

# DL11

DL11-E, C/D OFFLINE TEST  
MD-11-DDDLA-A

EP-DDDLB-A-DL-A  
COPYRIGHT © 1976.  
FICHE 1 OF 1

NOV 1976  
**digital**  
MADE IN USA

This microfiche card contains a grid of frames. The frames on the left side of the card contain data, likely test results or program listings, arranged in a structured format. The frames on the right side of the card are mostly blank, indicating that the data is concentrated in the left half of the card. The data in the frames appears to be organized into columns and rows, possibly representing a table or a list of test parameters and their corresponding values.

6/11/76

.REM 1

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DDDLA-A-D
PRODUCT NAME:	DL11-E,C/D 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

## 1. ABSTRACT

TWO SEPARATE DIAGNOSTIC PROGRAMS ARE PROVIDED FOR THE DL11-E (ASYNCHRONOUS MODEM INTERFACE), MAINDEC-11-DDDLA-A (DL11-E OFF LINE TESTS) AND MAINDEC-11-DYDLB-A (DL11-E ON LINE TESTS). THE OFF LINE TEST TESTS ALL DL11-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 DL11-C AND DL11-D CAN ALSO BE TESTED WITH THIS OFF LINE TEST. THESE ARE BOTH TESTED IN MAINTENANCE MODE AND ONLY THOSE TESTS MARKED C,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 DL11C+D CAN BE PERFORMED BY USING THE SELECT SWITCH OPTION (SR9). TEST 56 IS A DATA TEST WHICH CAN BE USED FOR CABLE TESTING DL11-D'S. WARNING--A FAILURE IN THIS TEST MAY OCCUR DUE TO A SPLIT BAUD RATE OF RCVTR/TXVTR.

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. DL11-E OR DL11-C OR DL11-D
- C. SPECIAL JUMPER CONNECTOR H315 (SEE DL11 MAINTENANCE MANUAL FOR DETAILED DESCRIPTION) IF DL11-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.

4. 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.

## 4.1 PRGO 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.  
DISCONNECT THE DL11-E FROM THE MODEM AND INSERT THE JUMPER CONNECTOR IN THE MODEM END OF THE CABLE, AND PRESS CONTINUE.  
NOTE, IF THE CABLE IS LEFT CONNECTED TO THE MODEM 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 0-5 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 TEST. AND WILL HALT AT LOCATION 3776.

LOAD THE LINE # AS REQUESTED AND PRESS CONTINUE.  
LINE NUMBER REFERS TO THE ADDRESSES TO WHICH THE DL11-E RESPONDS.

LINE 00 77561X	LINE 10 77571X	LINE 20 77601X	LINE 30 77611X
LINE 01 77562X	LINE 11 77572X	LINE 21 77602X	LINE 31 77612X
LINE 02 77563X	LINE 12 77573X	LINE 22 77603X	LINE 32 77613X
LINE 03 77564X	LINE 13 77574X	LINE 23 77604X	LINE 33 77614X
LINE 04 77565X	LINE 14 77575X	LINE 24 77605X	LINE 34 77615X
LINE 05 77656X	LINE 15 77576X	LINE 25 77606X	LINE 35 77616X
LINE 06 77567X	LINE 16 77577X	LINE 26 77607X	LINE 36 77617X
LINE 07 77570X	LINE 17 77600X	LINE 27 77610X	

- D. THE PROGRAM WILL NOW BEGIN TESTING THE DL11-E OR C/D YOU SELECTED.  
ALL DL11'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.  
THESE 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 SR6 WAS UP PROGRAM WILL HALT AT LOCATION  
252. PRESS CONTINUE FOR ANOTHER PASS.

#### 4.2 PRG1 - TRANSMITTER SCOPE LOOP

- A. LOAD ADDRESS = 000200 (RESTART = 000204)  
LOAD SR 0-2 = 1, 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 DL11-E WILL TRANSMIT  
AND THE DELAY IS THE TIME ELAPSED BETWEEN SUCCESSIVE TRANS-  
MISSIONS OF ONE CHARACTER. LOAD CHARACTER CODE IN  
SR15-SR8; SET DELAY TIME IN SR7-SR0.  
PRESS CONTINUE WHEN THIS DONE.

- C. THE PROGRAM WILL RUN WITHOUT ERROR OR END TYPEOUTS.

#### 4.3 PRG2 - RECEIVER SCOPE LOOP

- A. LOAD ADDRESS = 000200 (RESTART = 000204)  
LOAD SR 0-2 = 2, 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 NO. 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 SUCCES-  
SIVE CHARACTERS. LOAD CHARACTER CODE IN SR15-SR8;  
SET DELAY TIME IN SR7-SR0.  
PRESS CONTINUE WHEN THIS DONE.
- C. THE PROGRAM WILL NOW RUN WITHOUT ERROR OR END TYPEOUTS.

## 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-  
URES.

## 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 PRG0 - 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 MUST

BE INSERTED BEFORE STARTING IF DL11-E.

### 5.1.1 ROUTINE DESCRIPTIONS

ROUTINE	TESTS
AT0-AT3 AT4-AT27	ADDRESSABILITY OF CSRS & DBRS DIDDLES ALL BITS IN THE CSRS AND CHECKS THAT THEY CAN BE READ/WITTEN PROPERLY.
AT31-AT32 AT33 AT34	PROPER OPERATION OF RESET INSTRUCTION PROPER OPERATION OF READY BIT PROPER OPERATION OF TRANSMIT SPEED SELECTION
AT35-AT37	PROPER OPERATION OF DONE BIT
AT40	PROPER OPERATION RECEIVER SPEED SELECT
AT41	PROPER OPERATION OF DATA OVERRUN
AT42-AT52	PROPER OPERATION OF INTERRUPTS
AT53	READING RXCSR DOES NOT CLEAR DONE
AT54	ERROR CAUSES INTERRUPT
AT55	DATA TEST MAINTENANCE MODE
AT56	DATA TEST WITH JUMPER
AT57	PROPER OPERATION OF BREAK BIT

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

T005 PC=XXXX RXCSR=XXXX

T56 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 DL11-E IS INCORRECT  
A TRAP TO THE WRONG LOCATION WILL OCCUR AND AN ERROR MESSAGE  
WILL OCCUR.

### 5.1.3 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 DL11-E. IN ADDITION TO TESTING DL11-E LOGIC  
THE JUMPER ALSO TESTS CABLE WIRING TO/FROM THE DL11-E/DATA  
SET. THE FOLLOWING TESTS WILL FAIL IF THE CABLE IS NOT  
INSTALLED IN THE DL11-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 DL11-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 DL11-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 DL11-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 PRG0 ROUTINE 54 EXCEPT THAT  
THE USER SPECIFIES DL11-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

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 8  
DDDLAA.P11

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 0 ONLY), AND THE PC OF THE POWER FAIL ERROR CALL.

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

]











4

```

:EQUATE STATEMENTS
PSW=177776
SPBOT=1176
NOP=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
POPSP=5726
POPSP2=022626
PRTY7=340
PRTY6=300
PRTY5=240
PRTY4=200
PRTY3=140
PRTY2=100
PRTY1=40
PRTY0=0
TYPE=EMT+0
TYPES=EMT+1
STALL=EMT+2
ERROR=EMT+3
DATCHK=EMT+4
CHALT=EMT+5
STRXV=EMT+6
STTXV=EMT+7
EHALT=EMT+10
SRESET=EMT+11
SCOPE=EMT+12
SAVREG=EMT+13
RSTREG=EMT+14
ERROR1=EMT+15
DELAY=EMT+16
TIMERX=EMT+17
TIMETX=EMT+20
ATLAST=-1
CD=100000

```

```

;POP THE STACK. SAME AS TST (6)+
;POP STACK TWICE. SAME AS CMP (6)+.(6)+
;PRIORITY LEVEL DEFINITIONS

```

;FLAG FOR C/D TESTS

.LIST ME

```

      . = 172
SRPTRH: 177571          ;HIGH BYTE OF SWITCH REGISTER
SRPTR:  177570
SOFTSR: 000000
      . = 200
      JMP 2#STARTZ      ;GO TO START OF PROGRAM.
      . = 204
      JMP 2#RESTART
EOPHLT: . = 250
        HALT           ;THIS IS AN END OF PASS HALT; NOT AN ERROR HALT.
                          ;THIS HAPPENS ONLY IF SW6 IS UP. PRESS CONTINUE
                          ;TO GET ANOTHER PASS.
        RTS PC
      . = 1200

```

```

: DEVICE ADDRESS LIST
: LSB BIT0 IS SET TO A 1 BY MAPPER IF DEVICE NOT FOUND
: TO TEST THAT LINE NOT FOUND CLEAR BIT0 IN THAT DEVICE ADDRESS
: IN THIS TABLE AFTER MAPPING DONE
: *****

```

Device	Address	Line	Description
RXCRO:	175610	0	DEVICE ADDRESS (RXCSR)
RXCR1:	175620	1	DEVICE ADDRESS (RXCSR)
RXCR2:	175630	2	DEVICE ADDRESS (RXCSR)
RXCR3:	175640	3	DEVICE ADDRESS (RXCSR)
RXCR4:	175650	4	DEVICE ADDRESS (RXCSR)
RXCR5:	175660	5	DEVICE ADDRESS (RXCSR)
RXCR6:	175670	6	DEVICE ADDRESS (RXCSR)
RXCR7:	175700	7	DEVICE ADDRESS (RXCSR)
RXCR10:	175710	10	DEVICE ADDRESS (RXCSR)
RXCR11:	175720	11	DEVICE ADDRESS (RXCSR)
RXCR12:	175730	12	DEVICE ADDRESS (RXCSR)
RXCR13:	175740	13	DEVICE ADDRESS (RXCSR)
RXCR14:	175750	14	DEVICE ADDRESS (RXCSR)
RXCR15:	175760	15	DEVICE ADDRESS (RXCSR)
RXCR16:	175770	16	DEVICE ADDRESS (RXCSR)
RXCR17:	176000	17	DEVICE ADDRESS (RXCSR)
RXCR20:	176010	20	DEVICE ADDRESS (RXCSR)
RXCR21:	176020	21	DEVICE ADDRESS (RXCSR)
RXCR22:	176030	22	DEVICE ADDRESS (RXCSR)
RXCR23:	176040	23	DEVICE ADDRESS (RXCSR)
RXCR24:	176050	24	DEVICE ADDRESS (RXCSR)
RXCR25:	176060	25	DEVICE ADDRESS (RXCSR)
RXCR26:	176070	26	DEVICE ADDRESS (RXCSR)
RXCR27:	176100	27	DEVICE ADDRESS (RXCSR)
RXCR30:	176110	30	DEVICE ADDRESS (RXCSR)
RXCR31:	176120	31	DEVICE ADDRESS (RXCSR)
RXCR32:	176130	32	DEVICE ADDRESS (RXCSR)
RXCR33:	176140	33	DEVICE ADDRESS (RXCSR)
RXCR34:	176150	34	DEVICE ADDRESS (RXCSR)
RXCR35:	176160	35	DEVICE ADDRESS (RXCSR)
RXCR36:	176170	36	DEVICE ADDRESS (RXCSR)
XORADD:	177777	37	SPECIAL ADDRESS FOR XOR
RXEND:	177777	XX	DEVICE ADDRESS (RXCSR)

```

: CHARACTER LENGTH, PRIORITY, C/D MASK

```

: INITIALLY SET FOR DL11-E, PRIORITY=4, CHARACTER LENGTH=8  
: BIT 15 SET TO A 1 = THAT LINE HAS DL11-C OR DL11-D  
: EX: 140377 = DL11C OR DL11D, 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

```

*****
CMAS0: 040377      1 :LINE 0 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS1: 040377      :LINE 1 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS2: 040377      :LINE 2 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS3: 040377      :LINE 3 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS4: 040377      :LINE 4 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS5: 040377      :LINE 5 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS6: 040377      :LINE 6 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS7: 040377      :LINE 7 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS10: 040377     :LINE 10 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS11: 040377     :LINE 11 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS12: 040377     :LINE 12 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS13: 040377     :LINE 13 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS14: 040377     :LINE 14 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS15: 040377     :LINE 15 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS16: 040377     :LINE 16 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS17: 040377     :LINE 17 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS20: 040377     :LINE 20 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS21: 040377     :LINE 21 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS22: 040377     :LINE 22 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS23: 040377     :LINE 23 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS24: 040377     :LINE 24 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS25: 040377     :LINE 25 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS26: 040377     :LINE 26 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS27: 040377     :LINE 27 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS30: 040377     :LINE 30 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS31: 040377     :LINE 31 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS32: 040377     :LINE 32 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS33: 040377     :LINE 33 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS34: 040377     :LINE 34 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS35: 040377     :LINE 35 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS36: 040377     :LINE 36 CHARACTER MASK, PRIORITY, C/D FLAG
CMAS37: 040377     :LINE 37 SPECIAL ADDRESS FOR XOR

```

```

UMASK: 0           : MASK FOR DEVICE UT
RMASK: 0           : MASK FOR CHAR LENGTH FOR DEVICE UT
STLMSK: 177740    : MASK FOR MAX RANDOM STALL

```

```

RXCSR: 0           : RECEIVER UNDER TEST
RXBUF: 0           : RECEIVER BUFFER UNDER TEST
TXCSR: 0           : TRANSMITTER CSR UNDER TEST
TXBUF: 0           : TRANSMITTER BUFFER UNDER TEST
RXVTR: 0           : RECEIVER VECTOR UNDER TEST
RXLVL: 0           : RECEIVER PRIORITY LEVEL UT
TXVTR: 0           : TRANSMITTER VECTOR UNDER TEST
TXLVL: 0           : TRANSMITTER PRIORITY LEVEL UT

```

```

*****
TKS: 177560       : LSR CSR
TKB: 177562       : LSR BUFFER
TPS: 177564       : LSP CSR

```

TPB:	177566	:LSP BUFFER
TKVTR:	60	:LSR INTERRUPT VECTOR
TKLVL:	PRTY4	:LSR PRIORITY LEVEL
TPVTR:	64	:LSP INTERRUPT VECTOR
TPLVL:	PRTY4	:LSP PRIORITY LEVEL
PRGNUM:	OPEN	:CONTAINS CURRENT PROGRAM#
KSTART:	OPEN	:CURRENT PROGRAM START ADDRESS.
CURTST:	OPEN	:CONTAINS ADDR OF CURRENT TEST.
RTNNO:	OPEN	:CONTAINS CURRENT TEST #.
TNNO:	0	:CONTAINES EDITED TNUM
NXTST:	OPEN	:CONTAINS ADDR OF NEXT TEST.
ICTR:	OPEN	:CONTAINS CURRENT ITERATION COUNT
SCOPTR:	OPEN	:CONTAINS CURRENT SCOPE POINTER.
OLDPS:	0	:PS SAVED FROM TRAP TO EMT ROUTINE
FMAP:	0	:MAPPING FLAG, 1= MAPPING IN PROGRESS
PRGTAB:	PRG0	:PRG0 START ADDRESS
	PRG1	:PRG1 START ADDRESS
	PRG2	:PRG2 START ADDRESS
	PRG3	:PRG3 START ADDRESS
	PRG4	:PRG4 START ADDRESS
	INCRPG	:INCORRECT PROGRAM SELECTED
	INCRPG	
	INCRPG	
EMTTAB:	TYP	:POINTER TO TYPEOUT ROUTINE
	TYPS	:POINTER TO CHAINED MESSAGES ROUTINE
	STAL	:POINTER TO RANDOM STALL ROUTINE
	ERR	:POINTER TO ERROR ROUTINE
	DTCHK	
	OPEN	
	STLSRV	
	STLSPV	
	EHLT	:POINTER TO ERROR HALT ROUTINE.
	SRSETT	
	CHAINN	
	SAVRG	
	RSTRG	
	ERR1	
	DLY	
	TMAX	
	TMTX	
CRBUF:	OPEN	
CRBUFA:	OPEN	
CRBUFB:	OPEN	
CTRO:	OPEN	
CTR1:	OPEN	
CTR2:	OPEN	
CTR3:	OPEN	
CTR4:	OPEN	
CTR5:	OPEN	
CTR6:	OPEN	
CTR7:	OPEN	
TXCSRT:	OPEN	
RXCSRT:	OPEN	
RXBUFT:	OPEN	
FOUNDV:	0	

