT

TYPE
MS4
HALT
ISR %7, AQAS
MOV AQAST,CTR4

MS5
HALT
ISR %7, AQAS
MOV AQAST,CTR5
TYPE
MS6
HALT
JSR 07.AQAS
MOV AQAST,67A6
TYPE
MS7
HALT
JSR %7, AQAS
MOV AQAS, CTR7
JSR %7, CMPT
BR AQAB
ERROR1
ERXTIM
AQAB: SCOPE
AQAS: CLR AQAS
TSTB @TXCSR
BPL -4
TST @RXBUF
CLR @TXBUF
LOAD TXBUF
AQASA: JSR %7, TIME
INC AQAS
TSTB @RXCSR
BPL AQASA
DONE SET?
BRANCH IF DONE NOT SET
RTS %7
EXIT
AQAST: OPEN
ELAPSED TIME COUNTER

AT41: 100041
AT42: ADDRESS OF NEXT TEST
10: ITERATION COUNT
ARAA: SCOPE ENTRY POINT
X=X+1

: TEST CORRECT OPERATION OF DATA OVERRUN BIT (@XBUF BIT 14)
ARAA: JSR %7, ARAS
: OUTPUT CHARACTER AND WAIT 500 MSEC
JSR %7, ARAS
: OUTPUT CHARACTER AND WAIT 500 MSEC
MOV @RXBUF, RXBUFT
: SAVE RXBUF CONTENTS + CLEAR DONE
BIT @BIT14, RXBUFT
: SEE IF DATA OVERRUN BIT WAS SET
BNE .+6
: BRANCH IF BIT WAS SET
ERROR
SCOPE
TST RXBUFT
: SEE THAT ERROR BIT WAS SET (@XBUF BIT 15)
BNE .+6
: ERROR BIT FAILED TO SET WHEN OVERRUN SET
ERROR
SCOPE
BIT @BIT14, RXBUFT
: SEE THAT DATA OVERRUN WAS NOT
CLEARED WHEN RXBUF WAS READ
BNE .+6
: BRANCH IF SET
ERROR
SCOPE
READING RXBUF CLEARED DATA OVERRUN
JSR %7, ARAS
BIT @BIT15, RXBUFT
: OUTPUT CHAR + WAIT 500 MSEC
BEQ .+6
: TEST THAT ERROR CLEARED
ERROR
SCOPE
BIT @BIT14, RXBUFT
: TEST THAT OVERRUN CLEARED
BEQ .+4
ERROR
SCOPE
RSS
BIT2, @TXCSR
: SET MAINTENANCE BIT
; TEST THAT TRANSMITTER IS ABLE TO INTERRUPT, IF THE INTERRUPT IS SERVICED.
; WILL HAVE OCCURRED AT THE CORRECT VECTOR.

; TEST THAT READY DOES NOT CAUSE AN INTERRUPT WHEN THE PROCESSOR IS
; AT THE SAME PRIORITY AS THE TRANSMITTER INTERRUPT REQUEST LEVEL

; TEST THAT TRANSMITTER INTERRUPTS WHEN PROCESSOR IS AT PRIORITY ONE LEVEL
; LOWER THAN THE TRANSMITTER INTERRUPT PRIORITY.
AVAB:  
BIC #BIT6, TXCSR  ;DISABLE TX INTERRUPTS
MOV "XLVL, P5W  ;SET PROCESSOR PRIORITY TO ONE LEVEL
SUB #40, P5W  ;LOWER THAN TX PRIORITY
BIS #BIT6, TXCSR  ;ENABLE TX INTERRUPTS
NOP
ERROR  ;TX FAILED TO INTERRUPT
BR AVAC

P0SP2
BIC #BIT6, TXCSR  ;HERE IF INTERRUPT OCCURS. POP STOCK TWICE
SCOPE

IC0045  ;TEST NUMBER
AT46  ;ADDRESS OF NEXT TEST
1D0  ;ITERATION COUNT
AXAA  ;SCOPE ENTRY POINT
X=X+1

:"TEST THAT TRANSMITTER DOES NOT REinitialize AFTER THE INITIAL INTERRUPT HAS OCCURRED AND HAS BEEN SERVICED.
AXAA:  
STXV  ;SET TX INTERRUPT SERVICE TO AWAC
AVAC  
BIC #BIT6, TXCSR  ;DISABLE TX INTERRUPTS
CLR P5W  ;SET PROCESSOR PRIORITY TO 0
BIS #BIT6, TXCSR  ;ENABLE TX INTERRUPTS
NOP
ERROR  ;TRANSMITTER FAILED TO INTERRUPT
SCOPE
AVAB:  
BIC #BIT6, TXCSR  ;DISABLE TX INTERRUPTS
SCOPE
AVAC:  
MOV #AWAC, TXVTTR  ;HERE IF INTERRUPT OCCURS. CHANGE EXIT
MOV #AWAD, AX46  ;POINTER TO AWAD AND EXIT INTERRUPT?
AWAD:  
NOP
BR AWAB  ;OK IF NO INTERRUPT REOCChurs.
AWAB:  
P0SP2  ;HERE IF INTERRUPT REOCCURS
ERROR  ;TX REOCCURRED AFTER RTI
BR AWAB

IC0045  ;TEST NUMBER
AT46  ;ADDRESS OF NEXT TEST
1D0  ;ITERATION COUNT
AXAA  ;SCOPE ENTRY POINT
X=X+1

:"TEST THAT RECEIVER DONE BIT IS ABLE TO INTERRUPT. IF THE INTERRUPT IS SERVICED IT WILL HAVE OCCURRED AT THE CORRECT VECTOR.
ISR  
7,0VRAY  ;GO TO OVERLAY ROUTINE
STXV  ;SET RX INTERRUPT SERVICE TO AXAB
AXAB  
ISR  
%7, STXRD  ;SET RX DONE BIT
CLS P5W  ;SET PROCESSOR PRIORITY TO 0
BIS #BIT6, P5CSR  ;ENABLE RX INTERRUPTS
4CP  
ERROR  ;RX FAILED TO INTERRUPT
BR AVAC
AXAB: POPSP2
AXAC: BIC #BIT6, ARXCSR
      DISABLE INT EN
      SCOPE

4-47:
ATSO:
  TEST NUMBER
  ADDRESS OF NEXT TEST
  ITERATION COUNT
  * SCENE ENTRY POINT
  *

AXI:
  TEST THAT MODEM INTERRUPT BIT IS ABLE TO INTERRUPT. IF THE INTERRUPT IS
  SERVICED, IT WILL HAVE OCCURRED AT THE CORRECT VECTOR.
  ISR 7, OVERLAY: GO TO OVERLAY ROUTINE
  STRXV 7: SET RX INTERRUPT SERVICE TO AXAB

AXI:
  BIG #44, ARXCSR
  DISABLE MODEM INTERRUPTS
  CLR PSW
  SET PROCESSOR PRIORITY TO C
  SYS 44, ARXCSR
  ENABLE MODEM INTERRUPT, AQ TO SNC
  DELAY 5: M:
  ERROR BR AXIC
  ; MODEM FAILED TO INTERRUPT

AXI:
  POPSP2
AXIC: BIG #BIT5, ARXCSR
      DISABLE INT EN

4-50:
ATSO:
  TEST NUMBER
  ADDRESS OF NEXT TEST
  ITERATION COUNT
  * SCENE ENTRY POINT
  *

AXI:
  TEST THAT RECEIVER DOME BIT DOES NOT CAUSE AN INTERRUPT WHEN THE PROCESSOR
  IS AT THE SAME PRIORITY LEVEL AS THE RECEIVER INTERRUPT REQUEST LEVEL
  STRXV 7: SET RX INTERRUPT SERVICE TO AXAC

