.REM %

IDENTIFICATION

PRODUCT CODE: AC-E188B-MC

PRODUCT NAME: CVDLAB0 DLV11-J TEST

PRODUCT DATE: NOV 1978

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).
1.2 SYSTEM REQUIREMENTS.
1.3 RELATED DOCUMENTS AND STANDARDS.
1.4 DIAGNOSTIC HIERARCHY PREREQUISITES.
1.5 ASSUMPTIONS.

2.0 OPERATING INSTRUCTIONS.

2.1 LOADING AND STARTING PROCEDURES.
2.2 SPECIAL ENVIRONMENTS.
2.3 OPERATIONAL SWITCH SETTINGS
2.4 PROGRAM OPTIONS.
2.5 EXECUTION TIMES.
2.6 POWER FAIL.

3.0 ERROR INFORMATION.

3.1 ERROR REPORTING PROCEDURE.
3.2 ERROR HALTS.

4.0 PERFOMANCE AND PROGRESS REPORTS.

4.1 PERFORMANCE REPORTS.

5.0 DEVICE INFORMATION TABLES.

6.0 SUMMARY OF TESTS.

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

THIS DIAGNOSTIC IS A LOGIC TEST TO VERIFY THE OPERATION OF THE
DLV11-J SERIAL INTERFACE.
TESTING IS DONE IN TWO DISTINCT PHASES:

1. ALL SELECTED CHANNELS PER DLV11-J MODULE ARE TESTED INDIVIDUALLY

2. THE DLV11-J MODULE IS TESTED AS A WHOLE FOR CHANNEL INTERACTION
   PROBLEMS. THIS DIAGNOSTIC IS DESIGNED TO TEST AND DETECT
   ERRORS TO THE LOGIC LEVEL (NOT TO THE CHIP LEVEL).

THIS DIAGNOSTIC OPERATES UP TO 2 DLV11-J SERIAL LINE INTERFACES
CONFIGURED AT CONSECUTIVE BASE ADDRESSES.
THE PROGRAM WILL DO AUTO-SIZING IF THE DEVICE MAP '$DEVM' 0.
ALTERNATELY, THE OPERATOR CAN SELECT OR DESELECT INDIVIDUAL MODULES
& CHANNELS BY SETTING THE PROPER BITS IN '$DEVM' (SEE PROGRAM OPTIONS SEC. 2.4)
BEFORE RUNNING THE PROGRAM.
BY THIS METHOD, THE PROGRAM WILL BYPASS AUTO-SIZING AND TEST ONLY
THOSE MODULES/CHANNELS IT FINDS IN $DEVM.

IN EITHER CASE, THE PROGRAM WILL PRINT OUT ALL MODULES & CHANNELS
TO BE TESTED BEFORE PROCEEDING.

THE OPERATOR MUST INSTALL DATA WRAP AROUND CONNECTORS TO DO
DATA TESTING. TO BYPASS DATA TESTS, THE OPERATOR MUST MODIFY
'$USWR' (USER SWITCH REGISTER) , SEE PROGRAM OPTIONS SEC. 2.4)

THE DEFAULT ADDRESSES & VECTORS ARE AS FOLLOWS:

    177560 -CONSOLE INTERFACE DEVICE ADDRESS
    176500 -FIRST SERIAL CHANNEL ADDRESS OF UP TO 8 CONSECUTIVE
            SERIAL LINE DEVICES.

     60 - VECTOR FOR CONSOLE DEVICE INTERFACE.
    300 - VECTOR FOR FIRST OF 16 DEVICES.

THIS PROGRAM IS DESIGNED TO RUN ON ANY Q-BUS PDP-11 WITH 4K OF
MEMORY AND A DLV11-J (Q-BUS) MODULE. IT CAN RUN UNDER XXDP &
APT MONITORS, AND ON PROCESSORS WITH NO HARDWARE
SWITCH REGISTER.

1.2 SYSTEM REQUIREMENTS.

HARDWARE REQUIREMENTS:

ANY Q-BUS PDP-11 FAMILY PROCESSOR
4K MEMORY - MINIMUM
A SPECIAL DATA WRAP AROUND CONNECTOR OR EQUIVALENT
(REQ'D IF DATA WRAP AROUND TESTS DESIRED)

IF CHANNEL 3 IS CONFIGURED AS THE CONSOLE:
TESTS 6-12, 14-17, 21, 22 ARE BYPASSED.

IF DATA WRAP AROUND TESTS ARE BYPASSED:
TESTS 7-12, 14-17, 21, 22 ARE BYPASSED.

IF CHANNEL 3 IS CONFIGURED AS THE CONSOLE & A VT-100 IS
THE CONSOLE DEVICE, SWITCH THE VT-100 TO ''JUMP SCROLL'' MODE
TO AVOID LOOSING CHARACTERS ON THE SCREEN.

SOFTWARE REQUIREMENTS:

THIS DIAGNOSTIC IS DESIGNED TO RUN IN ANY OF THE
FOLLOWING WAYS:
STAND ALONE
WITH APT MONITOR
WITH XXDP MONITOR (CHAINABLE IF RENAMED TO .BIC EXTENSION)

THIS DIAGNOSTIC IS NOT DESIGNED TO RUN WITH THE
DIAGNOSTIC SUPERVISOR.

1.3 RELATED DOCUMENTS AND STANDARDS.

DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS	175-003-009-02
APT	MD-11-DZZMA
SYSMAC	MD-11-DZQAC

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES.
NO SPECIAL DIACNOSTICS ARE REQUIRED TO RUN BEFORE THIS, BUT
THE PROCESSOR, MEMORY, AND BUS ARE ASSUMED TO BE FULLY
OPERATIONAL.

1.5 ASSUMPTIONS.

THIS DIAGNOSTIC ASSUMES THAT THE OPERATOR HAS INITIALIZED
LOCATION '$USWR' AND '$DEVM' TO THE PROPER VALUES. (SEE SEC. 2.4)

2.0 OPERATING INSTRUCTIONS.

2.1 LOADING AND STARTING PROCEDURES.

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED MEDIA.

THERE ARE 2 STARTING LOCATIONS FOR THIS DIAGNOSTIC:

1. LOCATION 200 FOR ALL NORMAL TESTING STARTS & RESTARTS.

2. LOCATION 400 FOR INTERRUPT VECTOR DEBUG... OFFLINE ONLY.

THE 400 START SHOULD BE USED ONLY TO DEBUG FAULTY VECTOR ADDRESSES PUT OUT BY THE DV11-J DURING INTERRUPTS.
STARTING AT 400 WILL CAUSE THE ENTIRE VECTOR AREA, EXCEPT THAT NEEDED BY SYSMAC, TO BE OVERWRITTEN & TO POINT TO A COMMON SERVICE ROUTINE TO ENABLE LOOPING ON THE PROBLEM.

AS SOON AS TESTING STARTS, THE OPERATOR CAN CHANGE THE SWITCH REGISTER ONLY BY A 'BREAK' & MANUALLY LOADING LOCATION 176 (SWREG) WITH THE DESIRED CONTENTS (SEE SEC. 2.3) THEN DOING A 'P' TO PROCEED.

IT IS IMPORTANT THAT WHEN GOING BACK TO NORMAL TESTING A FRESH LOAD OF THE PROGRAM BE PERFORMED TO RESET THE VECTOR AREA.

THE USER CAN SELECT A SPECIFIC TEST TO BE EXECUTED BY SETTING BIT 8 IN SWREG AND THE TEST NUMBER (IN OCTAL) IN BITS <7:0>.
(NOTE: ALL TESTS PREVIOUS TO THE SELECTED ONE ARE EXECUTED WITHOUT ITERATIONS.)

2.2 SPECIAL ENVIRONMENTS.

THIS DIAGNOSTIC FOLLOWS THE STANDARD PROCEDURE FOR RUNNING UDER APT,XXDP MONITORS, AS DESCRIBED IN THEIR RESPECTIVE PROCEDURES MANUAL AND SYSMAC PACKAGE.

2.3 OPERATIONAL SWITCH SETTINGS

THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED FOR ALL OPERATIONAL SWITCH SETTINGS.
THIS CAN BE ACCOMPLISHED IN THE FOLLOWING WAY:

1) TYPE CONTROL G <^G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE:' SWR=XXXXXX NEW=' (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE 'NEW=' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

   A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED). LEADING ZEROS NEED NOT BE TYPED, AND IF MORE THAN 6 DIGITS ARE TYPED THE LAST 6 WILL BE USED. IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

   B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 3.

   C) IF THE INPUT CHARACTER IS NOT ONE OF THE CHARACTERS MENTIONED ABOVE THEN A QUESTION MARK (?) WILL BE TYPED FOLLOWED BY A CARRAGE RETURN AND A LINE FEED SEQUENCE THEN PROCEED FROM STEP 3 (ERASING ALL PREVIOUS INPUT).

4) THE DIAGNOSTIC WILL CONTINUE ON TYPING <CR>.

NOTE: BECAUSE OF FREQUENT BUS RESETS IN THE PROGRAM, IT MAY BE NECESSARY TO DO 'CONTROL-G' SEVERAL TIMES. ALTERNATELY, A 'BREAK' & MANUALLY LOADING LOC. 176, FOLLOWED BY A 'P' TO PROCEED WILL ALSO WORK.

SOFTWARE SWITCH REGISTER OPTIONS (SWREG)

```
BIT 15 SET - 100000 - HALT ON ERROR
    14 SET -  40000 - LOOP ON TEST  (TO BE USED ONLY WHILE TESTING IN PROGRESS)
    13 SET -  20000   INHIBIT ERROR TYPEOUTS
    12 SET -  10000 - ENABLE PERFORMANCE REPORTS
    11 SET -   4000 - INHIBIT ITERATIONS
    10 SET =   2000   BELL ON ERROR
     9 SET -   1000 - LOOP ON ERROR
     8 SET      400   LOOP ON TEST IN SWR<7:0>
     7:0               NUMBER OF TEST TO LOOP ON (USED WITH BIT 8)
                         (ALL TESTS PREVIOIUS TO THE SELECTED TEST
                         ARE EXECUTED FIRST WITH 1 ITERATION ONLY)
```

2.4 PROGRAM OPTIONS.

THIS PROGRAM REQUIRES THE ADDRESS OF THE FIRST RCSR (STORED AT '$BASE') AND ITS INTERRUPT VECTOR (STORED AT '$VECT1'). IT WILL BE ABLE TO ADDRESS ANY DLV11-J STARTING AT THE SPECIFIED BASE ADDRESS THRU 2 CONSECUTIVE MODULES.
EXAMPLES: $BASE: 176500 (DEFAULT)
$VECT1: 300 (DEFAULT)

LOCATION '$DEVM' IS USED AS A BIT MAP TO INDICATE WHICH UNIT NUMBERS ARE PRESENT AND WILL BE TESTED.

$DEVM (BITS 15 THRU 8 NOT USED)

								MODULE 2				MODULE 1				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	BIT
								3	2	1	0	3	2	1	0	CHANNEL

A BIT MAP CAN BE ENTERED AT '$DEVM' PRIOR TO STARTING THE PROGRAM. IF '$DEVM' IS LEFT AT ITS DEFAULT VALUE (0), THE PROGRAM WILL DO AUTO-SIZING BEGINNING AT THE DEVICE ADDRESS IN '$BASE' & WILL SETUP '$DEVM' ACCORDINGLY. IN EITHER CASE, THE PROGRAM WILL TYPE OUT A LIST OF ALL THE MODULES/CHANNELS TO BE TESTED BEFORE PROCEEDING.

NOTE: SIZING IS PERFORMED ONLY ONCE AT THE BEGINNING OF THE PROGRAM. IF $DEVM IS TO BE CHANGED, THE PROGRAM MUST BE RESTARTED AT 200 FOR IT TO BE EFFECTIVE. OTHERWISE, $DEVM WILL BE IGNORED.

OPTIONS

LOCATION $USWR CONTAINS ALL THE USER SELECTABLE OPTIONS. THE VALUES IN THIS WORD MUST CONFORM TO THE ACTUAL BOARD CONFIGURATION.

BIT POSITION	DEFINITION	$USWR DEFAULT VALUE	
0	# OF DATA BITS TRANSMITTED 0 7 BITS, 1 8 BITS	1	= 8 BITS
1	PARITY ENABLED	0	- NO
2	EVEN ODD PARITY	0	ODD
3	BREAK DETECTION ENABLED	1	YES
4	RUN DATA WRAP AROUND TESTS	1	= YES
<11:9>	CONSOLE DEVICE	1	- YES - CONSOLE ON MODULE 1, CHAN. 3

<11:9> 0 CONSOLE NOT ON DLV11-J
<11:9> 1 CONSOLE ON MODULE 1
<11:9> 2 CONSOLE ON MODULE 2

IMPORTANT:

1. FOR DIAGNOSTIC PURPOSES, ALL CHANNELS MUST BE CONFIGURED
   TO THE SAME BIT/WORD LENGTH.

2. THE 'SIZING' ROUTINE WILL SET THE APPROPRIATE CHAN 3 BIT
   IN $DEVM IF BITS <11:9> ARE SET IN $USWR.

3. IF RUNNING UNDER APT & THE CONSOLE IS ON THE DLV11-J, THE
   CONSOLE MUST NOT BE TESTED.
   THIS IS BECAUSE APT SENDS 'BREAKS' TO THE CONSOLE WHICH
   INTEREFERES WITH THE TESTS.

   FOR THIS CASE:

   A. ALLOW THE $DEVM TO STAY EITHER AT ITS DEFAULT OF 0
      OR CLEAR THE APPROPRIATE CHAN. 3 BIT.

   B. CLEAR BITS <11:9> IN $USWR SO THE CONSOLE BIT WILL NOT
      BE SET INTO $DEVM FOR TESTING.

SUMMARY OF DEVICE ADDRESSES & VECTORS.

			DEV ADDRESS (RCSR)	VECTORS REC	VECTORS XMIT
MODULE #1	CHANNEL	0	$BASE+00	$VECT1+00	$VECT1+4
		1	+10	+10	+14
		2	+20	+20	+24
		3	+30	+30	+34
MODULE #2	CHANNEL	0	+40	+40	+44
		1	+50	+50	+54
		2	+60	+60	+64
		3	+70	+70	+74
CONSOLE DEVICE		3	$TKS	TKVEC-60	TPVEC-64

SUMMARY OF USER LOCATIONS & DEFAULTS.

	LOC	DEFAULT	
$BASE	1250	176500	
$VECT1	1244	300	
$TKS	1144	177560	FOR CONSOLE DEVICE ON MODULE UNDER TEST
TKVEC		60	FOR CONSOLE DEVICE ON MODULE UNDER TEST
TPVEC		64	FOR CONSOLE DEVICE ON MODULE UNDER TEST
$USWR	1220	1031	SEE SEC. 2.4
$DEVM	1252	0	SEE SEC. 2.4
SWREG	176	0	SEE SEC. 2.3

2.5 EXECUTION TIMES.

EXECUTION TIMES FOR AN LSI-11 PROCESSOR WITH ONE DLV11-J MODULE AT SHIPMENT CONFIGURATION:

CH. 0,1,2 AT 9600 BAUD.
CH. 3 (CONSOLE) AT 300 BAUD.

ARE: FIRST PASS- 30 SEC
ADDITIONAL PASSES 90 SEC

THE TEST TIME IS BAUD RATE DEPENDENT; HIGHER BAUD RATES RESULT IN SHORTER PASS TIMES.

2.6 POWER FAIL.

AUTO START FROM POWER FAIL IS NOT IMPLEMENTED IN THIS PROGRAM.

3.0 ERROR INFORMATION.

3.1 ERROR REPORTING PROCEDURE.

SINCE THIS DIAGNOSTIC WAS DESIGNED TO FIT IN 4K OF MEMORY THE ERROR TYPEOUT IS VERY BRIEF. THE FORMAT OF THE ERROR TYPEOUT IS AS FOLLOWS:

TEST#_____,ERROR#_____,PC-_____,ADDRESS=______,VECTOR-_____

WHERE ALL VALUES TYPED ARE OCTAL.
THE ADDRESS AND VECTOR REFER TO THE FAILING CHANNEL.
FOR FURTHER INFORMATION THE LISTING MUST BE CONSULTED.
BITS 15,13,10 AND 9 OF THE SWITCH REGISTER (SWREG) CONTROL THE SEQUENCE OF EVENTS AFTER AN ERROR IS CAUGHT.

BIT 15 SET: CAUSES THE PROGRAM TO HALT IN THE ERROR ROUTINE. IF THE PROGRAM IS CONTINUED, IT WILL PROCEED FROM WHERE IT HALTED.

BIT 13 SET: DISABLES THE PRINTING OF THE ERROR MESSAGE.

BIT 10 SET: CAUSES THE BELL TO RING ON ERROR.

BIT 9 SET: CAUSES THE DIAGNOSTIC TO LOOP FROM BEGINNING OF TEST TO ERROR.

THE ERROR ROUTINE SUPPORTS THE CONTROL G <^G> FUNCTION.
REFER TO SECTION 2.3 FOR DETAILS.

3.2 ERROR HALTS.

THE ONLY HALT IN THIS DIAGNOSTIC IS IN THE ERROR ROUTINE, AND IS EXECUTED ONLY IF BIT 15 OF THE SWITCH REGISTER (SWREG) IS SET WHEN AN ERROR OCCURS.

4.0 PERFORMANCE AND PROGRESS REPORTS.

4.1 PERFORMANCE REPORTS. (BIT 12 SET IN THE SWITCH REGISTER 'SWREG')

AS EACH CHANNEL (4 CHANNELS/DLV11-J) COMPLETES ONE PASS OF THE DIAGNOSTIC THE FOLLOWING ITEMS ARE TYPED:

'CSR:____' : THE BASE ADDRESS OF THE LINE UNDER TEST
'VECTOR:___' : THE ASSOCIATED VECTOR
'ERRORS:__' : THE TOTAL NUMBER OF ERRORS ON THIS DEVICE ON THIS PASS.

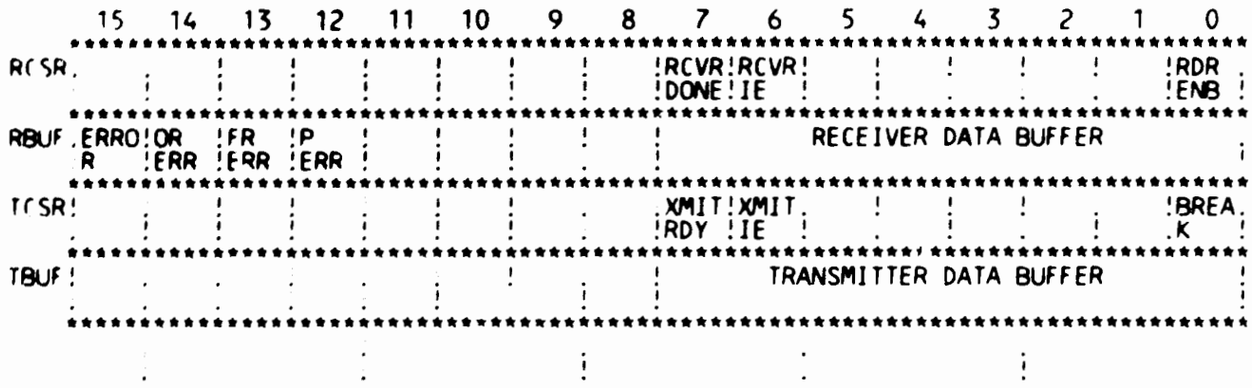
AFTER ALL MODULES & CHANNELS TO BE TESTED HAVE BEEN EXERCISED, AN END PASS STATEMENT IS TYPED:

'END PASS#_____.''

EXAMPLE OF PRINTOUT ASSUMING: 2 DLV11-J MODULES
ALL CHANNELS ENABLED
SEPERATE CONSOLE DEVICE
NO ERRORS

```
CVDLAB0 DLV11-J TEST

SWR- 000000     NEW- 10000<CR>                  (ENABLE PERFORMANCE REPORTS)

WILL TEST:

MODULE 1  CHANNEL 0  1  2  3
MODULE 2  CHANNEL 0  1  2  3

        ** PHASE 1 SUMMARY **
CSR: 176500, VECTOR: 000300, ERRORS: 0
CSR: 176510, VECTOR: 000310, ERRORS: 0
CSR: 176520, VECTOR: 000320, ERRORS: 0
CSR: 176530, VECTOR: 000330, ERRORS: 0

        ** PHASE 2 SUMMARY **
CSR: 176500, VECTOR: 000300, ERRORS: 0

        ** PHASE 1 SUMMARY **
CSR: 176540, VECTOR: 000340, ERRORS: 0
CSR: 176550, VECTOR: 000350, ERRORS: 0
CSR: 176560, VECTOR: 000360, ERRORS: 0
CSR: 176570, VECTOR: 000370, ERRORS: 0

        ** PHASE 2 SUMMARY **
CSR: 176540, VECTOR: 000340, ERRORS: 0

END PASS #      1
```

5.0 DEVICE REGISTERS.



NOTES:

1. RCSR AT BASE ADDRESS ($BASE)
   RBUF AT $BASE+2
   TCSR AT $BASE+4
   TBUF AT $BASE+6

2. BLANK BOXES INDICATE UNUSED AND RESERVED BIT POSITIONS. SEE THE LISTING FOR AN EXPLANATION OF THE BITS.

3. ORERR OVERRUN ERROR
   FRERR FRAMING ERROR
   PERR PARITY ERROR

N 1

6.0 SUMMARY OF TESTS AND SPECIAL SUBROUTINES.

PHASE 1 TESTS

TEST 1 ADDRESSABILITY

THIS TEST VERIFIES THAT ALL 4 REGISTERS OF THE CHANNEL UNDER TEST RESPOND TO THEIR ADDRESSES.

THE FOLLOWING 3 TESTS TEST ALL 'READ WRITE' BITS

TEST 2 BREAK - TCSR 0 SET, CLEAR, RESET

TEST 3 XMITIE - TCSR 6 SET, CLEAR, RESET

TEST 4 RCVRIE - RCSR 6 SET, CLEAR, RESET

TEST 5 XMITRDY - TCSR 7 - IS SET BY INIT

TEST 6 XMIT RDY - TCSR 7 - CLEARS WHEN TBUF IS LOADED WITH A CHARACTER AND THAT IT SETS WITHIN A REASONABLE AMOUNT OF TIME.