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 18  
 DDDLAAR.F11

```

LINENO: 0
TEMP: OPEN
TEMP1: 0
FTITLE: 0
FNONE: 0
TOPC: 0
FROMPC: 0
PASCNT: 0
STARTZ: MOV    #SPBOT,%6
          MOV    6,-(SP)          ;SAVE CURRENT VECTOR
          MOV    4,-(SP)
          MOV    #15,4
          TST   @SRPTR           ;SET UP TIME OUT VECTOR
                                   ;TRY TO REFERENCE THE
                                   ;HARDWARE SWITCH REGISTER
15:      BR    25                ;BRANCH IF NO TIME OUT TRAP OCCURRS
          MOV    #SOFTSR,SRPTR   ;CHANGE THE SWITCH REGISTER POINTER
                                   ;TO POINT TO A SOFTWARE SWITCH REGISTER
25:      CMP    (6)+,(6)+
          MOV    (6)+,4
          MOV    (6)+,6
          MOV    SRPTR,SRPTRH
          INC   SRPTRH
          CLR   @FTITLE
          MOV   @4,-(%6)
          MOV   #XORA,@4
          TST  @177060
          MOV  (%6)+,@4
          MOV  #174000,@XORADD
          MOV  #-1,@XORFLG
          TYPE
          MESS1
MESS1:   JMP   @START
          .ASCII <15><12>'YOU ARE ON AN XOR TESTER@'

XORFLG: .EVEN
        .WORD 0
XORA:   CMP   (%6)+,(%6)+
        MOV   (%6)+,@4
        MOV   #-1,@XORADD
        CLR  @XORFLG
        JMP  @START

START:  MOV   #SPBOT,%6          ;SET BOTTOM OF SP STACK.
        MOV   #PFAIL,24
        CLR  FOUNOV
        CLR  FMAP
        JSR  %7,CLACD           ;CLEAR DEVICE UT PARAMETERS
        JSR  %7,OVRLAY         ;OVERLAY TRAP AREA
        TST  FTITLE            ;TITLE PRINTED AND MAP MADE
        BNE  START1           ;YES, SKIP OVER THIS
  
```

```

TYPE
MTIT
INC
CLR
MOV
MAPA: MOV
CMP
BEQ
BIC
CLR
TST
NOP
BR
MAPNE: BIS
POPSP2
BR
MAPOK: MOV
JSR
TEMP1
MDEVAD
5
TYPE
MDEVAD
INC
BR
MAPEND: MOV
TST
BEQ
START1: MOV
START2: BIT
BNE
MOV
SUB
ASR
MOV
JSR
BR
START3: TST
CMP
BNE
MAPERR: TYPE
MNONE
TST
BEQ
JMP
CLR
HALT
START4: JMP
MOV
CLR
CLR
MOV
BIC
MOV
ASL
JMP
FTITLE
FNONE
: CLEAR DEVICE PRESENT FLAG
#MAPNE, MACHER
: SET UP NO DEVICE PRESENT RETURN
#RXCRD, %4
: SET UP DEVICE POINTER
(%4), #RXEND
: LAST DEVICE
MAPEND
: YES, EXIT
#BIT0, (4)
: CLEAR ODD ADDRESS
PSW
#(4)
: TEST DEVICE
MAPOK
#BIT0, (4)+
: NOT LIVING
MAPA
(4)+, TEMP1
: SAVE DEVICE ADDRESS FOR TYPING
%5, OACNV
FNONE
: SET HAVE DEVICE
MAPA
#ERTP, MACHER
: RESET TRAPS
FNONE
: ANY DEVICES PRESENT
MAPERR
: NO, ERROR
#RXCRD, %1
#BIT0, (1)
: IS DEVICE LIVING
START3
: NO, CHECK FOR END
%1, LINENO
: CALCULATE LINE NUMBER UNDER TEST
#RXCRD, LINENO
LINENO
(1), %1
: YES, LOAD AND EXIT
%7, FORMAD
START4
(1)+
: END OF TABLE
%1, #RXEND
: NO, LOOP
START2
#42
: MONITOR LOAD
+6
: NO, CONTINUE
PRGXTL
: YES, EXIT
FTITLE
START
#1, PASCNT
PSW
RTNNO
#SRPTR, %0
: (SR) TO R0
#177770, %0
: LIMIT (SR) TO BITS 3-0
%0, PRGNUM
: SAVE PROGRAM #
%0
#PRGTAB(0)
: GO TO SELECTED PROGRAM.

```

```

GETRDY: MOV      KSTART,NXTST      ;ADDR OF 1ST ROUTINE TO NXTST
GTRDYX: MOV      #ERTP,MACHER      ;RESET MACHER TRAP.
        MOV      #40,MACHER+2
        CLR      FMAP
        MOV      #SPBOT,%6         ;SET BOTTOM OF STACK.
        SRESET                                ;ISSUE RESET.
GTRDYA: CLR      PSM
        JSR      %7,FORWD           ;ROLL FORWARD TO "NEXT" ROUTINE.
        BIT      #BIT9,JSRPTR       ;CHECK SELECT ROUTINE SWITCH
        BNE      GTRDYC           ;BRANCH IF SELECT ROUTINE SWITCH IS SET.
        TST      UMASK             ;C/D DEVICE
        BPL      GTRDA1           ;NO, CONTINUE
        TST      RTNNO            ;THIS A C/D TEST
        BPL      GTRDYA           ;NO, DO NEXT TEST
GTRDA1: JMP      @CURTST           ;GO RUN CURRENT ROUTINE.
        BR      CHNB              ;NO GO. MANUAL RTN BYPASSED.
GTRDYC: MOV      @SRPTR,%0         ;(SR) TO R0
        BIC      #177600,%0        ;MASK UNDESIRED BITS
        CMPB     RTNNO,%0         ;COMPARE RTNNO TO (R0)
        BNE      GTRDYD           ;BRANCH IF ROUTINE NOT FOUND YET.
        JMP      @CURTST           ;GO RUN ROUTINE.
GTRDYD: CMP      #-1,NXTST        ;NO. CHECK FOR LAST ROUTINE.
        BNE      GTRDYA           ;BRANCH IF NOT LAST ROUTINE.
        JSR      %7,INCRTN         ;YES. INCORRECT ROUTINE SELECTED.
        BR      GETRDY            ;START OVER.

CHAINN: BIT      #BIT14,JSRPTR     ;CHECK FOR SCOPE OPTION.
CHNAB:  BEQ      CHNA              ;BRANCH IF SCOPE SW NOT SET.
        MOV      SCOPTR,%6         ;SET UP TO RETURN TO ROUTINE.
CHNA:   RTI
        TST      @XORFLG          ;RETURN TO ROUTINE.
        BPL      IS
        MOV      @#4,-(%6)
        MOV      @XOR,%4
        TST      @#177060         ;TEST FOR XOR
        MOV      (%6)+,%4
IS:     BIT      #BIT11,JSRPTR     ;TEST INHIBIT ITERATION SWITCH
        BNE      CHNAA           ;BRANCH IF INHIBIT ITERATION SW SET.
        DEC      ICTR             ;DECREMENT ITERATION COUNT.
        BNE      CHNAB           ;BRANCH IF COUNT NOT 0.
CHNAA:  POPSP2
        BIT      #BIT10,JSRPTR     ;POP STACK TWICE
        BEQ      CHNB
        MOV      RTNNO,%0
        BIC      #BIT15,%0
CHNB:   BIT      #BIT9,JSRPTR      ;CHECK SELECT ROUTINE SWITCH
        BNE      GETRDY          ;BRANCH IF SELECT RTN SW SET
        CMP      #-1,NXTST        ;LAST TEST?
        BNE      GTRDYX          ;BRANCH IF NOT LAST TEST.
        JSR      %7,PRGEND        ;PROGRAM END.
        BR      GETRDY
XOR:    CMP      (%6)+,(%6)+
        MOV      (%6)+,%4

```

MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 21  
 DDDLAA.P11

```

      BR      CHNAB
; INIT FOR C/D - WITHOUT JUMPER RESET STARTS ASSEMBLING CHARACTER SETTING DONE
; SET MAINT DELAY, CLEAR RX DONE
COINIT: TST   UMASK      ; C-D DEVICE
        BPL   CDINX     ; NO EXIT
        BIS   #BIT2, @TXCSR ; SET MAINT BIT
        DELAY 1500.     ; WAIT 1.5 SEC
CDINX:  TST   @RXBUF    ; CLEAR RX DONE
        RTS   %7

FORWD:  MOV   NXTST, %5 ; ADDR OF NEXT ROUTINE TO R5.
        MOV   (5)+, RTNNO ; GET NEXT ROUTINE NUMBER.
        MOV   (5)+, NXTST ; GET ADDR OF NEXT "NEXT" ROUTINE.
        MOV   (5)+, ICTR  ; GET ITERATION COUNT.
        MOV   (5)+, SCOPTR ; GET SCOPE LOOP ENTRY POINTER.
        MOV   %5, CURTST ; ADDR OF NOW CURRENT TEST TO CURTST.
        RTS   %7        ; EXIT FORWD SUBROUTINE.

EMTINT: MOV   @%6, -(6) ; GET SAVED PC.
        SUB   #2, @%6   ; DECREMENT PC BY 2.
EMTA:   ASL   @%6       ; EMT ARG X 2.
        BIC   #177001, @%6 ; REMOVE 7 MSB.
        ADD   #EMTTAB, @%6 ; FORM EMT RTN ADDR.
        MOV   @%6, @%6
        JMP   @%6+     ; GO TO EMT ROUTINE.

; SAVE REGS 0 TO 4 SUBROUTINE.
SAVRG:  MOV   (6)+, SVRPC ; SAVE PC AND PSW.
        MOV   (6)+, SVRPSW
        MOV   %4, -(6)   ; SAVE REGS 0 - 4
        MOV   %3, -(6)   ; IN STACK.
        MOV   %2, -(6)
        MOV   %1, -(6)
        MOV   %0, -(6)
        MOV   SVRPSW, -(6) ; RESTORE PC AND PSW.
        MOV   SVRPC, -(6)
        RTI              ; EXIT.
SVRPC:  OPEN
SVRPSW: OPEN

; RESTORE REGS 0 TO 4 SUBROUTINE.
RSTRG:  MOV   (6)+, RSTPC ; SAVE PC AND PSW.
        MOV   (6)+, RSTPSW
        MOV   (6)+, %0   ; RESTORE REGS 0 - 4
        MOV   (6)+, %1   ; FROM STACK.
        MOV   (6)+, %2
        MOV   (6)+, %3
        MOV   (6)+, %4
        MOV   RSTPSW, -(6) ; RESTORE PC AND PSW.
        MOV   RSTPC, -(6)

```

```

RSTPC: RTI ;EXIT
RSTPSW: 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,%6 ;SET UP EXIT
STPRA: MOV RXVTR,%1
        MOV #OPEN,(1)+ ;SET VECTOR ADDRESS
        MOV RXLVL,(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,%6 ;SET UP EXIT
STPPA: MOV TXVTR,%1
        MOV #OPEN,(1)+ ;SET VECTOR ADDRESS.
        MOV TXLVL,(1)+ ;SET PRIORITY
        RTI ;EXIT.
  
```

```

:ROUTINE TO ISSUE RESET.
SRSETT: MOV #52525,%0 ;DATA TO R0.
        COM %0 ;COMPLEMENT (R0).
        MOV %0,SRSETT+2 ;(R0) TO SRSETT+2.
        RESET ;ISSUE RESET. (R0) IS
        RTI ;DISPLAYED. EXIT.
  
```

```

:RANDOM NUMBER GENERATOR. ROUTINE EXITS WITH NUMBER IN REGISTER 0.
RNGEN: MOV RP1,%0
        ROL %0
        ROL %0
        ADD RP2,%0
        MOV %0,RP1
        ROL %0
        ROL %0
        ADD RP2,%0
        ROL %0
        ROL %0
        MOV %0,RP2
        MOV RP1,%0
        RTS %7 ;EXIT. NUMBER IN R0
  
```

RP1: 1233  
 RP2: 7622

```

:CLRCD - CLEAR CURRENT DEVICE PARAMETERS
CLRCD: CLR TXBUF
        CLR TXCSR
        CLR RXCSR
        CLR RXBUF
        CLR RXVTR
        CLR TXVTR
        CLR RXLVL
        CLR TXLVL
        RTS %7
  
```

```

;SUBROUTINE TO OUTPUT ASCII MESSAGE ON TELETYPE PRINTER.
TYP:   MOV    2%6,%0      ;GET ADDRESS THAT CONTAINS MESSAGE ADDRESS.
      ADD    #2,2%6      ;SET UP EXIT.
      MOV    2%0,%0      ;ADDRESS OF MESSAGE TO RO.
TYP A: MOV B   (0)+,TYPDAT ;GET CHARACTER
      CMP B  #100,TYPDAT ;CHECK FOR"@"CHARACTER
      BNE   TYP C        ;BRANCH IF NOT"@".
TYP C: RTI                ;TERMINATOR CHAR. DONE. EXIT.
      CMP B  #45,TYPDAT  ;CHECK FOR"%".
      BEQ   TYP F        ;BRANCH IF"%".
      CMP B  #43,TYPDAT  ;NOT"%".CHECK FOR"#".
      BEQ   TYP G        ;BRANCH IF "#".
      JSR   %7,TYP D     ;TYPE CHAR IN TYPDAT
      BR    TYP A
TYP D: MOV B  TYPDAT,2TPB ;OUTPUT CHARACTER TO PRINTER
      TST B  2TPS       ;WAIT FOR DONE FLAG.
      BPL   -4
      RTS   %7          ;EXIT
TYP F: MOV B  #15,TYPDAT ;MOVE CARRIAGE RETURN CODE TO TYPDAT
      JSR   %7,TYP D     ;GO TYPE CHAR.
TYP G: MOV B  #12,TYPDAT ;MOVE LF CODE TO TYPDAT.
      JSR   %7,TYP D     ;GO TYPE CHAR.
      BR    TYP A
TYPDAT: OPEN

```

```

;SUBROUTINE TO OUTPUT A SERIES OF ASCII MESSAGES ON TELETYPE PRINTER
TYP S: MOV    2%6,%0      ;GET ADDRESS THAT CONTAINS MESSAGE ADDRESS
      ADD    #2,2%6      ;UPDATE TO NEXT MESSAGE ADDRESS
      MOV    2%0,TYP S B ;ADDRESS OF MESSAGE TO TYP S B
      CMP    #-1,TYP S B ;CHECK FOR TERMINATOR
      BNE   TYP S A      ;BRANCH IF NOT TERMINATOR.
      RTI                ;TERMINATOR. EXIT
TYP S A: TYPE      ;CALL ON TYP SUB TO TYPE MESSAGE
TYP S B: OPEN      ;ADDRESS OF MESSAGE GOES HERE
      BR    TYP S       ;GO PROCESS NEXT MESSAGE

```

```

;OVERLAY VECTOR AREA
OVR LAY: MOV    #300,%1    ;GET DL11-E VECTOR BASE ADDRESS
      MOV    #302,%2
      MOV    #4,%3
OVR L Y A: MOV    %2,(1)+  ;LOAD VECTOR WITH IOT ERROR TRAP
      MOV    %3,(1)+
      ADD    #4,%2
      CMP    %1,#1000     ;ALL VECTORS BEEN LOADED
      BEQ   OVR L Y B
      BR    OVR L Y A
OVR L Y B: RTS    7      ;EXIT

```

```

;SUBROUTINE TO DELAY A SPECIFIED NUMBER OF MILLISECONDS
DLY:   MOV    2%6,DLCNT   ;GET DELAY COUNT ADDRESS.
      ADD    #2,2%6      ;SET UP EXIT ADDRESS
      MOV    2DLCNT,-(6)  ;DELAY COUNT TO STACK
      BEQ   DLY C
      CLR   PSW          ;SET PRIORITY 0
DLY A: MOV    #226,-(6)  ;1 MSEC COUNT TO STACK

```

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 24  
 DDDLAA.P11

```

DLYB:  DEC      @%6      ;DECREMENT 1 MSEC COUNT
      BNE      DLYB      ;BRANCH IF NOT 0.
      POPSP
      DEC      @%6      ;ZERO. UNCOVER MSECS. COUNT.
      BNE      DLYA      ;DECREMENT IT
      POPSP
      RTI
DLYC:  POPSP
      RTI
DLCNT:  OPEN          ;CONTAINS MILLISECONDS COUNT ADDRESS.
  
```

```

;SUBROUTINE TO STALL A RANDOM NUMBER OF MILLISECONDS. MAXIMUM STALL
;DETERMINED BY CONTENTS OF LOC STLMSK.
  
```

```

STAL:  JSR      %7,RNGEN ;GO GET RANDOM NUMBER.
      BIC      STLMSK,%0 ;# IN RD. APPLY STALL MASK,
      BEQ      STALB     ;BRANCH IF RESULT IS 0.
      MOV      %0,STALA
      DELAY
STALA:  OPEN          ;DELAY
      OPEN          ;DELAY COUNT
STALB:  RTI          ;DONE. EXIT.
  