AXAC:
  JSR 7, STRXD
  SET RX DOME BIT
  BIG #BIT6, ARXCSR
  DISABLE RX INTERRUPTS
  MVI RXLV, PSW
  SET PROCESSOR PRIORITY SAME AS RECEIVER'S
  BIG #BIT6, ARXCSR
  ENABLE RX INTERRUPTS
  NOP

AXAB:
  BIG #BIT6, ARXCSR
  OK IF NO INTERRUPT. DISABLE RX INTERRUPTS
  SCOPE
  SCOPE

AXAC:
  POPSP2
  HERE IF INTERRUPT OCCURS. POP STOCK TWICE
  ERXCR
  RX INTERRUPTED WITH PROCESSOR AT SAME
  BR AXAB
  PRIORITY AS THE RECEIVER

AT51:
  100051
  TEST NUMBER
  ADDRESS OF NEXT TEST
  ITERATION COUNT
  * SCENE ENTRY POINT
  *

AXAB:
  TEST THAT RECEIVER DOME BIT CAUSES INTERRUPT WHEN PROCESSOR IS AT PRIORITY
  LEVEL LOWER THAN THE RECEIVER'S INTERRUPT REQUEST LEVEL
  STRXV 7: SET RX INTERRUPT TO AXAB
**E05**

```
ARAB : 8 STRXD
AJSR : %7 STRXD
J SVC %B7, JRXCSR
: SET RX DONE BIT
J MV %RXLVL, PSW
: SET PROCESSOR PRIORITY ONE LEVEL
J SUB %#40, PSW
: LOWER THAN RECEIVER'S PRIORITY
J BIS %B16, JRXCSR
: ENABLE RX INTERRUPTS
J NOP
J ERROR RX FAILED TO INTERRUPT WITH PROCESSOR AT
J BR AZAC PRIORITY ONE LEVEL LOWER THAN RECEIVER'S
J POPSP2 HERE IF INTERRUPT OCCURS
J BIC %B16, JRXCSR DISABLE RX INTERRUPTS
J SCOPE SCOPE

***: *************************************************************
J A52: 10052 TEST NUMBER
J AT53 ADDRESS OF NEXT TEST
J 100 ITERATION COUNT
J AABA SCOPE ENTRY POINT
J X=X+1

: TEST THAT RECEIVER DOES NOT INTERRUPT AFTER THE INITIAL INTERRUPT HAS
J OCCURRED AND DONE BIT HAS NOT BEEN CLEARED
J JSR %7, STRXD
J AABABA SET RX DONE BIT
J J MV %RXLVL, PSW SET RX INTERRUPT SERVICE TO AABC
J J BIS %B16, JRXCSR DISABLE RX INTERRUPTS
J J BIS %B16, JRXCSR ENABLE RX INTERRUPTS
J J NOP ERROR RX FAILED TO INTERRUPT
J J BIC %B16, JRXCSR DISABLE RX INTERRUPTS
J J SCOPE SCOPE
J J MOV AABE, JRXVTR HERE IF INTERRUPT OCCURS, CHANGE SERVICE TO
J J MOV %AABB, %S HERE, SET EXIT POINTER TO AABB
J J RTI EXIT INTERRUPT SERVICE
J J NOP OK IF NO INTERRUPT OCCURS
J J BR ABB
J J BR ABBP HERE IF INTERRUPT REOCCURS
J J ERROR RX REINTERRUPTED AFTER RTI

***: *************************************************************
J A53: 10053 TEST NUMBER
J AT54 ADDRESS OF NEXT TEST
J 100 ITERATION COUNT
J AABA SCOPE ENTRY POINT
J X=X+1

: TEST THAT READING RXCSR DOES NOT CLEAR DONE BIT (RXCSR BIT 7)
J ABB: JSR %7 STRXD
J ABB: SET RX DONE BIT
J J MOV %RXCSR, RXCSR
J J SAVE CONTENT OF RXCSR
J J TST %RXCSR
J J SEE IF DONE BIT IS CLEAR
J J BMI ABBB
J J BRANCH IF DONE BIT IS NOT CLEAR
J J ABBB TST %RXBUF
J J CLEAR DONE BIT IF SET
J J SCOPE SCOPE

***: *************************************************************
J A54: 10054 TEST NUMBER
J A55 ADDRESS OF NEXT TEST
```
**F05**

```
100   IONG   :ITERATION COUNT
ACBA   :SCOPE ENTRY POINT
X=X+1

:TEST THAT DONE CAN CAUSE INT WITH ERROR SET
SINR
ACBB   :SET RX INTERRUPT SERVICE TO ACBB.
ACSR:  %7,STRXD :SET RX DONE BIT
JSR   %7,STRXD :SET RX DATA OFLOW
BIC   #BIT6,ARXCSR :DISABLE RX INTERRUPTS
CLR   PSW :SET PROCESSOR PRIORITY TO 0
BS     :BIT6,ARXCSR :ENABLE RX INTERRUPTS
POP    ERROR :RX DONE FAILED TO CAUSE INTERRUPT
BR    ACBC
ACBC:  POPSP2 :HERE IF INTERRUPT OCCURS. POP STACK TWICE
BIC   #BIT6,ARXCSR
SCOPE

ATS5:    100DSS :TEST NUMBER
A'56:    ADDR OF NEXT TEST
3.      IONG
ADDA   :SCOPE ENTRY COUNT
X=X+1

:DATA TEST USING NORMAL CONFIGURATION
JSR   %7,CINIT :INIT IF C-D DEVICE
ADDA:   JSR   5,DATTS :SCOPE

A'56:    56 :TEST NUMBER
ATS7:    ADDR OF NEXT TEST
3.      IONG
APBA   :SCOPE ENTRY COUNT
X=X+1

:DATA TEST USING JUMPER CONNECTOR.
:USES SPECIAL BINARY COUNT PATTERN FOR DATA. NO INTERRUPT.
JSR   7,INBIN :INITIALIZE BINARY COUNT PATTERN
APBA:   MOV   #1000,CTRO :SET CHARACTER COUNT TO 1000
APBB:   TIMETX :TIME OUT TX DONE
ERROR   ERROR DONE NOT SETTING
JSR   7,GBTINP :GET BINARY CHARACTER
MOV     %1,CRBUF :SAVE CHAR IN CRBUF AND
JSR   7,MASKIT :MASK OFF NON TRANSMITTED BITS
MOV     %1,ATXBUF :LOAD CHAR.
TIMETX :TIME OUT RX DONE
ERROR   ERROR DONE NOT SETTING
MOV     ARXBUF,CRBUF :LOAD RECEIVED DATA INTO CPBF
DATCHK  :CHECK DATA
DEC    CTRO :TESTED 1000 CHARACTERS
BNE    APBB :BRANCH IF NOT
SCOPE   YES, SCOPE

A'57:    57 :TEST NUMBER
A'60:    ADDR OF NEXT TEST
```
; TEST THAT RDR BUSY TURNS OFF RDR ENABLE
; WHEN RUN ON AN XOR TESTER
EXA:  RESET  ;RESET
INC  @RXCSR  ;SET RDR ENABLE, SEE IF RDE IS TURNED OFF BY RDR BUSY
MOV  B-10,3$+2  ;WAIT LOOP FOR XOR TESTER
BNE  3$  ;SHIP OUT CHAR.
MOV  B-50000,3$+2
TSTB  @RXCSR  ;TEST COMPLETE
BNE  5$  ;ALLOW TIME FOR RDR DONE TO SET
INC  B-10  ;FAILURE OF RDR DONE TO SET
6$:  ;FAILURE OF RDR DONE TO SET