TEST 7 OUTPUTTING A CHAR FROM TBUF (WITH WRAP AROUND CONNECTED) RESULTS IN RCVRDONE SETTING WITHIN A REASONABLE AMOUNT OF TIME AND THAT RESET CLEARS THE BIT.

TEST 10 RCVRDONE IS CLEARED BY SETTING READER ENABLE

TEST 11 RCVRDONE IS CLEARED BY READING RBUF

TEST 12 OVERRUN & ERROR BIT - RBUF 14

TEST 13 TRANSMITTER INTERRUPT LOGIC TEST

LOGICALLLY THIS IS 4 SEPARATE TESTS
A) DOES TRANSMITTER INTERRUPT LOGIC WORK
B) AT PRIORITY OF 0
C) AND ONLY ONCE
D) BUT NOT WITH INTERRUPT ENABLE CLEAR

TEST 14 RECEIVER INTERRUPT LOGIC TEST THIS TEST COVERS ALL OF THE RECEIVER SIDE OF THE INTERRUPT LOGIC IN CHARACTER MODE.

TEST 15 TEST DATA WRAP AROUND: FLAG MODE.

TEST 16 TEST DATA WRAP AROUND: INTERRUPT MODE.

TEST 17 TEST BREAK DETECTION LOGIC TRANSMIT KNOWN CHAR

A) TRANSMIT KNOWN CHAR WITH BREAK SET AND COMPARE RECEIVED WITH 0

B) TEST FOR FRAMING ERROR ON BREAK

C) IF PARITY IS ENABLED AND ODD PARITY IS SELECTED, CHECK TO BE SURE PARITY ERROR WAS GENERATED

D) IF PARITY IS ENABLED AND EVEN PARITY IS SELECTED, CHECK TO BE SURE NO PARITY ERROR OCCURRED

TEST 20 NOT A TEST - SEND BACK TO LOOP

PHASE 2 TESTS

TEST 21 TEST THAT CHANNELS INTERRUPT AT ASSIGNED PRIORITY

TEST 22 TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING.