```

```

;SUBROUTINE TO GENERATE RANDOM CHARACTER COUNT
  
```

```

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

```

;SUBROUTINE TO SKIP ON FLAG AND TIME OUT IF SKIP FAILS
  
```

```

TMRX:  MOV      RXCSR,SIOT ;SET UP RXCSR ADDRESS
      BR      TIME1
TMTX:  MOV      TXCSR,SIOT ;SET UP TXCSR ADDRESS
TIME1:  CLR      TIMER
TIME2:  INC      TIMER
      BEQ      TIMEX      ;BRANCH IF COUNTER OVERFLOW
      TSTB     @SIOT
      BPL     TIME2
      ADD     #2,@%6      ;SET UP EXIT RETURN
TIMEX:  RTI
TIMER:  0
SIOT:  0
  
```

```

;SUBROUTINE TO SELECT LINE
  
```

```

LINSEL: BIT      #BIT12,@SRPTR ;BRANCH IF SET
      BNE      LINS LX
      CLR      FOUNDV
      RTS      5
LINS LX: JSR      %7,OVRLAY
      JSR      %7,CLRCD
      TYPE
      LDLINE
      HALT
      MOV      @SRPTR,TEMP
      BIC      #177740,TEMP
  
```

M02

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 25  
DDDLAA.P11

MOV TEMP,LINENO ;SAVE FOR TYPING  
ASL TEMP

```

MOV      TEMP,%1
MOV      RXCR0(1),%1      ;GET RXCSR DEVICE ADDRESS
BIT      #BIT0,%1        ;IS DEVICE THERE
BEQ      LINB             ;YES
LINA:    TYPE             ;NO, REPORT
         MNOLIN
LINB:    BR              LINSLX
         JSR             %7,FORMAD
         CLR             PSW
         BIS             #BIT0,FMAP      ;SET MAPPING FLAG
         BIC             #BIT6,@TXCSR
         BIS             #BIT6,@TXCSR
         NOP
         NOP
         TST             RXVTR
         BEQ             LINA
         BIC             #BIT6,@TXCSR
         MOV             #PTY7,PSW
         JSR             5,OACNV        ;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
PTO:     OPEN
PT1:     OPEN
PIND:    OPEN
PTOP:    OPEN
PTIP:    OPEN

;SPECIAL BINARY COUNT PATTERN SUBROUTINE. EXITS WITH BIN CHAR IN RO
GTBIN:   MOV             PTO,PT1      ;PREVIOUS BIN CHAR TO PT1
         COM             PT1
         COM             RIND
         BNE             .+6
         INC             PT1
         BIC             #177400,PT1  ;MASK TO 8 BITS
         MOV             PT1,PTO      ;SAVE BIN CHAR IN PTO
         MOV             PT1,%0      ;BIN CHAR TO RO.
         RTS             %7           ;EXIT.
GTBINP:  MOV             PTO,PTIP     ;PREVIOUS BIN CHAR TO PTIP
         COM             PTIP
         COM             PIND
         BNE             .+6
         INC             PTIP
         BIC             #177400,PTIP ;MASK TO 8 BITS.
  
```

```

MOV PTIP,PTOP      ;SAVE BIN CHAR IN PTOP.
MOV PTIP,%1        ;BIN CHAR TO R1.
RTS %7             ;EXIT.

;OCTAL TO ASCII CONVERT ROUTINE
OACNV: SAVREG
MOV 2(5)+,OACNVX  ;GET OCTAL VALUE.
MOV (5)+,%1       ;GET DESTINATION ADDR.
MOV (5)+,%2       ;GET CONVERT COUNT.
ADD %2,%1         ;DEVELOP ADDR TO STORE 1ST CHAR.
OACNVA: MOV OACNVX,%3
BIC #177770,%3   ;ISOLATE LEAST SIGNIFICANT DIGIT.
ADD #60,%3       ;CONVERT DIGIT TO ASCII.
MOVB %3,-(1)     ;STORE ASCII CHARACTER.
BIC #7,OACNVX
ROR OACNVX
ROR OACNVX
ROR OACNVX
DEC %2           ;DONE ALL DIGITS?
BNE OACNVA       ;BRANCH IF NOT DONE.
RSTREG
RTS %5          ;DONE. EXIT.
OACNVX: OPEN

;SUBROUTINE TO MOVE A VARIABLE NUMBER OF BYTES.
BMOVE: SAVREG    ;SAVE REGS.
MOV (5)+,%1     ;GET FROM ADDRESS
MOV (5)+,%2     ;GET TO ADDRESS
MOV (5)+,%3     ;GET COUNT
BMOVA: MOVB (1)+,(2)+ ;MOVE BYTE
DEC %3         ;DECREMENT COUNT
BNE BMOVA      ;BRANCH IF NOT DONE.
RSTREG
RTS %5         ;RESTORE REGS.
;DONE EXIT

;BINARY TO DECIMAL ASCII CONVERT SUBROUTINE.
BDCNV: SAVREG
MOV #DECVAL,%0 ;SET UP ADDR TO STORE DECIMAL ASCII IN R0
MOV 2(5)+,%1   ;BINARY VALUE TO R1.
MOV (5)+,BDCNVC ;GET DEST ADDR
MOV (5)+,BDCNVD ;GET CHAR COUNT
MOV #ADTENP,%2 ;ADDR OF TEN POWER STRING TO R2.
MOV #5,CNVCTR  ;SET UP FOR 5 POWER CONVERSIONS.
BDCNVA: MOV (2)+,TENPWR ;MOVE POWER OF TEN VALUE TO TENPWR.
JSR %7,SUBTEN ;PERFORM CONVERSION
DEC CNVCTR     ;DONE 5 CONVERSIONS?
BNE BDCNVA    ;BRANCH IF NOT YET 5.
SUB BDCNVD,%0
MOV %0,BDCNVB
JSR %5,BMOVE

BDCNVB: 0
BDCNVC: 0
BDCNVD: 0
RSTREG
RTS %5
SUBTEN: CLR DIGIT ;YES, EXIT.
;CLEAR DIGIT

```

```

SUBTNA: SUB      TENPWR,%1      :SUBTRACT TEN POWER FROM BINARY VALUE.
        BCS      SUBTNB        :BRANCH IF UNSUCCESSFUL SUBTRACTION.
        INC      DIGIT
        BR       SUBTNA
SUBTNB: ADD      TENPWR,%1      :RESTORE SUBTRACTED VALUE.
        ADD      #60,DIGIT      :CONVERT (DIGIT) TO ASCII
        MOVB     DIGIT,(D)+     :MOVE ASCII CHAR TO DECVAL FIELD.
        RTS     %7             :EXIT.

CNVCTR: OPEN
DIGIT:  OPEN
TENPWR: OPEN
ADTENP: 10000.
        1000.
        100.
        10.
        1.
DECVAL: .BYTE  040,040,040,040,040,040

DATST:  BIC      #BIT1,DRXCSR   :CLEAR DATA TERM. READY
        BIS      #BIT2,DRXCSR   :SET MAINTENANCE BIT
        MOV      #100,CTRO      :GET CHARACTER COUNT
DATAA:  TSTB     DRXCSR         :WAIT FOR
        BPL     -4             :READY FLAG
        JSR     7,GTBINP       :GET CHARACTER
        MOVB    %1,CBUBA       :MOVE CHARACTER
        JSR     7,MASKIT       :MASK OFF NON TRANSMITTED BITS
        MOVB    %1,DRXBUF      :TRANSMIT CHARACTER
        TSTB     DRXCSR         :WAIT FOR
        BPL     -4             :DONE FLAG
        MOVB    DRXBUF,CBUBF    :GET RECEIVED CHARACTER
        DATCHK   :CHK DATA
        DEC      CTRO          :DECREMENT CHARACTER COUNT
        BNE     DATAA
        TST     (6)+          :POP STACK
        SCOPE

SETS:   TYPE
        ASETSR
        HALT
        RTS     %7           :COMMON HALT.
        :EXIT.
INCRTN: TYPE
        AINCRT
        HALT
        RTS     %7           :TYPE INCORRECT ROUTINE SELECTED.
        :COMMON HALT.
        :EXIT.
INCRPG: TYPE
        AINCPG
        HALT
        JMP
PRGEND: CLR      START
        MOV      FOUNDV
        MOV      PASCNT,ERRST+20 :STORE AWAY PASS COUNT
        MOV      LINENO,ERRST+22
        MOV      RXCSR,ERRST+24
        MOV      RXVTR,ERRST+26
        MOV      #60.,TEMP6
13:    RESET
        DEC     TEMP6

```

```

25:   BNE      15
      BIT     #BIT6, @SRPTR      ;HALT ON END OF PASS?
      BEQ     25
      JSR     PC, EOPHLT
      BIT     #BIT13, @SRPTR     ;INHIBIT PRINT SET?
      BNE     PRGEXT            ;BR IF SET
      JSR     %5, BDCNV
      PASCNT
      APCNT
      5
      JSR     %5, OACNV          ;CONVERT LINE NUMBER
      LINENO
      ACLIN
      2
      JSR     %5, OACNV          ;CONVERT RXCSR
      RXCSR
      APRXC
      5
      JSR     %5, OACNV          ;CONVERT VECTOR
      RXVTR
      APVEC
      4
      TYPE
      APGEND
      JMP     .+6

TEMP6: .WORD 0

PRGEXT: BIT     #BIT12, @SRPTR   ;LOCK ON LINE
      BEQ     PRGXT1            ;BR IF NOT SET
      INC     PASCNT
      BR     PRGXTL
PRGXT1: MOV     LINENO, TEMP     ;GET LINENO
      ASL     TEMP
PRGEC:  ADD     #2, TEMP         ;UPDATE LINE NUMBER
PRGEA:  MOV     TEMP, %1
      MOV     RXCR0(1), %1      ;GET RXCSR DEVICE ADDRESS
      CMP     #177777, %1       ;LAST ONE
      BNE     PRGEB            ;NO, CONTINUE
      INC     PASCNT
      CLR     LINENO
      CLR     TEMP
PRGXTL: MOV     @#42, %5
      BEQ     CONT
      RESET

LOGIC:  JSR     7, (5)

CONT:   BIT     #BIT12, @SRPTR   ;LOCK ON LINE
      BEQ     PRGEA            ;BRANCH IF NOT SET
      RTS
PRGEB:  BIT     #BIT0, %1        ;DEVICE THERE
      BNE     PRGEC            ;NO
      ASR     TEMP
      MOV     TEMP, LINENO

```

```

      JSR      %7,FORMAD
      RTS      %7          ;EXIT.

:CONDITIONAL ERROR HALT ROUTINE.
EHLT:  TST      JSRPTR    ;CHECK FOR HALT ON ERROR.
      BPL      EHLTA     ;BRANCH IF NO HALT DESIRED.
      HALT
EHLTA: RTI              ;HALT.
                        ;IN DATA LIGHTS.

:MASKIT - MASK DATA ACCORDING TO LINE NUMBER
MASKIT: MOV      UMASK,RMASK ;GET MASK
      BIC      #177000,RMASK ;REMOVE C/D FLAG+PRIORITY
      COM     RMASK
      BIC      RMASK,CBUBFA ;MASK DESIRED BITS
      RTS      7

:DATA CHECK ROUTINE, TEST ERROR BITS
DTCHK: MOV      CRXBUF,CBUBFB ;DID ANY ERROR BITS SET
      BIT      #170000,CBUBFB
      BNE     DTCHKX      ;YES, TYPE ERROR
      CMP     CRBUF,CBUBFA ;COMPARE EXPECTED AND RECEIVED
      BEQ     DTCHKA      ;CHARS. BRANCH IF SAME.
DTCHKX: MOV      CRUBFA,ERRST+6
      MOV     CRBUF,ERRST+10
      JSR     %5,0ACNV     ;GO TO OCTAL TO ASCII CONVERT.
      CRBUF   ;SOURCE ADDR.
      AWAS   ;DESTINATION ADDR.
      3      ;#OF DIGITS TO CONVERT.
      JSR     %5,0ACNV     ;GO TO OCTAL TO ASCII CONVERT.
      CRUBFA ;SOURCE ADDR.
      AASB   ;DESTINATION ADDR.
      3      ;#OF DIGITS TO CONVERT.
      JSR     %5,0ACNV
      CRUBFB
      ARXBUF
      6
      ERROR1
      ERDAT
DTCHKA: RTI

:ERROR HANDLER
ERR:   MOV      #-1,ERRB    ;SET UP ONE MESSAGE CALL.
      MOV     #240,ERRB+2
      CLR     ERRE
      BR      ERRA
ERR1:  MOV      %6,ERRB     ;DEVELOP ADDT'L MESSAGE ADDR.
      MOV     %ERRB,ERRB    ;STORE AT ERRB.
      MOV     #-1,ERRB+2
      MOV     #2,ERRE
ERRA:  BIT      #BIT13,JSRPTR ;INHIBIT ERROR PRINT?
      BNE     ERRC         ;BRANCH TO INHIBIT PRINT.
      MOV     %6,ERRD      ;DEVELOP CALLING ADDR.
      SUB     #2,ERRD
      MOV     RTNNO,TNNO
      BIC     #BIT15,TNNO
      JSR     %5,0ACNV     ;GO TO OCTAL TO ASCII CONVERT.

```

```

ERRD      :SOURCE ADDR.
APC       :DESTINATION ADDR.
6         :#OF DIGITS TO CONVERT.
JSR      %5,0ACNV :GO TO OCTAL TO ASCII CONVERT.
RXCSR    :SOURCE ADDR.
MAXNUM   :DESTINATION ADDR.
6         :#OF DIGITS TO CONVERT.
JSR      %5,0ACNV :GO TO OCTAL TO ASCII CONVERT.
RTNNO    :SOURCE ADDR.
ATNUMB   :DESTINATION ADDR.
3         :#OF DIGITS TO CONVERT.
TYPES    :TYPE:
ERRB:    OPEN    :ERROR HEADER,
-1       :ADDT'L ERROR MESSAGE IF ANY.
MOV      RTNNO,ERRST
MOV      ERRD,ERRST+2
MOV      RXCSR,ERRST+4
ERRC:    EHALT   :GO ERR HALT IF DESIRED..
ADD      ERRE,%6  :EXIT.
RTI
ERRD:    OPEN
ERRE:    OPEN
:
:ERROR TRAP HANDLER - TYPE TO AND FROM WHERE ERROR TRAP OCCURRED
ERTP:    MOV      PSW,OLDPS      ;SAVE OLD STATUS
MOV      #PRTY7,PSW
ASR      OLDPS
ASR      OLDPS
ASR      OLDPS
BIC      #177740,OLDPS
MOV      OLDPS,TOPC
MOV      %6,FROMPC      ;GET FROM PC
ERTPA:   JSR      %5,0ACNV
TOPC
MTG
6
JSR      %5,0ACNV
FROMPC
MFROM
5
TYPE
MERR
HALT
JMP      START
:
:MAPVEC - MAP VECTOR OR REPORT ERROR DEPENDING ON FMAP FLAG
MAPVEC:  MOV      %6,TOPC
POPS2
MOV      %6,FROMPC
SUB      #4,TOPC
TST      FMAP
BEQ      ERTPA      ;NOT MAPPING, REPORT ERROR
MOV      TOPC,TVTR  ;STORE VECTOR
SUB      #4,TOPC
MOV      TOPC,RVTR

```

```

      CLR      FMAP
      RTI

:FORMAD-FORM DEVICE AT ADDRESSES
FORMAD: MOV     %1,RXCSR
      ADD     #2,%1
      MOV     %1,RXBUF
      ADD     #2,%1
      MOV     %1,TXCSR
      ADD     #2,%1
      MOV     %1,TXBUF
      MOV     LINENO,TEMP      ;GET PRIORITY
      ASL     TEMP
      ADD     #CMASO,TEMP
      MOV     @TEMP,TEMP1
      MOV     TEMP1,UMASK
      SWAB    TEMP1
      ASL     TEMP1
      BIC     #177437,TEMP1
      MOV     TEMP1,RXLVL
      MOV     TEMP1,TXLVL
      RTS     %7

:DOTHIS - SELECTABLE TEST DECISION MAKER
DOTHIS: BIT     #BIT9,@SRPTR      ;IS SELECT TEST SWITCH SET
      BNE     GOBACK              ;RETURN TO TEST IF SW SET
      JMP     GTRDYX              ;GO TO NEXT TEST
GOBACK: RTS     %7

PFAIL: MOV     #PWRUP,24
      HALT
PWRUP: MOV     #PFAIL,24
      RESET
      MOV     #SPBOT,%6
      TYPE
      MPWRF
      ERROR
      BR      RESTART

:DECIDE IF VECTOR TO BE MAPPED AND MAP
TSTVEC: CMP     #0,FOUNDV          ;NEED VECTOR MAPPING
      BNE     TSTVEX              ;NO, EXIT
      JSR     %7,OVRLAY
      CLR     RXVTR
      CLR     PSW
      BIS     #BIT0,FMAP          ;SET MAPPING FLAG
      BIC     #BIT6,@TXCSR       ;CAUSE INTERRUPT
      BIS     #BIT6,@TXCSR
      NOP
      NOP
      TST     RXVTR              ;DID TRAP OCCUR?
      BNE     TSTVA              ;YES, OK
      BIT     #BIT13,@SRPTR
      BNE     TSTVEC
      TYPE                          ;NO, ERROR

```

```

INTER
ERROR
TSTVA:  JMP      TSTVEC
        BIC      #BIT6, @TXCSR
        MOV      #PRTY7, PSW      ;RAISE PRIORITY, RETURN
        INC      FOUNDV
TSTVEX: RTS      %7