A$:  ;SCOPE

A$:  ;SCOPE

A$:  ;ADDRESS OF NEXT TEST
10.  ;ITERATION COUNT
EXA  ;SCOPE ENTRY POINT
X=X+1  *

; TEST THAT WHEN RDR ENABLE IS SET THAT THE RXCSR DONE
; B$ IS CLEARED
EXA:  RESET  ;TEST NUMBER
PC, STRX  ;SET RXVR DONE
INC  @RXCSR  ;SET ENABLE
TSTB  @RXCSR  ;DONE SHOULDW CLEAR
BPL  3$  ;DONE NOT CLEAR
BNE  5$  ;ALLOW TIME FOR RDR DONE TO SET
INC  B-10
2$:  ;WAIT IOMIC. SEC. FOR XOR
BNE  3$  ;FAILURE OF RDR DONE TO SET
A$:  ;SCOPE

A$:  ;ADDRESS OF NEXT TEST
3.  ;ITERATION COUNT
EXA  ;SCOPE ENTRY POINT
X=X+1  *

EXA:  ;TEST NUMBER
TST  XORFLG  ;CHECKING JUMPER CONNECTIONS FOR XOR, RCVR
BPL  3$  ;ADDRESS OF NEXT TEST
MOV  B-1, @RXCSR  ;ITERATION COUNT
TST  @RXCSR
P  ;SCOPE

A$:  ;ADDRESS OF NEXT TEST
3.  ;ITERATION COUNT
EXA  ;SCOPE ENTRY POINT
X=X+1  *
EXBA: ;SCOPE ENTRY POINT
X=X+1;

;SAME AS ABOVE BUT FOR XMTR
TST XORFLG
BPL #5
MOV #17767, ATXCSR
TST ATXCSR

RESET

;SCOPE

;TEST NUMBER
ATLAST :ADDRESS OF NEXT TEST
10 :ITERATION COUNT
AQBA :SCOPE ENTRY POINT
X=X+1;

;TEST THAT WHEN TXCSR BIT 0 IS SET THAT THE OUTPUT DATA LINE
;IS PULLED TO A SPACE
AQBA: JSR #7, CDINIT ;INIT IF C-D DEVICE
BIS #BIT2, ATXCSR ;SET MAINTENANCE BIT IN TXCSR
BIS #BIT0, ATXCSR ;SET BREAK BIT
MOV #252, ATXBUF ;LOAD BUFFER
TIMEX
ERROR :TIME OUT RX DONE
CMPB @RXBUF, #0 ;CHARACTER RECEIVED SHOULD BE C
BEQ +4 ;CHARACTER OTHER THAN C
ERROR
RESET :ISSUE RESET
SCOPE
:SUBROUTINE TO SET RXCSR DONE BIT.
:RXCSR:
BIS BIPX RXCSR
CLR TXBFR
:LOAD TXBUF
:TIME OUT TX DONE ERROR DONE NOT SETTING
:EXIT.

:SUBROUTINE TO CHECK THAT CTR0 THROUGH CTR3 CONTAIN DESCENDING VALUES.
:B:
CMP CTR0,CTR1
BLO CMPCTR1
CMP CTR1,CTR2
BLO CMPCTR2
CMP CTR2,CTR3
BLO CMPCTR3
CMP CTR3,CTR4
BLO CMPCTR4
CMP CTR4,CTR5
BLO CMPCTR5
CMP CTR5,CTR6
BLO CMPCTR6
CMP CTR6,CTR7
BHI CMPCTR7
CMPCTR7: RTS
************
:PRG1 - TRANSMITTER SCOPE LOOP
************
:PRG1: TYPE
      PITIT
      JSR 5,LINSLX
      TYPE
      SELCAD
      HALT
;TYPE PROGRAM TITLE.
;GET LINE # FROM USER.
;SELECT CHAR AND DELAY.
;WAIT FOR USER.
:PRG1A:
      MOV A,SRPTR,PRG1B
      MOV A,SRPTRH,ATXBUF
;DELAY COUNT TO PRG1B.
;LOAD TXBUF.
;DELAY # OF MSEC'S. SET AT SR.
:PRG1B:
      OPEN
      BR PRG1A
;REPEAT.

************
:PRG2 - RECEIVER SCOPE LOOP.
************
:PRG2: TYPE
      PITIT
      JSR 5,LINSLX
      TYPE
      SELCAD
      HALT
;TYPE PROGRAM TITLE.
;GET LINE # FROM USER.
;SELECT CHAR AND DELAY.
;WAIT FOR USER.
:PRG2A:
      BIS #BIT2,ATXCSR
;SET MAINTENANCE BIT.
      MOV A,SRPTR,PRG2B
;DELAY COUNT TO PRG2B.
      MOV A,SRPTRH,ATXBUF
;LOAD TXBUF.
;DELAY # OF MSEC'S. SET IN SR.
:PRG2B:
      OPEN
      MOV A,RXBUF,%D
;RXBUF CONTENTS TO RD.
      RESET
;DISPLAY CONTENTS OF RXBUF IN RC.
      RESET
;BY ISSUING 5 RESET INSTRUCTIONS
      RESET
      RESET
      BR PRG2A
PRG3 - SINGLE CHARACTER MAINTENANCE MODE DATA TEST.

PRG3: TYPE

PRG3A: MOVB A5RPTRX, CRBUFA

PRG4: TYPE

PRG4A: JSR %7, INBIN

SUBROUTINE TO OUTPUT, RECEIVE, AND CHECK DATA WITH MAINTENANCE BIT SET.

MOUTIN: BIT #BIT7, A5RPTR

BNE +4

ERROR

ERROR DONE NOT SETTING

ERRORD

MOV CRBUFA, @TXBUF

JSR 7, MASKIT

TIMEX

ERROR

ERROR DONE NOT SETTING

MOV ARXBUF, CRBUF

DATCHK %7

RTS %7

EXIT.
; ASCII MESSAGES

; TITLE: .ASCII "DL11-E,C/D OFF LINE TEST - MAINDEC-11-DZDLA-D:"

.ASCII "MAP OF DEVICES PRESENT:"

MDEVAD: .ASCII " ";

MNONE: .ASCII "NONE FOUND:"

EMD: .ASCII "XT"

ATNUMB: .ASCII " PC="

APC: .ASCII " RXCSR="

MRXNUM: .ASCII " 0"

PO"IT: .ASCII "PRGO - INPUT-OUTPUT LOGIC TESTS:"

.ASCII "DISCONNECT DL11-E FROM MODEM"

.ASCII " AND CONNECT JUMPER TO CABLE:"

ATXCSR: .ASCII "TXCSR S/B:"

ATXSB: .ASCII " WAS:"

ATXWAS: .ASCII " 0"

ARXCSR: .ASCII "RXCSR S/B:"
ARXSB: .ASCII 'WAS: '
ARXWAS: .ASCII '0'
ERXTIM: .ASCII 'TX SPEEDS NOT IN ASCENDING ORDER. 0'
ERXTIM: .ASCII 'RX SPEEDS NOT IN ASCENDING ORDER. 0'
PITIT: .ASCII '%%PG1 - TRANSMITTER SCOPE LOOP2'
P2TIT: .ASCII '%%PG2 - RECEIVER SCOPE LOOP2'
SELCAD: .ASCII '%SET TEST CHAR CODE IN SR15-SRB, SET DELAY TIME IN SR7-SRA. 0'
ERDA*: .ASCII 'DATA S/B: '
AASB: .ASCII 'WAS: '
A~WAS: .ASCII '
ARXBUF: .ASCII 'RXBUF: '
ARXBUF: .ASCII '0'
ASETSR: .ASCII '%SET DESIRED SR OPTIONS. NORMAL OPERATION '
.ASCII 'IS WITH SR = 00000000'
AINCRT: .ASCII "INCORRECT ROUTINE SELECTED, PLACE CORRECT PROGRAM" 

.INI SR 0-2 AND PRESS CONTINUE." 