%

```
                        .TITLE  CVDLAA0 DLV11-J TEST
                        ;*COPYRIGHT (C) 1978
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY GREGORY GLEZMAN & GARY PAPAZIAN
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*

                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            LOOP ON TEST IN SWR<7:0>
                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWAPE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5      %5              ;;GENERAL REGISTER
000006                  R6      %6              ;;GENERAL REGISTER
000007                  R7      %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0     0               ;;PRIORITY LEVEL 0
```

```
                        
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7
                        
                        ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11    4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04    20
000010                  SW03    10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0
                        
                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
```

```
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;''T'' BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;''TRAP'' TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR


000001                  ADRS=   R1
176500                  DLADDR= 176500

                        ; THE FOLLOWING DEFINITIONS APPLY TO THE GLOBAL SUBS
177777                  SET= -1
000000                  CLR   0
000001                  TRUE  1
```

```
                        ;;*****************************************************************
                        ; RCSR REGISTER BIT NAMES
                        ;;*****************************************************************
                        ; UNUSED       BIT15
                        ; UNUSED       BIT14
                        ; UNUSED       BIT13
                        ; UNUSED       BIT12
                        ; UNUSED       BIT11
                        ; UNUSED       BIT10
                        ; UNUSED       BIT09
                        ; UNUSED       BIT08
000200                  RCVRDONE-      BIT07        ; RECEIVER DONE
000100                  RCVRIE-        BIT06        ; RECEIVER INTERRUPT ENABLE
                        ; UNUSED       BIT05
                        ; UNUSED       BIT04
                        ; UNUSED       BIT03
                        ; UNUSED       BIT02
                        ; UNUSED       BIT01
000001                  RDRRUN=        BIT00        ; READER RUN

                        ;;*****************************************************************
                        ; RBUF REGISTER BIT NAMES
                        ;;*****************************************************************
100000                  ERROR-         BIT15        ; ERROR INDICATOR
040000                  ORERR=         BIT14        ; OVERRUN ERROR
020000                  FRERR=         BIT13        ; FRAMING ERROR
010000                  PERR=          BIT12        ; PARITY ERROR
                        ; UNUSED       BIT11
                        ; UNUSED       BIT10
                        ; UNUSED       BIT09
                        ; UNUSED       BIT08
000200                  RDATA7=        BIT07        ; \
000100                  RDATA6=        BIT06        ;
000040                  RDATA5=        BIT05        ;  !
000020                  RDATA4=        BIT04        ;   \   RECEIVED DATA BITS
000010                  RDATA3=        BIT03        ;   /
000004                  RDATA2=        BIT02        ;  .
000002                  RDATA1=        BIT01        ; .
000001                  RDATA0-        BIT00        ; /
```

```
                        ;;*****************************************************************
                        ; TCSR REGISTER BIT NAMES
                        ;;*****************************************************************
                        ; UNUSED       BIT15
                        ; UNUSED       BIT14
                        ; UNUSED       BIT13
                        ; UNUSED       BIT12
                        ; UNUSED       BIT11
                        ; UNUSED       BIT10
                        ; UNUSED       BIT09
                        ; UNUSED       BIT08
000200                  XMITRDY-       BIT07       ; TRANSMITTER READY
000100                  XMITIE=        BIT06       ; TRANSMITTER INTERRUPT ENABLE
                        ; UNUSED       BIT05
                        ; UNUSED       BIT04
                        ; UNUSED       BIT03
                        ; UNUSED       BIT02
                        ; UNUSED       BIT01
000001                  BREAK=         BIT00       ; SEND BREAK (CONTINUOUS SPACE)


                        ;;*****************************************************************
                        ; TBUF REGISTER BIT NAMES
                        ;;*****************************************************************
                        ; UNUSED       BIT15
                        ; UNUSED       BIT14
                        ; UNUSED       BIT13
                        ; UNUSED       BIT12
                        ; UNUSED       BIT11
                        ; UNUSED       BIT10
                        ; UNUSED       BIT09
                        ; UNUSED       BIT08
000200                  TDATA7-        BIT07       ; \
000100                  TDATA6=        BIT06       ;
000040                  TDATA5=        BIT05       ;
000020                  TDATA4=        BIT04       ;  \ TRANSMITTER DATA BUFFER
000010                  TDATA3=        BIT03       ;  /
000004                  TDATA2=        BIT02       ; .
000002                  TDATA1=        BIT01       ; .
000001                  TDATA0=        BIT00       ; /


                        ;;*****************************************************************
                        ; FLAG BITS TO BE USE OR CLEARED IN $USWR.

000002                  PARITY  - 2
000004                  EVENODD - 4
000010                  BRK     - 10
000020                  WRAP    - 20
```

```
                        ;;*****************************************************************
                        .SBTTL  TRAP CATCHER

000000                          .=0
                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
000174                          .=174
000174  000000         DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000         SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  001334         JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM


                        ;
                        ; OFFLINE TESTING FOR INTERRUPT VECTOR PROBLEMS
                        ;
000400                          .=400

000400  005000                 CLR     R0
000402  012720  013610 1$:     MOV     #INTSRV,(R0)+   ;SET INTR HANDLER PTR
000406  012720  000340         MOV     #PR7,(R0)+      ;SET PRIORITY
000412  020027  000400         CMP     R0,#400         ;ALL DONE?
000416  001371                 BNE     1$              ;BR IF NO
000420  005037  000176         CLR     176             ;CLEAN UP SWREG
000424  000137  001334         JMP     START           ;GO DO IT
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
000430                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  014450                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000430                                  .=$SVPC                 ;; RESTORE PC
001000                          .=1000
                                .SBTTL  APT PARAMETER BLOCK

                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
001000                                  .$X=.   ;;SAVE CURRENT LOCATION
000024                                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
000044                                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
001000                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000031                  $TSTM:  .WORD   25.     ;;RUN TIM OF LONGEST TEST
001006  000144                  $PASTM: .WORD   100.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000144                  $UNITM: .WORD   100.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  00003C                          .WORD   $EMAIL-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                        .SBTTL  COMMON TAGS

                        ;;*****************************************************************
                        ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                        ;*USED IN THE PROGRAM.

                                .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                 .WORD   0
001102  000            $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000            $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000         $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000         $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000         $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000         $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000            $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001            $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000         $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000         $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000         $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000         $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000         $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                 .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                 .WORD   0
001134  000            $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000            $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                 .WORD   0
001140  177570         SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570         DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560         $TKS:   177560                  ;;TTY KBD STATUS
001146  177562         $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564         $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566         $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000            $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002            $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012            $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
001157  000            $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE' FLAG (BIT<07> 0 YES)
001160  000000         $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001162  000000         $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001164  177607  000377 $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001170  077            $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171  015            $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172  000012         $LF:    .ASCIZ  <12>            ;;LINE FEED
                        ;;*****************************************************************
                        .SBTTL  APT MAILBOX-ETABLE

                        ;;*****************************************************************
                        .EVEN
001174                  $MAIL:                          ;;APT MAILBOX
001174  000000         $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001176  000000         $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001200  000000         $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001202  000000         $PASS:  .WORD   APASS           ;;PASS COUNT
001204  000000         $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001206  000000         $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001210  000000         $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
```

```
  968  001212  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
  969  001214                      $ETABLE:                    ;;APT ENVIRONMENT TABLE
  970  001214      000             $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
  971  001215      000             $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
  972  001216  000000              $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
  973  001220  001031              $USWR:  .WORD   AUSWR   ;;USER SWITCHES
  974  001222  000000              $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
  975                                                     BITS 15-11=CPU TYPE
  976                                                             11/04=01,11/05=02,11/20=03,11/40=04,11/45-05
  977                                                             11/70=06,PDQ=07,Q=10
  978                                                     BIT 10=REAL TIME CLOCK
  979                                                     BIT  9=FLOATING POINT PROCESSOR
  980                                                     BIT  8=MEMORY MANAGEMENT
  981  001224      000             $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
  982  001225      000             $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
  983                                                     MEM.TYPE BYTE   --  (HIGH BYTE)
  984                                                             900 NSEC CORE=001
  985                                                             300 NSEC BIPOLAR-002
  986                                                             500 NSEC MOS=003
  987  001226  000000              $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
  988                                                       MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
  989  001230      000             $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
  990  001231      000             $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
  991  001232  000000              $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
  992  001234      000             $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
  993  001235      000             $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
  994  001236  000000              $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
  995  001240      000             $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
  996  001241      000             $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
  997  001242  000000              $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
  998  001244  000300              $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
  999  001246  000000              $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
 1000  001250  176500              $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
 1001  001252  000000              $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
 1002  001254                      $ETEND:
 1003                              .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT

001254                          $ERRTB:
                                ;;              GLOBAL DATA
001254  176500                      DLADD:  DLADDR
001256  000300                      DLVEC:  300
001260  176500                      RCSR:   DLADDR  +  0
001262  176502                      RBUF:   DLADDR  +  2
001264  176504                      TCSR:   DLADDR  +  4
001266  176506                      TBUF:   DLADDR  +  6
001270  000000                      I:      0
001272  000020                      .BLKW 20    ;FOR R5 STACK
001332  000000                      R5STACK:    .WORD 0
```

```
1029  001334                                  START:
1030                                          .SBTTL  INITIALIZE THE COMMON TAGS
1031                                          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1032  001334  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1033  001340  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1034  001342  022706  001140                          CMP     #SWR,R6 ;;DONE?
1035  001346  001374                                  BNE     .-6             ;;LOOP BACK IF NO
1036  001350  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1037                                          ;;INITIALIZE A FEW VECTORS
1038  001354  012737  016210  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1039  001362  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1040  001370  012737  016010  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1041  001376  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1042  001404  012737  017142  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1043  001412  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
1044  001420  013737  014416  014410                  MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
1045  001426  005037  001160                          CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
1046  001432  005037  001162                          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1047  001436  112737  000001  001115                  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1048  001444  012737  001444  001106                  MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1049  001452  012737  001452  001110                  MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1050                                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1051                                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1052  001460  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1053  001464  012737  001520  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1054  001472  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1055  001500  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1056  001506  022777  177777  177424                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1057  001514  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1058                                                                          ;;AND  THE HARDWARE SWR IS NOT   -1
1059  001516  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
1060  001520  012716  001526          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1061  001524  000002                                  RTI
1062  001526  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1063  001534  012737  000174  001142                  MOV     #DISPREG,DISPLAY
1064  001542  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1065
1066  001546  005037  001202                          CLR     $PASS           ;;CLEAR PASS COUNT
1067  001552  132737  000200  001215                  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1068  001560  001403                                  BEQ     67$             ;;YES,USE NON-APT SWITCH
1069  001562  012737  001216  001140                  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1070  001570                                  67$:
1071                                          .SBTTL  TYPE PROGRAM NAME
1072                                          ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1073  001570  005227  177777                          INC     #-1             ;;FIRST TIME?
1074  001574  001043                                  BNE     68$             ;;BRANCH IF NO
1075  001576  022737  014450  000042                  CMP     #$ENDAD,@#42    ;;ACT-11?
1076  001604  001437                                  BEQ     68$             ;;BRANCH IF YES
1077  001606  104401  001654                          TYPE    ,69$            ;;TYPE ASCIZ STRING
1078                                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1079  001612  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1080  001616  001012                                  BNE     70$             ;;BRANCH IF YES
1081  001620  123727  001214  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
1082  001626  001406                                  BEQ     70$             ;;BRANCH IF YES
1083  001630  023727  001140  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1084  001636  001005                                  BNE     71$             ;;BRANCH IF NO
```

```
1085  001640  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1086  001642  000403                          BR      71$
1087  001644  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1088  001652                          71$:
1089  001652  000414                          BR      68$             ;;GET OVER THE ASCIZ
1090                                  ;;69$:   .ASCIZ  <CRLF>*CVDLAB0 DLV11-J TEST*<CRLF>
1091  001704                          68$:
```

```
1092
1093
1094  001704  004737  012700                JSR     PC,SIZE         ;SIZE THE BUS IF $DEVM = 0
1095
1096  001710  005037  014166                CLR     PHASE2          ;INIT FOR TESTS
1097  001714  012737  000001  001206        MOV     #1,$UNIT
1098  001722  005037  014164                CLR     CHAN
1099  001726  005037  014170                CLR     P1CNT
1100
1101  001732  012706  001100        LOOP:   MOV     #STACK,SP       ;RESET STACK POINTER
1102  001736  004737  013620                JSR     PC,CYCLE        ;SETUP MODULE AND CHANNEL
1103                                                                ;               ADRS+2 -> NEXT VECTOR
1104                                                                                ;GET UNIT ADDRESS
1105  001742                                                                LET     DLADD :  (ADRS)+
1106  001742  012137  001254                MOV     (ADRS)+,DLADD
1107                                                                                ;GET UNIT VECTOR
1108  001746                                                                LET     DLVEC := (ADRS)
1109  001746  011137  001256                MOV     (ADRS),DLVEC
1110                                                                                ;RCSR = DLADD  +  0
1111  001752                                                                LET     RCSR  :- DLADD
1112  001752  013737  001254  001260        MOV     DLADD,RCSR
1113  001760                                                                LET     RBUF  := DLADD + #2
1114  001760  013737  001254  001262        MOV     DLADD,RBUF
1115  001766  062737  000002  001262        ADD     #2,RBUF
1116  001774                                                                LET     TCSR  :  DLADD + #4
1117  001774  013737  001254  001264        MOV     DLADD,TCSR
1118  002002  062737  000004  001264        ADD     #4,TCSR
1119  002010                                                                LET     TBUF  :  DLADD + #6
1120  002010  013737  001254  001266        MOV     DLADD,TBUF
1121  002016  062737  000006  001266        ADD     #6,TBUF
1122  002024                                                                LET     R5    :  #R5STACK
1123  002024  012705  001332                MOV     #R5STACK,R5
1124
1125  002030  012700  001000                MOV     #1000,R0        ;DELAY TO ALLOW PRINT TO FINISH
1126  002034  077001                        SOB     R0,.            ;BEFORE RESET
1127  002036                                                        BRESET
1128  002036  000005                        RESET
1129  002040                                                                LET     $DEVCT :  $DEVCT + #1
1130  002040  005237  001204                INC     $DEVCT
1131  002044                                                                IFB     PHASE2  EQ  #TRUE  THEN
1132  002044  123727  014166  000001        CMPB    PHASE2,#TRUE
1133  002052  001003                        BNE     $1
1134  002054  000137  007000                JMP     MODTST                      ;GO TO PHASE 2 TESTING
1135  002060                                                                ELSE
1136  002060  000402                        BR      $2
1137  002062                          $1:
1138  002062                                                                   LET $TSTNM :- #0
1139  002062  005037  001102                CLR     $TSTNM
1140  002066                                                                ENDIF
1141  002066                          $2:
```

```
1142                                    ;;*****************************************************************
1143                                    ;*TEST 1        ADDRESSABILITY
1144                                    ;*
1145                                    ;*      THIS TEST VERIFIES THAT ALL 4 REGISTERS OF THE CHANNEL UNDER TEST
1146                                    ;*      RESPOND TO THEIR ADDRESSES.
1147                                    ;*
1148                                    ;*****************************************************************
1149  002066  000004                   TST1:   SCOPE
1150  002070  012737  000002  001160           MOV     #2,$TIMES       ;;DO 2 ITERATIONS
1151  002076  012737  000001  001200           MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1152  002104                                                                   LET     ADRS := DLADD
1153  002104  013701  001254                   MOV     DLADD,ADRS
1154                                                                             ; SET UP INTERRUPT
1155  002110  012737  013610  000004           MOV     #INTSRV,ERRVEC
1156  002116  012737  000340  000006           MOV     #PR7,ERRVEC+2
1157  002124                                                                   LET     I :- #0
1158  002124  005037  001270                   CLR     I
1159  002130                                                                   REPEAT
1160  002130                           $3:
1161  002130                                                                       BGNSUB
1162  002130  012737  002136  001110           MOV     #64$,$LPERR
1163                                                                                 ;CLEAR FLAG
1164  002136                                                                         LET INTFLAG :  #0
1165  002136  005037  013616                   CLR     INTFLAG
1166
1167                                                                                 ;READ FLAG
1168  002142  005711                           TST     @ADRS
1169  002144                                                                         IF INTFLAG NE #0 THEN
1170  002144  005737  013616                   TST     INTFLAG
1171  002150  001401                           BEQ     $4
1172                                                                                 ; FATAL ERROR
1173  002152                                                                             ERRDF 1
1174  002152  104001                           ERROR   1
1175  002154                                                                         ENDIF
1176  002154                           $4:
1177  002154                                                                       ENDSUB
1178  002154                                                                       LET     I :- I + #2
1179  002154  062737  000002  001270           ADD     #2,I
1180  002162                                                                       LET     ADRS :- DLADD + I
1181  002162  013701  001254                   MOV     DLADD,ADRS
1182  002166  063701  001270                   ADD     I,ADRS
1183  002172                                                                   UNTIL I EQ #8.
1184  002172  023727  001270  000010           CMP     I,#8.
1185  002200  001353                           BNE     $3
```

```
1186
1187  002202  032777  177476  177050          BIT     #177476,@RCSR   ;CHECK THAT ALL UNUSED BITS ARE 0
1188  002210  001401                          BEQ     1$
1189  002212  104027                          ERROR   27              ;RCSR HAS UNUSED BITS SET
1190
1191  002214  032777  007400  177040  1$:     BIT     #7400,@RBUF
1192  002222  001401                          BEQ     2$
1193  002224  104030                          ERROR   30              ;RBUF HAS UNUSED BITS SET
1194
1195  002226  032777  177476  177030  2$:     BIT     #177476,@TCSR
1196  002234  001401                          BEQ     3$
1197  002236  104031                          ERROR   31              ;TCSR HAS UNUSED BITS SET
1198
1199  002240  032777  177400  177020  3$:     BIT     #177400,@TBUF
1200  002246  001401                          BEQ     4$
1201  002250  104032                          ERROR   32              ;TBUF HAS ;UNUSED BITS SET
1202
1203  002252  012737  000006  000004  4$:     MOV     #6,ERRVEC       ;RESTORE
1204  002260  005037  000006                  CLR     ERRVEC+2
1205
```

```
                                ;;*****************************************************************
                                ;* THE FOLLOWING 3 TESTS  TEST ALL 'READ WRITE' BITS
                                ;;*****************************************************************


                                ;;*****************************************************************
                                ;*TEST 2        BREAK - TCSR 0 SET, CLEAR, RESET
                                ;*               THIS BIT IS THE ONLY ONE IN THIS POSITION
                                ;*               THAT IS READ AND WRITE.
                                ;;*****************************************************************
002264  000004                  TST2:   SCOPE
002266  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
002274  012737  000002  001200          MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

002302                                                          IF #BRK NOTSETIN $USWR THEN
002302  032737  000010  001220          BIT     #BRK,$USWR
002310  001001                        BNE     $5
002312                                                              EXIT                ; BREAK NOT INSTALLED
002312  000452                        BR      TST3            ;;EXIT THIS TEST
002314                                                          ENDIF
002314                          $5:

                                                        ; SEE IF IT IS CLEAR
002314                                                          BGNSUB
002314  012737  002322  001110          MOV     #64$,$LPERR

002322                                                              IF      #BREAK SETIN @TCSR THEN
002322  032777  000001  176734          BIT     #BREAK,@TCSR
002330  001401                        BEQ     $6

                                                                    ; BREAK DID NOT RESET IN TCSR
002332                                                              ERRHRD 2
002332  104002                        ERROR   2
002334                                                              ENDIF
002334                          $6:
002334                                                          ENDSUB

                                                        ; TRY TO SET BREAK BIT
002334                                                          BGNSUB
002334  012737  002342  001110          MOV     #64$,$LPERR
002342                                                              LET     @TCSR := @TCSR SET.BY #BREAK
002342  052777  000001  176714          BIS     #BREAK,@TCSR

                                                                    ; STUCK TO 0
002350                                                              IF      #BREAK NOTSETIN @TCSR THEN
002350  032777  000001  176706          BIT     #BREAK,@TCSR
002356  001001                        BNE     $7

                                                                    ; BREAK DID NOT SET IN TCSR
002360                                                              ERRHRD 3
002360  104003                        ERROR   3
002362                                                              ENDIF
002362                          $7:
002362                                                          ENDSUB

                                                        ; TRY TO CLEAR A SET BIT
002362                                                          BGNSUB
002362  012737  002370  001110          MOV     #0'$,$LPERR
```

```
  262  002370
 1263  002370  042777  000001  176666          BIC     #BREAK,@TCSR
 1264
 1265  002376
 1266  002376  032777  000001  176660          BIT     #BREAK,@TCSR
 1267  002404  001401                          BEQ     $10
 1268
 1269  002406
 1270  002406  104004                          ERROR   4
 1271  002410
 1272  002410                          $10:
 1273  002410
 1274
 1275
 1276  002410
 1277  002410  012737  002416  001170          MOV     #64$,$LPERR
 1278
 1279  002416
 1280  002416  052777  000001  176640          BIS     #BREAK,@TCSR
 1281
 1282  002424
 1283  002424  000005                          RESET
 1284  002426
 1285  002426  032777  000001  176630          BIT     #BREAK,@TCSR
 1286  002434  001401                          BEQ     $11
 1287
 1288  002436
 1289  002436  104005                          ERROR   5
 1290  002440
 1291  002440                          $11:
 1292  002440
```

```
                                        LET     @TCSR := @TCSR CLR.BY #BREAK

                                            ; SHOULD HAVE CLEARED
                                        IF      #BREAK SETIN @TCSR THEN

                                            ; BREAK  DID NOT CLEAR IN TCSR
                                            ERRHRD 4

                                        ENDIF

                                    ENDSUB

                                            ; NOW SEE IF RESET CLEARS IT
                                    BGNSUB

                                        LET     @TCSR :- @TCSR SET.BY #BREAK

                                            ; ISSUE BUS RESET
                                        BRESET

                                        IF      #BREAK SETIN @TCSR THEN