:RESTART ROUTINE
RESTART:MOV  PRGNUM,%D
        ASL   %D
        JMP  @RSTART(D)      ;GO RESTART SELECTED PROGRAM

RSTART:  PRG0A      ;PROGRAM 0 RESTART ADDRESS
        PRG1A      ;PROGRAM 1 RESTART ADDRESS
        PRG2A      ;PROGRAM 2 RESTART ADDRESS
        PRG3A      ;PROGRAM 3 RESTART ADDRESS
        PRG4A      ;PROGRAM 4 RESTART ADDRESS
        INCRPG
        INCRPG
        INCRPG

:PRG0 - INPUT-OUTPUT LOGIC TESTS
PRG0:   MOV      #ATO,KSTART
        TST     @#42
        BNE    PRGOB      ;MONITOR LOAD
        TYPE   PRGOB      ;YES, START TEST
        POTIT  ;TYPE TITLE AND INSTRUCTIONS
        HALT
PRGOB:  JSR     7,SETSR
        JSR     5,LINSEL      ;GO GET LINE # FROM USER
PRGOA:  JMP     GETRDY      ;GET STARTED.
        X=-1

:*****
ATO:   10000      ;TEST NUMBER *
        AT1      ;ADDRESS OF NEXT TEST *
        1000.    ;ITERATION COUNT *
        AAA      ;SCOPE ENTRY POINT *
        X=X+1
:*****
:TEST ABILITY TO REFERENCE RECEIVER CSR WITHOUT TRAPPING
AAA:   MOV      #AAE,MACHER      ;SET UP MACHINE ERROR TRAP.
        CLR     @RXCSR          ;REFERENCE RXCSR
AAB:   SCOPE
AAE:   POPSP2
        ERROR
        BR      AAB              ;TRAPPED WHEN REFERENCING RXCSR.
:*****
AT1:   100001     ;TEST NUMBER *
        AT2      ;ADDRESS OF NEXT TEST *
        1000.    ;ITERATION COUNT *
        ABA      ;SCOPE ENTRY POINT *
        X=X+1
:*****

```

```

:TEST ABILITY TO REFERENCE RECEIVER BUFFER WITHOUT TRAPPING
ABA:  MOV      #ABE,MACHER      ;SET UP MACHINE ERROR TRAP.
      TST      @XORFLG
      BMI      ABB
      TST      @RXBUF          ;REFERENCE RXBUF
ABB:  SCOPE                    ;OK IF NO TRAP SCOPE
ABE:  POPSP2
      ERROR                    ;TRAPPED WHEN REFERENCING RXBUF
      BR       ABB

:*****
AT2:  100002                   ;TEST NUMBER *
      AT3                    ;ADDRESS OF NEXT TEST *
      1000.                   ;ITERATION COUNT *
      ACA                    ;SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST ABILITY TO REFERENCE TRANSMITTER CSR WITHOUT TRAPPING.
ACA:  MOV      #ACE,MACHER      ;SET UP MACHINE ERROR TRAP.
      TST      @TXCSR          ;REFERENCE TXCSR
ACB:  SCOPE                    ;SCOPE
ACE:  POPSP2
      ERROR                    ;TRAPPED WHEN REFERENCING TXCSR
      BR       ACB

:*****
AT3:  100003                   ;TEST NUMBER *
      AT4                    ;ADDRESS OF NEXT TEST *
      1000.                   ;ITERATION COUNT *
      ADA                    ;SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST ABILITY TO REFERENCE TRANSMITTER BUFFER WITHOUT TRAPPING
ADA:  MOV      #ADE,MACHER      ;SET UP MACHINE ERROR TRAP.
      TST      @TXBUF          ;REFERENCE TX BUF.
ADB:  SCOPE                    ;SCOPE
ADE:  POPSP2
      ERROR                    ;TRAPPED WHEN REFERENCING TXBUF
      BR       ADB

:*****
AT4:  100004                   ;TEST NUMBER *
      AT5                    ;ADDRESS OF NEXT TEST *
      10.                     ;ITERATION COUNT *
      AEA                    ;SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT TXCSR BIT 0 (BREAK) CAN BE SET AND CLEARED
:AND THAT RESET CLEARS IT
AEA:  BIT      #BIT0,@TXCSR     ;SEE IF BIT IS CLEAR
      BEQ      AEB             ;BR IF CLEAR
      ERROR                    ;RESET DID NOT CLEAR IT
      BR       AED
AEB:  BIS      #BIT0,@TXCSR     ;SET TXCSR BIT 0
      BIT      #BIT0,@TXCSR     ;DID IT SET
      BNE      AEC             ;YES, GO ON
      ERROR                    ;TXCSR BIT0 FAILED TO SET
      BR       AED
AEC:  BIC      #BIT0,@TXCSR     ;CLEAR TXCSR BIT 0
  
```

```

    BIT      #BIT0,@TXCSR ;DID IT CLEAR
    BEQ      AED
AED:  ERROR   ;TXCSR BIT 0 DID NOT CLEAR
    BIS      #BIT0,@TXCSR
    SRESET   ;ISSUE RESET TO CLEAR
    SCOPE

:*****
ATS:  100005 ;TEST NUMBER *
      AT6    ;ADDRESS OF NEXT TEST *
      10.    ;ITERATION COUNT *
      AGA    ;SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT TXCSR BIT2 CAN BE SET, CLEARED, AND THAT RESET CLEARS IT.
AGA:  BIT      #BIT2,@TXCSR ;SEE IF TXCSR BIT2 IS CLEAR.
      BEQ      AGB          ;BRANCH IF BIT IS CLEAR.
      ERROR   ;RESET DID NOT CLEAR TXCSR BIT2
      BR      AGD
AGB:  BIS      #BIT2,@TXCSR ;SET TXCSR BIT2.
      BIT      #BIT2,@TXCSR ;SEE IF BIT IS SET.
      BNE     AGC          ;BRANCH IF BIT IS SET.
      ERROR   ;TXCSR BIT2 FAILED TO SET.
      BR      AGD
AGC:  BIC      #BIT2,@TXCSR ;CLEAR TXCSR BIT2
      BIT      #BIT2,@TXCSR ;SEE IF BIT IS CLEAR.
      BEQ      AGD
AGD:  ERROR   ;TXCSR BIT2 FAILED TO CLEAR.
      BIS      #BIT2,@TXCSR ;SET TXCSR BIT2.
      SRESET   ;ISSUE RESET TO CLEAR BIT.
      SCOPE

:*****
AT6:  100006 ;TEST NUMBER *
      AT7    ;ADDRESS OF NEXT TEST *
      10.    ;ITERATION COUNT *
      AJA    ;SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT TXCSR BIT6 CAN BE SET, CLEARED, AND THAT RESET CLEARS IT.
AJA:  MOV      #PRTY7,PSW ;SET PRIORITY 7.
      BIT      #BIT6,@TXCSR ;SEE IF TXCSR BIT6 IS CLEAR.
      BEQ      AJB          ;BRANCH IF BIT IS CLEAR.
      ERROR   ;RESET DID NOT CLEAR TXCSR BIT6
      BR      AJD
AJB:  BIS      #BIT6,@TXCSR ;SET TXCSR BIT6.
      BIT      #BIT6,@TXCSR ;SEE IF BIT IS SET.
      BNE     AJC          ;BRANCH IF BIT IS SET.
      ERROR   ;TXCSR BIT6 FAILED TO SET.
      BR      AJD
AJC:  BIC      #BIT6,@TXCSR ;CLEAR TXCSR BIT6
      BIT      #BIT6,@TXCSR ;SEE IF BIT IS CLEAR.
      BEQ      AJD
AJD:  ERROR   ;TXCSR BIT6 FAILED TO CLEAR.
      BIS      #BIT6,@TXCSR ;SET TXCSR BIT6.
      SRESET   ;ISSUE RESET TO CLEAR BIT.
      SCOPE
:*****

```

```

AT7: 100007 ; TEST NUMBER *
      AT10 ; ADDRESS OF NEXT TEST *
      100. ; ITERATION COUNT *
      AKA ; SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT TXCSR BIT 7 (READY BIT) IS SET UPON ENTERING ROUTINE AND
:THAT IT CAN BE READ RELIABLY.
AKA: TSTB @TXCSR ; SEE IF TXCSR BIT 7 IS SET.
      BMI AKB ; BRANCH IF SET.
      ERROR ; TXCSR BIT 7 NOT SET.
      SRESET ; ISSUE RESET TO CLEAR BIT IF ERROR
      SCOPE ; SCOPE
AKB:
:*****
AT10: 10 ; TEST NUMBER *
      AT11 ; ADDRESS OF NEXT TEST *
      100. ; ITERATION COUNT *
      ALA ; SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT RXCSR BIT 1 CAN BE SET + CLEARED
ALA: BIC #BIT1,@RXCSR ; SET RXCSR BIT1
      BIS #BIT1,@RXCSR ; SEE IF BIT IS SET
      BIT #BIT1,@RXCSR ; BRANCH IF SET
      BNE ALY ; RXCSR BIT 1 FAILED TO SET
      ERROR
      BR ALZ
ALY: BIC #BIT1,@RXCSR ; CLEAR RXCSR BIT 1
      BIT #BIT1,@RXCSR ; SEE IF BIT IS CLEAR
      BEQ ALZ
      ERROR ; RXCSR BIT 1 FAILED TO CLEAR
      SCOPE ; SCOPE
ALZ:
:*****
AT11: 11 ; TEST NUMBER *
      AT12 ; ADDRESS OF NEXT TEST *
      10. ; ITERATION COUNT *
      APA ; SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT RXCSR BIT2 IS CLEAR AND CAN BE READ RELIABLY.
APA: BIT #BIT2,@RXCSR ; SEE IF RXCSR BIT2 IS CLEAR.
      BEQ APB ; BRANCH IF BIT IS CLEAR.
      ERROR ; RXCSR BIT2 IS NOT CLEAR.
      BR APD
APB: BIS #BIT2,@RXCSR ; SET RXCSR BIT2
      BIT #BIT2,@RXCSR ; SEE IF BIT IS SET
      BNE APCX ; BRANCH IF SET
      ERROR ; RXCSR BIT2 FAILED TO SET
      BR APD
APCX: BIC #BIT2,@RXCSR ; CLEAR RXCSR BIT2
      BIT #BIT2,@RXCSR ; SEE IF BIT IS CLEAR
      BEQ APD
      ERROR ; RXCSR BIT2 FAILED TO CLEAR
      BIS #BIT2,@RXCSR ; SET BIT
      SRESET ; ISSUE RESET TO CLEAR BIT
      SCOPE
  
```

```

*****
AT12: 12 ;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, JRXCSR ;SEE IF RXCSR BIT3 IS CLEAR.
      BEQ AQB ;BRANCH IF BIT IS CLEAR.
      ERROR ;RESET DID NOT CLEAR RXCSR BIT3
AQB: BR AQA
      BIS #BIT3, JRXCSR ;SET RXCSR BIT3.
      BIT #BIT3, JRXCSR ;SEE IF BIT IS SET.
      BNE AQC ;BRANCH IF BIT IS SET.
      ERROR ;RXCSR BIT3 FAILED TO SET.
AQC: BR AQA
      BIC #BIT3, JRXCSR ;CLEAR RXCSR BIT3
      BIT #BIT3, JRXCSR ;SEE IF BIT IS CLEAR.
      BEQ AQA
      ERROR ;RXCSR BIT3 FAILED TO CLEAR.
AQA: BIS #BIT3, JRXCSR ;SET RXCSR BIT3.
      SRESET ;ISSUE RESET TO CLEAR BIT.
      SCOPE ;SCOPE
*****
AT13: 13 ;TEST NUMBER *
      AT14 ;ADDRESS OF NEXT TEST *
      10. ;ITERATION COUNT *
      ARA ;SCOPE ENTRY POINT *
      X=X+1 ;
*****
;TEST THAT RXCSR BITS CAN BE SET, CLEARED, AND THAT RESET CLEARS IT.
ARA: MOV #PTY7, PSW ;PTY7 TO INHIBIT ANY INT
      BIT #BITS, JRXCSR ;SEE IF RXCSR BITS IS CLEAR.
      BEQ ARB ;BRANCH IF BIT IS CLEAR.
      ERROR ;RESET DID NOT CLEAR RXCSR BITS
ARB: BR ARA
      BIS #BITS, JRXCSR ;SET RXCSR BITS.
      BIT #BITS, JRXCSR ;SEE IF BIT IS SET.
      BNE ARC ;BRANCH IF BIT IS SET.
      ERROR ;RXCSR BITS FAILED TO SET.
ARC: BR ARA
      BIC #BITS, JRXCSR ;CLEAR RXCSR BITS
      BIT #BITS, JRXCSR ;SEE IF BIT IS CLEAR.
      BEQ ARA
      ERROR ;RXCSR BIT4 FAILED TO CLEAR.
ARA: BIS #BITS, JRXCSR ;SET RXCSR BITS.
      SRESET ;ISSUE RESET TO CLEAR BIT.
      SCOPE ;SCOPE
*****
AT14: 100014 ;TEST NUMBER *
      AT15 ;ADDRESS OF NEXT TEST *
      10. ;ITERATION COUNT *
      ASA ;SCOPE ENTRY POINT *
      X=X+1 ;
*****

```

:TEST THAT RXCSR BIT6 CAN BE SET, CLEARED, AND THAT RESET CLEARS IT.

```

ASA:  MOV    #PRTY7,PSW      ;SET PRIORITY 7.
      BIT    #BIT6,@RXCSR   ;SEE IF RXCSR BIT6 IS CLEAR.
      BEQ    ASB            ;BRANCH IF BIT IS CLEAR.
      ERROR  ASD            ;RESET DID NOT CLEAR RXCSR BIT6
      BR     ASD
ASB:  BIS    #BIT6,@RXCSR   ;SET RXCSR BIT6.
      BIT    #BIT6,@RXCSR   ;SEE IF BIT IS SET.
      BNE    ASC            ;BRANCH IF BIT IS SET.
      ERROR  ASD            ;RXCSR BIT6 FAILED TO SET.
      BR     ASD
ASC:  BIC    #BIT6,@RXCSR   ;CLEAR RXCSR BIT6
      BIT    #BIT6,@RXCSR   ;SEE IF BIT IS CLEAR.
      BEQ    ASD            ;RXCSR BIT6 FAILED TO CLEAR.
      ERROR  ASD
ASD:  BIS    #BIT6,@RXCSR   ;SET RXCSR BIT6.
      SRESET ;ISSUE RESET TO CLEAR BIT.
      SCOPE ;SCOPE

```

```

*****
AT15: 15                ;TEST NUMBER *
      AT16             ;ADDRESS OF NEXT TEST *
      100.             ;ITERATION COUNT *
      ATA              ;SCOPE ENTRY POINT *
      X=X+1

```

:TEST THAT RXCSR BIT7 IS CLEAR AND CAN BE READ RELIABLY.

```

ATA:  BIT    #BIT7,@RXCSR   ;SEE IF RXCSR BIT7 IS CLEAR.
      BEQ    ATB            ;BRANCH IF BIT IS CLEAR.
      ERROR  ASD            ;RXCSR BIT7 IS NOT CLEAR.
      SRESET ;RESET IF ERROR
      SCOPE ;SCOPE
ATB:  SCOPE

```

```

*****
AT16: 16                ;TEST NUMBER *
      AT17             ;ADDRESS OF NEXT TEST *
      100.             ;ITERATION COUNT *
      AXA              ;SCOPE ENTRY POINT *
      X=X+1

```

:TEST THAT RXCSR BIT10 IS CLEAR AND CAN BE READ RELIABLY.

```

AXA:  BIT    #BIT10,@RXCSR ;SEE IF RXCSR BIT10 IS CLEAR.
      BEQ    AXB            ;BRANCH IF BIT IS CLEAR.
      ERROR  ASD            ;RXCSR BIT10 IS NOT CLEAR.
      SRESET ;RESET BIT IF ERROR
      SCOPE ;SCOPE
AXB:  SCOPE

```

```

*****
AT17: 100017           ;TEST NUMBER *
      AT20             ;ADDRESS OF NEXT TEST *
      100.             ;ITERATION COUNT *
      AYA              ;SCOPE ENTRY POINT *
      X=X+1

```

:TEST THAT RXCSR BIT11 IS CLEAR AND CAN BE READ RELIABLY.