AINCPG: .ASCII "INVALID PROGRAM SELECTED." 

APGEND: .BYTE 207 
.ASCII "END PASS = " 

APCNT: .ASCII " LINE = " 

ACLIN: .ASCII " RXCSR = " 

APRXC: .ASCII " VECTOR = " 

APVEC: .ASCII " 

P3TIT: .ASCII "%PRG3-SINGLE CHAR MAINT MODE DATA TEST." 

P4TIT: .ASCII "%PRG4-SPEC BIN COUNT MAINT MODE DATA TEST." 

SELCAR: .ASCII "SET TEST CHAR CODE IN SR15-SR8."
LINE: .ASCII "LOAD LINE NO INTO SR 0-23"

LINE: .ASCII "LINE NO."
SELINE: .ASCII "WAS SELECTED"

MSETRX: .ASCII "RECEIVER SPEED CHECK"
MSETTX: .ASCII "TRANSMIT SPEED CHECK"

MSETO: .ASCII "SET CLOCK SWITCHES TO POSITION, THEN PRESS CONTINUE"

MTERM: .ASCII "ERROR - UNEXPECTED TRAP"

MTRAPPED: .ASCII "TRAPPED TO"
MT0: .ASCII ""
MTRAPPED FROM PC: .ASCII "TRAPPED FROM PC"

MFROM: .ASCII "A"
MNOLEM: .ASCII "NO DEVICE PRESENT - THIS LINE NO. B"

INTER: .ASCII "NO INTERRUPT"
MCS: .ASCII "CS = 03"
MCS: .ASCII "CS = 10"
"MS2: .ASCII "%CS = 20"
"MS3: .ASCII "%CS = 30"
"MS4: .ASCII "%CS = 40"
"MS5: .ASCII "%CS = 50"
"MS6: .ASCII "%CS = 60"
"MS7: .ASCII "%CS = 70"
"MPURF: .ASCII "%RECOVERED FROM POWER FAILURE2"