                                            ; BREAK DID NOT RESET IN TCSR
                                            ERRHRD 5

                                        ENDIF

                                    ENDSUB
```

```
  293
 1294                                   ;;*****************************************************************
 1295                                   ;*TEST 3          XMITIE - TCSR 6        SET, CLEAR, RESET
 1296                                   ;;*****************************************************************
 1297  002440  000004                  TST3:   SCOPE
 1298  002442  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1299  002450  012737  000003  001200           MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 1300                                                                           ; USE PRIORITY OF 7
 1301  002456                                   SETPRI #PR7
 1302
 1303                                                   ; SEE IF IT IS CLEAR
 1304  002470                                                                   BGNSUB
 1305  002470  012737  002476  001110           MOV     #65$,$LPERR
 1306
 1307  002476                                                                       IF      #XMITIE SETIN @TCSR THEN
 1308  002476  032777  000100  176560           BIT     #XMITIE,@TCSR
 1309  002504  001401                           BEQ     $12
 1310                                                                               ; XMITIE DID NOT RESET IN TCSR
 1311  002506                                                                       ERRHRD 12
 1312  002506  104012                           ERROR   12
 1313  002510                                                                   ENDIF
 1314  002510                                  $12:
 1315  002510                                                                 ENDSUB
 1316
 1317                                                   ; TRY TO SET XMITIE BIT
 1318  002510                                                                   BGNSUB
 1319  002510  012737  002516  001110           MOV     #64$,$LPERR
 1320  002516                                                                       LET     @TCSR :  @TCSR SET.BY #XMITIE
 1321  002516  052777  000100  176540           BIS     #XMITIE,@TCSR
 1322                                                                               ; STUCK TO 0
 1323  002524                                                                       IF      #XMITIE NOTSETIN @TCSR THEN
 1324  002524  032777  000100  176532           BIT     #XMITIE,@TCSR
 1325  002532  001001                           BNE     $13
 1326                                                                               ; XMIT DID NOT SET IN TCSR
 1327  002534                                                                       ERRHRD 13
 1328  002534  104013                           ERROR   13
 1329  002536                                                                   ENDIF
 1330  002536                                  $13:
 1331  002536                                                                 ENDSUB
 1332
 1333                                                   ; TRY TO CLEAR A SET BIT
 1334  002536                                                                   BGNSUB
 1335  002536  012737  002544  001110           MOV     #64$,$LPERR
 1336
 1337  002544                                                                       LET     @TCSR :  @TCSR CLR.BY #XMITIE
 1338  002544  042777  000100  176512           BIC     #XMITIE,@TCSR
 1339                                                                               ; SHOULD HAVE CLEARED
 1340  002552                                                                       IF      #XMITIE SETIN @TCSR THEN
 1341  002552  032777  000100  176504           BIT     #XMITIE,@TCSR
 1342  002560  001401                           BEQ     $14
 1343                                                                               ;  XMIT DID NOT CLEAR IN TCSR
 1344  002562                                                                       ERRHRD 14
 1345  002562  104014                           ERROR   14
 1346  002564                                                                   ENDIF
 1347  002564                                  $14:
 1348  002564                                                                 ENDSUB
```

```
1349
1350                                                    ; NOW SEE IF RESET CLEARS IT
1351  002564                                                                    BGNSUB
1352  002564  012737  002572  001110          MOV     #64$,$LPERR
1353
1354  002572                                                                        LET     @TCSR := @TCSR SET.BY #XMITIE
1355  002572  052777  000100  176464          BIS     #XMITIE,@TCSR
1356                                                                                ; ISSUE BUS RESET
1357  002600                                                                        BRESET
1358  002600  000005                          RESET
1359  002602                                                                        IF      #XMITIE SETIN @TCSR THEN
1360  002602  032777  000100  176454          BIT     #XMITIE,@TCSR
1361  002610  001401                          BEQ     $15
1362                                                                                ; XMIT DID NOT RESET IN TCSR
1363  002612                                                                        ERRHRD 15
1364  002612  104015                          ERROR   15
1365  002614                                                                        ENDIF
1366  002614                          $15:
1367  002614                                                                    ENDSUB
```

```
 1368
 1369                                   ;;*****************************************************************
 1370                                   ;*TEST 4          RCVRIE - RCSR 6        SET, CLEAR, RESET
 1371                                   ;*               THIS BIT IS THE ONLY ONE IN THIS POSITION
 1372                                   ;*               THAT IS READ AND WRITE.
 1373                                   ;*****************************************************************
 1374  002614  000004                  TST4:   SCOPE
 1375  002616  012737  000010  001160           MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1376  002624  012737  000004  001200           MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 1377                                                           ; SEE IF IT IS CLEAR
 1378  002632                                                            BGNSUB
 1379  002632  012737  002640  001110           MOV     #64$,$LPERR
 1380
 1381  002640                                                                IF      #RCVRIE SETIN @RCSR THEN
 1382  002640  032777  000100  176412           BIT     #RCVRIE,@RCSR
 1383  002646  001401                           BEQ     $16
 1384
 1385  002650                                                                    ; RCVRIE DID NOT RESET IN RCSR
 1386  002650  104035                                                           ERRHRD 35
 1387  002652                                                                ENDIF
 1388  002652                           $ 6:
 1389  002652                                                            ENDSUB
 1390
 1391                                                           ; TRY TO SET RCVRIE BIT
 1392  002652                                                            BGNSUB
 1393  002652  012737  002660  001110           MOV     #64$,$LPERR
 1394  002660                                                                LET     @RCSR :  @RCSR SET.BY #RCVRIE
 1395  002660  052777  000100  176372           BIS     #RCVRIE,@RCSR
 1396                                                                    ; STUCK TO 0
 1397  002666                                                            IF      #RCVRIE NOTSETIN @RCSR THEN
 1398  002666  032777  000100  176364           BIT     #RCVRIE,@RCSR
 1399  002674  001001                           BNE     $17
 1400
 1401  002676                                                                    ; RCVRIE DID NOT SET IN RCSR
 1402  002676  104036                                                           ERRHRD 36
 1403  002700                                                                ENDIF
 1404  002700                           $17:
 1405  002700                                                            ENDSUB
 1406
 1407                                                           ; TRY TO CLEAR A SET BIT
 1408  002700                                                            BGNSUB
 1409  002700  012737  002706  001110           MOV     #64$,$LPERR
 1410
 1411  002706                                                                LET     @RCSR := @RCSR CLR.BY #RCVRIE
 1412  002706  042777  000100  176344           BIC     #RCVRIE,@RCSR
 1413                                                                    ; SHOULD HAVE CLEARED
 1414  002714                                                            IF      #RCVRIE SETIN @RCSR THEN
 1415  002714  032777  000100  176336           BIT     #RCVRIE,@RCSR
 1416  002722  001401                           BEQ     $20
 1417
 1418  002724                                                                    ; RCVRIE DID NOT CLEAR IN RCSR
 1419  002724  104037                                                           ERRHRD 37
 1420  002726                                                                ENDIF
 1421  002726                           $20:
 1422  002726                                                            ENDSUB
 1423
```

```
1424                                                    ; NOW SEE IF RESET CLEARS IT
1425  002726                                                BGNSUB
1426  002726  012737  002734  001110          MOV     #64$,$LPERR
1427
1428  002734                                                LET     @RCSR := @RCSR SET.BY #RCVRIE
1429  002734  052777  000100  176316          BIS     #RCVRIE,@RCSR
1430                                                        ; ISSUE BUS RESET
1431  002742                                                BRESET
1432  002742  000005                          RESET
1433  002744                                                IF      #RCVRIE SETIN @RCSR THEN
1434  002744  032777  000100  176306          BIT     #RCVRIE,@RCSR
1435  002752  001401                          BEQ     $21
1436                                                        ; RCVRIE DID NOT RESET IN RCSR
1437  002754                                                ERRHRD 40
1438  002754  104040                          ERROR   40
1439  002756                                                ENDIF
1440  002756                          $21:
1441  002756                                              ENDSUB
```

```
1442
1443                                    ;;*****************************************************************
1444                                    ;*TEST 5          TEST THAT XMITRDY - TCSR 7 - IS SET BY INIT
1445                                    ;;*****************************************************************
1446  002756  000004                    TST5:   SCOPE
1447  002760  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1448  002766  012737  000005  001200            MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1449
1450  002774                                                                    BGNSUB
1451  002774  012737  003002  001110            MOV     #64$,$LPERR
1452
1453  003002                                                                        IF      #XMITRDY NOTSETIN @TCSR THEN
1454  003002  032777  000200  176254            BIT     #XMITRDY,@TCSR
1455  003010  001002                            BNE     $22
1456
1457                                                                               ;RESET SHOULD HAVE SET BIT.
1458                                                                               ;XMITRDY DID NOT SET IN TCSR (AFTER RESE
1459  003012                                                                       ERRHRD 42
1460  003012  104042                            ERROR   42
1461                                                                               ;ISSUE ANOTHER RESET
1462  003014                                                                       BRESET
1463  003014  000005                            RESET
1464  003016                                                                    ENDIF
1465  003016                                $22:
1466                                                                               ;ALLOW LOOPING ON ERROR
1467  003016                                                                    ENDSUB
```

```
                                ;;*****************************************************************
                                ;*TEST 6         TEST THAT XMIT RDY - TCSR 7 - CLEARS
                                ;*              WHEN TBUF IS LOADED WITH A CHARACTER
                                ;*              AND THAT IT SETS WITHIN A REASONABLE AMOUNT OF TIME.
                                ;;*****************************************************************
003016  000004          TST6:   SCOPE
003020  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
003026  012737  000006  001200          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

003034                                                          IF CONSOL EQ #TRUE THEN
003034  023727  013604  000001          CMP     CONSOL,#TRUE
003042  001001                          BNE     $23
003044                                                              EXIT
003044  000521                          BR      TST7            ;;EXIT THIS TEST
003046                                                          ENDIF
003046                          $23:

003046                                                          LET PASS := #0  ;INIT COUNT OF TIMES THRU
003046  005037  003302                  CLR     PASS
003052                                                          LOOP            ; START OF LOOP
003052                          $24:
                                                                ;  MAX OF 2 TIMES THRU
003052                                                      LET ERRORFLAG :  #0
003052  005037  003304                  CLR     ERRORFLAG
003056                                                       LET EXITFLAG :  #0
003056  005037  003306                  CLR     EXITFLAG
                                            ; LOAD TBUF WITH ONE CHARACTER
                                            ; WAIT FOR READY TO SET
                                            ; (SHOULD BE VERY SHORT WAIT
                                            ; SINCE UART DOUBLE BUFFERS ITS INPUT)

                                                            ;SEND A CHARACTER
003062                                                      LET @TBUF :B= #0
003062  105077  176200                  CLRB    @TBUF
                                                            ;WAIT A MAXIMUM
                                                            ;OF 50 MSEC FOR
                                                            ;XMIT RDY TO SET IN TCSR
003066                                                      CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
003066  010546                          MOV     R5,-(SP)
003070  012745  177777                  MOV     #SET,-(R5)
003074  013745  001264                  MOV     TCSR,-(R5)
003100  012745  000200                  MOV     #XMITRDY,-(R5)
003104  012745  000500                  MOV     #500,-(R5)
003110  004737  012446                  JSR     PC,TIMER
003114  012605                          MOV     (SP)+,R5
                                                            ;TIMER RETURNS AN ERROR IF BIT DID
                                                            ;NOT MEET CONDITION WITHIN TIME LIMIT
003116                                                      IF.ERROR THEN
003116  103003                          BCC     $26
                                                            ;XMIT RDY DID NOT SET IN TCSR
                                                            ERRHRD 66
003120
003120  104066                          ERROR   66
003122  000137  003310                  JMP     TST7
003126                                                      ENDIF
003126                          $26:
```

```
1524  003126  105077  176134          CLRB    @TBUF   ;SHIP 1'ST CHAR
1525  003132  105777  176126  1$:     TSTB    @TCSR   ;WAIT FOR RDY
1526  003136  100375                  BPL     1$
1527                                        ; LOAD TBUF WITH A SECOND CHARACTER (TO DOUBLE BUFFER)
1528                                        ; CHECK IMMEDIATELY THAT XMITRDY IS CLEAR
1529                                        ; AND THEN WAIT FOR IT TO SET
1530
1531                                                    ;SEND SECOND CHARACTER
1532  003140                                          LET @TBUF :B= #0
1533  003140  105077  176122          CLRB    @TBUF
1534  003144  000240                          NOP            ; GIVE IT TIME TO CLEAR
1535                                                    ; XMITRDY SHOULD HAVE CLEARED UPON
1536                                                    ; RECEIPT OF A CHARACTER
1537  003146                                          IF #XMITRDY SETIN @TCSR THEN
1538  003146  032777  000200  176110  BIT     #XMITRDY,@TCSR
1539  003154  001404                  BEQ     $27
1540                                                    ; XMITRDY DID NOT CLEAR IN TCSR
1541                                                    ; WILL RESULT IN ERR 67 IF FAILS 2X
1542  003156                                            LET ERRORFLAG := #SET
1543  003156  012737  177777  003304  MOV     #SET,ERRORFLAG
1544                                                            ; DEFER ERROR TYPEOUT
1545
1546  003164                                          ELSE
1547  003164  000416                  BR      $30
1548  003166                  $27:
1549                                                    ;WAIT A MAXIMUM
1550                                                    ;OF 100 MSEC FOR
1551                                                    ;XMIT RDY TO SET IN TCSR
1552  003166                                          CALL TIMER IN <#1000,#XMITRDY,TCSR,#SET>
1553  003166  010546                  MOV     R5,-(SP)
1554  003170  012745  177777          MOV     #SET,-(R5)
1555  003174  013745  001264          MOV     TCSR,-(R5)
1556  003200  012745  000200          MOV     #XMITRDY,-(R5)
1557  003204  012745  001000          MOV     #1000,-(R5)
1558  003210  004737  012446          JSR     PC,TIMER
1559  003214  012605                  MOV     (SP)+,R5
1560  003216                                          IF.ERROR THEN
1561  003216  103001                  BCC     $31
1562                                                    ;XMIT RDY DID NOT SET IN TCSR
1563  003220                                            ERRHRD 70
1564  003220  104070                  ERROR   70
1565  003222                                          ENDIF
1566  003222                  $31:
1567  003222                                        ENDIF ; OF DEFERED ERROR CALL
1568  003222                  $30:
1569  003222                                        IF ERRORFLAG EQ #SET THEN
1570  003222  023727  003304  177777  CMP     ERRORFLAG,#SET
1571  003230  001013                  BNE     $32
1572  003232                                          LET PASS := PASS + #1
1573  003232  005237  003302          INC     PASS
1574  003236                                          IF PASS GT #1 THEN
1575  003236  023727  003302  000001  CMP     PASS,#1
1576  003244  003404                  BLE     $33
1577                                                    ; CALL ERROR IF 2ND TRY
1578                                                    ; ON XMIT RDY NOT CLEARING
1579  003246                                            ERRHRD 67
```

```
 1580  003246  104067                          ERROR   67                              LET EXITFLAG := #SET
 1581  003250
 1582  003250  012737  177777  003306          MOV     #SET,EXITFLAG                 ENDIF
 1583  003256
 1584  003256                                $33:                                  ELSE            ; NO ERROR
 1585  003256
 1586  003256  000403                          BR      $34
 1587  003260                                $32:                                      LET EXITFLAG := #SET
 1588  003260
 1589  003260  012737  177777  003306          MOV     #SET,EXITFLAG                 ENDIF
 1590  003266
 1591  003266                                $34:                                EXIF        EXITFLAG EQ #SET
 1592  003266
 1593  003266  023727  003306  177777          CMP     EXITFLAG,#SET
 1594  003274  001401                          BEQ     $25                         ENDLOOP
 1595  003276
 1596  003276  000665                          BR      $24
 1597  003300                                $25:                                      EXIT ; SKIP AROUND FLAG WORDS
 1598  003300
 1599  003300  000403                          BR      TST7            ;;EXIT THIS TEST
 1600  003302  000000                        PASS:     0
 1601  003304  000000                        ERRORFLAG: 0
 1602  003306  000000                        EXITFLAG:  0
```

```
1603
1604                                    ;;*****************************************************************
1605                                    ;*TEST 7         TEST THAT OUTPUTTING A CHAR FROM TBUF (WITH WRAP CONNECTED)
1606                                    ;*              RESULTS IN RCVRDONE SETTING WITHIN A REASONABLE AMOUNT OF TIME
1607                                    ;*              AND THAT RESET CLEARS THE BIT.
1608                                    ;*****************************************************************
1609  003310  000004                    TST7:   SCOPE
1610  003312  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1611  003320  012737  000007  001200            MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1612
1613  003326                                                            IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
1614  003326  032737  000020  001220            BIT     #WRAP,$USWR
1615  003334  001404                           BEQ     $35
1616  003336  023727  013604  000001            CMP     CONSOL,#TRUE
1617  003344  001001                           BNE     $36
1618  003346                            $35:
1619  003346                                                                EXIT
1620  003346  000434                           BR      TST10           ;;EXIT THIS TEST
1621  003350                                                            ENDIF
1622  003350                            $36:
1623
1624  003350                                                            BGNSUB
1625  003350  012737  003356  001110            MOV     #64$,$LPERR
1626                                                ; SEND A CHARACTER AND LET IT WRAP AROUND
1627
1628  003356                                                                LET @TBUF :B= #0
1629  003356  105077  175704                   CLRB    @TBUF
1630
1631                                                                       ; WAIT A MAXIMUM OF 50 MSEC
1632                                                                       ; FOR RCVR DONE TO SET IN
1633                                                                       ; RCSR
1634  003362                                                               CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
1635  003362  010546                           MOV     R5,-(SP)
1636  003364  012745  177777                   MOV     #SET,-(R5)
1637  003370  013745  001260                   MOV     RCSR,-(R5)
1638  003374  012745  000200                   MOV     #RCVRDONE,-(R5)
1639  003400  012745  000500                   MOV     #500,-(R5)
1640  003404  004737  012446                   JSR     PC,TIMER
1641  003410  012605                           MOV     (SP)+,R5
1642
1643                                                                          ;DIDN'T SET IN TIME
1644  003412                                                               IF.ERROR THEN
1645  003412  103001                           BCC     $37
1646                                                                          ; RCVRDONE DID NOT SET IN RCSR
1647  003414                                                                  ERRHRD 71
1648  003414  104071                           ERROR   71
1649  003416                                                               ENDIF
1650  003416                            $37:
1651
1652  003416                                                            ENDSUB
1653
1654  003416                                                            BGNSUB
1655  003416  012737  003424  001110            MOV     #64$,$LPERR
1656                                                ; NOW THAT IT IS SET SEE IF IT CAN BE RESET
1657  003424                                                                BRESET
1658  003424  000005                           RESET
```

```
1659
1660  003426                                                    IF #RCVRDONE SETIN @RCSR THEN
1661  003426  032777  000200  175624          BIT     #RCVRDONE,@RCSR
1662  003434  001401                          BEQ     $40
1663                                                            ; RCVRDONE DID NOT RESET IN RCSR.
1664  003436                                                    ERRHRD 72
1665  003436  104072                          ERROR   72
1666  003440                                                    ENDIF
1667  003440                  $40:
1668  003440                                                ENDSUB
```

```
1669
1670                                    ;;*********************************************************************
1671                                    ;*TEST 10        TEST THAT RCVRDONE IS CLEARED BY SETTING READER ENABLE
1672                                    ;;*********************************************************************
1673  003440  000004                   TST10:  SCOPE
1674  003442  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1675  003450  012737  000010  001200            MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1676
1677  003456                                                            IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
1678  003456  032737  000020  001220            BIT     #WRAP,$USWR
1679  003464  001404                           BEQ     $41
1680  003466  023727  013604  000001            CMP     CONSOL,#TRUE
1681  003474  001001                           BNE     $42
1682  003476                           $41:
1683  003476                                                                EXIT
1684  003476  000433                           BR      TST11           ;;EXIT THIS TEST
1685  003500                                                            ENDIF
1686  003500                           $42:
1687
1688  003500                                                            BGNSUB
1689  003500  012737  003506  001110            MOV     #64$,$LPERR
1690                                                  ; OUTPUT A CHARACTER AND WAIT FOR XMITRDY TO SET.
1691
1692                                                                        ; OUTPUT A CHARACTER
1693  003506                                                            LET @TBUF :B- #0
1694  003506  105077  175554                   CLRB    @TBUF
1695                                                                    ; WAIT MAXIMUM OF 500 MSEC
1696                                                                    ; FOR RCVRDONE TO SET IN
1697                                                                    ; RCSR
1698  003512                                                            CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
1699  003512  010546                           MOV     R5,-(SP)
1700  003514  012745  177777                   MOV     #SET,-(R5)
1701  003520  013745  001260                   MOV     RCSR,-(R5)
1702  003524  012745  000200                   MOV     #RCVRDONE,-(R5)
1703  003530  012745  000500                   MOV     #500,-(R5)
1704  003534  004737  012446                   JSR     PC,TIMER
1705  003540  012605                           MOV     (SP)+,R5
1706                                                                        ; DID IT BECOME READY?
1707  003542                                                            IF.ERROR  THEN
1708  003542  103001                           BCC     $43
1709                                                                        ;RCVRDONE DID NOT SET IN RCSR
1710  003544                                                                ERRHRD 25
1711  003544  104025                           ERROR   25
1712                                                                        ; SET IT BACK TO CONTINUE
1713  003546                                                            ENDIF
1714  003546                           $43:
1715  003546                                                            ENDSUB
1716
1717                                                  ;NOW THAT IT IS SET LETS SEE IF SETTING
1718                                                  ;READER ENABLE CLEARS RCVRDONE
1719
1720  003546                                                            LET @RCSR := @RCSR SET.BY #RDRRUN
1721  003546  052777  000001  175504            BIS     #RDRRUN,@RCSR
1722  003554                                                                IF #RCVRDONE SETIN @RCSR THEN
1723  003554  032777  000200  175476            BIT     #RCVRDONE,@RCSR
1724  003562  001401                           BEQ     $44
```

```
1725
1726  003564                                                    ;RCVRDONE DID NOT CLEAR IN RCSR
1727  003564  104026                        ERROR   26          ERRHRD 26
1728                                                            ; SET IT BACK TO CONTINUE
1729  003566                                                    ENDIF
1730  003566                  $44:
```

```
1731
1732                                    ;;*****************************************************************
1733                                    ;*TEST 11        TEST THAT RCVRDONE IS CLEARED BY READING RBUF
1734                                    ;;*****************************************************************
1735  003566  000004                    TST11:  SCOPE
1736  003570  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1737  003576  012737  000011  001200            MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1738
1739  003604                                                            IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
1740  003604  032737  000020  001220            BIT     #WRAP,$USWR
1741  003612  001404                            BEQ     $45
1742  003614  023727  013604  000001            CMP     CONSOL,#TRUE
1743  003622  001001                            BNE     $46
1744  003624                            $45:
1745  003624                                                                  EXIT
1746  003624  000432                            BR      TST12           ;;EXIT THIS TEST
1747  003626                                                                  ENDIF
1748  003626                            $46:
1749
1750  003626                                                            BGNSUB
1751  003626  012737  003634  001110            MOV     #64$,$LPERR
1752                                                  ; OUTPUT A CHARACTER AND WAIT FOR RCVRDONE TO SET.
1753
1754                                                                        ; OUTPUT A CHARACTER
1755  003634                                                            LET @TBUF :B- #0
1756  003634  105077  175426                    CLRB    @TBUF
1757                                                                    ; WAIT MAXIMUM OF 500 MSEC
1758                                                                    ; FOR RCVRDONE TO SET IN
1759                                                                    ; RCSR
1760  003640                                                            CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
1761  003640  010546                            MOV     R5,-(SP)
1762  003642  012745  177777                    MOV     #SET,-(R5)
1763  003646  013745  001260                    MOV     RCSR,-(R5)
1764  003652  012745  000200                    MOV     #RCVRDONE,-(R5)
1765  003656  012745  000500                    MOV     #500,-(R5)
1766  003662  004737  012446                    JSR     PC,TIMER
1767  003666  012605                            MOV     (SP)+,R5
1768                                                                        ; DID IT BECOME READY?
1769  003670                                                            IF.ERROR  THEN
1770  003670  103001                            BCC     $47
1771                                                                        ;RCVRDONE DID NOT SET IN RCSR
1772  003672                                                                ERRHRD 73
1773  003672  104073                            ERROR   73
1774                                                                        ; SET IT BACK TO CONTINUE
1775  003674                                                            ENDIF
1776  003674                            $47:
1777  003674                                                        ENDSUB
1778
1779                                                  ; NOW THAT IT IS SET LETS SEE IF READING THE