```

AYA:  BIT    #BIT11,@RXCSR ;SEE IF RXCSR BIT11 IS CLEAR.
      BEQ    AYB            ;BRANCH IF BIT IS CLEAR.
      ERROR  ASD            ;RXCSR BIT11 IS NOT CLEAR.
      SRESET ;RESET BIT IF ERROR

```

```

AYB: SCOPE ;SCOPE
:*****
AT20: 100020 ;TEST NUMBER *
      AT21 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      AZA ;SCOPE ENTRY POINT *
      X=X+1 ;
:*****
:TEST THAT RXCSR BIT14 IS CLEAR AND CAN BE READ RELIABLY.
AZA: BIT #BIT14,@RXCSR ;SEE IF RXCSR BIT14 IS CLEAR.
      BEQ AZB ;BRANCH IF BIT IS CLEAR.
      ERROR ;RXCSR BIT14 IS NOT CLEAR.
      SRESET ;RESET BIT IF ERROR
AZB: SCOPE ;SCOPE
:*****
AT21: 100021 ;TEST NUMBER *
      AT22 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      AAAA ;SCOPE ENTRY POINT *
      X=X+1 ;
:*****
:TEST THAT RXCSR BIT15 IS CLEAR AND CAN BE READ RELIABLY.
AAAA: BIT #BIT15,@RXCSR ;SEE IF RXCSR BIT15 IS CLEAR.
       BEQ AAAB ;BRANCH IF BIT IS CLEAR.
       ERROR ;RXCSR BIT15 IS NOT CLEAR.
       SRESET ;RESET BIT IF ERROR
AAAB: SCOPE ;SCOPE
:
:ALL PREVIOUS TESTS MUST HAVE BEEN RUN SUCCESSFULLY PRIOR
:TO RUNNING THE FOLLOWING TESTS. ALSO, THE JUMPER CONNECTOR
:MUST BE INSERTED IN THE DL11-E CABLE IN PLACE OF THE MODEM. COMMENTS
:REFER TO OPERATION WITH JUMPER INSERTED.
:*****
AT22: 22 ;TEST NUMBER *
      AT23 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      AFBA ;SCOPE ENTRY POINT *
      X=X+1 ;
:*****
:TEST THAT CARRIER DETECT SETS AND CLEARS WHEN DATA TERMINAL
:READY SETS AND CLEARS.
AFBA: BIS #BIT1,@RXCSR ;SET DATA TERMINAL READY
      BIT #BIT12,@RXCSR ;TEST CARRIER DETECT
      BNE AFBB ;SHOULD BE SET
      ERROR ;WASN'T
      BR AFBC
AFBB: BIC #BIT1,@RXCSR ;CLEAR DATA TERMINAL READY
      BIT #BIT12,@RXCSR ;TEST CARRIER DETECT
      BEQ AFBC
      ERROR ;WAS SET, ERROR
AFBC: SCOPE
:*****
AT23: 23 ;TEST NUMBER *
      AT24 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
  
```

```

AGBA                                ;SCOPE ENTRY POINT
X=X+1
:*****
:TEST THAT MODEM INTERRUPT (BIT 15) SETS WHEN CARRIER DETECT
:CHANGES STATE, AND IS CLEARED WHEN RXCSR IS READ.
AGBA:  BIC      #BIT1, @RXCSR      ;CLEAR DATA TERMINAL READY
        MOV      @RXCSR, RXCSRT    ;READ RXCSR
        BIT      #BIT15, @RXCSR   ;TEST MODEM INTERRUPT
        BEQ      AGBB             ;WAS CLEAR GO TO AGBB
        ERROR   ;WASN'T CLEAR
        BR       AGBE             ;GO TO SCOPE
AGBB:  BIS      #BIT1, @RXCSR      ;SETTING DATA TERMINAL READY
        ;CAUSES CARRIER DETECT TO SET
        ;WHICH CAUSES MODEM INTERRUPT TO SET
        MOV      @RXCSR, RXCSRT    ;MOVE RXCSR TO TEMPORARY LOCATION
        BIT      #BIT15, RXCSRT   ;TEST MODEM INTERRUPT
        BNE      AGBC             ;SHOULD BE SET GO TO AGBC
        ERROR   ;WAS CLEAR
        BR       AGBE             ;GO TO SCOPE
AGBC:  BIT      #BIT15, @RXCSR     ;MODEM INTERRUPT BIT SHOULD
        ;HAVE BEEN CLEARED
        BEQ      AGBD             ;IT WAS GO TO AGBD
        ERROR   ;IT WASN'T
        BR       AGBE             ;GO TO SCOPE
AGBD:  BIC      #BIT1, @RXCSR      ;CLEARING DATA TERMINAL READY
        ;CAUSES CARRIER DETECT TO CLEAR
        ;BUT MODEM INTERRUPT WILL SET
        MOV      @RXCSR, RXCSRT    ;MOV RXCSR TO TEMPORARY LOCATION
        BIT      #BIT15, RXCSRT   ;TEST MODEM INTERRUPT
        BNE      AGBE             ;SHOULD BE SET
        ERROR   ;IT WASN'T
        BR       AGBE
AGBE:  SCOPE                      ;SCOPE
:*****
AT24:  24                          ;TEST NUMBER
        AT25                      ;ADDRESS OF NEXT TEST
        100                        ;ITERATION COUNT
        AJBA                          ;SCOPE ENTRY POINT
        X=X+1
:*****
:TEST THAT CLEAR TO SEND (BIT13) SETS/CLEARs WHEN DATA TERMINAL
:READY SETS/CLEARs.
AJBA:  BIC      #BIT1, @RXCSR      ;CLEAR DATA TERMINAL READY
        BIT      #BIT13, @RXCSR   ;TEST CLEAR TO SEND
        BEQ      AJBB             ;CLEAR TO SEND SHOULD BE CLEAR
        ERROR   ;
        BR       AJBD
AJBB:  BIS      #BIT1, @RXCSR      ;SET DATA TERMINAL READY
        BIT      #BIT13, @RXCSR   ;TEST CLEAR TO SEND
        BNE      AJBC             ;BRANCH IF SET
        ERROR   ;CLEAR TO SEND SHOULD BE SET
        BR       AJBD
AJBC:  BIC      #BIT1, @RXCSR      ;CLEAR DATA TERMINAL READY
        BIT      #BIT13, @RXCSR   ;TEST CLEAR TO SEND
        BEQ      AJBD             ;CLEAR TO SEND SHOULD BE CLEAR
        ERROR   ;
  
```



```

ALBB:  BR      ALBD
      BIS      #BIT3,@RXCSR      ;SET SUPERVISORY XMIT DATA
      BIT      #BIT10,@RXCSR     ;TEST SUPERVISORY RECEIVE DATA
      BNE      ALBC              ;SHOULD HAVE BEEN SET
      ERROR
ALPC:  BR      ALBD
      BIC      #BIT3,@RXCSR      ;CLEAR SUPERVISORY XMIT DATA
      BIT      #BIT10,@RXCSR     ;TEST SUPERVISORY RECEIVE DATA
      BEQ      ALBD              ;SHOULD HAVE BEEN CLEAR
      ERROR
ALBD:  SCOPE                      ;SCOPE
:*****
AT30:  30                          ;TEST NUMBER *
      AT31                      ;ADDRESS OF NEXT TEST *
      100.                      ;ITERATION COUNT *
      AMBA                      ;SCOPE ENTRY POINT *
      X=X+1                      ;
:*****
:TEST THAT SUP REC DATA TRANSITIONS SET MODEM INTERRUPT
AMBA:  BIC      #BIT3,@RXCSR      ;CLEAR SUP REC
      BIS      #BIT3,@RXCSR      ;SET SUP REC
      BIT      #BIT15,@RXCSR     ;TEST MODEM INTERRUPT
      BNE      AMBB              ;MODEM INTERRUPT SHOULD BE SET
      ERROR
AMBB:  BR      AMBE
      BIT      #BIT15,@RXCSR     ;MODEM INTERRUPT SHOULD BE
      BEQ      AMBC              ;CLEARED BY PREVIOUS READ
      ERROR
AMBC:  BR      AMBE
      BIC      #BIT3,@RXCSR      ;1-0 TRANS OF SUP REC DATA
      BIT      #BIT15,@RXCSR     ;TEST MODEM INTERRUPT
      BNE      AMBD              ;SHOULD BE SET
      ERROR
AMBD:  BR      AMBE
      BIS      #BIT3,@RXCSR      ;0-1 TRANS OF SUP REC DATA
      BIT      #BIT15,@RXCSR     ;TEST MODEM INTERRUPT
      BNE      AMBE              ;SHOULD BE SET
      ERROR
AMBE:  SCOPE
:*****
AT31:  100031                      ;TEST NUMBER *
      AT32                      ;ADDRESS OF NEXT TEST *
      10.                      ;ITERATION COUNT *
      ABAA                      ;SCOPE ENTRY POINT *
      X=X+1                      ;
:*****
:TEST THAT RESET CLEARS ALL TXCSR BITS, AND SETS BIT 7 (READY)
ABAA:  MOV      #PTY7,PSW          ;SET PRIORITY 7.
      MOV      #-1,@TXCSR        ;SET ALL POSSIBLE BITS IN TXCSR
      SRESET                      ;ISSUE RESET TO CLEAR BITS
      CMP      #BIT7,@TXCSR      ;SEE IF ONLY BIT 7 IS SET.
      BEQ      ABAB              ;BRANCH IF ONLY BIT 7 IS SET
      MOV      @TXCSR,TXCST      ;SAVE CONTENTS OF TXCSR
      MOV      #BIT7,TEMP        ;MOVE EXPECTED TXCSR TO TEMP.
      JSR      %5,0ACNV          ;GO TO OCTAL TO ASCII CONVERT.
      TEMP                      ;SOURCE ADDR.
  
```

```

        ATXSB          ;DESTINATION ADDR.
        6              ;#OF DIGITS TO CONVERT.
        JSR           %5,0ACNV ;GO TO OCTAL TO ASCII CONVERT.
        TXCSRT        ;SOURCE ADDR.
        ATXWAS        ;DESTINATION ADDR.
        6              ;#OF DIGITS TO CONVERT.
        ERROR1        ;RESET FAILED TO CLEAR ALL BITS EXCEPT
        ATXCSR        ;BIT 7 - SEE PRINTOUT
ABAB:   SCOPE         ;SCOPE
*****
AT32:   32            ;TEST NUMBER *
        AT33          ;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           #PRTY7,PSM ;SET PRIORITY 7
        BIC           #BIT1,RXCSR ;CLEAR DATA TERM.READY
        MOV           #-1,RXCSR  ;SET ALL POSSIBLE BITS IN RXCSR
        BIS           #4,ATXCSR  ;SET MAINT BIT
        CLR           @TXBUF     ;TRANSMIT A CHAR
        TIMETX        ;TIME OUT TX DONE
        ERROR        ;ERROR DONE NOT SETTING
        MOV           #1,@TXBUF  ;TRANSMIT ANOTHER CHAR.
        TIMERX        ;TIME OUT RX DONE
        ERROR        ;ERROR DONE NOT SETTING
        SRESET       ;ISSUE RESET TO CLEAR BITS.
        MOV           @RXCSR,RXCSRT ;MOVE RXCSR CONTENTS TO RXCSRT
        CMP           #30002,RXCSRT ;SEE IF ONLY BITS 1,12,13 SET
        BEQ          ACAA       ;BRANCH IF ONLY BITS 1,12,13 SET.
        MOV           #30002,TEMP
        JSR           %5,0ACNV   ;GO TO OCTAL TO ASCII CONVERT.
        TEMP         ;SOURCE ADDR.
        ARXSB        ;DESTINATION ADDR.
        6              ;#OF DIGITS TO CONVERT.
        JSR           %5,0ACNV   ;GO TO OCTAL TO ASCII CONVERT.
        RXCSRT       ;SOURCE ADDR.
        ARXWAS       ;DESTINATION ADDR.
        6              ;#OF DIGITS TO CONVERT.
        ERROR1        ;RESET FAILED TO CLEAR ALL BITS EXCEPT
        ARXCSR        ;BIT 0. SEE ERROR PRINTOUT.
ACAB:   BIC           #BIT1,@RXCSR ;CLEAR DATA TERM. READY
        SCOPE         ;SCOPE
*****
AT33:   100033        ;TEST NUMBER *
        AT34          ;ADDRESS OF NEXT TEST *
        10.           ;ITERATION COUNT *
        ADAA         ;SCOPE ENTRY POINT *
        X=X+1
*****
;TEST THAT LOADING TXBUF (TRANSMITTER BUFFER) CLEARS TXCSR BIT 7 (READY)
;AND WITHOUT MAINT SET THAT TXDONE SETS READY
ADAA:   CLR           @TXBUF     ;LOAD TXBUF
        TIMETX        ;TIME OUT TX DONE

```

```

      ERROR          ;ERROR, DONE NOT SETTING
      CLR           @TXBUF      ;LOAD TX BUF
      TSTB         @TXCSR      ;TEST TXCSR BIT 7 (READY BIT)
      BPL          ADAB        ;BRANCH IF BIT NOT SET.
      ERROR        ;ERROR. LOADING TXBUF FAILED TO CLEAR READY.
      BR           ADAC
ADAB:  TIMETX       ;WAIT FOR DONE
      ERROR        ;DONE NEVER SET
      BIT         @BIT7,@TXCSR
      BNE         .+4
ADAC:  ERROR        ;READY DID NOT SET
      SRESET
      SCOPE        ;SCOPE.
:*****
AT34:  100034      ;TEST NUMBER *
      AT35        ;ADDRESS OF NEXT TEST *
      1           ;ITERATION COUNT *
      AIAA        ;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,DOTHIS   ;TEST IF THIS TEST SELECTED
      TYPES
      MSETTX
      MSETC
      MS0
      -1
      HALT
      JSR         %7,AIAS      ;OUTPUT CHAR AND TIME.
      MOV        AIAST,CTRO    ;MOVE ELAPSED TIME TO CTRO.
      TYPE
      MS1
      HALT
      JSR         %7,AIAS      ;OUTPUT CHAR AND TIME.
      MOV        AIAST,CTR1    ;MOVE ELAPSED TIME TO CTR1.
      TYPE
      MS2
      HALT
      JSR         %7,AIAS      ;OUTPUT CHAR AND TIME.
      MOV        AIAST,CTR2    ;MOVE ELAPSED TIME TO CTR2.
      TYPE
      MS3
      HALT
      JSR         %7,AIAS      ;OUTPUT CHAR AND TIME.
      MOV        AIAST,CTR3    ;MOVE ELAPSED TIME TO CTR3.
      TYPE
      MS4
      HALT
      JSR         %7,AIAS      ;OUTPUT CHAR AND TIME.
      MOV        AIAST,CTR4    ;MOVE ELAPSED TIME TO CTR4.
      TYPE
      MS5
      HALT
      JSR         %7,AIAS      ;OUTPUT CHAR AND TIME.
      MOV        AIAST,CTR5    ;MOVE ELAPSED TIME TO CTR5
  
```

```

TYPE
MS6
HALT
JSR %7 AIAS ;OUTPUT CHAR AND TIME
MOV AIAST,CTR6 ;MOVE ELAPSED TIME TO CTR6
TYPE
MS7
HALT
JSR %7 AIAS ;OUTPUT CHAR AND TIME
MOV AIAST,CTR7 ;MOVE ELAPSED TIME TO CTR7
JSR %7 CMPT ;CHECK THAT CTR0 THROUGH CTR7 CONTAIN
BR AIAF ;DESCENDING VALUES
ERROR1 ;TRANSMIT SPEEDS NOT ARRANGED IN
ETXTIM ;ASCENDING ORDER.
SCOPE ;SCOPE
AIAF:
AIAS: CLR AIAST ;CLEAR ELAPSED TIME COUNTER.
TSTB @TXCSR ;WAIT FOR TX READY.
BPL -4
CLR @TXBUF
TSTB @TXCSR
BPL -4
AIASA: CLR @TXBUF ;LOAD TXBUF.
JSR %7 TIME ;WAIT 75 US
INC AIAST ;INCREMENT ELAPSED TIME COUNTER.
TSTB @TXCSR ;READY SET?
BPL AIASA ;BRANCH IF READY NOT SET.
RTS %7 ;EXIT.