.EVEN
.174000

ERRST: .WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.WORD 0
.END
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2038</td>
</tr>
<tr>
<td>B</td>
<td>2042</td>
</tr>
<tr>
<td>C</td>
<td>2046</td>
</tr>
<tr>
<td>D</td>
<td>2050</td>
</tr>
<tr>
<td>E</td>
<td>2054</td>
</tr>
<tr>
<td>F</td>
<td>2058</td>
</tr>
<tr>
<td>G</td>
<td>2062</td>
</tr>
<tr>
<td>H</td>
<td>2066</td>
</tr>
<tr>
<td>I</td>
<td>2070</td>
</tr>
<tr>
<td>J</td>
<td>2074</td>
</tr>
<tr>
<td>K</td>
<td>2078</td>
</tr>
<tr>
<td>L</td>
<td>2082</td>
</tr>
<tr>
<td>M</td>
<td>2086</td>
</tr>
<tr>
<td>N</td>
<td>2090</td>
</tr>
<tr>
<td>O</td>
<td>2094</td>
</tr>
<tr>
<td>P</td>
<td>2098</td>
</tr>
<tr>
<td>Q</td>
<td>2102</td>
</tr>
<tr>
<td>R</td>
<td>2106</td>
</tr>
<tr>
<td>S</td>
<td>2110</td>
</tr>
<tr>
<td>T</td>
<td>2114</td>
</tr>
<tr>
<td>U</td>
<td>2118</td>
</tr>
<tr>
<td>V</td>
<td>2122</td>
</tr>
<tr>
<td>W</td>
<td>2126</td>
</tr>
<tr>
<td>X</td>
<td>2130</td>
</tr>
<tr>
<td>Y</td>
<td>2134</td>
</tr>
<tr>
<td>Z</td>
<td>2138</td>
</tr>
<tr>
<td>USER SYMBOLS</td>
<td>VALUE</td>
</tr>
<tr>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>0:106</td>
<td>0:120</td>
</tr>
<tr>
<td>0:144</td>
<td>0:148</td>
</tr>
<tr>
<td>0:163</td>
<td>0:167</td>
</tr>
<tr>
<td>0:186</td>
<td>0:190</td>
</tr>
<tr>
<td>0:191</td>
<td>0:195</td>
</tr>
<tr>
<td>0:315</td>
<td>0:319</td>
</tr>
<tr>
<td>0:344</td>
<td>0:348</td>
</tr>
<tr>
<td>0:363</td>
<td>0:367</td>
</tr>
<tr>
<td>0:392</td>
<td>0:396</td>
</tr>
<tr>
<td>0:515</td>
<td>0:519</td>
</tr>
<tr>
<td>0:544</td>
<td>0:548</td>
</tr>
<tr>
<td>0:563</td>
<td>0:567</td>
</tr>
<tr>
<td>0:592</td>
<td>0:596</td>
</tr>
<tr>
<td>0:221</td>
<td>0:225</td>
</tr>
<tr>
<td>0:241</td>
<td>0:245</td>
</tr>
<tr>
<td>0:261</td>
<td>0:265</td>
</tr>
<tr>
<td>0:301</td>
<td>0:305</td>
</tr>
<tr>
<td>0:306</td>
<td>0:309</td>
</tr>
<tr>
<td>0:316</td>
<td>0:319</td>
</tr>
<tr>
<td>0:326</td>
<td>0:329</td>
</tr>
<tr>
<td>0:331</td>
<td>0:334</td>
</tr>
<tr>
<td>0:336</td>
<td>0:339</td>
</tr>
<tr>
<td>0:341</td>
<td>0:344</td>
</tr>
<tr>
<td>0:354</td>
<td>0:357</td>
</tr>
<tr>
<td>0:371</td>
<td>0:374</td>
</tr>
<tr>
<td>0:386</td>
<td>0:389</td>
</tr>
<tr>
<td>0:407</td>
<td>0:411</td>
</tr>
<tr>
<td>0:416</td>
<td>0:419</td>
</tr>
<tr>
<td>0:424</td>
<td>0:428</td>
</tr>
<tr>
<td>0:444</td>
<td>0:448</td>
</tr>
<tr>
<td>0:464</td>
<td>0:468</td>
</tr>
<tr>
<td>0:484</td>
<td>0:488</td>
</tr>
<tr>
<td>0:504</td>
<td>0:508</td>
</tr>
<tr>
<td>0:524</td>
<td>0:528</td>
</tr>
<tr>
<td>0:544</td>
<td>0:548</td>
</tr>
<tr>
<td>0:564</td>
<td>0:568</td>
</tr>
<tr>
<td>0:584</td>
<td>0:588</td>
</tr>
<tr>
<td>0:604</td>
<td>0:608</td>
</tr>
<tr>
<td>0:624</td>
<td>0:628</td>
</tr>
<tr>
<td>0:644</td>
<td>0:648</td>
</tr>
<tr>
<td>0:664</td>
<td>0:668</td>
</tr>
<tr>
<td>0:684</td>
<td>0:688</td>
</tr>
<tr>
<td>0:704</td>
<td>0:708</td>
</tr>
<tr>
<td>0:724</td>
<td>0:728</td>
</tr>
<tr>
<td>0:744</td>
<td>0:748</td>
</tr>
<tr>
<td>0:764</td>
<td>0:768</td>
</tr>
<tr>
<td>0:784</td>
<td>0:788</td>
</tr>
<tr>
<td>0:804</td>
<td>0:808</td>
</tr>
<tr>
<td>0:824</td>
<td>0:828</td>
</tr>
<tr>
<td>0:844</td>
<td>0:848</td>
</tr>
<tr>
<td>0:864</td>
<td>0:868</td>
</tr>
<tr>
<td>0:884</td>
<td>0:888</td>
</tr>
<tr>
<td>0:904</td>
<td>0:908</td>
</tr>
<tr>
<td>0:924</td>
<td>0:928</td>
</tr>
<tr>
<td>0:944</td>
<td>0:948</td>
</tr>
<tr>
<td>0:964</td>
<td>0:968</td>
</tr>
<tr>
<td>0:984</td>
<td>0:988</td>
</tr>
<tr>
<td>Symbol</td>
<td>Value</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>220</td>
<td>123</td>
</tr>
<tr>
<td>2526</td>
<td>2529</td>
</tr>
<tr>
<td>2489</td>
<td>2494</td>
</tr>
<tr>
<td>2499</td>
<td>2504</td>
</tr>
<tr>
<td>2509</td>
<td>2514</td>
</tr>
<tr>
<td>2521</td>
<td>2527</td>
</tr>
<tr>
<td>2531</td>
<td></td>
</tr>
<tr>
<td>1927</td>
<td>1929</td>
</tr>
<tr>
<td>2256</td>
<td>2565</td>
</tr>
<tr>
<td>1978</td>
<td>1980</td>
</tr>
<tr>
<td>1944</td>
<td>1949</td>
</tr>
<tr>
<td>1952</td>
<td>1954</td>
</tr>
<tr>
<td>1972</td>
<td>1975</td>
</tr>
<tr>
<td>1969</td>
<td>1974</td>
</tr>
<tr>
<td>1977</td>
<td>1979</td>
</tr>
<tr>
<td>1988</td>
<td>1998</td>
</tr>
<tr>
<td>1961</td>
<td>1970</td>
</tr>
<tr>
<td>2074</td>
<td>2082</td>
</tr>
<tr>
<td>3055</td>
<td>3063</td>
</tr>
<tr>
<td>1581</td>
<td>3052</td>
</tr>
<tr>
<td>2251</td>
<td>3076</td>
</tr>
<tr>
<td>2237</td>
<td>2108</td>
</tr>
<tr>
<td>1866</td>
<td>1868</td>
</tr>
<tr>
<td>1886</td>
<td>1909</td>
</tr>
<tr>
<td>1910</td>
<td>1933</td>
</tr>
<tr>
<td>1924</td>
<td>1958</td>
</tr>
<tr>
<td>1939</td>
<td>1963</td>
</tr>
<tr>
<td>1952</td>
<td>1964</td>
</tr>
<tr>
<td>1957</td>
<td>1968</td>
</tr>
<tr>
<td>1793</td>
<td>1795</td>
</tr>
<tr>
<td>1807</td>
<td>1809</td>
</tr>
<tr>
<td>2035</td>
<td>2037</td>
</tr>
<tr>
<td>2047</td>
<td>2049</td>
</tr>
<tr>
<td>2074</td>
<td>2076</td>
</tr>
<tr>
<td>2108</td>
<td>2110</td>
</tr>
<tr>
<td>2135</td>
<td>2137</td>
</tr>
<tr>
<td>2154</td>
<td>2156</td>
</tr>
<tr>
<td>2170</td>
<td>2172</td>
</tr>
<tr>
<td>2207</td>
<td>2209</td>
</tr>
<tr>
<td>Symbol</td>
<td>Usage 1</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
</tr>
<tr>
<td>B1C</td>
<td>000001</td>
</tr>
<tr>
<td>B1C1</td>
<td>000002</td>
</tr>
<tr>
<td>B1C2</td>
<td>000003</td>
</tr>
<tr>
<td>B1C3</td>
<td>000004</td>
</tr>
<tr>
<td>B1C4</td>
<td>000005</td>
</tr>
<tr>
<td>B1C5</td>
<td>000006</td>
</tr>
<tr>
<td>B1C6</td>
<td>000007</td>
</tr>
<tr>
<td>B1C7</td>
<td>000008</td>
</tr>
<tr>
<td>B1C8</td>
<td>000009</td>
</tr>
<tr>
<td>B1C9</td>
<td>000010</td>
</tr>
<tr>
<td>B1C10</td>
<td>000011</td>
</tr>
<tr>
<td>B1C11</td>
<td>000012</td>
</tr>
<tr>
<td>B1C12</td>
<td>000013</td>
</tr>
<tr>
<td>B1C13</td>
<td>000014</td>
</tr>
<tr>
<td>B1C14</td>
<td>000015</td>
</tr>
<tr>
<td>B1C15</td>
<td>000016</td>
</tr>
<tr>
<td>Symbol</td>
<td>Offset</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>MAIN</td>
<td>001342</td>
</tr>
<tr>
<td>MAD</td>
<td>001343</td>
</tr>
<tr>
<td>MAR</td>
<td>001344</td>
</tr>
<tr>
<td>MAR2</td>
<td>001345</td>
</tr>
<tr>
<td>MAR3</td>
<td>001346</td>
</tr>
<tr>
<td>MAR4</td>
<td>001347</td>
</tr>
<tr>
<td>MAR5</td>
<td>001348</td>
</tr>
<tr>
<td>MAR6</td>
<td>002360</td>
</tr>
<tr>
<td>MAR7</td>
<td>002362</td>
</tr>
<tr>
<td>MAR8</td>
<td>002364</td>
</tr>
<tr>
<td>MAR9</td>
<td>002366</td>
</tr>
<tr>
<td>MAR10</td>
<td>002368</td>
</tr>
<tr>
<td>MAR11</td>
<td>002370</td>
</tr>
<tr>
<td>MAR12</td>
<td>002372</td>
</tr>
<tr>
<td>MAR13</td>
<td>002374</td>
</tr>
<tr>
<td>MAR14</td>
<td>002376</td>
</tr>
<tr>
<td>MAR15</td>
<td>002378</td>
</tr>
<tr>
<td>MAR16</td>
<td>002380</td>
</tr>
<tr>
<td>MAR17</td>
<td>002382</td>
</tr>
<tr>
<td>MAR18</td>
<td>002384</td>
</tr>
<tr>
<td>MAR19</td>
<td>002386</td>
</tr>
<tr>
<td>MAR20</td>
<td>002388</td>
</tr>
<tr>
<td>MAR21</td>
<td>002390</td>
</tr>
<tr>
<td>MAR22</td>
<td>002392</td>
</tr>
<tr>
<td>MAR23</td>
<td>002394</td>
</tr>
<tr>
<td>MAR24</td>
<td>002396</td>
</tr>
<tr>
<td>MAR25</td>
<td>002398</td>
</tr>
<tr>
<td>MAR26</td>
<td>002400</td>
</tr>
<tr>
<td>MAR27</td>
<td>002402</td>
</tr>
<tr>
<td>MAR28</td>
<td>002404</td>
</tr>
<tr>
<td>MAR29</td>
<td>002406</td>
</tr>
<tr>
<td>MAR30</td>
<td>002408</td>
</tr>
<tr>
<td>MAR31</td>
<td>002410</td>
</tr>
<tr>
<td>MAR32</td>
<td>002412</td>
</tr>
<tr>
<td>MAR33</td>
<td>002414</td>
</tr>
<tr>
<td>MAR34</td>
<td>002416</td>
</tr>
<tr>
<td>MAR35</td>
<td>002418</td>
</tr>
<tr>
<td>MAR36</td>
<td>002420</td>
</tr>
<tr>
<td>MAR37</td>
<td>002422</td>
</tr>
<tr>
<td>MAR38</td>
<td>002424</td>
</tr>
<tr>
<td>MAR39</td>
<td>002426</td>
</tr>
<tr>
<td>MAR40</td>
<td>002428</td>
</tr>
<tr>
<td>MAR41</td>
<td>002430</td>
</tr>
<tr>
<td>MAR42</td>
<td>002432</td>
</tr>
<tr>
<td>MAR43</td>
<td>002434</td>
</tr>
<tr>
<td>MAR44</td>
<td>002436</td>
</tr>
<tr>
<td>MAR45</td>
<td>002438</td>
</tr>
<tr>
<td>MAR46</td>
<td>002440</td>
</tr>
<tr>
<td>MAR47</td>
<td>002442</td>
</tr>
<tr>
<td>MAR48</td>
<td>002444</td>
</tr>
<tr>
<td>MAR49</td>
<td>002446</td>
</tr>
<tr>
<td>MAR50</td>
<td>002448</td>
</tr>
<tr>
<td>MAR51</td>
<td>002450</td>
</tr>
<tr>
<td>MAR52</td>
<td>002452</td>
</tr>
<tr>
<td>MAR53</td>
<td>002454</td>
</tr>
<tr>
<td>MAR54</td>
<td>002456</td>
</tr>
<tr>
<td>MAR55</td>
<td>002458</td>
</tr>
<tr>
<td>MAR56</td>
<td>002460</td>
</tr>
<tr>
<td>MAR57</td>
<td>002462</td>
</tr>
<tr>
<td>MAR58</td>
<td>002464</td>
</tr>
<tr>
<td>MAR59</td>
<td>002466</td>
</tr>
<tr>
<td>MAR60</td>
<td>002468</td>
</tr>
<tr>
<td>MAR61</td>
<td>002470</td>
</tr>
<tr>
<td>MAR62</td>
<td>002472</td>
</tr>
<tr>
<td>MAR63</td>
<td>002474</td>
</tr>
<tr>
<td>MAR64</td>
<td>002476</td>
</tr>
<tr>
<td>MAR65</td>
<td>002478</td>
</tr>
<tr>
<td>MAR66</td>
<td>002480</td>
</tr>
<tr>
<td>MAR67</td>
<td>002482</td>
</tr>
<tr>
<td>MAR68</td>
<td>002484</td>
</tr>
<tr>
<td>MAR69</td>
<td>002486</td>
</tr>
<tr>
<td>MAR70</td>
<td>002488</td>
</tr>
<tr>
<td>MAR71</td>
<td>002490</td>
</tr>
<tr>
<td>MAR72</td>
<td>002492</td>
</tr>
<tr>
<td>MAR73</td>
<td>002494</td>
</tr>
<tr>
<td>MAR74</td>
<td>002496</td>
</tr>
<tr>
<td>MAR75</td>
<td>002498</td>
</tr>
<tr>
<td>MAR76</td>
<td>002500</td>
</tr>
<tr>
<td>MAR77</td>
<td>002502</td>
</tr>
<tr>
<td>MAR78</td>
<td>002504</td>
</tr>
<tr>
<td>MAR79</td>
<td>002506</td>
</tr>
<tr>
<td>MAR80</td>
<td>002508</td>
</tr>
<tr>
<td>MAR81</td>
<td>002510</td>
</tr>
<tr>
<td>MAR82</td>
<td>002512</td>
</tr>
<tr>
<td>MAR83</td>
<td>002514</td>
</tr>
<tr>
<td>MAR84</td>
<td>002516</td>
</tr>
<tr>
<td>MAR85</td>
<td>002518</td>
</tr>
<tr>
<td>MAR86</td>
<td>002520</td>
</tr>
<tr>
<td>MAR87</td>
<td>002522</td>
</tr>
<tr>
<td>MAR88</td>
<td>002524</td>
</tr>
<tr>
<td>MAR89</td>
<td>002526</td>
</tr>
<tr>
<td>MAR90</td>
<td>002528</td>
</tr>
</tbody>
</table>