1780                                                  ; BUFFER CLEARS RCVRDONE.
1781
1782                                                                        ;READ BUFFER
1783  003674                                                            LET R0 :B= @RBUF
1784  003674  117700  175362                    MOVB    @RBUF,R0
1785
1786  003700                                                            IF #RCVRDONE SETIN @RCSR THEN
```

```
 787  003700  032777  000200  175352          BIT     #RCVRDONE,@RCSR
1788  003706  001401                          BEQ     $50
1789                                                                ;RCVRDONE DID NOT CLEAR IN RCSR
1790  003710                                                        ERRHRD 74
1791  003710  '04074                         ERROR   74
1792                                                                ; SET IT BACK TO CONTINUE
1793  003712                                                      ENDIF
1794  003712                          $50:
```

```
                                ;;*****************************************************************
                                ;*TEST 12        TEST THE OVERRUN & ERROR BITS - RBUF 14
                                ;;*****************************************************************
003712  000004                  TST12:  SCOPE
003714  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
003722  012737  000012  001200          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                                                IF CONSOL EQ #0 THEN
003730  005737  013604                  TST     CONSOL
003734  001007                          BNE     $51
                                                                    IF #WRAP SETIN $USWR THEN
003736  032737  000020  001220          BIT     #WRAP,$USWR
003744  001401                          BEQ     $52
                                                                        ;; NULL ---EXECUTE TEST
                                                                    ELSE
003746  000401                          BR      $53
003750                          $52:
                                                                        EXIT
003750  000524                          BR      TST13           ;;EXIT THIS TEST
                                                                    ENDIF
003752                          $53:
                                                                ELSE
003752  000401                          BR      $54
003754                          $51:
                                                                    EXIT
003754  000522                          BR      TST13           ;;EXIT THIS TEST
                                                                ENDIF
003756                          $54:


                                                                BGNSUB
003756  012737  003764  001110          MOV     #64$,$LPERR
                                            ;OUTPUT 2 CHARACTERS
                                            ;THIS SHOULD AN CAUSE OVERRUN ERROR.

                                                                    ;OUTPUT 1 CHARACTER
003764                                                          LET @TBUF :B- #0
003764  105077  175276                  CLRB    @TBUF

003770                                                          CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
003770  010546                          MOV     R5,-(SP)
003772  012745  177777                  MOV     #SET,-(R5)
003776  013745  001260                  MOV     RCSR,-(R5)
004002  012745  000200                  MOV     #RCVRDONE,-(R5)
004006  012745  000500                  MOV     #500,-(R5)
004012  004737  012446                  JSR     PC,TIMER
004016  012605                          MOV     (SP)+,R5

                                                                    ;DID IT SET IN TIME?
004020                                                          IF.ERROR THEN
004020  103001                          BCC     $55
                                                                    ;RCVRDONE DID NOT SET IN RCSR
004022                                                              ERRHRD 16
004022  104016                  $55:    ERROR   16
```

```
1851  004024                                                          ENDIF
1852  004024                          $55:
1853
1854                                                                      ;OUTPUT 2ND CHARACTER
1855  004024                                                          LET @TBUF :B= #0
1856  004024  105077  175236                 CLRB    @TBUF
1857                                                                      ;LET OVERRUN HAPPEN
1858  004030                                                            WAITMS 300.
1859  004030  010546                         MOV     R5,-(SP)
1860  004032  012745  000454                 MOV     #300.,-(R5)
1861  004036  004737  012620                 JSR     PC,WAIT
1862  004042  012605                         MOV     (SP)+,R5
1863
1864                                                                      ;READ BUFFER AND ERROR BITS
1865  004044                                                          LET R4 :- @RBUF
1866  004044  017704  175212                 MOV     @RBUF,R4
1867
1868                                                                      ;IT DIDN'T SET
1869  004050                                                          IF #ORERR NOTSETIN R4 THEN
1870  004050  032704  040000                 BIT     #ORERR,R4
1871  004054  001002                         BNE     $56
1872                                                                      ;ORERR DID NOT SET IN RBUF
1873  004056                                                            ERRHRD 101
1874  004056  *04101                         ERROR   101
1875
1876                                                                      ;NO USE COMPOUNDING ERRORS
1877  004060                                                            EXIT
1878  004060  000460                         BR      TST13           ;;EXIT THIS TEST
1879  004062                                                          ENDIF
1880  004062                          $56:
1881  004062                                                        ENDSUB
1882
1883                                  ;NOW SEE IF ERROR BIT SET WITH OVERRUN  ERROR:
1884  004062                                                        BGNSUB
1885  004062  012737  004070  001110         MOV     #64$,$LPERR
1886  004070                                                          IF #ERROR NOTSETIN R4 THEN
1887  004070  032704  100000                 BIT     #ERROR,R4
1888  004074  001002                         BNE     $57
1889
1890                                                                      ;ERROR DID NOT SET IN RBUF
189*  004076                                                            ERRHRD 102
1892  004076  104102                         ERROR   102
1893
1894                                                                      ;-WHEN ORERR SET.
1895                                                                      ;GET OUT NOW.
1896  004100                                                            EXIT
1897  004100  000450                         BR      TST13           ;;EXIT THIS TEST
1898  004102                                                          ENDIF
1899  004102                          $57:
1900  004102                                                        ENDSUB
1901
1902  004102                                                        BGNSUB
1903  004102  012737  004110  001110         MOV     #64$,$LPERR
1904                                      ;CHECK REAL RBUF TO SEE IF ORERR IS STILL SET.
1905
1906  004110                                                          IF #ORERR NOTSETIN @RBUF THEN
```

```
1907  004110  032777  040000  175144          BIT     #ORERR,@RBUF
1908  004116  001002                          BNE     $60
1909
1910                                                                ;READING RBUF CLEARED ORERR.
1911  004120                                                        ERRHRD 103
1912  004120  104103                          ERROR   103
1913                                                                ;SKIP REST OF TEST
1914  004122                                                        EXIT
1915  004122  000437                          BR      TST13          ;;EXIT THIS TEST
1916  004124                                                              ENDIF
1917  004124                          $60:
1918  004124                                                      ENDSUB
1919
1920  004124                                                      BGNSUB
1921  004124  012737  004132  001110          MOV     #64$,$LPERR
1922                                              ;READING RBUF ABOVE SHOULD ENABLE ERROR TO BE CLEARED
1923                                              ;BY NEXT TRANSFER.
1924                                              ;NOW SEE IF THEY CLEAR WHEN ANOTHER CHAR. IS RECEIVED
1925
1926                                                                ;SEND A CHARACTER AROUND.
1927  004132                                                    LET @TBUF :B= #0
1928  004132  105077  175130                  CLRB    @TBUF
1929  004136                                                      CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
1930  004136  010546                          MOV     R5,-(SP)
1931  004140  012745  177777                  MOV     #SET,-(R5)
1932  004144  013745  001260                  MOV     RCSR,-(R5)
1933  004150  012745  000200                  MOV     #RCVRDONE,-(R5)
1934  004154  012745  000500                  MOV     #500,-(R5)
1935  004160  004737  012446                  JSR     PC,TIMER
1936  004164  012605                          MOV     (SP)+,R5
1937
1938                                                                ;DID IT SET IN TIME?
1939  004166                                                      IF.ERROR THEN
1940  004166  103001                          BCC     $61
1941                                                                ;RCVRDONE DID NOT SET IN RCSR
1942  004170                                                        ERRHRD 20
1943  004170  104020                          ERROR   20
1944  004172                                                      ENDIF
1945  004172                          $61:
1946
1947  004172                                                          IF #ORERR SETIN @RBUF THEN
1948  004172  032777  040000  175062          BIT     #ORERR,@RBUF
1949  004200  001402                          BEQ     $62
1950                                                                ;ORERR DID NOT CLEAR IN RBUF
1951  004202                                                        ERRHRD 104
1952  004202  104104                          ERROR   104
1953
1954                                                                ;-AFTER RECEIVING ANOTHER CHAR
1955                                                                ;SKIP AROUND REST
1956  004204                                                        EXIT
1957  004204  000406                          BR      TST13          ;;EXIT THIS TEST
1958  004206                                                              ENDIF
1959  004206                          $62:
1960
1961  004206                                                          IF #ERROR SETIN @RBUF THEN
1962  004206  032777  100000  175046          BIT     #ERROR,@RBUF
```

```
1963  004214  001401                    BEQ     $63
1964                                                        ;ERROR DID NOT CLEAR IN RBUF
1965  004216                                                ERRHRD 105
1966  004216  104105                    ERROR   105
1967
1968  004220                                                ENDIF
1969  004220                        $63:
1970  004220                                                ENDSUB
1971  004220                                                EXIT
1972  004220  000400                    BR      TST13       ;;EXIT THIS TEST
```

```
                        ;;*****************************************************************
                        ;*TEST 13       TRANSMITTER INTERRUPT LOGIC TEST
                        ;*              LOGICALLY THIS IS 4 SEPARATE TESTS
                        ;*              A) DOES TRANSMITTER INTERRUPT LOGIC WORK
                        ;*              B) AT PRIORITY OF 0
                        ;*              C) AND ONLY ONCE
                        ;*              D) BUT NOT WITH INTERRUPT ENABLE CLEAR
                        ;;*****************************************************************
004222  000004          TST13:  SCOPE
004224  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004232  012737  000013  001200          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                                                        ;CLEAR 'INTERRUPT OCCURED' FLAG
                                                                LET INTFLAG :  #0
004240
004240  005037  013616                  CLR     INTFLAG

                                                                        ;GET VECTOR ADDRESS
                                                                LET R3 :- DLVEC
004244
004244  013703  001256                  MOV     DLVEC,R3

                                                                        ;FOR THE TRANSMITTER
                                                                LET R3 := R3 + #4
004250
004250  062703  000004                  ADD     #4,R3

                                                                        ;SET VECTOR TO POINT TO TRANS.SRV AT PRI
004254  012723  013610                  MOV     #INTSRV,(R3)+
004260  012713  000340                  MOV     #PR7,(R3)
                                                                BGNSUB
004264
004264  012737  004272  001110          MOV     #64$,$LPERR
                                                                ;; MAKE SURE THAT TRANSMITTER READY IS SET
                                                                CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
004272
004272  010546                          MOV     R5,-(SP)
004274  012745  177777                  MOV     #SET,-(R5)
004300  013745  001264                  MOV     TCSR,-(R5)
004304  012745  000200                  MOV     #XMITRDY,-(R5)
004310  012745  000500                  MOV     #500,-(R5)
004314  004737  012446                  JSR     PC,TIMER
004320  012605                          MOV     (SP)+,R5


                                                                        ;CLEAR INTERRUPT ENABLE
                                                                    LET @TCSR :- @TCSR CLR.BY #XMITIE
004322
004322  042777  000100  174734          BIC     #XMITIE,@TCSR

                                                                        ;SET IT TO 0
004330                          SETPRI #PR0


                                                                        ;NOW SET I.E. BIT
                                                                    LET @TCSR :  @TCSR SET.BY #XMITIE
004342
004342  052777  000100  174714          BIS     #XMITIE,@TCSR

                                                                CALL TIMER IN <#500,#1,#INTFLAG,#SET>
004350
004350  010546                          MOV     R5,-(SP)
004352  012745  177777                  MOV     #SET,-(R5)
004356  012745  013616                  MOV     #INTFLAG,-(R5)
004362  012745  000001                  MOV     #1,-(R5)
004366  012745  000500                  MOV     #500,-(R5)
004372  004737  012446                  JSR     PC,TIMER
004376  012605                          MOV     (SP)+,R5
```

```
2029
2030                                                                    ;DID IT SET IN TIME?
2031  004400                                                          IF.ERROR THEN
2032  004400  103001                    BCC     $64
2033                                                                    ;INTERRUPT DID NOT OCCUR
2034  004402                                                            ERRHRD 106
2035  004402  104106                    ERROR   106
2036  004404                                                          ENDIF
2037  004404                          $64:
2038
2039                                                                    ;LET POSSIBLE 2'ND INTERR OCCUR
2040  004404                                                          WAITMS 500.
2041  004404  010546                    MOV     R5,-(SP)
2042  004406  012745  000764            MOV     #500.,-(R5)
2043  004412  004737  012620            JSR     PC,WAIT
2044  004416  012605                    MOV     (SP)+,R5
2045
2046                                                                    ;DID EXACTLY 1 INTERRUPT OCCUR
2047  004420                                                          IF INTFLAG GT #1 THEN
2048  004420  023727  013616  000001    CMP     INTFLAG,#1
2049  004426  003401                    BLE     $65
2050                                                                    ;TRANSMITTER INTERRUPTED TWICE
2051  004430                                                              ERRHRD 107
2052  004430  104107                    ERROR   107
2053  004432                                                          ENDIF
2054  004432                          $65:
2055  004432                                                          ENDSUB
2056                                                          ;INTERRUPT WITHOUT INTERRUPT ENABLE SET
2057  004432                                                          BGNSUB
2058  004432  012737  004440  001110    MOV     #64$,$LPERR
2059                                                                    ;CLEAR INTERRUPT ENABLE
2060  004440                                                          LET @TCSR :- @TCSR CLR.BY #XMITIE
2061  004440  042777  000100  174616    BIC     #XMITIE,@TCSR
2062                                                                    ;CLEAR 'INTERRUPT OCCURED' FLAG
2063  004446                                                          LET INTFLAG := #0
2064  004446  005037  013616            CLR     INTFLAG
2065                                                      ;NO INTERRUPTS SHOULD OCCUR, PSW STILL AT 0.
2066                                                      ;DARE IT TO HAPPEN
2067  004452                                                              LET @TBUF :- #0
2068  004452  005077  174610            CLR     @TBUF
2069                                                                  ;SEE IF INT FLAG EVER SETS
2070  004456                                                          CALL TIMER IN <#1000,#1,#INTFLAG,#SET>
2071  004456  010546                    MOV     R5,-(SP)
2072  004460  012745  177777            MOV     #SET,-(R5)
2073  004464  012745  013616            MOV     #INTFLAG,-(R5)
2074  004470  012745  000001            MOV     #1,-(R5)
2075  004474  012745  001000            MOV     #1000.,-(R5)
2076  004500  004737  012446            JSR     PC,TIMER
2077  004504  012605                    MOV     (SP)+,R5
2078                                                                  ;DID IT SET DURING TIMER?
2079  004506  103401                    BCS     1$                    ;BR IF NO
2080                                                                    ;INTERR STILL OCCURED WITH IE DISABLED
2081  004510                                                            ERRHRD 110
2082  004510  104110                    ERROR   110
2083  004512                          1$:
```

```
2084
2085                                    ;;*****************************************************************
2086                                    ;*TEST 14        RECEIVER INTERRUPT LOGIC TEST
2087                                    ;*              THIS TEST COVERS ALL OF THE RECEIVER
2088                                    ;*              SIDE OF THE INTERRUPT LOGIC IN
2089                                    ;*              CHARACTER MODE.
2090                                    ;*****************************************************************
2091  004512  000004                    TST14:  SCOPE
2092  004514  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
2093  004522  012737  000014  001200            MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2094  004530                                                            IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
2095  004530  032737  000020  001220            BIT     #WRAP,$USWR
2096  004536  001404                            BEQ     $66
2097  004540  023727  013604  000001            CMP     CONSOL,#TRUE
2098  004546  001002                            BNE     $67
2099  004550                                $66:
2100  004550  000137  005046                    JMP     TST15                   ;       EXIT TEST
2101  004554                                                            ENDIF
2102  004554                                $67:
2103
2104                                                                            ;CLEAR INTERRUPT OCCURED FLAG
2105                                                                            ;SET UP RECEIVER INTER.VECTOR
2106  004554                                                            SETVEC DLVEC,#INTSRV,#PR7
2107  004554  010146                            MOV     R1,-(SP)
2108  004556  013701  001256                    MOV     DLVEC,R1
2109  004562  012721  013610                    MOV     #INTSRV,(R1)+
2110  004566  012711  000340                    MOV     #PR7,(R1)
2111  004572  012601                            MOV     (SP)+,R1
2112                                            ;PRIORITY 0 AND MULTIPLE INTERRUPT TEST.-RCVRIE
2113  004574                                                            BGNSUB
2114  004574  012737  004602  001110            MOV     #64$,$LPERR
2115  004602                                                            LET INTFLAG :- #0
2116  004602  005037  013616                    CLR     INTFLAG
2117                                                                            ;CLEAR INTERRUPTS
2118  004606                                                                LET @RCSR :- @RCSR CLR.BY #RCVRIE
2119  004606  042777  000100  174444            BIC     #RCVRIE,@RCSR
2120                                                                            ;CHANGE PRIORITY
2121                                                                            ;..TO 0
2122  004614                                        SETPRI #PR0
2123
2124                                                                            ;SEND A CHARACTER
2125  004626                                                                LET @TBUF :B= #0
2126  004626  105077  174434                    CLRB    @TBUF
2127                                                                            ;WAIT A MAXIMUM
2128                                                                    ;OF 50 MSEC FOR
2129                                                                    ;XMIT RDY TO SET IN TCSR
2130  004632                                                            CALL TIMER IN <#500,#RCVRDONE,TCSR,#SET>
2131  004632  010546                            MOV     R5,-(SP)
2132  004634  012745  177777                    MOV     #SET,-(R5)
2133  004640  013745  001264                    MOV     TCSR,-(R5)
2134  004644  012745  000200                    MOV     #RCVRDONE,-(R5)
2135  004650  012745  000500                    MOV     #500,-(R5)
2136  004654  004737  012446                    JSR     PC,TIMER
2137  004660  012605                            MOV     (SP)+,R5
2138                                                                            ;SET INTERRUPT ENABLE
2139  004662                                                                LET @RCSR :- @RCSR SET.BY #RCVRIE
```

```
2140  004662  052777  000100  174370          BIS     #RCVRIE,@RCSR
2141                                                            ;LET IT COME IN.
2142
2143  004670                                                  CALL TIMER IN <#500,#1,#INTFLAG,#SET>
2144  004670  010546                          MOV     R5,-(SP)
2145  004672  012745  177777                  MOV     #SET,-(R5)
2146  004676  012745  013616                  MOV     #INTFLAG,-(R5)
2147  004702  012745  000001                  MOV     #1,-(R5)
2148  004706  012745  000500                  MOV     #500,-(R5)
2149  004712  004737  012446                  JSR     PC,TIMER
2150  004716  012605                          MOV     (SP)+,R5
2151
2152                                                            ;DID IT SET IN TIME?
2153  004720                                                  IF.ERROR THEN
2154  004720  103001                          BCC     $70
2155                                                            ;INTERRUPT DID NOT OCCUR
2156  004722                                                    ERRHRD 111
2157  004722  104111                          ERROR   111
2158  004724                                                  ENDIF
2159  004724                          $70:
2160                                                        ;LET POSSIBLE 2'ND INTERR OCCUR
2161  004724                                                WAITMS 500
2162  004724  010546                          MOV     R5,-(SP)
2163  004726  012745  000500                  MOV     #500,-(R5)
2164  004732  004737  012620                  JSR     PC,WAIT
2165  004736  012605                          MOV     (SP)+,R5
2166
2167                                                            ;EXACTLY 1 INTERRUPT?
2168  004740                                                  IF INTFLAG GT #1 THEN
2169  004740  023727  013616  000001          CMP     INTFLAG,#1
2170  004746  003401                          BLE     $71
2171                                                            ;RECEIVER INTERRUPTED TWICE
2172  004750                                                      ERRHRD 112
2173  004750  104112                          ERROR   112
2174  004752                                                  ENDIF
2175  004752                          $71:
2176  004752                                                  ENDSUB
2177
2178
2179
2180
2181
2182
2183                                                    ;INTERRUPT WITHOUT IE SET.
2184  004752                                                  BGNSUB
2185  004752  012737  004760  001110          MOV     #64$,$LPERR
2186
2187                                                            ;CLEAR INTERRUPT
2188  004760                                                  LET @RCSR :- @RCSR CLR.BY #RCVRIE
2189  004760  042777  000100  174272          BIC     #RCVRIE,@RCSR
2190                                                            ;CLEAR INTERRUPT FLAG
2191  004766                                                  LET INTFLAG := #0
2192  004766  005037  013616                  CLR     INTFLAG
2193                                                            ; SEND A CHARACTER
2194  004772                                                  LET @TBUF :B= #0
2195  004772  105077  174270                  CLRB    @TBUF
```

```
                                                                ; DARE IT
                                                               CALL TIMER IN <#500,#1,#INTFLAG,#CLR>
004776
004776  010546                        MOV     R5,-(SP)
005000  012745  000000                MOV     #CLR,-(R5)
005004  012745  013616                MOV     #INTFLAG,-(R5)
005010  012745  000001                MOV     #1,-(R5)
005014  012745  000500                MOV     #500,-(R5)
005020  004737  012446                JSR     PC,TIMER
005024  012605                        MOV     (SP)+,R5
                                                                ;DID IT CLEAR IN TIME?
005026                                                         IF.ERROR THEN
005026  103001                        BCC     $72
                                                                ;INTERR STILL OCCURED WITH IE DISABLED
                                                                ERRHRD 113
005030
005030  104113                        ERROR   113
005032                                                         ENDIF
005032                          $72:
005032                                                         ENDSUB
005032                          SETPPI  #PR7    ;RAISE CPU PRIORITY
005032                                                         ;CLEAR THE WORLD
                                                                BRESET
005044
005044  000005                        RESET
```

```
2221
2222                                    ;;*****************************************************************
2223                                    ;*TEST 15        TEST DATA  WRAP AROUND: FLAG MODE
2224                                    ;;*****************************************************************
2225  005046  000004                   TST15:  SCOPE
2226  005050  012737  000001  001160           MOV     #1,$TIMES       ;;DO 1 ITERATION
2227  005056  012737  000015  001200           MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2228  005064                                                                   IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
2229  005064  032737  000020  001220           BIT     #WRAP,$USWR
2230  005072  001404                          BEQ     $73
2231  005074  023727  013604  000001          CMP     CONSOL,#TRUE
2232  005102  001001                          BNE     $74
2233  005104                           $73:
2234                                                                           ;CAN'T TEST WITHOUT A WRAP
2235  005104                                                                   EXIT
2236  005104  000506                          BR      TST16           ;;EXIT THIS TEST
2237  005106                                                                   ENDIF
2238  005106                           $74:
2239  005106                                                                   BRESET
2240  005106  000005                          RESET
2241                                                                           ;BINARY COUNT PATTERN
2242  005110                                                                   INCR R2 FROM #0 TO #377 BY #1
2243  005110  005002                          CLR     R2
2244  005112  000401                          BR      $75
2245  005114                           $76:
2246  005114  005202                          INC     R2
2247  005116                           $75:
2248  005116  020227  000377                  CMP     R2,#377
2249  005122  003077                          BGT     $77
2250
2251
2252
2253                                                                           ; MAKE SURE IT'S READY
2254  005124                                                                   CALL TIMER IN <#500,#XMITRDY,TCSR,#SET>
2255  005124  010546                          MOV     R5,-(SP)
2256  005126  012745  177777                  MOV     #SET,-(R5)
2257  005132  013745  001264                  MOV     TCSR,-(R5)
2258  005136  012745  000200                  MOV     #XMITRDY,-(R5)
2259  005142  012745  000500                  MOV     #500,-(R5)
2260  005146  004737  012446                  JSR     PC,TIMER
2261  005152  012605                          MOV     (SP)+,R5
2262  005154                                                                   IF.ERROR THEN
2263  005154  103002                          BCC     $100
2264                                                                           ; TRANSMITTER NEVER BECAME READY
2265  005156                                                                   ERRHRD 123
2266  005156  104123                          ERROR   123
2267  005160                                                                   EXIT
2268  005160  000460                          BR      TST16           ;;EXIT THIS TEST
2269  005162                                                                   ENDIF
2270  005162                           $100:
2271
2272                                                                           ;START IT ON ITS WAY
2273  005162                                                                   LET @TBUF :B= R2
2274  005162  110277  174100                  MOVB    R2,@TBUF
2275                                                                           ; NOW WAIT FOR RECIEVER DONE
2276  005166                                                                   CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
```

```
2277  005166  010546                        MOV     R5,-(SP)
2278  005170  012745  177777                MOV     #SET,-(R5)
2279  005174  013745  001260                MOV     RCSR,-(R5)
2280  005200  012745  000200                MOV     #RCVRDONE,-(R5)
2281  005204  012745  000500                MOV     #500,-(R5)
2282  005210  004737  012446                JSR     PC,TIMER
2283  005214  012605                        MOV     (SP)+,R5
2284  005216                                                          IF.ERROR THEN
2285  005216  103002                        BCC     $101
2286  005220                                                              ERRHRD 124
2287  005220  104124                        ERROR   124
2288                                                                  ; RECIEVER NEVER BECAME READY
2289  005222                                                          EXIT
2290  005222  000437                        BR      TST16           ;;EXIT THIS TEST
2291  005224                                                          ENDIF
2292  005224                          $101:
2293
2294                                                                  ;RETRIEVE