TIME: MOV #15.,%D
TIM1: DEC %D
BNE TIM1
RTS %7

AIAST: OPEN
*****
AT35: 35 ;TEST NUMBER *
AT36 ;ADDRESS OF NEXT TEST *
10. ;ITERATION COUNT *
ALAA ;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 MSECS, AND
;THAT RESET INSTRUCTION CLEARS THE DONE BIT

ALAA: BIS #BIT2,@TXCSR ;SET MAINTENANCE BIT
CLR @TXBUF ;LOAD TXBUF
DELAY 500 ;WAIT 500 MSECS.
TSTB @RXCSR ;SEE IF DONE BIT IS SET
BMI ALAB ;BRANCH IF DONE BIT IS SET
ERROR ;DONE BIT FAILED TO SET
BR ALAC
ALAB: SRESET ;ISSUE RESET TO CLEAR DONE BIT
TSTB @RXCSR ;SEE IF DONE BIT IS CLEARED
BPL ALAC ;BRANCH IF DONE BIT IS CLEARED
  
```

```

ALAC: ERROR ;RESET FAILED TO CLEAR DONE BIT
      SCOPE ;SCOPE
*****
AT36: 100036 ;TEST NUMBER *
      AT37 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      AMAA ;SCOPE ENTRY POINT *
      X=X+1 ;
*****
;TEST THAT DONE BIT (RXCSR BIT 7) IS CLEARED BY READING RXBUF.
;DONE SET BY OUTPUTTING CHARACTER WITH MAINTENANCE BIT SET (TXCSR BIT 2)
AMAA: BIS #BIT2,@TXCSR ;SET MAINTENANCE BIT (TXCSR BIT 2)
      CLR @TXBUF ;LOAD TXBUF
      TIMERX ;WAIT FOR DONE BIT TO SET.
      ERROR
      TST @RXBUF ;READ RXBUF TO CLEAR DONE BIT
      TSTB @RXCSR ;SEE IF DONE BIT IS CLEAR
      BPL AMAC ;BRANCH IF DONE BIT IS CLEAR
      ERROR ;READING RXBUF FAILED TO CLEAR DONE BIT
AMAC: SCOPE ;SCOPE
*****
AT37: 100037 ;TEST NUMBER *
      AT40 ;ADDRESS OF NEXT TEST *
      100. ;ITERATION COUNT *
      A0AA ;SCOPE ENTRY POINT *
      X=X+1 ;
*****
;TEST THAT RECEIVER ACTIVE SETS WHEN CHAR STARTS AND
;CLEARS WHEN RECEIVER DONE SETS
A0AA: JSR %7,CDINIT ;INIT IF C-D DEVICE
      BIS #BIT2,@TXCSR ;SET MAINT
      CLR @TXBUF ;TRANSMIT CHAR
A0AB: CLR TEMP ;CLEAR BUSY INDICATOR
      BIT #BIT11,@RXCSR ;IS RECEIVER ACTIVE SET
      BEQ A0AB1 ;BRANCH IF CLEAR
A0AB1: INC TEMP ;YES, REMEMBER THAT
      TSTB @RXCSR ;SEE IF DONE SET
      BPL A0AB
      CMP TEMP,#0 ;DID RECEIVER ACTIVE SET
      BNE A0AC ;RECEIVER ACTIVE NEVER SET
      ERROR
A0AC: BR A0AD ;DID DONE CLEAR ACTIVE
      BIT #BIT11,@RXCSR
      BEQ A0AD
      ERROR ;NO, RECEIVER ACTIVE DID NOT CLEAR
A0AD: TST @RXBUF ;CLEAR RX DONE
      SCOPE
*****
AT40: 40 ;TEST NUMBER *
      AT41 ;ADDRESS OF NEXT TEST *
      1. ;ITERATION COUNT *
      A0AA ;SCOPE ENTRY POINT *
      X=X+1 ;
*****
;TEST THAT RECEIVE SPEEDS ARE ARRANGED IN ASCENDING ORDER BY CHECKING THAT TIME
;ELAPSED TO DONE BIT SETTING (RXCSR BIT 7) DECREASES AS A HIGHER SPEED

```

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 47  
DDDLAA.P11

: THIS IS NOT DONE IN MAINTENANCE MODE TX AND RX  
: POTS MUST BE STEPPED TOGETHER  
: IS SELECTED.

```
AQAA: JSR      %7,DOTHIS      ;CHECK IF THIS TEST TO BE DONE
      TYPES
      MSETRX
      MSETC
      MSD
      -1
      HALT
      JSR      %7,AQAS        ;OUTPUT CHARACTER AND TIME DONE BIT
      MOV      AQAST,CTRO     ;MOVE ELAPSED TIME TO CTRO
      TYPE
      MS1
      HALT
      JSR      %7,AQAS        ;OUTPUT CHARACTER AND TIME DONE BIT
      MOV      AQAST,CTR1     ;MOVE ELAPSED TIME TO CTR1
      TYPE
      MS2
      HALT
      JSR      %7,AQAS        ;OUTPUT CHARACTER AND TIME DONE BIT.
      MOV      AQAST,CTR2     ;MOVE ELAPSED TIME TO CTR2.
```

J04

MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 48  
ODDLAA.F11

TYPE

K04

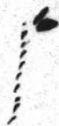
.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 49  
DDDLAA.F11

```
MS3  
HALT  
JSR    %7,AQAS      ;OUTPUT CHARACTER AND TIME DONE BIT  
MOV    AQAST,CTR3   ;MOVE ELAPSED TIME TO CTR3.  
TYPE  
MS4  
HALT  
JSR    %7,AQAS  
MOV    AQAST,CTR4  
TYPE  
MS5  
HALT  
JSR    %7,AQAS  
MOV    AQAST,CTR5
```

L04

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 50  
DDDLAA.P11

TYPE  
MS6  
HALT



MO4

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 51  
DDDLAA.P11

JSR %7 AQAS  
MOV AQAST,CTR6



```

      CLR      @TXBUF      ;LOAD TXBUF
      DELAY    500.        ;DELAY 500 MSECS
      RTS      %7          ;EXIT
*****
AT42: 100042      ;TEST NUMBER *
      AT43      ;ADDRESS OF NEXT TEST *
      10.        ;ITERATION COUNT *
      ATAA      ;SCOPE ENTRY POINT *
      X=X+1
*****
;TEST THAT TRANSMITTER IS ABLE TO INTERRUPT. IF THE INTERRUPT IS SERVICED,
;IT WILL HAVE OCCURRED AT THE CORRECT VECTOR.
      JSR      7,OVRLAY    ;GO TO OVER LAY ROUTINE
      STTXV     ;SET TX INTERRUPT SERVICE
      ATAC      ;TO ATAC
ATAA: BIC      @BIT6,@TXCSR ;DISABLE TX INTERRUPT
      CLR      PSM        ;SET PROCESSOR PRIORITY TO 0
      BIS      @BIT6,@TXCSR ;ENABLE TX INTERRUPT
      NOP
      ERROR     ;READY DID NOT CAUSE AN INTERRUPT
ATAB: BIC      @BIT6,@TXCSR
ATAAC: BIC     @BIT6,@TXCSR ;SCOPE
      POPSP2    ;HERE IF INT, DISABLE TX INT
      BR       ATAB
*****
AT43: 100043      ;TEST NUMBER *
      AT44      ;ADDRESS OF NEXT TEST *
      1000.     ;ITERATION COUNT *
      AUAA      ;SCOPE ENTRY POINT *
      X=X+1
*****
;TEST THAT READY DOES NOT CAUSE AN INTERRUPT WHEN THE PROCESSOR IS
;AT THE SAME PRIORITY AS THE TRANSMITTER INTERRUPT REQUEST LEVEL
      STTXV     ;SET TX INTERRUPT SERVICE TO
AUAA: MOV      TXLVL,PSM   ;SET PROCESSOR PRIORITY SAME AS TX PRIORITY
      BIC      @BIT6,@TXCSR ;ENABLE TX INTERRUPTS
      BIS      @BIT6,@TXCSR
AUAB: NOP
      BIC      @BIT6,@TXCSR ;OK IF NO INTERRUPT OCCURS. DISABLE INTERRUPTS
      SCOPE
AUAC: POPSP2    ;HERE IF INTERRUPT OCCURS. POP STOCK TWICE
      ERROR     ;TX INTERRUPTED WITH PROCESSOR AT SAME
      BR       AUAB      ;PRIORITY AS THE TRANSMITTER
*****
AT44: 100044      ;TEST NUMBER *
      AT45      ;ADDRESS OF NEXT TEST *
      10.        ;ITERATION COUNT *
      AVAA      ;SCOPE ENTRY POINT *
      X=X+1
*****
;TEST THAT TRANSMITTER INTERRUPTS WHEN PROCESSOR IS AT PRIORITY ONE LEVEL
;LOWER THAN THE TRANSMITTER INTERRUPT PRIORITY.
      STTXV     ;SET TX INTERRUPT SERVICE TO AVAB
  
```

```

AVAB:  AVAB
      BIC    #BIT6,@TXCSR    ;DISABLE TX INTERRUPTS
      MOV    TXLVL,PSW      ;SET PROCESSOR PRIORITY TO ONE LEVEL
      SUB    #40,PSW        ;LOWER THAN TX PRIORITY
      BIS    #BIT6,@TXCSR    ;ENABLE TX INTERRUPTS
      NOP
      ERROR  ;TX FAILED TO INTERRUPT
      BR     AVAC
AVAB:  POPSP2
AVAC:  BIC    #BIT6,@TXCSR    ;HERE IF INTERRUPT OCCURS. POP STOCK TWICE
      SCOPE  ;DISABLE TX INTERRUPTS
      SCOPE  ;SCOPE
*****
AT45:  100045 ;TEST NUMBER *
      AT46  ;ADDRESS OF NEXT TEST *
      100.  ;ITERATION COUNT *
      AXAA  ;SCOPE ENTRY POINT *
      X=X+1
*****
;TEST THAT TRANSMITTER DOES NOT REINTERRUPT AFTER THE INITIAL INTERRUPT HAS
;OCCURRED AND HAS BEEN SERVICED.
AWAA:  STTXV ;SET TX INTERRUPT SERVICE TO AWAC
      AWAC
      BIC    #BIT6,@TXCSR    ;DISABLE TX INTERRUPTS
      CLR    PSW            ;SET PROCESSOR PRIORITY TO 0
      BIS    #BIT6,@TXCSR    ;ENABLE TX INTERRUPTS
      NOP
      ERROR  ;TRANSMITTER FAILED TO INTERRUPT
AWAB:  BIC    #BIT6,@TXCSR    ;DISABLE TX INTERRUPTS
      SCOPE  ;SCOPE
AWAC:  MOV    #AWAE,@TXVTR    ;HERE IF INTERRUPT OCCURS. CHANGE EXIT
      MOV    #AWAD,@%6       ;POINTER TO AWAD AND EXIT INTERRUPT
      RTI
AWAD:  NOP ;OK IF NO INTERRUPT REOCCURS.
      BR     AWAB
AWAE:  POPSP2
      ERROR  ;TX REINTERRUPTED AFTER RTI
      BR     AWAB
*****
AT46:  100046 ;TEST NUMBER *
      AT47  ;ADDRESS OF NEXT TEST *
      10.   ;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.
      JSR    7,OVRLAY        ;GO TO OVERLAY ROUTINE
      STRXV ;SET RX INTERRUPT SERVICE TO AXAB
      AXAB
AXAA:  JSR    %7,STRXD        ;SET RX DONE BIT
      BIC    #BIT6,@RXCSR    ;DISABLE RX INTERRUPTS
      CLR    PSW            ;SET PROCESSOR PRIORITY TO 0
      BIS    #BIT6,@RXCSR    ;ENABLE RX INTERRUPTS
      NOP
      ERROR  ;RX FAILED TO INTERRUPT
      BR     AXAC
  
```

```

AXAB: POPSP2           ;HERE IF INTERRUPT OCCURS
AXAC: BIC      #BIT6, @RXCSR ;DISABLE INT EN
      SCOPE           ;SCOPE
:*****
AT47:  47             ;TEST NUMBER *
      AT50           ;ADDRESS OF NEXT TEST *
      10.            ;ITERATION COUNT *
      AX1A           ;SCOPE ENTRY POINT *
      X=X+1         ;
:*****
:TEST THAT MODEM INTERRUPT BIT IS ABLE TO INTERRUPT. IF THE INTERRUPT IS
:SERVICED IT WILL HAVE OCCURRED AT THE CORRECT VECTOR.
      JSR      7, OVRLAY ;GO TO OVERLAY ROUTINE
      STRXV           ;SET RX INTERRUPT SERVICE TO AXAB
AX1A:  BIC      #44, @RXCSR ;DISABLE MODEM INTERRUPTS
      CLR      PSW     ;SET PROCESSOR PRIORITY TO 0
      BIS      #44, @RXCSR ;ENABLE MODEM INTERRUPTS, RG TO SND
      DELAY      5
      ERROR           ;MODEM FAILED TO INTERRUPT
      BR      AX1C
AX1B:  POPSP2           ;HERE IF INTERRUPT OCCURS
AX1C:  BIC      #BIT5, @RXCSR ;DISABLE INT EN
      SCOPE
:*****
AT50:  10050         ;TEST NUMBER *
      AT51           ;ADDRESS OF NEXT TEST *
      1000.          ;ITERATION COUNT *
      AYAA           ;SCOPE ENTRY POINT *
      X=X+1         ;
:*****
:TEST THAT RECEIVER DONE BIT DOES NOT CAUSE AN INTERRUPT WHEN THE PROCESSOR
:IS AT THE SAME PRIORITY LEVEL AS THE RECEIVER INTERRUPT REQUEST LEVEL
      STRXV           ;SET RX INTERRUPT SERVICE TO AYAC
      AYAC
      JSR      %7, STRXD ;SET RX DONE BIT
AXAA:  BIC      #BIT6, @RXCSR ;DISABLE RX INTERRUPTS
      MOV      RXLVL, PSW ;SET PROCESSOR PRIORITY SAME AS RECEIVER'S
      BIS      #BIT6, @RXCSR ;ENABLE RX INTERRUPTS
      NOP
AXAB:  BIC      #BIT6, @RXCSR ;OK IF NO INTERRUPT. DISABLE RX INTERRUPTS
      SCOPE
AXAC:  POPSP2           ;HERE IF INTERRUPT OCCURS. POP STOCK TWICE
      ERROR           ;RX INTERRUPTED WITH PROCESOR AT SAME
      BR      AYAB     ;PRIORITY AS THE RECEIVER
:*****
AT51:  10051         ;TEST NUMBER *
      AT52           ;ADDRESS OF NEXT TEST *
      10.            ;ITERATION COUNT *
      AZAA           ;SCOPE ENTRY POINT *
      X=X+1         ;
:*****
:TEST THAT RECEIVER DONE BIT CAUSES INTERRUPT WHEN PROCESSOR IS AT PRIORITY
:ONE LEVEL LOWER THAN THE RECEIVER'S INTERRUPT REQUEST LEVEL
      STRXV           ;SET RX INTERRUPT TO AZAB
  
```

```

AZAB: JSR      %7,STRXD      ;SET RX DONE BIT
      BIC      #BIT6,@RXCSR ;DISABLE RX INTERRUPTS
      MOV      RXLVL,PSW    ;SET PROCESSOR PRIORITY ONE LEVEL
      SUB      #40,PSW     ;LOWER THAN RECEIVER'S PRIORITY
      BIS      #BIT6,@RXCSR ;ENABLE RX INTERRUPTS
      NOP
      ERROR
      BR       AZAC        ;RX FAILED TO INTERRUPT WITH PROCESSOR AT
                          ;PRIORITY ONE LEVEL LOWER THAN RECEIVER'S
AZAB: POPSP2
AZAC: BIC      #BIT6,@RXCSR ;DISABLE RX INTERRUPTS
      SCOPE
:*****
AT52: 100052      ;TEST NUMBER *
      AT53      ;ADDRESS OF NEXT TEST *
      100.      ;ITERATION COUNT *
      ABBA      ;SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT RECEIVER DOES NOT INTERRUPT AFTER THE INITIAL INTERRUPT HAS
:OCCURED AND DONE BIT HAS NOT BEEN CLEARED
AABA: JSR      %7,STRXD      ;SET RX DONE BIT
      STRXV    ;SET RX INTERRUPT SERVICE TO AABC
      AABC
      BIC      #BIT6,@RXCSR ;DISABLE RX INTERRUPTS
      BIS      #BIT6,@RXCSR ;ENABLE RX INTERRUPTS
      NOP
      ERROR
      BIC      #BIT6,@RXCSR ;DISABLE RX INTERRUPTS
AABB: SCOPE
      MOV      #AABE,@RXVTR ;HERE IF INTERRUPT OCCURS. CHANGE SERVICE TO
      MOV      #AABD,@%6    ;AABE, SET EXIT POINTER TO AABD
      RTI      ;EXIT INTERRUPT SERVICE
AABD: NOP
      BR       AABB        ;OK IF NO INTERRUPT REOCCURS
AABE: POPSP2
      ERROR
      BR       AABB        ;HERE IF INTERRUPT REOCCURS
                          ;RX REINTERRUPTED AFTER RTI
:*****
AT53: 100053      ;TEST NUMBER *
      AT54      ;ADDRESS OF NEXT TEST *
      100.      ;ITERATION COUNT *
      ABBA      ;SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT READING RXCSR DOES NOT CLEAR DONE BIT (RXCSR BIT 7 )
AABA: JSR      %7,STRXD      ;SET RX DONE BIT
      MOV      @RXCSR,RXCST ;SAVE CONTENT OF RXCSR
      TSTB    @RXCSR        ;SEE IF DONE BIT IS CLEAR
      BMI     ABBB          ;BRANCH IF DONE BIT IS NOT CLEAR
      ERROR
AABB: TST      @RXBUF        ;CLEAR DONE BIT IF SET
      SCOPE
:*****
AT54: 100054      ;TEST NUMBER *
      AT55      ;ADDRESS OF NEXT TEST *

```