Symbol Cross Reference Table -- User Symbols

- **I06**
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMTRA</td>
<td>003000</td>
</tr>
<tr>
<td>EMINT</td>
<td>002766</td>
</tr>
<tr>
<td>EMHDR</td>
<td>001614</td>
</tr>
<tr>
<td>ERO</td>
<td>015003</td>
</tr>
<tr>
<td>ERO2</td>
<td>000243</td>
</tr>
<tr>
<td>ERR</td>
<td>054950</td>
</tr>
<tr>
<td>ERR2</td>
<td>005590</td>
</tr>
<tr>
<td>ERR3</td>
<td>005669</td>
</tr>
<tr>
<td>ERR4</td>
<td>005665</td>
</tr>
<tr>
<td>ERRF</td>
<td>005663</td>
</tr>
<tr>
<td>ERROR1</td>
<td>104015</td>
</tr>
<tr>
<td>ERR2</td>
<td>017400</td>
</tr>
<tr>
<td>ERR3</td>
<td>005472</td>
</tr>
<tr>
<td>ERR4</td>
<td>005662</td>
</tr>
<tr>
<td>ERR5</td>
<td>005732</td>
</tr>
<tr>
<td>ERR6</td>
<td>015345</td>
</tr>
<tr>
<td>ERR7</td>
<td>055926</td>
</tr>
<tr>
<td>EXA</td>
<td>014024</td>
</tr>
<tr>
<td>EXB</td>
<td>014704</td>
</tr>
<tr>
<td>EXC</td>
<td>014130</td>
</tr>
<tr>
<td>EXT</td>
<td>013740</td>
</tr>
<tr>
<td>FMAP1</td>
<td>001427</td>
</tr>
<tr>
<td>FMAP2</td>
<td>001649</td>
</tr>
<tr>
<td>FORM1</td>
<td>006046</td>
</tr>
<tr>
<td>FORM2</td>
<td>002734</td>
</tr>
<tr>
<td>FOUN</td>
<td>001612</td>
</tr>
<tr>
<td>FROM</td>
<td>001630</td>
</tr>
<tr>
<td>FTR</td>
<td>001643</td>
</tr>
<tr>
<td>GETROD</td>
<td>002403</td>
</tr>
<tr>
<td>GBACK</td>
<td>006206</td>
</tr>
<tr>
<td>GENC</td>
<td>003646</td>
</tr>
<tr>
<td>GIB</td>
<td>002064</td>
</tr>
<tr>
<td>GIBP</td>
<td>002539</td>
</tr>
<tr>
<td>GIRDA1</td>
<td>002472</td>
</tr>
<tr>
<td>GIRDY1</td>
<td>002442</td>
</tr>
<tr>
<td>GIRDY2</td>
<td>002500</td>
</tr>
<tr>
<td>GIRDY3</td>
<td>002522</td>
</tr>
<tr>
<td>GIRDY4</td>
<td>002542</td>
</tr>
<tr>
<td>ICT</td>
<td>014949</td>
</tr>
<tr>
<td>INCE</td>
<td>001454</td>
</tr>
<tr>
<td>INC2</td>
<td>001643</td>
</tr>
<tr>
<td>INCR</td>
<td>001409</td>
</tr>
<tr>
<td>INCP</td>
<td>004736</td>
</tr>
<tr>
<td>INCRT</td>
<td>004726</td>
</tr>
<tr>
<td>INTER</td>
<td>017011</td>
</tr>
<tr>
<td>MSTAR</td>
<td>002496</td>
</tr>
<tr>
<td>MNE</td>
<td>016677</td>
</tr>
<tr>
<td>MUNA</td>
<td>005432</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>VALUE</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>LINB</td>
<td>004050</td>
</tr>
<tr>
<td>LINENO</td>
<td>001614</td>
</tr>
<tr>
<td>LISCL</td>
<td>003742</td>
</tr>
<tr>
<td>LMSLX</td>
<td>003760</td>
</tr>
<tr>
<td>LOGIC</td>
<td>001530</td>
</tr>
<tr>
<td>MACH</td>
<td>000000</td>
</tr>
<tr>
<td>MANUAL</td>
<td>100000</td>
</tr>
<tr>
<td>MAPIO</td>
<td>001210</td>
</tr>
<tr>
<td>MAPD1</td>
<td>002226</td>
</tr>
<tr>
<td>MAPD2</td>
<td>002226</td>
</tr>
<tr>
<td>MAPD3</td>
<td>002132</td>
</tr>
<tr>
<td>MAPD4</td>
<td>002132</td>
</tr>
<tr>
<td>MAPD5</td>
<td>001566</td>
</tr>
<tr>
<td>MAPD6</td>
<td>002176</td>
</tr>
<tr>
<td>MAPR2</td>
<td>005770</td>
</tr>
<tr>
<td>MAPR3</td>
<td>005770</td>
</tr>
<tr>
<td>MASK II</td>
<td>003512</td>
</tr>
<tr>
<td>MEVAD</td>
<td>014746</td>
</tr>
<tr>
<td>MESS1</td>
<td>001770</td>
</tr>
<tr>
<td>MFRM</td>
<td>016235</td>
</tr>
<tr>
<td>MNOLIN</td>
<td>016746</td>
</tr>
<tr>
<td>MONE</td>
<td>014766</td>
</tr>
<tr>
<td>MOUTIN</td>
<td>014566</td>
</tr>
<tr>
<td>MPURF</td>
<td>017127</td>
</tr>
<tr>
<td>MAXMIN</td>
<td>015034</td>
</tr>
<tr>
<td>MSETC</td>
<td>016550</td>
</tr>
<tr>
<td>MSETX</td>
<td>016474</td>
</tr>
<tr>
<td>MS01</td>
<td>016582</td>
</tr>
<tr>
<td>MS02</td>
<td>017027</td>
</tr>
<tr>
<td>MS03</td>
<td>017047</td>
</tr>
<tr>
<td>MS04</td>
<td>017067</td>
</tr>
<tr>
<td>MS05</td>
<td>017077</td>
</tr>
<tr>
<td>MS06</td>
<td>017107</td>
</tr>
<tr>
<td>MS07</td>
<td>017117</td>
</tr>
<tr>
<td>MS08</td>
<td>017117</td>
</tr>
<tr>
<td>MTEX</td>
<td>016636</td>
</tr>
<tr>
<td>M08</td>
<td>014636</td>
</tr>
<tr>
<td>M09</td>
<td>016703</td>
</tr>
<tr>
<td>M0P</td>
<td>000240</td>
</tr>
<tr>
<td>M0XSI</td>
<td>004996</td>
</tr>
<tr>
<td>OACNV</td>
<td>004322</td>
</tr>
<tr>
<td>OACRNA</td>
<td>004336</td>
</tr>
<tr>
<td>OACNYX</td>
<td>004806</td>
</tr>
<tr>
<td>OLDP5</td>
<td>001470</td>
</tr>
<tr>
<td>OPEN</td>
<td>000000</td>
</tr>
<tr>
<td>OVR0</td>
<td>005616</td>
</tr>
<tr>
<td>OVR0A</td>
<td>003632</td>
</tr>
<tr>
<td>OVR0B</td>
<td>003632</td>
</tr>
<tr>
<td>OVR0B</td>
<td>003632</td>
</tr>
<tr>
<td>OVR0B</td>
<td>003632</td>
</tr>
<tr>
<td>PASCNT</td>
<td>001632</td>
</tr>
<tr>
<td>PC</td>
<td>000000</td>
</tr>
<tr>
<td>PFAIL</td>
<td>004210</td>
</tr>
<tr>
<td>PQ</td>
<td>004200</td>
</tr>
<tr>
<td>POPSP</td>
<td>005726</td>
</tr>
</tbody>
</table>