2295  005224                                                        LET R3 := @RBUF
2296  005224  017703  174032                MOV     @RBUF,R3
2297                                                                  ;CHECK FOR ERROR DURING TRANSFER
2298  005230                                                         IF #ERROR SETIN R3 THEN
2299  005230  032703  100000                BIT     #ERROR,R3
2300  005234  001401                        BEQ     $102
2301                                                                   ;ERROR BIT SET IN HIGH BYTE OF RBUF
2302  005236                                                           ERRHRD 200
2303  005236  104200                        ERROR   200
2304  005240                                                          ENDIF
2305  005240                          $102:
2306
2307
2308                                                                  ;COMPARE DATA
2309  005240                                                      IF R3 NE R2 THEN
2310  005240  020302                        CMP     R3,R2
2311  005242  001415                        BEQ     $103
2312  005244                                                        IF #BIT0 NOTSETIN $USWR THEN
2313  005244  032737  000001  001220        BIT     #BIT0,$USWR
2314  005252  001006                        BNE     $104
2315  005254                                                            IF R2 LT #200 THEN
2316  005254  020227  000200                CMP     R2,#200
2317  005260  002002                        BGE     $105
2318                                                                       ;DATA COMPARE ERR IN 7 BIT WORD
2319  005262                                                               ERRHRD 117
2320  005262  104117                        ERROR   117
2321  005264                                                               EXIT
2322  005264  000416                        BR      TST16           ;;EXIT THIS TEST
2323  005266                                                            ENDIF
2324  005266                          $105:
2325  005266                                                        ELSE
2326  005266  000402                        BR      $106
2327  005270                          $104:
2328                                                                  ;DATA COMP ERR IN 8 BIT WORD
2329  005270                                                          ERRHRD 17
2330  005270  104017                        ERROR   17
2331  005272                                                          EXIT
2332  005272  000413                        BR      TST16           ;,EXIT THIS TEST
```

```
2333  005274                                                              ENDIF
2334  005274                        $106:
2335  005274                                                            ELSE IF #BIT0 NOTSETIN $USWR AND R2 GT #177 THEN
2336  005274  000411                       BR      $107
2337  005276                        $103:
2338  005276  032737  000001  001220       BIT     #BIT0,$USWR
2339  005304  001005                       BNE     $110
2340  005306  020227  000177               CMP     R2,#177
2341  005312  003402                       BLE     $110
2342                                                                      ;GETTING 8 BITS ON 7 BIT XMIT
2343                                                                      ;MAKE SURE $USWR SETUP CORRECTLY.
2344  005314                                                              ERRHRD 22
2345  005314  104022                       ERROR   22
2346  005316                                                              EXIT
2347  005316  000401                       BR      TST16           ;;EXIT THIS TEST
2348  005320                                                              ENDIF
2349  005320                        $110:
2350  005320                        $107:
2351  005320                                                        ENDINC  ;R2
2352  005320  000675                       BR      $76
2353  005322                        $77:
```

```
2354
2355                                    ;;*****************************************************************
2356                                    ;*TEST 16        TEST DATA WRAP AROUND: INTERRUPT MODE
2357                                    ;;*****************************************************************
2358  005322  000004                   TST16:  SCOPE
2359  005324  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
2360  005332  012737  000016  001200            MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2361  005340                                                                    IF #WRAP NOTSETIN $USWR OR CONSOL EQ #TRUE THEN
2362  005340  032737  000020  001220            BIT     #WRAP,$USWR
2363  005346  001404                           BEQ     $111
2364  005350  023727  013604  000001            CMP     CONSOL,#TRUE
2365  005356  001002                           BNE     $112
2366  005360                           $111:
2367  005360  000137  005772                   JMP     TST17                       ;EXIT TEST
2368  005364                                                                    ENDIF
2369  005364                           $112:
2370
2371                                            ; THIS TEST WILL RUN BOTH TRANSMITTER AND
2372                                            ; RECIEVER AT FULL SPEED TESTING
2373                                            ; THE ABILITY OF THE MODULE
2374                                            ; TO HANDLE INTERRUPTS FROM BOTH SIDES AT ONCE.
2375                                            ;
2376                                            ;DOUBLE BUFFERING IS NOT FULLY TESTED BECAUSE OF
2377                                            ;APT CONSIDERATIONS. I.E. 'BREAK' FROM APT
2378                                            ;CAUSES OVERRUN ERRORS. THEREFORE TRANSMIT INTR IS
2379                                            ;ENABLED ONLY AFTER THE RECVR HAS OBTAINED THE LAST WORD
2380
2381                                            ;  THIS TEST WILL TRANSFER A MAXIMUM OF 400(8)
2382                                            ; CHARACTERS THROUGH THE MODULE, BUT IF AN ERROR
2383                                            ; IS DETECTED BY THE TEST A PREMATURE SHUTDOWN OCCURS.
2384
2385                                                                            ;CHANGE PRIORITY
2386                                                                            ;..TO 0
2387  005364                               SETPRI #PR0
2388                                                                            ;GET VECTOR ADDRESS
2389  005376                                                                LET R1 := DLVEC
2390  005376  013701  001256                    MOV     DLVEC,R1
2391                                                                            ;RCVR VECTOR
2392  005402                                                                LET (R1)+ := #REC
2393  005402  012721  005664                    MOV     #REC,(R1)+
2394  005406                                                                LET (R1)+ := #PR7
2395  005406  012721  000340                    MOV     #PR7,(R1)+
2396                                                                            ;POINT TO TRANSMITTER VECTOR
2397                                                                            ;AND SET IT UP ALSO
2398  005412                                                                LET (R1)+ := #TRAN
2399  005412  012721  005626                    MOV     #TRAN,(R1)+
2400  005416                                                                LET (R1)  :  #PR7
2401  005416  012711  000340                    MOV     #PR7,(R1)
2402
2403                                                                            ; CLEAR ERROR COUNTER
2404  005422                                                                LET ERRCNT := #0
2405  005422  005037  005622                    CLR     ERRCNT
2406  005426                                                                LET DATA :  #0  ;XMIT DATA
2407  005426  005037  005662                    CLR     DATA
2408  005432                                                                IF #BIT0 NOTSETIN $USWR THEN
2409  005432  032737  000001  001220            BIT     #BIT0,$USWR
```

```
2410  005440  001004                          BNE     $113
2411  005442                                                          LET NUMBER := #200
2412  005442  012737  000200  005624          MOV     #200,NUMBER
2413  005450                                                          ELSE
2414  005450  000403                          BR      $114
2415  005452                          $113:
2416  005452                                                          LET NUMBER := #400
2417  005452  012737  000400  005624          MOV     #400,NUMBER
2418  005460                                                          ENDIF
2419  005460                          $114:
2420
2421  005460                                                          BRESET  ;SET UP ALL REGISTERS
2422  005460  000005                          RESET
2423
2424                                                                          ;SET I.E. IN TRANSMITTER
2425  005462                                                          LET @TCSR := @TCSR SET.BY #XMITIE
2426  005462  052777  000100  173574          BIS     #XMITIE,@TCSR
2427                                                                          ;AND RECEIVER
2428  005470                                                          LET @RCSR := @RCSR SET.BY #RCVRIE
2429  005470  052777  000100  173562          BIS     #RCVRIE,@RCSR
2430
2431
2432                                                                  ;NOW WE WAIT
2433  005476                                                          REPEAT
2434  005476                          $115:
2435  005476                                                          UNTIL DATA EQ NUMBER OR ERRCNT GT #0
2436  005476  023737  005662  005624          CMP     DATA,NUMBER
2437  005504  001403                          BEQ     $116
2438  005506  005737  005622                  TST     ERRCNT
2439  005512  003771                          BLE     $115
2440  005514                          $116:
2441
2442                                                                  ;NOW LETS CHECK.
2443                                                                  ;TURN OFF ALL INTR ENABLE
2444  005514                                                          LET @TCSR := @TCSR CLR.BY #XMITIE
2445  005514  042777  000100  173542          BIC     #XMITIE,@TCSR
2446  005522                                                          LET @RCSR := @RCSR CLR.BY #RCVRIE
2447  005522  042777  000100  173530          BIC     #RCVRIE,@RCSR
2448  005530                                                          IF ERRCNT NE #0 THEN
2449  005530  005737  005622                  TST     ERRCNT
2450  005534  001431                          BEQ     $117
2451  005536                                                              IF #ERROR SETIN RHLD THEN
2452  005536  032737  100000  005770          BIT     #ERROR,RHLD
2453  005544  001424                          BEQ     $120
2454  005546                                                                  IF #ORERR SETIN RHLD THEN
2455  005546  032737  040000  005770          BIT     #ORERR,RHLD
2456  005554  001402                          BEQ     $121
2457                                                                              ;OVERRUN ERROR
2458  005556                                                                      ERRHRD 220
2459  005556  104220                          ERROR   220
2460  005560                                                                  ELSE IF #FRERR SETIN RHLD THEN
2461  005560  000415                          BR      $122
2462  005562                          $121:
2463  005562  032737  020000  005770          BIT     #FRERR,RHLD
2464  005570  001402                          BEQ     $123
2465                                                                              ;FRAMING ERROR
```

```
2466  005572                                                        ERRHRD 221
2467  005572  104221                    ERROR   221
2468  005574                                                      ELSE IF #PERR SETIN RHLD THEN
2469  005574  000407                    BR      $124
2470  005576                  $123:
2471  005576  032737  010000  005770    BIT     #PERR,RHLD
2472  005604  001402                    BEQ     $125
2473                                                              ;PARITY ERROR
2474  005606                                                      ERRHRD 222
2475  005606  104222                    ERROR   222
2476  005610                                                    ELSE
2477  005610  000401                    BR      $126
2478  005612                  $125:
2479                                                              ;UNKNOWN ERROR
2480  005612                                                      ERRHRD 24
2481  005612  104024                    ERROR   24
2482  005614                                                    ENDIF
2483  005614                  $126:
2484  005614                  $124:
2485  005614                  $122:
2486  005614                                                  ELSE
2487  005614  000401                    BR      $127
2488  005616                  $120:
2489                                                                ;DATA COMPARE ERROR
2490  005616                                                        ERRHRD 120
2491  005616  104120                    ERROR   120
2492  005620                                                  ENDIF
2493  005620                  $127:
2494  005620                                                ENDIF
2495  005620                  $117:
2496  005620                                                EXIT    ;SKIP OVER SUPPORT ROUTINES & STORAGE
2497  005620  000464                    BR      TST17           ;;EXIT THIS TEST
2498
2499
2500  005622  000000          ERRCNT: 0
2501  005624  000000          NUMBER: 0
```

```
                                ;;*****************************************************************
                                ;TRANSMIT INTERRUPT HANDLER
                                ;;*****************************************************************
005626                                                  BGNSRV  TRAN
005626                          TRAN:
005626                                                      IF DATA NE NUMBER AND ERRCNT EQ #0 THEN
005626  023737  005662  005624          CMP     DATA,NUMBER
005634  001406                        BEQ     $130
005636  005737  005622                TST     ERRCNT
005642  001003                        BNE     $130
                                                                ;SHIP OUT WORD
005644                                                          LET @TBUF :- DATA
005644  013777  005662  173414          MOV     DATA,@TBUF
005652                                                      ENDIF
005652                          $130:
                                                            ;STOP INTERR, NOT EXER DOUBL BUFFER
005652                                                      LET @TCSR :- @TCSR CLR.BY #XMITIE
005652  042777  000100  173404          BIC     #XMITIE,@TCSR
005660                                                  ENDSRV
005660  000002                        RTI

005662  000000                  DATA:   0

                                ;;*****************************************************************
                                ;RECEIVER INTERRUPT HANDLER
                                ;;*****************************************************************
005664                                                  BGNSRV  REC
005664                          REC:
                                                            ;GET CHAR
005664                                                      LET RHLD :- @RBUF
005664  017737  173372  005770          MOV     @RBUF,RHLD
                                                            ;CHECK ERROR
005672                                                      IF #ERROR SETIN RHLD OR RHLD NE  DATA THEN
005672  032737  100000  005770          BIT     #ERROR,RHLD
005700  001004                        BNE     $131
005702  023737  005770  005662          CMP     RHLD,DATA
005710  001411                        BEQ     $132
005712                          $131:
                                                               ;STOP ALL INTERR PROC & GET OUT
005712                                                         LET DATA := NUMBER
005712  013737  005624  005662          MOV     NUMBER,DATA
005720                                                         LET @RCSR := @RCSR CLR.BY #RCVRIE
005720  042777  000100  173332          BIC     #RCVRIE,@RCSR
005726                                                         LET ERRCNT := ERRCNT + #1
005726  005237  005622                INC     ERRCNT
005732                                                      ELSE
005732  000415                        BR      $133
005734                          $132:
                                                               LET DATA := DATA + #1
005734  005237  005662                INC     DATA
005740                                                            IF DATA EQ NUMBER THEN
005740  023737  005662  005624          CMP     DATA,NUMBER
005746  001004                        BNE     $134
005750                                                               LET @RCSR :  @RCSR CLR.BY #RCVRIE
005750  042777  000100  173302          BIC     #RCVRIE,@RCSR
```

```
2558  005756                                                          ELSE
2559  005756  000403                    BR      $135
2560  005760                    $134:
2561                                                                    ;ALLOW NEXT XMIT INTERR
2562  005760                                                            LET @TCSR := @TCSR SET.BY #XMITIE
2563  005760  052777  000100  173276    BIS     #XMITIE,@TCSR
2564  005766                                                          ENDIF
2565  005766                    $135:
2566  005766                                                  ENDIF
2567  005766                    $133:
2568  005766                                              ENDSRV
2569  005766  000002                    RTI
2570
2571  005770  000000            RHLD:   0
2572
```

```
                                        ;;*****************************************************************
                                        ;*TEST 17        TEST BREAK DETECTION LOGIC
                                        ;*              TRANSMIT KNOWN CHAR WITH BREAK SET
                                        ;*              AND COMPARE RECEIVED WITH 0.
                                        ;*              FRAMING ERROR WILL ALSO BE CHECKED
                                        ;*              IF ERROR BITS ARE ENABLED.
                                        ;;*****************************************************************
005772  000004                TST17:  SCOPE
005774  012737  000010  001160        MOV     #10,$TIMES      ;;DO 10 ITERATIONS
006002  012737  000017  001200        MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                                                ;DONT DO THIS TEST IF 'BREAK' GENERATION
                                                                ;ENABLED, ELSE WILL HALT TO CONSOLE ODT.
                                                                ;DO IF BREAK 'DETECTION' IS ENABLED.
006010                                                          IF #WRAP NOTSETIN $USWR OR #BRK NOTSETIN $USWR THEN
006010  032737  000020  001220        BIT     #WRAP,$USWR
006016  001404                      BEQ     $136
006020  032737  000010  001220        BIT     #BRK,$USWR
006026  001003                      BNE     $137
006030                          $136:
006030  000137  006502              JMP     TST20                           ;EXIT TEST
006034                                                          ELSE
006034  000406                      BR      $140
006036                          $137:
006036                                                                  IF CONSOL EQ #TRUE THEN
006036  023727  013604  00000'      CMP     CONSOL,#TRUE
006044  001002                      BNE     $141
                                                                           ;CAN'T TEST CONSOLE
006046  000137  006502              JMP     TST20                        ;EXIT TEST
006052                                                                  ENDIF
006052                          $141:
006052                                                          ENDIF
006052                          $140:
006052                                                          BGNSUB
006052  012737  006060  001110        MOV     #64$,$LPERR
006060                                                                LET ERRCHK := #0        , CLEAR ERROR WORD
006060  005037  006500              CLR     ERRCHK
                                                                        ;SET BREAK BIT
006064                                                           LET @TCSR :- @TCSR SET.BY #BREAK
006064  052777  000001  173'72        BIS     #BREAK,@TCSR
                                                                        ;NON-ZERO CHAR. '*'
006072                                                          LET @TBUF :- #125
006072  012777  000125  173166        MOV     #125,@TBUF
                                                                        ;WAIT FOR DONE
006100                                                          CALL TIMER IN <#500,#RCVRDONE,RCSR,#SET>
006100  010546                      MOV     R5,-(SP)
006102  012745  177777              MOV     #SET,-(R5)
006106  013745  001260              MOV     RCSR,-(R5)
006112  012745  000200              MOV     #RCVRDONE,-(R5)
006116  012745  000500              MOV     #500,-(R5)
006122  004737  012446              JSR     PC,TIMER
006126  012605                      MOV     (SP)+,R5
006130
006130  103001                      BCC     $'2                     IF.ERROR THEN
                                                                        ; RECIEVER DONE DID NOT SE'
```

```
2629  006132
2630  006132  104115                          ERROR   115
2631  006134
2632  006134                        $142:
2633  006134
2634  006134  017700  173122                  MOV     @RBUF,R0
2635  006140
2636  006140  105700                          TSTB    R0
2637  006142  001403                          BEQ     $143
2638
2639  006144
2640  006144  052737  000001  006500          BIS     #BIT0,ERRCHK
2641  006152
2642  006152                        $143:
2643  006152
2644  006152  032700  020000                  BIT     #FRERR,R0
2645  006156  001003                          BNE     $144
2646  006160
2647  006160  052737  000002  006500          BIS     #BIT1,ERRCHK
2648  006166
2649  006166                        $144:
2650  006166
2651  006166  032737  000002  001220          BIT     #PARITY,$USWR
2652  006174  001421                          BEQ     $145
2653
2654  006176
2655  006176  032737  000004  001220          BIT     #EVENODD,$USWR
2656  006204  001007                          BNE     $146
2657
2658  006206
2659  006206  032700  010000                  BIT     #PERR,R0
2660  006212  001003                          BNE     $147
2661
2662  006214
2663  006214  052737  000004  006500          BIS     #BIT2,ERRCHK
2664  006222
2665  006222                        $147:
2666  006222
2667  006222  000406                          BR      $150
2668  006224                        $146:
2669  006224
2670  006224  032700  010000                  BIT     #PERR,R0
2671  006230  001403                          BEQ     $151
2672  006232
2673  006232  052737  000010  005622          BIS     #BIT3,ERRCNT
2674  006240
2675  006240                        $151:
2676  006240
2677  006240                        $150:
2678  006240
2679  006240                        $145:
2680
2681  006240
2682  006240  000005                          RESET
2683  006242  032777  170000  173012          BIT     #^C70000,@RBUF
2684  006250  001401                          BEQ     1$
```

```
    ERRHRD 115
ENDIF

LET R0 := @RBUF

IFB R0 NE #0 THEN

        ; BREAK DID NOT EQUAL 0
    LET ERRCHK :- ERRCHK SET.BY #BIT0

ENDIF

IF #FRERR NOTSETIN R0 THEN

        LET ERRCHK :  ERRCHK SET.BY #BIT1

ENDIF

IF #PARITY SETIN $USWR THEN

            ;ODD PARITY ENABLED
    IF #EVENODD NOTSETIN $USWR THEN

            ;BREAK SHOULD GENERATE A PARITY ERRO
        IF #PERR NOTSETIN R0 THEN

                ;NO PAR ERROR WHEN THERE SHOULD
             LET ERRCHK :  ERRCHK SET.BY #BIT2

        ENDIF

    ELSE

        IF #PERR SETIN R0 THEN

            LET ERRCNT :- ERRCNT SET.BY #BIT3

        ENDIF

    ENDIF

ENDIF

BRESET  ;CLEAN UP
```

```
2685  006252  104033                        ERROR   33              ;RESET DID NOT CLEAR ERROR,FR ERR,OR PERR IN RBUF
2686  006254                          1$:
2687
2688  006254                                                          IF #BIT0 SETIN ERRCHK THEN
2689  006254  032737  000001  006500        BIT     #BIT0,ERRCHK
2690  006262  001401                        BEQ     $152
2691  006264                                                              ERRHRD 121  ;BREAK ERROR
2692  006264  104121                        ERROR   121
2693  006266                                                          ENDIF
2694  006266                          $152:
2695  006266                                                          IF #BIT1 SETIN ERRCHK THEN
2696  006266  032737  000002  006500        BIT     #BIT1,ERRCHK
2697  006274  001401                        BEQ     $153
2698  006276                                                              ERRHRD 122  ; FRAMING ERROR
2699  006276  104122                        ERROR   122
2700  006300                                                          ENDIF
2701  006300                          $153:
2702  006300                                                          IF #BIT2 SETIN ERRCHK THEN
2703  006300  032737  000004  006500        BIT     #BIT2,ERRCHK
2704  006306  001401                        BEQ     $154
2705  006310                                                              ERRHRD  235
2706  006310  104235                        ERROR   235
2707                                                                        ;NO PARITY ERROR WHEN
2708                                                                        ;THERE SHOULD BE
2709  006312                                                          ENDIF
2710  006312                          $154:
2711  006312                                                          IF #BIT3 SETIN ERRCHK THEN
2712  006312  032737  00001C  006500        BIT     #BIT3,ERRCHK
2713  006320  001401                        BEQ     $155
2714  006322                                                              ERRHRD 236
2715  006322  104236                        ERROR   236
2716                                                                        ;PARITY ERROR SHOULD NOT HAVE
2717                                                                        ;OCCURED WITH EVEN PARITY
2718                                                                        ;ENABLED AND BREAK SET
2719  006324                                                          ENDIF
2720  006324                          $155:
2721  006324                                                  ENDSUB
2722  006324                                                  BGNSUB
2723  006324  012737  006332  001110        MOV     #64$,$LPERR
2724                                                                        ;SET BREAK BIT
2725  006332                                                          LET @TCSR := @TCSR SET.BY #BREAK
2726  006332  052777  000001  172724        BIS     #BREAK,@TCSR
2727
2728  006340                                                          CALL TIMER IN <#500,#BREAK,TCSR,#SET>
2729  006340  010546                        MOV     R5,-(SP)
2730  006342  012745  177777                MOV     #SET,-(R5)
2731  006346  013745  001264                MOV     TCSR,-(R5)
2732  006352  012745  000001                MOV     #BREAK,-(R5)
2733  006356  012745  000500                MOV     #500,-(R5)
2734  006362  004737  012446                JSR     PC,TIMER
2735  006366  012605                        MOV     (SP)+,R5
2736
2737                                                                        ;DID IT SET?
2738  006370                                                          IF.ERROR THEN
2739  006370  103001                        BCC     $456
2740                                                                        ;BREAK DID NOT SET
```

```
2741  006372                                                          ERRHRD 21
2742  006372  104021                    ERROR   21
2743  006374                                                        ENDIF
2744  006374                    $156:
2745
2746                                                                  ;CLEAR  BREAK BIT
2747  006374                                                        LET @TCSR := @TCSR CLR.BY #BREAK
2748  006374  042777  000001  172662    BIC     #BREAK,@TCSR
2749  006402                                                          WAITMS 100.
2750  006402  010546                    MOV     R5,-(SP)
2751  006404  012745  000144            MOV     #100.,-(R5)
2752  006410  004737  012620            JSR     PC,WAIT
2753  006414  012605                    MOV     (SP)+,R5
2754
2755
2756                                                                  ;READ RBUF TO CLEAR ERRORS & REC DONE
2757  006416                                                        LET R0 := @RBUF
2758  006416  017700  172640            MOV     @RBUF,R0
2759                                                                  ;SEND CHAR
2760  006422                                                        LET @TBUF := #125
2761  006422  012777  000125  172636    MOV     #125,@TBUF
2762                                                                  ;WAIT FOR DONE BIT
2763  006430                                                        CALL  TIMER IN <#500,#RCVRDONE,RCSR,#SET>
2764  006430  010546                    MOV     R5,-(SP)
2765  006432  012745  177777            MOV     #SET,-(R5)
2766  006436  013745  001260            MOV     RCSR,-(R5)
2767  006442  012745  000200            MOV     #RCVRDONE,-(R5)
2768  006446  012745  000500            MOV     #500,-(R5)
2769  006452  004737  012446            JSR     PC,TIMER
2770  006456  012605                    MOV     (SP)+,R5
2771  006460                                                        IF.ERROR THEN
2772  006460  103001                    BCC     $157
2773                                                                  ;RECEIVER NEVER CAME READY
2774  006462                                                          ERRHRD 230
2775  006462  104230                    ERROR   230
2776  006464                                                        ENDIF
2777  006464                    $157:
2778
2779                                                                  ;WAS CHAR AFTER BREAK RECEIVED
2780  006464                                                        IFB @RBUF NE #125 THEN
2781  006464  127727  172572  000125    CMPB    @RBUF,#125
2782  006472  001401                    BEQ     $160
2783                                                                  ;CHAR AFTER BREAK NOT RECEIVED CORRECTLY
2784  006474                                                          ERRHRD 231
2785  006474  104231                    ERROR   231
2786  006476                                                        ENDIF
2787  006476                    $160:
2788  006476                                                    ENDSUB
2789  006476                                                    EXIT
2790  006476  000401                    BR      TST20           ;;EXIT THIS TEST
2791  006500  000000            ERRCHK: .WORD 0
```

```
                                ;;*****************************************************************
                                ;*TEST 20       NOT A TEST - SEND BACK TO LOOP
                                ;;*****************************************************************
006502  000004                  TST20:  SCOPE
006504  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
006512                                                                  IF #BIT12 SETIN @SWR THEN
006512  032777  010000  172420          BIT     #BIT12,@SWR
006520  001513                          BEQ     $161
006522                                                                      IF PHASE2 EQ #TRUE  THEN
006522  023727  014166  000001          CMP     PHASE2,#TRUE
006530  001021                          BNE     $162
006532                                  TYPTXT  <<CRLF>/        ** PHASE 2 SUMMARY **/<CRLF>>
006572                                                                        ELSE
006572  000425                          BR      $163
006574                                  $162:
006574                                                                          IF P1CNT EQ #0 THEN
006574  005737  014170                  TST     P1CNT
006600  001022                          BNE     $164
006602                                  TYPTXT  <<CRLF>/        ** PHASE 1 SUMMARY **/<CRLF>>
006642                                                                              LET P1CNT := P1CNT + #1
006642  005237  014170                  INC     P1CNT
006646                                                                          ENDIF
006646                                  $164:
006646                                                                      ENDIF
006646                                  $163:
006646                                  TYPTXT  <*CSR: *>
006662                                  TYPOCT  DLADD
006670                                  TYPTXT  <*,VECTOR: *>
006710                                  TYPOCT  DLVEC
006716                                  TYPTXT  <*,ERRORS: *>
006736                                  TYPDEC  $ERTTL
006744  104401  001171                  TYPE    ,$CRLF
006750                                                              ENDIF
006750                                  $161:
006750  005037  001112                  CLR     $ERTTL          ; RESET FOR NEXT DEVICE/PASS
006754                                                              IF PHASE2 EQ #TRUE THEN
006754  023727  014166  000001          CMP     PHASE2,#TRUE
006762  001004                          BNE     $165
006764                                                                LET PHASE2 :- #0
006764  005037  014166                  CLR     PHASE2
006770                                                                LET $UNIT :  $UNIT + #1
006770  005237  001206                  INC     $UNIT
006774                                                              ENDIF
006774                                  $165:
006774  000137  001732                  JMP     LOOP            ; BACK UP TO THE BEGINNING
```

```
                                ;*****************************************************************
                                ;*TEST 21       TEST THAT CHANNELS INTR AT ASSIGNED PRIORITY
                                ;*             INTERRUPTS WILL BE ENABLED ON ALL ACTIVE CHANNELS.
                                ;*             RECEIVER AND TRANSMITTER. THEN WE'LL CHECK TO
                                ;*             SEE IF THEY INTERRUPTED IN THE ASSIGNED SEQUENCE.
                                ;*****************************************************************
007000  000004                  TST21:  SCOPE
007002  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
007010  012737  000021  001200          MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX


                                                                        ;CLEAR OUT INTERRUPT TABLE
007016                                                          LET R0 := #INTRTABLE
007016  012700  007744                  MOV     #INTRTABLE,R0
007022                                                          REPEAT
007022                          $166:
007022                                                              LET (R0)+ := #0
007022  005020                          CLR     (R0)+
007024                                                          UNTIL R0 EQ #TABEND
007024  020027  007764                  CMP     R0,#TABEND
007030  001374                          BNE     $166


                                                                    ;SET PRIORITY TO 7
007032                                  SETPRI #PR7


                                                                    ;SET UP ALL INTERRUPT VECTORS
007044                                                          LET R0 := DLVEC
007044  013700  001256                  MOV     DLVEC,R0
007050                                                          LET R1 := #RCVR0SRV
007050  012701  007424                  MOV     #RCVR0SRV,R1
007054                                                          REPEAT
007054                          $167:
007054                                                              LET (R0)+ := R1
007054  010120                          MOV     R1,(R0)+
007056                                                              LET (R0)+ := #PR7
007056  012720  000340                  MOV     #PR7,(R0)+
007062                                                              LET R1 := R1 + #30
007062  062701  000030                  ADD     #30,R1
007066                                                          UNTIL R1 EQ #SRVEND
007066  020127  007724                  CMP     R1,#SRVEND
007072  001370                          BNE     $167


                                                                    ;ENABLE INTERRUPTS ON ALL ACTIVE LINES,
                                                                    ;ALSO, XMIT CHAR'S TO PRIME RECEIVERS

                                                                ;COPY MASK
007074                                                          LET CHMASK := MASK
007074  013737  014162  007764          MOV     MASK,CHMASK
007102                                                          LET CHCNT := #0
007102  005037  007766                  CLR     CHCNT
007106                                                          LET R0 := DLADD
007106  013700  001254                  MOV     DLADD,R0
007112                                                          REPEAT
007112                          $170:
```

```
 2894  007112
 2895  007112  032737  000001  007764          BIT     #BIT0,CHMASK
 2896  007120  001430                          BEQ     $171
 2897
 2898  007122
 2899  007122  052710  000100                  BIS     #BIT6,(R0)
 2900  007126
 2901  007126  062700  000004                  ADD     #4,R0
 2902
 2903  007132
 2904  007132  052710  000100                  BIS     #BIT6,(R0)
 2905
 2906  007136
 2907  007136  012760  000252  000002          MOV     #252,2(R0)
 2908
 2909  007144
 2910  007144  162700  000004                  SUB     #4,R0
 2911
 2912  007150
 2913  007150  010546                          MOV     R5,-(SP)
 2914  007152  012745  177777                  MOV     #SET,-(R5)
 2915  007156  010045                          MOV     R0,-(R5)
 2916  007160  012745  000200                  MOV     #RCVRDONE,-(R5)
 2917  007164  012745  000500                  MOV     #500,-(R5)
 2918  007170  004737  012446                  JSR     PC,TIMER
 2919  007174  012605                          MOV     (SP)+,R5
 2920
 2921  007176
 2922  007176  103001                          BCC     $172
 2923
 2924  007200
 2925  007200  104023                          ERROR   23
 2926  007202
 2927  007202                          $172:
 2928  007202
 2929  007202                          $171:
 2930
 2931  007202
 2932  007202  062700  000010                  ADD     #10,R0
 2933
 2934  007206
 2935  007206  006237  007764                  ASR     CHMASK
 2936  007212
 2937  007212  005237  007766                  INC     CHCNT
 2938  007216
 2939  007216  023727  007766  000003          CMP     CHCNT,#3
 2940  007224  001007                          BNE     $173
 2941  007226
 2942  007226  004737  013530                  JSR     PC,TSTCON
 2943  007232
 2944  007232  005737  013604                  TST     CONSOLE
 2945  007236  001402                          BEQ     $174
 2946  007240
 2947  007240  005237  007766                  INC     CHCNT
 2948  007244
 2949  007244                          $174:
```

```
IF #BIT0 SETIN CHMASK THEN


    ;SET RCSR IE
    LET (R0) := (R0) SET.BY #BIT6

    LET R0 := R0 + #4

    ;SET XCSR IE
    LET (R0) := (R0) SET.BY #BIT6

    ;LOAD XBUF
    LET 2(R0) :  #252

    ;GO BACK TO RCSR
    LET R0 := R0 - #4

    ;LOOK FOR RCVR DONE
    CALL TIMER IN <#500,#RCVRDONE,R0,#SET>




            ;DID IT SET IN TIME?
    IF.ERROR THEN

            ;RCVRDONE DID NOT SET IN RCSR
            ERRHRD 23

    ENDIF

ENDIF


    LET R0 :  R0 + #10

;SETUP FOR NEXT CHANNEL
LET CHMASK := CHMASK SHIFT -1

LET CHCNT :  CHCNT + #1

IF CHCNT EQ #3 THEN


    CALL TSTCON

    IF CONSOLE NE #0 THEN


            LET CHCNT := CHCNT + #1

    ENDIF
```

```
                                                        ENDIF
                                $173:
                                                        UNTIL CHCNT EQ #4
007244  023727  007766  000004            CMP     CHCNT,#4
007252  001317                            BNE     $170

                                        ;ALL XMIT & REC INTERRUPTS SHOULD BE
                                        ;IN CONTENTION.
                                        ;RECEIVE INTERRUPTS HAVE PRIORITY
                                        ;OVER XMIT INTERRUPTS.
                                        ;LOW CHANNELS HAVE PRIORITY OVER HIGH CHANNELS.
                                        ;THEREFORE, ONCE CPU PRIORITY IS LOWERED,
                                        ;INTERRUPTS S/B IN THE ORDER SHOWN IN THE
                                        ;CHANNEL IDENTIFIER TABLE (RCHO:)

                                                        ;SET UP POINTER FOR SERVICE ROUTINE
007254                                                  LET R4 := #INTRTABLE
007254  012704  007744                MOV     #INTRTABLE,R4

                                                        ;LET EM GO
007260                                SETPRI  #PR0

                                                        ;DISABLE ALL INTERRUPTS
007272                                                  BRESET
007272  000005                        RESET

007274                                                  LET TCNT := #0
007274  005037  007772                CLR     TCNT
                                                        ;CLEAR ERR FLAG
007300                                                  LET PRIERR := #0
007300  005037  007770                CLR     PRIERR
007304                                                  LET R0 := #INTRTABLE
007304  012700  007744                MOV     #INTRTABLE,R0
                                                        ;SETUP EXPECTED VALUE
007310                                                  LET R1 :  #INTRTABLE+2
007310  012701  007746                MOV     #INTRTABLE+2,R1
007314  013737  014162  007764        MOV     MASK,CHMASK     ;GET # OF CHANNELS IN TCNT

007322  006237  007764        1$:     ASR     CHMASK
007326  103002                        BCC     2$
007330  005237  007772                INC     TCNT            ;GET TOTAL CHANNEL CT
007334  005737  007764        2$:     TST     CHMASK
007340  001370                        BNE     1$

007342  004737  013530                JSR     PC,TSTCON       ;CORRECT CHANNEL CT FOR CONSOLE
007346  005737  013604                TST     CONSOLE
007352  001402                        BEQ     3$
007354  005337  007772                DEC     TCNT
007360  006337  007772        3$:     ASL     TCNT            ;APPLY X2 MULT FACTOR

007364  005337  007772        4$:     DEC     TCNT
007370  001410                        BEQ     5$
007372  023727  007772  177777        CMP     TCNT,#-1        ;ALL 1'S ?
007400  001404                        BEQ     5$              ;BR IF YES
007402  022120                        CMP     (R1)+,(R0)+     ;R1-R0
007404  003367                        BGT     4$              ;BR IF R1>R0
```

```
 3006  007406  005237  007770                  INC     PRIERR          ;ELSE ERROR
 3007
 3008  007412                            5$:
 3009  007412                                                          IF PRIERR NE #0 THEN
 3010  007412  005737  007770                  TST     PRIERR
 3011  007416  001401                          BEQ     $175
 3012  007420                                                              ERRHRD 250
 3013  007420  104250                          ERROR   250
 3014                                                                    ;CHANNELS DID NOT INTR ACCORDING TO
 3015                                                                    ;ASSIGNED PRIORITY. THE BYTE ENTRIES
 3016                                                                    ;IN INTRTABLE: SHOULD BE IN THE ORDER
 3017                                                                    ;THAT THEY APPEAR IN THE CHANNEL #
 3018                                                                    ;TABLE .(EXCLUDING CHANNELS NOT ACTIVE)
 3019  007422                                                          ENDIF
 3020  007422                            $175:
 3021
 3022  007422                                                          EXIT
 3023  007422  000565                          BR      TST22           ;;EXIT THIS TEST
```

```
                                ;;*****************************************************************
                                ;*              START OF SERVICE ROUTINES
                                ;;*****************************************************************

007424                          RCVR0SRV:                               BGNSRV RCVR0SRV
                                                                            ;PUT INTR IDENTIFIER IN INTRTABLE
                                                                        LET (R4)+ := RCH0
007424 013724 007724                    MOV     RCH0,(R4)+
                                                                            ;GENERATE CSR ADDRESS
007430                                                                  LET TEMP :  RCSR
007430 013737 001260 007774             MOV     RCSR,TEMP
007436                                                                  LET TEMP := TEMP + #0
007436 062737 000000 007774             ADD     #0,TEMP
                                                                            ;ONE INTR IS ALL WE WANT FROM HERE
007444                                                                  LET @TEMP :  @TEMP CLR.BY #BIT6
007444 042777 000100 000322             BIC     #BIT6,@TEMP
007452                                                                  ENDSRV
007452 000002                           RTI

007454                          XMIT0SRV:                               BGNSRV  XMIT0SRV
                                                                            ;PUT INTR IDENTIFIER IN INTRTABLE
007454                                                                  LET (R4)+ :  TCH0
007454 013724 007734                    MOV     TCH0,(R4)+
                                                                            ;GENERATE CSR ADDRESS
007460                                                                  LET TEMP := RCSR
007460 013737 001260 007774             MOV     RCSR,TEMP
007466                                                                  LET TEMP := TEMP + #4
007466 062737 000004 007774             ADD     #4,TEMP
                                                                            ;ONE INTR IS ALL WE WANT FROM HERE
007474                                                                  LET @TEMP := @TEMP CLR.BY #BIT6
007474 042777 000100 000272             BIC     #BIT6,@TEMP
007502                                                                  ENDSRV
007502 000002                           RTI

007504                          RCVR1SRV:                               BGNSRV  RCVR1SRV
                                                                            ;PUT INTR IDENTIFIER IN INTRTABLE
007504                                                                  LET (R4)+ := RCH1
007504 013724 007726                    MOV     RCH1,(R4)+
                                                                            ;GENERATE CSR ADDRESS
007510                                                                  LET TEMP := RCSR
007510 013737 001260 007774             MOV     RCSR,TEMP
007516                                                                  LET TEMP := TEMP + #10
007516 062737 000010 007774             ADD     #10,TEMP
                                                                            ;ONE INTR IS ALL WE WANT FROM HERE
007524                                                                  LET @TEMP := @TEMP CLR.BY #BIT6
007524 042777 000100 000242             BIC     #BIT6,@TEMP
007532                                                                  ENDSRV
007532 000002                           RTI

007534                          XMIT1SRV:                               BGNSRV  XMIT1SRV
```

```
3080
3081  007534
3082  007534  013724  007736          MOV     TCH1,(R4)+
3083
3084  007540
3085  007540  013737  001260  007774  MOV     RCSR,TEMP
3086  007546
3087  007546  062737  000014  007774  ADD     #14,TEMP
3088
3089  007554
3090  007554  042777  000100  000212  BIC     #BIT6,@TEMP
3091  007562
3092  007562  000002                  RTI
3093
3094  007564                                                  BGNSRV  RCVR2SRV
3095  007564                  RCVR2SRV:
3096
3097  007564
3098  007564  013724  007730          MOV     RCH2,(R4)+
3099
3100  007570
3101  007570  013737  001260  007774  MOV     RCSR,TEMP
3102  007576
3103  007576  062737  000020  007774  ADD     #20,TEMP
3104
3105  007604
3106  007604  042777  000100  000162  BIC     #BIT6,@TEMP
3107  007612                                                  ENDSRV
3108  007612  000002                  RTI
3109
3110  007614                                                  BGNSRV  XMIT2SRV
3111  007614                  XMIT2SRV:
3112
3113  007614
3114  007614  013724  007740          MOV     TCH2,(R4)+
3115
3116  007620
3117  007620  013737  001260  007774  MOV     RCSR,TEMP
3118  007626
3119  007626  062737  000024  007774  ADD     #24,TEMP
3120
3121  007634
3122  007634  042777  000100  000132  BIC     #BIT6,@TEMP
3123  007642                                                  ENDSRV
3124  007642  000002                  RTI
3125
3126  007644                                                  BGNSRV RCVR3SRV
3127  007644                  RCVR3SRV:
3128
3129  007644
3130  007644  013724  007732          MOV     RCH3,(R4)+
3131
3132  007650
3133  007650  013737  001260  007774  MOV     RCSR,TEMP
3134  007656
3135  007656  062737  000030  007774  ADD     #30,TEMP

                                                        ;PUT INTR IDENTIFIER IN INTRTABLE
                                                    LET (R4)+ := TCH1

                                                        ;GENERATE CSR ADDRESS
                                                    LET TEMP := RCSR

                                                    LET TEMP := TEMP + #14

                                                        ;ONE INTR IS ALL WE WANT FROM HERE
                                                    LET @TEMP := @TEMP CLR.BY #BIT6

                                                        ;PUT INTR IDENTIFIER IN INTRTABLE
                                                    LET (R4)+ := RCH2

                                                        ;GENERATE CSR ADDRESS
                                                    LET TEMP := RCSR

                                                    LET TEMP := TEMP + #20

                                                        ;ONE INTR IS ALL WE WANT FROM HERE
                                                    LET @TEMP := @TEMP CLR.BY #BIT6

                                                        ;PUT INTR IDENTIFIER IN INTRTABLE
                                                    LET (R4)+ := TCH2

                                                        ;GENERATE CSR ADDRESS
                                                    LET TEMP := RCSR

                                                    LET TEMP := TEMP + #24

                                                        ;ONE INTR IS ALL WE WANT FROM HERE
                                                    LET @TEMP :  @TEMP CLR.BY #BIT6

                                                        ;PUT INTR IDENTIFIER IN INTRTABLE
                                                    LET (R4)+ := RCH3

                                                        ;GENERATE CSR ADDRESS
                                                    LET TEMP := RCSR

                                                    LET TEMP := TEMP + #30
```

```
                                                                    ;ONE INTR IS ALL WE WANT FROM HERE
                                                                LET @TEMP := @TEMP CLR.BY #BIT6
007664  042777  000100  000102          BIC     #BIT6,@TEMP
                                                        ENDSRV
007672  000002                          RTI
                                                        BGNSRV  XMIT3SRV
007674                          XMIT3SRV:
                                                                    ;PUT INTR IDENTIFIER IN INTRTABLE
                                                                LET (R4)+ := TCH3
007674  013724  007742                  MOV     TCH3,(R4)+
                                                                    ;GENERATE CSR ADDRESS
                                                                LET TEMP := RCSR
007700  013737  001260  007774          MOV     RCSR,TEMP
                                                                LET TEMP := TEMP + #34
007706  062737  000034  007774          ADD     #34,TEMP
                                                                    ;ONE INTR IS ALL WE WANT FROM HERE
                                                                LET @TEMP :  @TEMP CLR.BY #BIT6
007714  042777  000100  000052          BIC     #BIT6,@TEMP
                                                        ENDSRV
007722  000002                          RTI
007724                          SRVEND:
```

```
3158
3159
3160
3161                                    ;**CHANNEL IDENTIFIER TABLE**
3162                                    ;THE ENTRIES FROM THIS
3163                                    ;TABLE ARE PLACED IN THE INTRTABLE
3164                                    ;IN THE ORDER THAT THE INTR'S
3165                                    ;OCCUR. (EXCLUDING NON-ACTIVE CH'S)
3166  007724  000000                RCH0:   .WORD 0
3167  007726  000001                RCH1:   .WORD 1
3168  007730  000002                RCH2:   .WORD 2
3169  007732  000003                RCH3:   .WORD 3
3170  007734  000010                TCH0:   .WORD 10
3171  007736  000011                TCH1:   .WORD 11
3172  007740  000012                TCH2:   .WORD 12
3173  007742  000013                TCH3:   .WORD 13
3174
3175                                    ;THIS TABLE WILL CONTAIN ENTRIES
3176                                    ;REPRESENTING EACH INTR IN THE ORDER
3177                                    ;THAT IT OCCURED
3178                                    ; S/B IN THE ABOVE ORDER
3179                                    ;
3180  007744  000010                INTRTABLE:      .BLKW   8.
3181  007764                        TABEND:
3182
3183  007764  000000                CHMASK: .WORD 0         ;BITS 3-0 INDICATE WHICH CH'S ARE ACTIVE FOR THIS TEST
3184  007766  000000                CHCNT:  .WORD 0         ;CHANNEL COUNTER
3185  007770  000000                PRIERR: .WORD 0         ;ERROR WORD, CONTAINS NO. OF INTR OUT OF ORDER
3186  007772  000000                TCNT:   .WORD 0         ;TABLE PASS CTR
3187  007774  000000                TEMP:   .WORD 0         ;FOR SERVICE ROUTINES
3188
```

```
                        ;;*****************************************************************
                        ;*TEST 22       TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING
                        ;*             IN THIS WE'LL ENABLE INTERRUPTS ON ALL CHANNELS
                        ;*             FOR THIS DLV11-J. THEN WE'LL XMIT AN INCREMENTING
                        ;*             DATA PATERN VIA INTERRUPTS AND RECORD THE RECEIVER
                        ;*             INTR. IN THE RECEIVER STATUS TABLE.
                        ;*
                        ;*      NOTE:  DOUBLE BUFFERING CANNOT BE TESTED AT ITS MAX SPEED
                        ;*             BECAUSE OF APT CONSIDERATIONS. I.E. APT SENDS
                        ;*             'BREAKS' WHICH CAUSE OVERRUN ERRORS. THEREFORE
                        ;*             THE XMIT IE IS NOT ENABLED AGAIN UNTIL THE RECVR
                        ;*             HAS OBTAINED THE PREVIOUS WORD.
                        ;*****************************************************************
007776 000004           TST22:  SCOPE
010000 012737 000001 001160     MOV     #1,$TIMES       ;;DO 1 ITERATION
010006 012737 000022 001200     MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                                            ;CLR ERROR WORD,
010C14                                                  LET ERWRD :  #0
010C14 005037 012366            CLR     ERWRD


                                                            ;CLEAR OUT RECEIVER TABLES
010020                                                  LET R0 := #CHOTAB
010020 012700 011360            MOV     #CHOTAB,R0
010024                                                  REPEAT
010024                  $176:
010024                                                      LET (R0)+ :  #0
010024 005020                   CLR     (R0)+
010026                                                  UNTIL R0 EQ #STATEND
010026 020027 012360            CMP     R0,#STATEND
010032 001374                   BNE     $176


                                                            ;SET UP ALL VECTORS
010034                                                  LET R1 := #ROSRV
010034 012701 010660            MOV     #ROSRV,R1
010040                                                  LET R0 := DLVEC
010040 013700 001256            MOV     DLVEC,R0
010044                                                  REPEAT
010044                  $177:
010044                                                      LET (R0)+ := R1
010044 010120                   MOV     R1,(R0)+
010046                                                      LET (R0)+ :- #PR7
010046 012720 000340            MOV     #PR7,(R0)+
010052                                                      LET R1 :- R1 + #54
010052 062701 000054            ADD     #54,R1
010056                                                      LET (R0)+ :- R1
010056 010120                   MOV     R1,(R0)+
010060                                                      LET (R0)+ :- #PR7
010060 012720 000340            MOV     #PR7,(R0)+
010064                                                      LET R1 := R1 + #44
010064 062701 000044            ADD     #44,R1
010070                                                  UNTIL R1 EQ #SEREND
010070 020127 011360            CMP     R1,#SEREND
010074 001363                   BNE     $177


                                                            ;SET PRIORTY TO 7
010076                          SETPRI  #PR7
```

```
3245
3246
3247
3248  010110                                                    ;ENABLE INTR'S ON ALL ACTIVE LINES
3249  010110  013700  001254          MOV     DLADD,R0        ;THIS MODULE
3250                                                        LET R0 := DLADD
3251  010114                                                    ;INITIALIZE
3252  010114  013737  014162  012360  MOV     MASK,CHMSK      LET CHMSK := MASK
3253  010122                                                  LET CHCTR := #0
3254  010122  005037  012362          CLR     CHCTR
3255                                                        ;ACTIVE CHANNEL CTR
3256  010126                                                  LET ACTCH := #0
3257  010126  005037  012364          CLR     ACTCH
3258  010132                                                  REPEAT
3259  010132                  $200:
3260  010132                                                      IF #BIT0 SETIN CHMSK THEN
3261  010132  032737  000001  012360  BIT     #BIT0,CHMSK
3262  010140  001413                  BEQ     $201
3263                                                              ;# OF ACTIVE CH
3264  010142                                                      LET ACTCH := ACTCH + #1
3265  010142  005237  012364          INC     ACTCH
3266                                                              ;SET RCSR IE
3267  010146                                                      LET (R0) := (R0) SET.BY #BIT6
3268  010146  052710  000100          BIS     #BIT6,(R0)
3269  010152                                                      LET R0 := R0 + #4
3270  010152  062700  000004          ADD     #4,R0
3271                                                              ;SET XCSR IE
3272  010156                                                      LET (R0) := (R0) SET.BY #BIT6
3273  010156  052710  000100          BIS     #BIT6,(R0)
3274  010162                                                      LET R0 := R0 + #4
3275  010162  062700  000004          ADD     #4,R0
3276  010166                                                  ELSE
3277  010166  000402                  BR      $202
3278  010170                  $201:
3279  010170                                                      LET R0 := R0 + #10
3280  010170  062700  000010          ADD     #10,R0
3281  010174                                                  ENDIF
3282  010174                  $202:
3283  010174                                                  LET CHMSK := CHMSK SHIFT -1
3284  010174  006237  012360          ASR     CHMSK
3285
3286
3287  010200                                                  LET CHCTR := CHCTR + #1
3288  010200  005237  012362          INC     CHCTR
3289  010204                                                  IF CHCTR EQ #3 THEN
3290  010204  023727  012362  000003  CMP     CHCTR,#3
3291  010212  001007                  BNE     $203
3292  010214                                                      CALL TSTCON
3293  010214  004737  013530          JSR     PC,TSTCON
3294  010220                                                      IF CONSOLE NE #0 THEN
3295  010220  005737  013604          TST     CONSOLE
3296  010224  001402                  BEQ     $204
3297  010226                                                        LET CHCTR := CHCTR + #1
3298  010226  005237  012362          INC     CHCTR
3299  010232                                                      ENDIF
3300  010232                  $204:
```

```
3301  010232                                                        ENDIF
3302  010232                          $203:
3303  010232                                                    UNTIL CHCTR EQ #4
3304  010232  023727  012362  000004          CMP     CHCTR,#4
3305  010240  001334                          BNE     $200
3306
3307                                                            ;INIT BUFFER POINTERS FOR SERVICE ROUTINES
3308  010242                                                    LET R0 := #CH0TAB
3309  010242  012700  011360                  MOV     #CH0TAB,R0
3310  010246                                                    LET R1 := #CH1TAB
3311  010246  012701  011560                  MOV     #CH1TAB,R1
3312  010252                                                    LET R2 := #CH2TAB
3313  010252  012702  011760                  MOV     #CH2TAB,R2
3314  010256                                                    LET R3 :  #CH3TAB
3315  010256  012703  012160                  MOV     #CH3TAB,R3
3316
3317                                                            ;FILL XBUF WORDS
3318                                                            ;FROM TMP0 THRU TMP3E
3319                                                            ;START WORD CH 0 ALWAYS 0
3320  010262                                                    LET TMP0 := #0
3321  010262  005037  012372                  CLR     TMP0
3322                                                            ;OFFSET TO INDEX
3323  010266                                                    LET R4 :- #0
3324  010266  005004                          CLR     R4
3325                                                            ;PTR TO TABLE TO BE FILLED
3326  010270                                                    LET R5 := #TMP0E
3327  010270  012705  012374                  MOV     #TMP0E,R5
3328  010274                                                    REPEAT
3329  010274                          $205:
3330  010274                                                      IF #BIT0 NOTSETIN $USWR THEN
3331  010274  032737  000001  001220          BIT     #BIT0,$USWR
3332  010302  001005                          BNE     $206
3333  010304                                                        LET (R5)+ :  TABL7(R4)
3334  010304  016425  012426                  MOV     TABL7(R4),(R5)+
3335  010310                                                        LET (R5)+ :  TABL7(R4)
3336  010310  016425  012426                  MOV     TABL7(R4),(R5)+
3337  010314                                                      ELSE
3338  010314  000404                          BR      $207
3339  010316                          $206:
3340  010316                                                        LET (R5)+ :- TABL8(R4)
3341  010316  016425  012436                  MOV     TABL8(R4),(R5)+
3342  010322                                                        LET (R5)+ := TABL8(R4)
3343  010322  016425  012436                  MOV     TABL8(R4),(R5)+
3344  010326                                                      ENDIF
3345  010326                          $207:
3346  010326                                                        LET R4 :  R4 + #2
3347  010326  062704  000002                  ADD     #2,R4
3348  010332                                                    UNTIL R4 EQ #10
3349  010332  020427  000010                  CMP     R4,#10
3350  010336  001356                          BNE     $205
3351
3352                                                            ;INIT TMP4
3353  010340                                                    ;WILL BE DECR BY XMIT SERVICE ROUTINES TILL 0
3354  010340  013737  012364  012414          MOV     ACTCH,TMP4
3355                                                            ;SET PRIORITY TO 0
3356  010346                                  SETPRI  #PR0
```

```
3357
3358  010360                                                    REPEAT
3359  010360                          $210:
3360  010360                                                    UNTIL TMP4 EQ #0
3361  010360  005737  012414                TST     TMP4
3362  010364  001375                        BNE     $210