```

100.          : ITERATION COUNT *
ACBA          : SCOPE ENTRY POINT *
X=X+1        : *
:*****
:TEST THAT DONE CAN CAUSE INT WITH ERROR SET
STRXV        : SET RX INTERRUPT SERVICE TO ACBB.
ACBA: JSR     %7,STRXD      : SET RX DONE BIT
        JSR     %7,STRXD      : SET RX DATA OFLOW
        BIC     #BIT6,DRXCSR   : DISABLE RX INTERRUPTS
        CLR     PSW          : SET PROCESSOR PRIORITY TO 0
        BIS     #BIT6,DRXCSR   : ENABLE RX INTERRUPTS
        NOP
        ERROR      : RX DONE FAILED TO CAUSE INTERRUPT
ACBB: BR     ACBC
ACBC: POPSP2      : HERE IF INTERRUPT OCCURS. POP STOCK TWICE
        BIC     #BIT6,DRXCSR
        SCOPE
:*****
AT55: 100055      : TEST NUMBER *
        AT56      : ADDRESS OF NEXT TEST *
        3         : ITERATION COUNT *
        ADDA      : SCOPE ENTRY POINT *
        X=X+1
:*****
:DATA TEST USING NORMAL CONFIGURATION
ACBA: JSR     %7,CDINIT      : INIT IF C-D DEVICE
        JSR     5,DATTST
        SCOPE
:*****
AT56: 56         : TEST NUMBER *
        AT57      : ADDRESS OF NEXT TEST *
        3         : ITERATION COUNT *
        APBA      : SCOPE ENTRY POINT *
        X=X+1
:*****
:DATA TEST USING JUMPER CONNECTOR.
:USES SPECIAL BINARY COUNT PATTERN FOR DATA. NO INTERRUPT.
APBA: JSR     7,INBIN      : INITIALIZE BINARY COUNT PATTERN
APBB: MOV     #1000.,CTRO   : SET CHARACTER COUNT TO 1000
        TIMETX      : TIME OUT TX DONE
        ERROR      : ERROR DONE NOT SETTING
        JSR     7,GTBINP    : GET BINARY CHARACTER
        MOV     %1,CRBUFA   : SAVE CHAR IN CRBUFA AND
        JSR     7,MASKIT    : MASK OFF NON TRANSMITTED BITS
        MOV     %1,RTXBUF   : LOAD CHAR.
        TIMERX      : TIME OUT RX DONE
        ERROR      : ERROR DONE NOT SETTING
        MOV     DRXBUF,CRBUF : LOAD RECEIVED DATA INTO CRBUF
        DATCHK      : CHECK DATA
        DEC     CTRO        : TESTED 1000 CHARACTERS
        BNE     APBB        : BRANCH IF NOT
        SCOPE      : YES. SCOPE
:*****
AT57: 57         : TEST NUMBER *
        AT60      : ADDRESS OF NEXT TEST *

```

```

3.          ; ITERATION COUNT *
EXTA       ; SCOPE ENTRY POINT *
X=X+1     ; *
:*****
:TEST THAT RDR BUSY TURNS OFF RDR ENABLE
:WHEN RUN ON AN XOR TESTER
EXTA:  RESET          ; RESET
      INC            @RXCSR      ; SET RDR ENABLE, SEE IF RDE IS TURNED OFF BY RDR BUSY
      MOV            #-10,3$+2
2$:    INC            3$+2      ; WAIT LOOP FOR XOR TESTER
      BNE            2$
      CLR            @TXBUF      ; SHIP OUT CHAR.
      MOV            #-50000,3$+2
5$:    TSTB          @RXCSR      ; TEST COMPLETE
      BMI            6$
      INC            #-10        ; ALLOW TIME FOR RDR DONE TO SET
      BNE            5$
      ERROR          ; FAILURE OF RDR DONE TO SET
6$:    SCOPE
:*****
AT60:    60           ; TEST NUMBER *
      AT61           ; ADDRESS OF NEXT TEST *
      10.           ; ITERATION COUNT *
      EXA           ; SCOPE ENTRY POINT *
      X=X+1
:*****
:TEST THAT WHEN RDR ENABLE IS SET THAT THE RXCSR DONE
:BIT IS CLEARED
EXA:    RESET
      JSR            PC,STRXD    ; SET RCVR DONE
      INC            @RXCSR      ; SET ENABLE
      TSTB          @RXCSR      ; DONE SHOULD CLEAR
      BPL            1$
      ERROR          ; DONE NOT CLEAR
1$:    MOV            #-10,3$+2
3$:    INC            #-10
      BNE            3$
      SCOPE
:*****
AT61:    61           ; TEST NUMBER *
      AT62           ; ADDRESS OF NEXT TEST *
      3.            ; ITERATION COUNT *
      EXAA          ; SCOPE ENTRY POINT *
      X=X+1
:*****
EXAA:    TST            XORFLG    ; CHECKING JUMPER CONNECTIONS FOR XOR, RCVR
      BPL            3$
      MOV            #-1,@RXCSR
      TST            @RXCSR
      RESET
3$:    SCOPE
:*****
AT62:    62           ; TEST NUMBER *
      AT63           ; ADDRESS OF NEXT TEST *
      3.            ; ITERATION COUNT *

```

```

EXBA                                ;SCOPE ENTRY POINT *
X=X+1                                *
:*****
EXBA:  TST    XORFLG                ;SAME AS ABOVE BUT FOR XMTR
      BPL    4$
      MOV    #177677,@TXCSR
      TST    @TXCSR
      RESET
4$:    SCOPE
:*****
AT63:  100063                       ;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,@TXCSR          ;SET MAINTENANCE BIT IN TXCSR
      BIS    #BIT0,@TXCSR          ;SET BREAK BIT
      MOV    #252,@TXBUF           ;LOAD BUFFER
      TIMERX                          ;TIME OUT RX DONE
      ERROR                          ;ERROR DONE NOT SETTING
      CMPB   @RXBUF,#0              ;CHARACTER RECEIVED SHOULD BE 0
      BEQ    .+4
      ERROR                          ;CHARACTER OTHER THAN 0
      SRESET                         ;ISSUE RESET
      SCOPE
  
```

```

:SUBROUTINE TO SET RXCSR DONE BIT.
STRXD:  BIS      #BIT2, @TXCSR      :SET MAINTENANCE BIT.
        CLR      @TXBUF            :LOAD TXBUF.
        TIMERX                      :TIME OUT TX DONE
        ERROR                      :ERROR DONE NOT SETTING
        RTS      %7                :EXIT.
:SUBROUTINE TO CHECK THAT CTR0 THROUGH CTR3 CONTAIN DESCENDING VALUES.
CMPT:   CMP      CTR0, CTR1
        BLOS     CMPTNG
        CMP      CTR1, CTR2
        BLOS     CMPTNG
        CMP      CTR2, CTR3
        BLOS     CMPTNG
        CMP      CTR3, CTR4
        BLOS     CMPTNG
        CMP      CTR4, CTR5
        BLOS     CMPTNG
        CMP      CTR5, CTR6
        BLOS     CMPTNG
        CMP      CTR6, CTR7
        BHI      CMPTOK
CMPTNG: ADD      #2, @%6
CMPTOK: RTS      %7

```

```

:*****
:PRG1 - TRANSMITTER SCOPE LOOP
:*****
PRG1:  TYPE                ;TYPE PROGRAM TITLE.
      PITIT
      JSR      5,LINSLX    ;GO GET LINE # FROM USER
      TYPE                ;TYPE SELECT CHAR AND DELAY.
      SELCAD
      HALT
PRG1A: MOVB      @SRPTR,PRG1B ;WAIT FOR USER.
      MOVB      @SRPTRH,@TXBUF ;DELAY COUNT TO PRG1B.
      DELAY                    ;LOAD TXBUF.
      OPEN                    ;DELAY # OF MSECS. SET AT SR.
PRG1B: BR      PRG1A        ;REPEAT.
:*****
:PRG2 - RECEIVER SCOPE LOOP.
:*****
PRG2:  TYPE                ;TYPE PROGRAM TITLE.
      P2TIT
      JSR      5,LINSLX    ;GO GET LINE # FROM USER
      TYPE                ;TYPE SELECT CHAR AND DELAY.
      SELCAD
      HALT
PRG2A: BIS      #BIT2,@TXCSR ;WAIT FOR USER.
      MOVB      @SRPTR,PRG2B ;SET MAINTENANCE BIT.
      MOVB      @SRPTRH,@TXBUF ;DELAY COUNT TO PRG2B.
      DELAY                    ;LOAD TXBUF.
      OPEN                    ;DELAY # OF MSECS. SET IN SR.
PRG2B: MOV      @RXBUF,%0    ;RXBUF CONTENTS TO RD.
      RESET                    ;DISPLAY CONTENTS OF RXBUF (IN RD).
      RESET                    ;BY ISSUING 5 RESET INSTRUCTIONS
      RESET
      RESET
      RESET
      BR      PRG2A

```