* Values indicate specific conditions or states within the system.
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>POPS</td>
<td>522626</td>
</tr>
<tr>
<td>PRRG</td>
<td>005156</td>
</tr>
<tr>
<td>PRRGEB</td>
<td>1495</td>
</tr>
<tr>
<td>PRRGEC</td>
<td>1494</td>
</tr>
<tr>
<td>PRRGEN</td>
<td>1063</td>
</tr>
<tr>
<td>PRRGEXT</td>
<td>1465</td>
</tr>
<tr>
<td>PRRGNUM</td>
<td>849</td>
</tr>
<tr>
<td>PRRRTAB</td>
<td>859</td>
</tr>
<tr>
<td>PRRXTL</td>
<td>1000</td>
</tr>
<tr>
<td>PRRXTL1</td>
<td>1489</td>
</tr>
<tr>
<td>PRG</td>
<td>006454</td>
</tr>
<tr>
<td>PRG0A</td>
<td>1697</td>
</tr>
<tr>
<td>PRG0B</td>
<td>1711</td>
</tr>
<tr>
<td>PRGC</td>
<td>014352</td>
</tr>
<tr>
<td>PRGC1</td>
<td>1650</td>
</tr>
<tr>
<td>PRGC1B</td>
<td>2965</td>
</tr>
<tr>
<td>PRGD</td>
<td>014412</td>
</tr>
<tr>
<td>PRGEB</td>
<td>1699</td>
</tr>
<tr>
<td>PRGEB1</td>
<td>1699</td>
</tr>
<tr>
<td>PRGEB2</td>
<td>1699</td>
</tr>
<tr>
<td>PRGEB3</td>
<td>1699</td>
</tr>
<tr>
<td>PRGEB4</td>
<td>1699</td>
</tr>
<tr>
<td>PRGY0</td>
<td>00000</td>
</tr>
<tr>
<td>PRVY1</td>
<td>00000</td>
</tr>
<tr>
<td>PRVY2</td>
<td>00000</td>
</tr>
<tr>
<td>PRVY3</td>
<td>00000</td>
</tr>
<tr>
<td>PRVY4</td>
<td>00000</td>
</tr>
<tr>
<td>PRVY5</td>
<td>00000</td>
</tr>
<tr>
<td>PRVY7</td>
<td>000340</td>
</tr>
<tr>
<td>PSE</td>
<td>177776</td>
</tr>
<tr>
<td>PTO</td>
<td>004174</td>
</tr>
<tr>
<td>PTOB</td>
<td>1326</td>
</tr>
<tr>
<td>PTT</td>
<td>004176</td>
</tr>
<tr>
<td>PTT1</td>
<td>1327</td>
</tr>
<tr>
<td>PURUP</td>
<td>006220</td>
</tr>
<tr>
<td>PGTI</td>
<td>015046</td>
</tr>
<tr>
<td>PGTIT</td>
<td>015407</td>
</tr>
<tr>
<td>PSTIT</td>
<td>015447</td>
</tr>
<tr>
<td>PSTT</td>
<td>01623</td>
</tr>
<tr>
<td>PTT</td>
<td>016273</td>
</tr>
<tr>
<td>PRMSK</td>
<td>003666</td>
</tr>
<tr>
<td>PREST</td>
<td>006370</td>
</tr>
<tr>
<td>RIND</td>
<td>004172</td>
</tr>
<tr>
<td>RPRSK</td>
<td>014104</td>
</tr>
<tr>
<td>RPRST</td>
<td>003670</td>
</tr>
<tr>
<td>RPROG</td>
<td>003286</td>
</tr>
<tr>
<td>RPT</td>
<td>003274</td>
</tr>
<tr>
<td>RPT1</td>
<td>1154</td>
</tr>
<tr>
<td>RPT2</td>
<td>003276</td>
</tr>
<tr>
<td>RSTART</td>
<td>006402</td>
</tr>
<tr>
<td>RSTDC</td>
<td>003114</td>
</tr>
<tr>
<td>RSTPS</td>
<td>003116</td>
</tr>
<tr>
<td>REG</td>
<td>104014</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
</tr>
<tr>
<td>REG</td>
<td>003060</td>
</tr>
<tr>
<td>TNNO</td>
<td>001466</td>
</tr>
<tr>
<td>RXBUF</td>
<td>001412</td>
</tr>
<tr>
<td>RXBUFT</td>
<td>001610</td>
</tr>
<tr>
<td>RXCRD</td>
<td>001200</td>
</tr>
<tr>
<td>RXCR1</td>
<td>001202</td>
</tr>
<tr>
<td>RXCR10</td>
<td>001220</td>
</tr>
<tr>
<td>RXCR12</td>
<td>001244</td>
</tr>
<tr>
<td>RXCR20</td>
<td>001240</td>
</tr>
<tr>
<td>RXCR30</td>
<td>001260</td>
</tr>
<tr>
<td>RXCR32</td>
<td>001262</td>
</tr>
<tr>
<td>RXCR33</td>
<td>001264</td>
</tr>
<tr>
<td>RXCR34</td>
<td>001270</td>
</tr>
<tr>
<td>RXCR36</td>
<td>001272</td>
</tr>
<tr>
<td>RXCR4</td>
<td>001100</td>
</tr>
<tr>
<td>RXCR5</td>
<td>001120</td>
</tr>
<tr>
<td>RXCR6</td>
<td>001214</td>
</tr>
<tr>
<td>RXCR7</td>
<td>001216</td>
</tr>
<tr>
<td>RXCSR</td>
<td>001410</td>
</tr>
<tr>
<td>RXCSRT</td>
<td>001606</td>
</tr>
<tr>
<td>RXEND</td>
<td>001300</td>
</tr>
<tr>
<td>RXVLT</td>
<td>001420</td>
</tr>
<tr>
<td>RXVTR</td>
<td>001420</td>
</tr>
<tr>
<td>SAVREG</td>
<td>004013</td>
</tr>
<tr>
<td>SAVRGC</td>
<td>003020</td>
</tr>
<tr>
<td>SCOPE</td>
<td>2226</td>
</tr>
<tr>
<td>Symbol</td>
<td>Value</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>SCOPTR</td>
<td>001466</td>
</tr>
<tr>
<td>SELCA</td>
<td>016704</td>
</tr>
<tr>
<td>SELCB</td>
<td>016736</td>
</tr>
<tr>
<td>SELC</td>
<td>1312</td>
</tr>
<tr>
<td>SETSR</td>
<td>003740</td>
</tr>
<tr>
<td>SOFTSR</td>
<td>00176</td>
</tr>
<tr>
<td>SPP</td>
<td>001174</td>
</tr>
<tr>
<td>SPBOT</td>
<td>001174</td>
</tr>
<tr>
<td>SRESET</td>
<td>104011</td>
</tr>
<tr>
<td>SRPTR</td>
<td>00174</td>
</tr>
<tr>
<td>SPPTRAH</td>
<td>00172</td>
</tr>
<tr>
<td>SRSETT</td>
<td>003110</td>
</tr>
<tr>
<td>STAL</td>
<td>003622</td>
</tr>
<tr>
<td>STALL</td>
<td>003642</td>
</tr>
<tr>
<td>STALB</td>
<td>003644</td>
</tr>
<tr>
<td>STALL = 104002</td>
<td>713</td>
</tr>
<tr>
<td>START</td>
<td>002052</td>
</tr>
<tr>
<td>STARTZ</td>
<td>001634</td>
</tr>
<tr>
<td>START1</td>
<td>002242</td>
</tr>
<tr>
<td>START2</td>
<td>002246</td>
</tr>
<tr>
<td>START3</td>
<td>003052</td>
</tr>
<tr>
<td>START4</td>
<td>002342</td>
</tr>
<tr>
<td>STLM</td>
<td>001406</td>
</tr>
<tr>
<td>SLMSP</td>
<td>003154</td>
</tr>
<tr>
<td>SLR</td>
<td>003120</td>
</tr>
<tr>
<td>STPRA</td>
<td>003176</td>
</tr>
<tr>
<td>STPXO</td>
<td>004374</td>
</tr>
<tr>
<td>STRXV = 104006</td>
<td>717</td>
</tr>
<tr>
<td>STXXV = 104007</td>
<td>718</td>
</tr>
<tr>
<td>SUBTEN</td>
<td>004530</td>
</tr>
<tr>
<td>SUBTEN</td>
<td>004534</td>
</tr>
<tr>
<td>SUBTIME</td>
<td>00150</td>
</tr>
<tr>
<td>SVRCP</td>
<td>003054</td>
</tr>
<tr>
<td>SVRPSW</td>
<td>003056</td>
</tr>
<tr>
<td>TEMP</td>
<td>001616</td>
</tr>
<tr>
<td>TEMP1</td>
<td>001620</td>
</tr>
<tr>
<td>TEMPC</td>
<td>00126</td>
</tr>
<tr>
<td>TEMPC</td>
<td>004579</td>
</tr>
<tr>
<td>TEMPER</td>
<td>004579</td>
</tr>
<tr>
<td>TIME</td>
<td>001612</td>
</tr>
<tr>
<td>TIMER</td>
<td>003736</td>
</tr>
<tr>
<td>TIMER</td>
<td>003710</td>
</tr>
<tr>
<td>TIME2</td>
<td>003714</td>
</tr>
<tr>
<td>TIME2</td>
<td>001616</td>
</tr>
<tr>
<td>TKB</td>
<td>001432</td>
</tr>
<tr>
<td>TKLB</td>
<td>001432</td>
</tr>
<tr>
<td>TKS</td>
<td>001430</td>
</tr>
<tr>
<td>583</td>
<td>569</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>595</td>
<td>597</td>
</tr>
<tr>
<td>623</td>
<td>625</td>
</tr>
<tr>
<td>651</td>
<td>653</td>
</tr>
<tr>
<td>679</td>
<td>681</td>
</tr>
<tr>
<td>707</td>
<td>709</td>
</tr>
<tr>
<td>735</td>
<td>737</td>
</tr>
<tr>
<td>763</td>
<td>765</td>
</tr>
<tr>
<td>791</td>
<td>793</td>
</tr>
</tbody>
</table>

CROSS REFERENCE TABLE -- USER SYMBOLS
<table>
<thead>
<tr>
<th>Macro</th>
<th>1719</th>
<th>1733</th>
<th>1749</th>
<th>1763</th>
<th>1777</th>
<th>1802</th>
<th>1826</th>
<th>1851</th>
<th>1865</th>
<th>1984</th>
<th>1908</th>
<th>1922</th>
<th>1957</th>
<th>1995</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>9415</td>
<td>2008</td>
<td>2021</td>
<td>2034</td>
<td>2058</td>
<td>2072</td>
<td>2105</td>
<td>2133</td>
<td>2162</td>
<td>2175</td>
<td>2199</td>
<td>2227</td>
<td>2255</td>
<td>2283</td>
</tr>
<tr>
<td>Value</td>
<td>9415</td>
<td>2008</td>
<td>2021</td>
<td>2034</td>
<td>2058</td>
<td>2072</td>
<td>2105</td>
<td>2133</td>
<td>2162</td>
<td>2175</td>
<td>2199</td>
<td>2227</td>
<td>2255</td>
<td>2283</td>
</tr>
</tbody>
</table>
ERRJRS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

#DDLAA DDLLAA SEQ/SOL/CRF/DS:ERFZ/EN:ABS=DSKM:DDDLAA.P11
RUN TIME: 15.16 5 SECONDS
RUN TIME REAL: 50.34=1.4
CORE USE: 12K 123 PAGES.