```

:*****
:PRG3 - SINGLE CHARACTER MAINTENANCE MODE DATA TEST.
:*****
PRG3:  TYPE                ;TYPE PROGRAM TITLE.
      P3TIT
      JSR      5,LINSLX    ;GO GET LINE # FROM USER
      TYPE                ;TYPE: SELECT CHARACTER.
      SELCAR
      HALT
PRG3A: MOVB      @SRPTRH,CRBUFA ;MOVE DATA CHAR TO CRBUFA.
      JSR      %7,MOUTIN    ;GO OUTPUT, RECEIVE, AND CHECK DATA.
      BR       PRG3A
:*****
:PRG4 - SPECIAL BINARY COUNT MAINTENANCE MODE DATA TEST.
:*****
PRG4:  TYPE                ;TYPE PROGRAM TITLE.
      P4TIT
      JSR      5,LINSLX    ;GO GET LINE # FROM USER
      JSR      %7,INBIN    ;INITIALIZE BINARY COUNT.
PRG4A: JSR      %7,GTBINP   ;GET BINARY CHARACTER.
      MOVB     %1,CRBUFA   ;SAVE AT CRBUFA.
      JSR      %7,MOUTIN   ;GO OUTPUT, RECEIVE, AND CHECK DATA.
      BR       PRG4A      ;REPEAT.
;SUBROUTINE TO OUTPUT, RECEIVE, AND CHECK DATA WITH MAINTENANCE BIT SET.
MOUTIN: BIT      #BIT7,@SRPTR ;SEE IF BIT 7 IS SET.
      BNE     .+4         ;BRANCH IF SET.
      STALL    ;SET. DO A RANDOM STALL.
      TIMETX   ;TIME OUT TX DONE
      ERROR    ;ERROR DONE NOT SETTING
      BIS      #BIT2,@TXCSR ;SET MAINTENANCE BIT.
      MOV      CRBUFA,@TXBUF ;LOAD TXBUF.
      JSR      7,MASKIT    ;MASK OFF NON TRANSMITTED BITS
      TIMERX   ;TIME OUT RX DONE
      ERROR    ;ERROR DONE NOT SETTING
      MOV      @RXBUF,CRBUF ;MOVE CHAR IN RX BUFFER TO CRBUF.
      DATCHK   ;COMPARE EXPECTED AND RECEIVED DATA
      RTS      %7         ;EXIT.

```

;ASCII MESSAGES

MTIT: .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 '%T'  
ATNUMB: .ASCII ' PC= '

APC: .ASCII ' RXCSR= '

MRXNUM: .ASCII ' %'

POTIT: .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 ' %'

ARXCSR: .ASCII 'RXCSR S/B: '



AINCRT: .ASCII '%INCORRECT ROUTINE SELECTED, PLACE CORRECT PROGRAM'

.ASCII '%IN 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.␣'

LDLINE: .ASCII '%LOAD LINE NO. INTO SR Q-23'

ALINE: .ASCII ' LINE NO.'

SELIN: .ASCII ' WAS SELECTED'

MSETRX: .ASCII '%RECEIVER SPEED CHECK'

MSETTX: .ASCII '%TRANSMIT SPEED CHECK'

MSETC: .ASCII '%SET CLOCK SWITCHES TO POSITION, THEN PRESS CONTINUE.'

MTERR: .ASCII '%ERROR - UNEXPECTED TRAP'

.ASCII '%TRAPPED TO '

MTO: .ASCII ' '

.ASCII '%TRAPPED FROM PC '

MFROM: .ASCII ' '

MNOLIN: .ASCII '%NO DEVICE PRESENT - THIS LINE NO.'

INTER: .ASCII '%NO INTERRUPT'

MSD: .ASCII '%CS = 0'

MSI: .ASCII '%CS = 1'





.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 70  
 ODDLAA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

AINCPG	016065	1449	3158#							
AINCRT	015744	1445	3143#							
AJA	007046	1830	1834#							
AJB	007070	1836	1839#							
ALBA	010266	2113	2118#							
ALBB	010310	2120	2123#							
ALBC	010332	2125	2128#							
ALBD	010352	2122	2127	2130	2132#					
ALC	007112	1841	1844#							
ALD	007132	1838	1843	1846	1848#					
AKA	007154	1855	1860#							
AKB	007166	1861	1864#							
AKBA	010364	2137	2142#							
AKBC	010412	2145	2147#							
ALA	007200	1869	1873#							
ALAA	011636	2394	2401#							
ALAB	011666	2406	2409#							
ALAC	011700	2408	2411	2413#						
ALBA	010542	2179	2184#							
ALBB	010564	2186	2189#							
ALBC	010606	2191	2194#							
ALBD	010626	2188	2193	2196	2198#					
ALINE	016442	1315	3204#							
ALY	007230	1876	1879#							
ALZ	007250	1878	1881	1883#						
AMAA	011712	2418	2423#							
AMAC	011744	2429	2431#							
AMBA	010640	2203	2207#							
AMBB	010670	2210	2213#							
AMBC	010704	2214	2217#							
AMBD	010726	2219	2222#							
AMBE	010746	2212	2216	2221	2224	2226#				
AOAA	011762	2436	2442#							
AOAB	012000	2445#	2449							
AOAB1	012014	2446	2448#							
AOAC	012036	2451	2454#							
AOAD	012050	2453	2455	2457#						
AOBA	010444	2156	2160#							
AOBB	010466	2162	2165#							
AOBC	010510	2167	2170#							
AOBD	010530	2164	2169	2172	2174#					
APA	007262	1888	1892#							
APB	007276	1893	1896#							
APBA	013652	2823	2829#							
APBB	013660	2830#	2841							
APC	015015	1573	3054#							
APCNT	016135	1468	3167#							
APCX	007320	1898	1901#							
APD	007340	1895	1900	1903	1905#					
APGEND	016120	1483	3163#							
APRXC	016171	1476	3173#							
APVEC	016212	1480	3176#							
QQA	007362	1912	1916#							
QQA	012066	2463	2471#							
QQA	012312	2516	2519#							
QQA	012314	2478	2483	2488	2493	2498	2503	2508	2513	2521#



AT35	011626	2315	2391#
AT36	011702	2392	2415#
AT37	011746	2416	2433#
AT4	006636	1755	1778#
AT40	012056	2434	2460#
AT41	012360	2461	2533#
AT42	012524	2534	2571#
AT43	012612	2572	2593#
AT44	012670	2594	2613#
AT45	012754	2614	2634#
AT46	013052	2635	2660#
AT47	013136	2661	2682#
AT5	006736	1779	1803#
AT50	013220	2683	2704#
AT51	013302	2705	2725#
AT52	013372	2726	2747#
AT53	013470	2748	2773#
AT54	013530	2774	2788#
AT55	013614	2789	2809#
AT56	013636	2810	2820#
AT57	013730	2821	2844#
AT6	007036	1804	1827#
AT60	014014	2845	2867#
AT61	014064	2868	2886#
AT62	014120	2887	2899#
AT63	014154	2900	2912#
AT7	007144	1828	1852#
AUAA	012626	2596	2603#
AUAB	012652	2607#	2611
AUAC	012662	2602	2609#
AVAA	012704	2616	2623#
AVAB	012742	2622	2630#
AVAC	012744	2629	2631#
AWAA	012764	2637	2642#
AWAB	013014	2649#	2655
AWAC	013024	2643	2651#
AWAD	013040	2652	2654#
AWAE	013044	2651	2656#
AWAS	015623	1542	3127#
AXA	007724	1999	2003#
AXAA	013076	2663	2672#
AXAB	013124	2670	2678#
AXAC	013126	2677	2679#
AXB	007740	2004	2007#
AX1A	013156	2685	2693#
AX1B	013206	2692	2700#
AX1C	013210	2699	2701#
AYA	007752	2012	2016#
AYAA	013240	2707	2715#
AYAB	013264	2719#	2723
AYAC	013274	2713	2721#
AYB	007766	2017	2020#
AZA	010000	2025	2029#
AZAA	013322	2728	2736#
AZAB	013360	2734	2743#
AZAC	013362	2742	2744#

2658











RSTREG=	104014	723#	1368	1380	1401									
RSTRG	003060	879	1115#											
RTNNO	001456	852#	1006*	1025	1031	1056	1082*	1569	1580	1587	2524	2542	2551	2557
RXBUF	001412	832#	1077	1174*	1433	1533	1634*	1744	2427	2457				
		2561	2785	2838	2926	2984	3026							
RXBUFT	001610	898#	2542*	2543	2547									
RXCRO	001200	753#	961	984	988	1293	1496							
RXCR1	001202	754#												
RXCR10	001220	761#												
RXCR11	001222	762#												
RXCR12	001224	763#												
RXCR13	001226	764#												
RXCR14	001230	765#												
RXCR15	001232	766#												
RXCR16	001234	767#												
RXCR17	001236	768#												
RXCR2	001204	755#												
RXCR20	001240	769#												
RXCR21	001242	770#												
RXCR22	001244	771#												
RXCR23	001246	772#												
RXCR24	001250	773#												
RXCR25	001252	774#												
RXCR26	001254	775#												
RXCR27	001256	776#												
RXCR3	001206	756#												
RXCR30	001260	777#												
RXCR31	001262	778#												
RXCR32	001264	779#												
RXCR33	001266	780#												
RXCR34	001270	781#												
RXCR35	001272	782#												
RXCR36	001274	783#												
RXCR4	001210	757#												
RXCR5	001212	758#												
RXCR6	001214	759#												
RXCR7	001216	760#												
RXCSR	001410	831#	1173*	1265	1422*	1431	1455	1475	1576	1589	1632*	1728*	1873*	1874*
		1875	1879*	1880	1892	1896*	1897	1901*	1902	1905*	1916	1920*	1921	1925*
		1926	1929*	1941	1945*	1946	1950*	1951	1954*	1966	1970*	1971	1975*	1976
		1979*	1990	2003	2016	2029	2042	2062*	2063	2067*	2068	2081*	2082	2083
		2087*	2090	2095	2100*	2103	2118*	2119	2123*	2124	2128*	2129	2142*	2143*
		2144	2147*	2148	2160*	2161	2165*	2166	2170*	2171	2184*	2185	2189*	2190
		2194*	2195	2207*	2208*	2209	2213	2217*	2218	2222*	2223	2263*	2264*	2273
		2287*	2405	2410	2428	2445	2448	2454	2528	2672*	2674*	2679*	2693*	2695*
		2701*	2715*	2717*	2719*	2736*	2739*	2744*	2758*	2759*	2762*	2781	2782	2799*
		2801*	2806*	2854*	2860	2877*	2878	2894*	2895					
		897#	2082*	2090*	2091	2103*	2104	2273*	2274	2282	2781*			
RXCSRT	001606	785#	962	994										
RXEND	001300	836#	1134	1177*	1647*	2716	2737							
RXLVL	001422	835#	1132	1175*	1306	1456	1479	1627*	1672*	1679	2764*			
RXVTR	001420	722#	1353	1373	1384									
SAVREG=	104013	878	1100#											
SAVRG	003020	721#	1438	1729	1745	1759	1773	1801	1825	1850	1864	1883	1907	1931
SCOPE =	104012	1956	1981	1994	2007	2020	2033	2046	2071	2108	2132	2151	2174	2198
		2226	2252	2288	2312	2370	2413	2431	2458	2519	2546	2550	2555	2560



TKVTR	001440	845#																		
TMRX	003672	882	1265#																	
TMTX	003702	883	1267#																	
TNNO	001460	853#	1569*	1570*																
TOPC	001626	905#	1603*	1606	1619*	1622*	1625	1626*	1627											
TPB	001436	844#	1195*																	
TPLVL	001446	848#																		
TPS	001434	843#	1196																	
TPVTR	001444	847#																		
TSTVA	006346	1680	1687#																	
TSTVEC	006244	1129	1138	1669#	1682	1686														
TSTVEX	006366	1670	1690#																	
TXBUF	001416	834#	1171*	1430*	1638*	1772	2266*	2269*	2298*	2301*	2375*	2378*	2402*	2424*						
		2443*	2525*	2566*	2835*	2858*	2923*	2934*	2966*	2981*	3022*									
TXCSR	001414	823#	1074*	1172*	1267	1302*	1303*	1308*	1423*	1425	1636*	1675*	1676*	1687*						
		1758	1786	1790*	1791	1795*	1796	1799*	1810	1814*	1815	1819*	1820	1823*						
		1935	1839*	1840	1844*	1845	1848*	1860	2236*	2238	2240	2265*	2302	2308						
		2373	2376	2381	2401*	2423*	2442*	2522	2565*	2582*	2584*	2587*	2589*	2604*						
		2605*	2607*	2623*	2626*	2631*	2644*	2646*	2649*	2907*	2908	2921*	2922*	2933*						
		2979*	3021*																	
TXCSRT	001604	896#	2240*	2247																
TXLVL	001426	838#	1143	1178*	1648*	2603	2624													
TXVTR	001424	837#	1141	1176*	1625*	2651*														
TYP	003342	867	1182#																	
TYPA	003352	1185#	1194	1203																
TYPC	003370	1187	1189#																	
TYPD	003416	1193	1195#	1200	1202															
TYPDAT	003462	1185*	1186	1189	1191	1195	1199*	1201*	1204#											
TYPE =	104000	711#	929	956	977	996	1213	1285	1296	1314	1440	1444	1448	1482						
		1613	1663	1683	1712	2331	2336	2341	2346	2351	2356	2361	2480	2485						
		2490	2495	2500	2505	2510	2559	2962	2973	2976	2995	2998	3007							
TYPES =	104001	712#	1583	2323	2472															
TYPF	003434	1190	1199#																	
TYPG	003446	1192	1201#																	
TYPS	003464	868	1207#	1215																
TYPSA	003510	1211	1213#																	
TYPSB	003512	1209*	1210	1214#																
UMASK	001402	827#	1023	1072	1526	1643*														
X =	000063	1718#	1719	1724#	1733	1738#	1749	1754#	1763	1768#	1777	1782#	1802	1807#						
		1826	1831#	1851	1856#	1865	1870#	1884	1889#	1908	1913#	1932	1937#	1957						
		1962#	1982	1987#	1995	2000#	2008	2013#	2021	2026#	2034	2039#	2053	2058#						
		2072	2077#	2109	2114#	2133	2138#	2152	2157#	2175	2180#	2199	2204#	2227						
		2232#	2253	2258#	2289	2294#	2313	2318#	2390	2395#	2414	2419#	2432	2437#						
		2459	2464#	2532	2537#	2570	2575#	2592	2597#	2612	2617#	2633	2638#	2659						
		2664#	2681	2686#	2703	2708#	2724	2729#	2746	2751#	2772	2777#	2787	2793#						
		2808	2813#	2819	2824#	2843	2848#	2866	2871#	2885	2890#	2898	2903#	2911						
		2916#																		
XOR	002676	1046	1066#																	
XORA	002026	924	940#																	
XORADD	001276	784#	927*	942*																
XORFLG =	002024	928#	938#	943*	1043	1742	2892	2905												
	017430	422#	439	441#	443	445	447	449	451	453	455	457	459	461						
		463	465	467	469	471	473	475	477	479	481	483	485	487						
		489	491	493	495	497	499	501	503	505	507	509	511	513						
		515	517	519	521	523	525	527	529	531	533	535	537	539						
		541	543	545	547	549	551	553	555	557	559	561	563	565						



TSTA	1947	1719	1733	1749	1763	1777	1802	1826	1851	1865	1884	1908	1932	1957	1982
	2008	2021	2034	2053	2072	2109	2133	2152	2175	2199	2227	2253	2289	2313	2338
	2414	2432	2459	2532	2570	2592	2612	2633	2659	2681	2703	2724	2746	2772	2800
TSTA	1947	1719	1733	1749	1763	1777	1802	1826	1851	1865	1884	1908	1932	1957	1982
	2008	2021	2034	2053	2072	2109	2133	2152	2175	2199	2227	2253	2289	2313	2338
	2414	2432	2459	2532	2570	2592	2612	2633	2659	2681	2703	2724	2746	2772	2800

ADD	1094	1131	1140	1157	1161	1183	1208	1223	1231	1273	1357	1360	1408	1409	1494
ASL	1591	1633	1635	1637	1641	2953									
ASR	1010	1092	1291	1493	1640	1645	1694								
BFS	989	1514	1599	1600	1601										
BFS	1405														
BEQ	963	983	999	1040	1055	1190	1192	1225	1233	1249	1258	1270	1295	1307	1462
	1489	1503	1510	1537	1624	1787	1797	1811	1821	1836	1846	1881	1893	1903	1917
	1927	1942	1952	1967	1977	1991	2004	2017	2030	2043	2069	2084	2097	2120	2130
	2149	2162	2172	2186	2196	2214	2239	2275	2446	2455	2558	2562	2927		
BHI	2952														
BIC	964	1008	1030	1057	1093	1248	1257	1289	1302	1308	1338	1347	1359	1362	1422
	1527	1529	1570	1602	1646	1675	1687	1795	1819	1844	1873	1879	1901	1925	1950
	1975	2067	2081	2100	2118	2128	2142	2147	2160	2170	2184	2194	2207	2217	2263
	2287	2582	2587	2589	2604	2607	2623	2631	2644	2649	2672	2679	2693	2701	2715
	2719	2736	2744	2758	2762	2799	2806								
BIS	969	1074	1301	1303	1423	1674	1676	1790	1799	1814	1823	1839	1848	1874	1896
	1905	1920	1929	1945	1954	1970	1979	2062	2087	2123	2143	2165	2189	2208	2222
	2265	2401	2423	2442	2565	2584	2605	2626	2646	2674	2695	2717	2739	2759	2801
	2921	2922	2933	2979	3021										
BIT	985	1021	1039	1049	1054	1059	1279	1294	1461	1464	1488	1509	1512	1534	1565
	1653	1681	1786	1791	1796	1810	1815	1820	1835	1840	1845	1875	1880	1892	1897
	1902	1916	1921	1926	1941	1946	1951	1966	1971	1976	1990	2003	2016	2029	2042
	2063	2068	2083	2091	2095	2104	2119	2124	2129	2144	2148	2161	2166	2171	2185
	2190	2195	2209	2213	2218	2223	2308	2445	2454	2543	2551	2557	2561	3016	
BLOS	2940	2942	2944	2946	2948	2950									
BMT	1743	1861	2406	2548	2783	2861									
BNE	955	986	995	1022	1032	1035	1050	1052	1060	1062	1197	1211	1237	1240	1280
	1336	1345	1367	1379	1394	1436	1460	1465	1498	1513	1535	1566	1654	1670	1680
	1682	1711	1792	1816	1841	1876	1898	1922	1947	1972	2064	2092	2105	2125	2145
	2167	2191	2210	2219	2224	2309	2387	2451	2544	2553	2841	2857	2863	2883	3017
BPL	1024	1026	1044	1073	1197	1272	1426	1432	1521	2303	2374	2377	2382	2411	2429
	2449	2523	2529	2879	2893	2906									
BR	914	968	971	980	992	1028	1037	1064	1068	1194	1203	1215	1226	1266	1298
	1407	1491	1560	1666	1732	1748	1762	1776	1789	1794	1813	1818	1838	1843	1878
	1895	1900	1919	1924	1944	1949	1969	1974	2066	2086	2094	2099	2107	2122	2127
	2164	2169	2188	2193	2212	2216	2221	2305	2367	2408	2453	2516	2591	2611	2629
	2655	2658	2677	2699	2723	2742	2768	2771	2804	2969	2990	3003	3014		
CLR	922	943	950	951	959	965	1001	1005	1006	1016	1019	1171	1172	1173	1174
	1175	1176	1177	1178	1234	1268	1281	1300	1403	1452	1500	1501	1559	1628	1672
	1673	1728	2266	2298	2301	2372	2375	2378	2402	2424	2443	2444	2521	2525	2566
	2583	2645	2673	2694	2800	2858	2934								
CMP	917	940	962	994	1034	1061	1066	1210	1224	1497	1536	1669	2238	2274	2450
	2939	2941	2943	2945	2947	2949	2951								
CMPB	1031	1186	1189	1191	2926										
COM	1148	1334	1335	1343	1344	1528									
DEC	1051	1236	1239	1366	1378	1393	1435	1459	2386	2840					
EMT	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725
	726	727													
HALT	440	742	1002	1058	1287	1442	1446	1450	1522	1615	1659	1714	2328	2333	2338
	2343	2348	2353	2358	2363	2477	2482	2487	2492	2497	2502	2507	2512	2964	2978
	3000														
INC	921	958	979	1269	1337	1346	1406	1490	1499	1689	2380	2447	2527	2854	2856
	2862	2877	2882												
JMP	737	739	931	944	1000	1003	1011	1027	1033	1096	1451	1484	1616	1655	1686
	1695	1717													
JSR	952	953	973	991	1020	1036	1063	1129	1138	1193	1200	1202	1247	1256	1283

	1284	1299	1310	1320	1392	1397	1427	1429	1463	1466	1470	1474	1478	1505	1516
	1540	1544	1548	1571	1575	1579	1605	1609	1671	1715	1716	2242	2246	2277	2281
	2323	2329	2334	2339	2344	2349	2354	2359	2364	2366	2379	2441	2471	2478	2483
	2488	2493	2498	2503	2508	2513	2515	2526	2540	2541	2556	2579	2668	2671	2690
	2714	2735	2755	2780	2797	2798	2816	2817	2828	2832	2834	2876	2920	2961	2975
	3002	3009	3010	3010	3011	3013	3023								
MOV	908	909	910	911	915	918	919	920	923	924	926	927	928	941	942
	948	949	960	961	972	981	984	987	990	1004	1007	1009	1013	1014	1015
	1017	1029	1041	1045	1046	1048	1056	1067	1081	1082	1083	1084	1085	1086	1089
	1091	1095	1100	1101	1102	1103	1104	1105	1106	1107	1108	1115	1116	1117	1118
	1119	1120	1121	1122	1123	1130	1132	1133	1134	1139	1141	1142	1143	1147	1149
	1154	1158	1164	1165	1182	1184	1207	1209	1218	1219	1220	1221	1222	1230	1232
	1235	1250	1259	1265	1267	1288	1290	1292	1293	1309	1319	1333	1339	1340	1342
	1348	1349	1354	1355	1356	1358	1374	1375	1376	1385	1386	1387	1388	1389	1390
	1391	1396	1424	1453	1454	1455	1456	1457	1492	1495	1496	1502	1515	1526	1533
	1538	1539	1557	1558	1561	1562	1563	1564	1567	1569	1587	1588	1589	1597	1598
	1603	1604	1619	1621	1625	1627	1632	1634	1636	1638	1639	1642	1643	1647	1648
	1658	1660	1662	1688	1693	1709	1727	1741	1757	1771	1834	1940	1965	2082	2090
	2103	2235	2236	2240	2241	2262	2264	2269	2273	2276	2330	2335	2340	2345	2350
	2355	2360	2365	2385	2479	2484	2489	2494	2499	2504	2509	2514	2542	2603	2624
	2651	2652	2716	2737	2764	2765	2781	2829	2855	2859	2881	2894	2907	2923	2984
	3022	3026													
MOVE	1185	1195	1199	1201	1361	1377	1410	1428	1430	1433	2833	2835	2838	2965	2966
	2980	2981	3001	3012											
NOP	967	1304	1305	1506	1507	1508	1677	1678	2585	2606	2627	2647	2654	2675	2718
	2740	2760	2767	2802											
RESET	1150	1458	1504	1661	2853	2875	2996	2909	2985	2986	2987	2988	2989		
ROL	1155	1156	1159	1160	1162	1163									
ROR	1363	1364	1365												
RTI	1042	1109	1124	1135	1144	1151	1188	1212	1242	1253	1274	1523	1554	1592	1629
	2653	2766													
RTS	745	1078	1087	1166	1179	1198	1227	1260	1282	1316	1324	1341	1350	1369	1381
	1402	1411	1443	1447	1511	1517	1530	1649	1656	1690	2383	2388	2530	2569	2937
	2954	3028													
SUB	988	1090	1395	1404	1568	1622	1626	2625	2738						
SWAB	1644														
TST	912	925	954	966	982	993	998	1023	1025	1043	1047	1072	1077	1306	1437
	1520	1623	1679	1710	1742	1744	1758	1772	2427	2457	2524	2547	2785	2892	2895
	2905	2908													
TSTB	1196	1271	1425	1431	1860	2302	2373	2376	2381	2405	2410	2428	2448	2522	2528
	2782	2860	2878												
.ASCII	932	3032	3040	3045	3048	3051	3052	3054	3057	3059	3065	3070	3076	3078	3081
	3083	3085	3088	3090	3096	3102	3108	3113	3123	3125	3127	3128	3130	3132	3139
	3143	3152	3158	3164	3167	3170	3173	3176	3178	3185	3193	3199	3204	3206	3209
	3213	3217	3226	3230	3233	3235	3239	3241	3247	3250	3252	3254	3256	3258	3260
	3262	3264	3266												
.BYTE	1420	3163													
.FNABL	4	396													
.FNND	3286														
.FVFN	937	3272													
.LIST	4	396	731												
.MACR	947														
.MLIST	4	396													
.REM															
.REPT	44														
.WORD	938	1486	3274	3275	3276	3277	3278	3279	3280	3281	3282	3283	3284	3285	

G07

.MAIN. MACY11 27(732) 10-SEP-76 09:54 PAGE 87  
DDDLAA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

\*DDDLAA,DDDLAA,SEQ/SOL/CRF/DS:ERFZ/EN:ABS=DSKM:DDDLAA.P11  
RUN-TIME: 10 16 5 SECONDS  
RUN-TIME RATIO: 50/34=1.4  
CORE USED: 12K (23 PAGES)