3363
3364                                                            ;WAIT FOR RECV'S TO CATCH UP
3365  010366                                                    LET TMP4 :  #100000
3366  010366  012737  100000  012414        MOV     #100000,TMP4
3367  010374                                                    REPEAT
3368  010374                          $211:
3369  010374                                                       LET TMP4 :- TMP4 - #1
3370  010374  005337  012414                DEC     TMP4
3371  010400                                                    UNTIL TMP4 EQ #0
3372  010400  005737  012414                TST     TMP4
3373  010404  001373                        BNE     $211
3374
3375                                                               ;DISABLE ALL INTR'S
3376  010406                                                    BRESET
3377  010406  000005                        RESET
3378
3379                                                               ;CHECK RECEIVER TABLE
3380                                                               ;RESET TMP0,1,2,3
3381  010410                                                       LET TMP0 := #0
3382  010410  005037  012372                CLR     TMP0
3383  010414                                                       LET TMP1 := TMP0E
3384  010414  013737  012374  012376        MOV     TMP0E,TMP1
3385  010422                                                       LET TMP2 :- TMP1E
3386  010422  013737  012400  012402        MOV     TMP1E,TMP2
3387  010430                                                       LET TMP3 := TMP2E
3388  010430  013737  012404  012406        MOV     TMP2E,TMP3
3389
3390  010436                                                       ;INITIALIZE
3391  010436  005037  012362                CLR     CHCTR         LET CHCTR :- #0
3392  010442
3393  010442  013737  014162  012360        MOV     MASK,CHMSK    LET CHMSK :  MASK
3394  010450                                                    REPEAT
3395  010450                          $212:
3396  010450                                                      IF #BIT0 SETIN CHMSK THEN
3397  010450  032737  000001  012360        BIT     #BIT0,CHMSK
3398  010456  001436                        BEQ     $213
3399  010460
3400  010460  013700  012362                MOV     CHCTR,R0          LET R0 := CHCTR
3401
3402  010464                                                          ;OFFSET TO INDEX (2X)
3403  010464  006300                        ASL     R0                LET R0 := R0 SHIFT +1
3404
3405  010466                                                          ;GET TABLE LOC FOR THAT CH.
3406  010466  016002  012416                MOV     TABL6(R0),R2      LET R2 := TABL6(R0)
3407
3408  010472                                                          ;OFFSET TO INDEX (4X)
3409  010472  006300                        ASL     R0                LET R0 := R0 SHIFT +1
3410
3411  010474                                                          ;GET EXPECTED STARTING WORD
3412  010474  016001  012372                MOV     TMP0(R0),R1       LET R1 := TMP0(R0)
```

```
3413  010500
3414  010500  062700  000002              ADD     #2,R0
3415
3416  010504
3417  010504  016003  012372              MOV     TMP0(R0),R3
3418
3419  010510
3420  010510                        $214:
3421  010510
3422  010510  032712  100000              BIT     #BIT15,(R2)
3423  010514  001403                      BEQ     $215
3424  010516
3425  010516  052737  000400  012366      BIS     #BIT8,ERWRD
3426  010524
3427  010524                        $215:
3428
3429  010524
3430  010524  121201                      CMPB    (R2),R1
3431  010526  001402                      BEQ     $216
3432  010530
3433  010530  005237  012366              INC     ERWRD
3434  010534
3435  010534                        $216:
3436
3437  010534
3438  010534  005201                      INC     R1
3439
3440  010536
3441  010536  062702  000002              ADD     #2,R2
3442
3443  010542
3444  010542  020103                      CMP     R1,R3
3445  010544  001403                      BEQ     $217
3446  010546  005737  012366              TST     ERWRD
3447  010552  001756                      BEQ     $214
3448  010554                        $217:
3449  010554
3450  010554                        $213:
3451  010554
3452  010554  005237  012362              INC     CHCTR
3453  010560
3454  010560  006237  012360              ASR     CHMSK
3455  010564
3456  010564  023727  012362  000003      CMP     CHCTR,#3
3457  010572  001010                      BNE     $220
3458  010574
3459  010574  004737  013530              JSR     PC,TSTCON
3460  010600
3461  010600  023727  013604  000001      CMP     CONSOLE,#TRUE
3462  010606  001002                      BNE     $221
3463
3464  010610
3465  010610  005237  012362              INC     CHCTR
3466  010614
3467  010614                        $221:
3468  010614
```

```
        LET R0 := R0 + #2

        ;GET MAX WORD FOR THAT CH.
        LET R3 := TMP0(R0)

        REPEAT

            IF #BIT15 SETIN (R2) THEN

                LET ERWRD := ERWRD SET.BY #BIT8

            ENDIF

                ;CHECK FOR DATA COMPARE ERROR
            IFB (R2) NE R1 THEN

                LET ERWRD := ERWRD + #1

            ENDIF

                ;BUMP EXPECTED DATA WORD
                LET R1 :- R1 + #1

                ;BUMP TABLE PTR
                LET R2 :- R2 + #2

        UNTIL R1 EQ R3 OR ERWRD NE #0



ENDIF

 LET CHCTR :- CHCTR + #1

 LET CHMSK :  CHMSK SHIFT -1

      IF CHCTR EQ #3 THEN

          CALL TSTCON

         IF CONSOLE EQ #TRUE THEN

              ;EXIT
              LET CHCTR := CHCTR + #1

         ENDIF

      ENDIF
```

```
3469  010614                          $220:
3470  010614                                                  UNTIL CHCTR EQ #4 OR ERWRD NE #0
3471  010614  023727  012362  000004          CMP     CHCTR,#4
3472  010622  001403                          BEQ     $222
3473  010624  005737  012366                  TST     ERWRD
3474  010630  001707                          BEQ     $212
3475  010632                          $222:
3476
3477  010632                                                  IF #BIT8 SETIN ERWRD THEN
3478  010632  032737  000400  012366          BIT     #BIT8,ERWRD
3479  010640  001401                          BEQ     $223
3480                                                              ;ERROR FLAG UP AFTER TRANSFER
3481  010642                                                      ERRHRD  270
3482  010642  104270                          ERROR   270
3483  010644                                                  ENDIF
3484  010644                          $223:
3485  010644                                                  IFB ERWRD NE #0 THEN
3486  010644  105737  012366                  TSTB    ERWRD
3487  010650  001401                          BEQ     $224
3488                                                              ;DATA COMPARE ERROR
3489  010652                                                      ERRHRD  271
3490  010652  104271                          ERROR   271
3491  010654                                                  ENDIF
3492  010654                          $224:
3493
3494
3495  010654  000137  006502                  JMP     TST20                   ;EXIT TEST
```

```
                        ;;*****************************************************************
                        ;*              START OF SERVICE ROUTINES
                        ;;*****************************************************************

010660                  ROSRV:                                  BGNSRV ROSRV
010660                                                              LET TMP := RCSR
010660  013737  001260  012370          MOV     RCSR,TMP
010666                                                              LET TMP := TMP + #2
010666  062737  000002  012370          ADD     #2,TMP
010674                                                              LET (R0)+ :  @TMP
010674  017720  001470                  MOV     @TMP,(R0)+
010700                                                              IF TMP0 NE TMP0E THEN
010700  023737  012372  012374          CMP     TMP0,TMP0E
010706  001407                  BEQ     $225
                                                                        ;GO TO XCSR
010710                                                                  LET TMP :  TMP + #2
010710  062737  000002  012370          ADD     #2,TMP
                                                                        ;ENABLE XMIT INTERR
010716                                                                  LET @TMP :  @TMP SET.BY #BIT6
010716  052777  000100  001444          BIS     #BIT6,@TMP
                                                                    ELSE
010724  000402                  BR      $226
010726                  $225:
                                                                        ;ALL DONE
010726                                                                  LET TMP4 :- TMP4 - #1
010726  005337  012414                  DEC     TMP4
                                                                    ENDIF
010732                  $226:
010732                                                          ENDSRV
010732  000002                  RTI

010734                                                          BGNSRV XOSRV
010734                  XOSRV:
010734                                                              LET TMP :- RCSR
010734  013737  001260  012370          MOV     RCSR,TMP
010742                                                              LET TMP := TMP + #6
010742  062737  000006  012370          ADD     #6,TMP
010750                                                              LET @TMP :  TMP0
010750  013777  012372  001412          MOV     TMP0,@TMP
010756                                                              LET TMP0 := TMP0 + #1
010756  005237  012372                  INC     TMP0
                                                                        ;GO BACK TO XCSR
010762                                                              LET TMP := TMP - #2
010762  162737  000002  012370          SUB     #2,TMP
                                                                        ;DISABLE XMIT INTERRUPTS,
                                                                        ;NOT EXER DOUBLE BUFFERING
010770                                                              LET @TMP := @TMP CLR.BY #BIT6
010770  042777  000100  001372          BIC     #BIT6,@TMP
010776                                                          ENDSRV
010776  000002                  RTI

011000                                                          BGNSRV R1SRV
011000                  R1SRV:
011000                                                              LET TMP := RCSR
```

```
3552  011000  013737  001260  012370          MOV     RCSR,TMP
3553  011006
3554  011006  062737  000012  012370          ADD     #12,TMP
3555  011014
3556  011014  017721  001350                  MOV     @TMP,(R1)+
3557  011020
3558  011020  023737  012376  012400          CMP     TMP1,TMP1E
3559  011026  001407                          BEQ     $227
3560
3561  011030
3562  011030  062737  000002  012370          ADD     #2,TMP
3563
3564  011036
3565  011036  052777  000100  001324          BIS     #BIT6,@TMP
3566  011044
3567  011044  000402                          BR      $230
3568  011046                          $227:
3569
3570  011046
3571  011046  005337  012414                  DEC     TMP4
3572  011052
3573  011052                          $230:
3574  011052
3575  011052  000002                          RTI
3576
3577  011054
3578  011054                          X1SRV:
3579  011054
3580  011054  013737  001260  012370          MOV     RCSR,TMP
3581  011062
3582  011062  062737  000016  012370          ADD     #16,TMP
3583  011070
3584  011070  013777  012376  001272          MOV     TMP1,@TMP
3585  011076
3586  011076  005237  012376                  INC     TMP1
3587
3588  011102
3589  011102  162737  000002  012370          SUB     #2,TMP
3590
3591
3592  011110
3593  011110  042777  000100  001252          BIC     #BIT6,@TMP
3594  011116
3595  011116  000002                          RTI
3596
3597  011120
3598  011120                          R2SRV:
3599  011120
3600  011120  013737  001260  012370          MOV     RCSR,TMP
3601  011126
3602  011126  062737  000022  012370          ADD     #22,TMP
3603  011134
3604  011134  017722  001230                  MOV     @TMP,(R2)+
3605  011140
3606  011140  023737  012402  012404          CMP     TMP2,TMP2E
3607  011146  001407                          BEQ     $231
```

```
        LET TMP := TMP + #12

        LET (R1)+ := @TMP

        IF TMP1 NE TMP1E THEN


            ;GO TO XCSR
            LET TMP :  TMP + #2

            ;ENABLE XMIT INTERR
            LET @TMP := @TMP SET.BY #BIT6

        ELSE


            ;ALL DONE
            LET TMP4 :  TMP4 - #1

        ENDIF

    ENDSRV


    BGNSRV X1SRV

        LET TMP :  RCSR

        LET TMP :  TMP + #16

        LET @TMP :  TMP1

        LET TMP1 :- TMP1 + #1

            ;GO BACK TO XCSR
        LET TMP := TMP - #2

            ;DISABLE XMIT INTERRUPTS,
            ;NOT EXER DOUBLE BUFF
        LET @TMP := @TMP CLR.BY #BIT6

    ENDSRV


    BGNSRV R2SRV

        LET TMP :- RCSR

        LET TMP :- TMP + #22

        LET (R2)+ :- @TMP

        IF TMP2 NE TMP2E THEN
```

```
3608
3609  011150
3610  011150  062737  000002  012370          ADD     #2,TMP
3611
3612  011156
3613  011156  052777  000100  001204          BIS     #BIT6,@TMP
3614  011164
3615  011164  000402                          BR      $232
3616  011166                          $231:
3617
3618  011166
3619  011166  005337  012414                  DEC     TMP4
3620  011172
3621  011172                          $232:
3622  011172
3623  011172  000002                          RTI
3624
3625  011174
3626  011174                          X2SRV:
3627  011174
3628  011174  013737  001260  012370          MOV     RCSR,TMP
3629  011202
3630  011202  062737  000026  012370          ADD     #26,TMP
3631  011210
3632  011210  013777  012402  001152          MOV     TMP2,@TMP
3633  011216
3634  011216  005237  012402                  INC     TMP2
3635
3636  011222
3637  011222  162737  000002  012370          SUB     #2,TMP
3638
3639
3640  011230
3641  011230  042777  000100  001132          BIC     #BIT6,@TMP
3642  011236
3643  011236  000002                          RTI
3644
3645  011240
3646  011240                          R3SRV:
3647  011240
3648  011240  013737  001260  012370          MOV     RCSR,TMP
3649  011246
3650  011246  062737  000032  012370          ADD     #32,TMP
3651  011254
3652  011254  017723  001110                  MOV     @TMP,(R3)+
3653  011260
3654  011260  023737  012406  012410          CMP     TMP3,TMP3E
3655  011266  001407                          BEQ     $233
3656
3657  011270
3658  011270  062737  000002  012370          ADD     #2,TMP
3659
3660  011276
3661  011276  052777  000100  001064          BIS     #BIT6,@TMP
3662  011304
3663  011304  000402                          BR      $234
```

```
        ;GO TO XCSR
        LET TMP := TMP + #2

        ;ENABLE XMIT INTERR
        LET @TMP := @TMP SET.BY #BIT6

    ELSE

        ;ALL DONE
        LET TMP4 :- TMP4 - #1

    ENDIF

ENDSRV

BGNSRV X2SRV

    LET TMP :- RCSR

    LET TMP :  TMP + #26

    LET @TMP :  TMP2

    LET TMP2 :- TMP2 + #1

        ;GO BACK TO XCSR
    LET TMP :- TMP - #2

        ;DISABLE XMIT INTERRUPTS,
        ;NOT EXER DOUBLE BUFF.
    LET @TMP :  @TMP CLR.BY #BIT6

ENDSRV

BGNSRV R3SRV

    LET TMP :  RCSR

    LET TMP :  TMP + #32

    LET (R3)+ :  @TMP

    IF TMP3 NE TMP3E THEN

        ;GO TO XCSR
        LET TMP :- TMP + #2

        ;ENABLE XMIT INTERR
        LET @TMP := @TMP SET.BY #BIT6

    ELSE
```

```
3664  011306                          $233:
3665                                                                    ;ALL DONE
3666  011306                                                            LET TMP4 :  TMP4 - #1
3667  011306  005337  012414                  DEC     TMP4
3668  011312                                                        ENDIF
3669  011312                          $234:
3670  011312                                                    ENDSRV
3671  011312  000002                  RTI
3672
3673  011314                                                    BGNSRV X3SRV
3674  011314                          X3SRV:
3675  011314                                                        LET TMP :- RCSR
3676  011314  013737  001260  012370          MOV     RCSR,TMP
3677  011322                                                        LET TMP :- TMP + #36
3678  011322  062737  000036  012370          ADD     #36,TMP
3679  011330                                                        LET @TMP :  TMP3
3680  011330  013777  012406  001032          MOV     TMP3,@TMP
3681  011336                                                        LET TMP3 := TMP3 + #1
3682  011336  005237  012406                  INC     TMP3
3683                                                                ;GO BACK TO XCSR
3684  011342                                                        LET TMP :- TMP - #2
3685  011342  162737  000002  012370          SUB     #2,TMP
3686                                                                ;DISABLE XMIT INTERRUPTS,
3687                                                                ;NOT EXER DOUBLE BUFF
3688  011350                                                        LET @TMP :- @TMP CLR.BY #BIT6
3689  011350  042777  000100  001012          BIC     #BIT6,@TMP
3690  011356                                                    ENDSRV
3691  011356  000002                  RTI
3692  011360                          SEREND:
```

```
3693
3694                                    ;** RECEIVER STATUS TABLES **
3695                                                                ;7 BIT WDS      8 BIT WDS
3696  011360  000100                    CH0TAB: .BLKW 100           ;  0- 37          0- 77
3697  011560  000100                    CH1TAB: .BLKW 100           ; 40- 77        100-177
3698  011760  000100                    CH2TAB: .BLKW 100           ;100-137        200-277
3699  012160  000100                    CH3TAB: .BLKW 100           ;140-177        300-377
3700  012360                            STATEND:
3701  012360  000000                    CHMSK:  .WORD 0             ;MASK OF ACTIVE CHANNELS
3702  012362  000000                    CHCTR:  .WORD 0             ;CHANNEL CTR
3703  012364  000000                    ACTCH:  .WORD 0             ;ACTIVE CHANNEL CTR
3704  012366  000000                    ERWRD:  .WORD 0             ;ERROR WORD, LOW BYTE IS NO. OF DATA ERRORS
3705                                                                ;HIGH BYTE INDICATES ERROR FLAG DETECTED
3706
3707  012370  000000                    TMP:    .WORD 0             ;TEMP STORAGE FOR SERVICE ROUTINES
3708
3709  012372  000000                     MP0:   0                   ;START WORD FOR CH 0
3710  012374  000000                    TMP0E:  0                   ;END   WORD FOR CH 0
3711  012376  000000                    TMP1:   0                   ;START              1
3712  012400  000000                    TMP1E:  0                   ;END                1
3713  012402  000000                    TMP2:   0                   ;START              2
3714  012404  000000                    TMP2E:  0                   ;END                2
3715  012406  000000                    TMP3:   0                   ;START              3
3716  012410  000002                    TMP3E:  .BLKW 2             ;END                3
3717
3718  012414  000000                    TMP4:   .WORD
3719
3720  012416  011360                    TABL6:  CH0TAB
3721  012420  011560                            CH1TAB
3722  012422  011760                            CH2TAB
3723  012424  012160                            CH3TAB
3724
3725  012426  000040                    TABL7:  40                  ;7 BIT WORD TABLE
3726  012430  000100                            100
3727  012432  000140                            140
3728  012434  000200                            200
3729
3730  012436  000100                    TABL8:  100                 ;8 BIT WORD TABLE
3731  012440  000200                            200
3732  012442  000300                            300
3733  012444  000400                            400
3734
3735
```

```
                        ;;*****************************************************************
                        ;* ROUTINE:TIMER
                        ;*       THIS ROUTINE IS USED TO TEST THE STATUS OF ANY BIT
                        ;*       IN ANY REGISTER.
                        ;*       INPUTS:
                        ;*             HOWLONG    THE MAXIMUM AMOUNT OF TIME TO SPEND IN
                        ;*                        THIS ROUTINE.
                        ;*             WHICHBIT     A MASK WITH THE BIT(S) SET THAT ARE
                        ;*                          TO BE CHECKED.
                        ;*             REG     A POINTER TO THE REGISTER TO BE CHECKED
                        ;*             SETCLR  THE DESIRED RESULTS
                        ;*                     EITHER #SET OR #CLEAR
                        ;*       OUTPUT:
                        ;*             THE 'C' BIT IS SET TO INDICATE AN ERROR
                        ;*             BUT IT IS TESTED BY THE IF.ERROR STATEMENT
                        ;*
                        ;*
                        ;*             NOTE::  THE USE OF (R5) IS PART OF THE LINKAGE
                        ;*                     MECHANISM BETWEEN THE CALLER AND THE CALLED
                        ;;*****************************************************************

                                                ROUTINE TIMER <HOWLONG,WHICHBIT,REG,SETCLR>
012446                  TIMER:
                                                    LET     REGSAV :  REG(R5) ; GET POINTER TO REGIST
012446  016537  000004  012606        MOV     REG(R5),REGSAV
                                                    LET     TIMSAV :  HOWLONG(R5) ; SAVE HOWLONG FOR
012454  016537  000000  012610        MOV     HOWLONG(R5),TIMSAV
                                                    LET     FLAG :B= #0 ; INITIALIZE THE EXIT FLAG
012462  105037  012612                CLRB    FLAG


                                                ;       START OF AN INFINITE LOOP

                                                LOOP

012466                  $237:
                                                     ; TEST TO SEE IF WHICHBIT IS SET
012466  036577  000002  000112        BIT     WHICHBIT(R5),@REGSAV
012474  001003                        BNE     $241
                                                    IF      WHICHBIT(R5)  NOTSETIN @REGSAV THEN
012476
012476  105037  012613                CLRB    HOLDSC
                                                        LET     HOLDSC :B- #0
012502
012502  000403                        BR      $242
                                                    ELSE
012504                  $241:
012504
012504  112737  177777  012613        MOVB    #SET,HOLDSC
                                                        LET     HOLDSC :B  #SET ; REMEMBER THIS
012512
012512                  $242:
                                                    ENDIF

                                                    ; NOW SEE IF THAT WAS WHAT WE WANTED
012512                                              IFB     HOLDSC EQ SETCLR(R5) THEN
012512  123765  012613  000006        CMPB    HOLDSC,SETCLR(R5)
012520  001003                        BNE     $243
                                                     ; JUST THE THING WE NEEDED
```

```
3792  012522
3793  012522  112737  000001  012612          MOVB    #TRUE,FLAG                      LET     FLAG :B= #TRUE
3794  012530                                                                          ENDIF
3795  012530                                  $243:
3796
3797  012530                                                                  EXIFB   FLAG EQ #TRUE OR TIMSAV LE #0
3798  012530  123727  012612  000001          CMPB    FLAG,#TRUE
3799  012536  001414                          BEQ     $240
3800  012540  005737  012610                  TST     TIMSAV
3801  012544  003411                          BLE     $240
3802                                                                              ; ONE WAY OR THE OTHER, WE ARE DONE
3803                                                                    ; IF WE ARE STILL HERE THEN HANG AROUND A WHILE
3804
3805  012546                                                                      WAITMS  1       ;WAIT FOR 1 MILLI-SECONDS
3806  012546  010546                          MOV     R5,-(SP)
3807  012550  012745  000001                  MOV     #1,-(R5)
3808  012554  004737  012620                  JSR     PC,WAIT
3809  012560  012605                          MOV     (SP)+,R5
3810  012562                                                                      LET     TIMSAV :  TIMSAV - #1 ; COUNTING DOWN
3811  012562  005337  012610                  DEC     TIMSAV
3812  012566                                                                  ENDLOOP                 ; CONTINUED AT THE TOP
3813  012566  000737                          BR      $237
3814  012570                                  $240:
3815
3816                                                  ; ONLY 2 WAYS TO GET HERE
3817                                                  ;       1). WE RAN OUT OF TIME---ERROR .
3818                                                  ;       2). THE BIT IS IN THE CORRECT CONDITION--GOOD .
3819
3820  012570                                                                  IFB     FLAG EQ #TRUE THEN
3821  012570  123727  012612  000001          CMPB    FLAG,#TRUE
3822  012576  001001                          BNE     $244
3823  012600                                                                      RETURN  NO.ERROR    ; GOOD
3824  012600  000405                          BR      $235
3825  012602                                                                  ENDIF
3826  012602                                  $244:                           RETURN  ERROR           ; BAD
3827  012602
3828  012602  000261                          SEC
3829  012604  000404                          BR      $236
3830
3831  012606  000000                  REGSAV: .WORD 0
3832  012610  000000                  TIMSAV: .WORD 0
3833  012612     000                  FLAG:   .BYTE 0
3834  012613     000                  HOLDSC: .BYTE 0
3835                                                                  ; WE ARE DONE  GO BACK HOME
                                                                                    ENDRTN
3836  012614
3837  012614                                  $235:
3838  012614  000241                          CLC
3839  012616                                  $236:
3840  012616  000207                          RTS     PC
```

```
3841
3842                                    ;;*****************************************************************
3843                                    ;* ROUTINE:WAIT
3844                                    ;*       THIS ROUTINE IS USED TO DELAY EXECUTION OF THE
3845                                    ;*       MAIN  PROGRAM  FOR A SPECIFIED AMOUNT OF TIME.
3846                                    ;*       THIS   IS   ACCOMPLISHED   BY  INCREMENTING  A
3847                                    ;*       REGISTER UP TO A LIMIT.  THE INNER LOOP IS SET
3848                                    ;*       TO APPROXIMATE 1 MILLI SEC.
3849                                    ;;*****************************************************************
3850  012620                                                ROUTINE WAIT    <TIME>
3851  012620                            WAIT:
3852  012620                                                        PUSH    <R1,R2,R3>
3853  012626                                                        LET R1 := TIME(R5)
3854  012626  016501  000000                    MOV     TIME(R5),R1
3855  012632                                                        INCRU R2 FROM #1 TO R1 BY #1
3856  012632  012702  000001                    MOV     #1,R2
3857  012636  000402                            BR      $247
3858  012640                            $250:
3859  012640  062702  000001                    ADD     #01,R2
3860  012644                            $247:
3861  012644  020201                            CMP     R2,R1
3862  012646  101010                            BHI     $251
3863  012650                                                        INCR R3 FROM #0 TO #100 BY #1
3864  012650  005003                            CLR     R3
3865  012652  000401                            BR      $252
3866  012654                            $253:
3867  012654  005203                            INC     R3
3868  012656                            $252:
3869  012656  020327  000100                    CMP     R3,#100
3870  012662  003001                            BGT     $254
3871  012664                                                        ENDINC
3872  012664  000773                            BR      $253
3873  012666                            $254:
3874  012666                                                        ENDINC
3875  012666  000764                            BR      $250
3876  012670                            $251:
3877  012670                                                        POP     <R3,R2,R1>
3878  012676                                                ENDRTN
3879  012676                            $245:
3880  012676                            $246:
3881  012676  000207                            RTS     PC
```

```
                                        .SBTTL  ROUTINE TO SIZE THE BUS

012700 005737 001252            SIZE:   TST     $DEVM           ;BR IF INITIALLY SET
012704 001060                           BNE     4$              ;BYPASS AUTO-SIZING

012706 012737 013610 000004     1$:     MOV     #INTSRV,ERRVEC
012714 012737 000340 000006             MOV     #PR7,ERRVEC+2

012722 013737 001250 013514             MOV     $BASE,TMP6
012730 012737 000001 013512             MOV     #1,SFTREG

012736 005037 013616            2$:     CLR     INTFLAG
012742 005777 000546                    TST     @TMP6           ;SEE IF THERE
012746 005737 013616                    TST     INTFLAG         ;GOT INTERRUPT?
012752 001003                           BNE     3$              ;BR IF YES  (NO DEVICE)
012754 053737 013512 001252             BIS     SFTREG,$DEVM    ;ELSE THERE, SET MAP

012762 062737 000010 013514     3$:     ADD     #10,TMP6        ;GO TO NEXT CHANNEL
012770 006337 013512                    ASL     SFTREG
012774 001360                           BNE     2$              ;BR IF MORE TO GO

012776 012737 000006 000004             MOV     #6,ERRVEC       ;RESTORE
013004 005037 000006                    CLR     ERRVEC+2

013010 005737 001252                    TST     $DEVM           ;ANYONE THERE?
013014 001014                           BNE     4$              ;BR IF YES
013016                                  TYPTXT  <<CRLF>!NO DEV TO TEST!>
013044 000777                           BR      .               ;LOOP HERE TILL PROBLEM FIXED
                                                                ;& RESTART AT 200
013046 105737 001221            4$:     TSTB    $USWR+1         ;SEE IF ANY CONSOLE
013052 001434                           BEQ     6$              ;BR IF NONE

013054 010046                           MOV     R0,-(SP)        ;SAVE R0
013056 113700 001221                    MOVB    $USWR+1,R0      ;GET CONSOLE DEVICE X2
013062 020027 000010                    CMP     R0,#10
013066 003005                           BGT     5$              ;BR IF TOO HI
013070 056037 013516 001252             BIS     TABL1-2(R0),$DEVM       ;SET CONSOLE BIT ON MAP
013076 012600                           MOV     (SP)+,R0        ;RESTORE R0
013100 000421                           BR      6$

013102                          5$:     TYPTXT  <<CRLF>!CONSOLE IN $USWR WRONG!>
013140 000137 014362                    JMP     $EOP                    ;START ALL OVER
                                                                        ;STACK WILL RESET IN 'LOOP'
013144 013737 001252 013502     6$:     MOV     $DEVM,MASK1
013152 013737 001252 013504             MOV     $DEVM,MASK2
013160 013737 001252 013506             MOV     $DEVM,MASK3
013166 013737 001252 013510             MOV     $DEVM,MASK4

013174 042737 177760 013502             BIC     #^C<17>,MASK1           ;SAVE ONLY MODULE 1 CHANNELS
013202 042737 177417 013504             BIC     #^C<360>,MASK2          ;                 2
013210 042737 170377 013506             BIC     #^C<7400>,MASK3         ;                 3
013216 042737 007777 013510             BIC     #^C<170000>,MASK4       ;                 4

013224 006237 013504                    ASR     MASK2
```

```
 3938  013230  006237  013504              ASR     MASK2
 3939  013234  006237  013504              ASR     MASK2
 3940  013240  006237  013504              ASR     MASK2                   ;RIGHT JUSTIFY MODULE 2 CHANNELS
 3941
 3942  013244  000337  013506              SWAB    MASK3                   ;                       3
 3943
 3944  013250  000337  013510              SWAB    MASK4
 3945  013254  006237  013510              ASR     MASK4
 3946  013260  006237  013510              ASR     MASK4
 3947  013264  006237  013510              ASR     MASK4
 3948  013270  006237  013510              ASR     MASK4                   ;                       4
 3949
 3950  013274                              TYPTXT  <<CRLF>.WILL TEST:.<CRLF>>
 3951
 3952  013320  012700  013502              MOV     #MASK1,R0
 3953  013324  012737  000061  013474      MOV     #61,MODNUM              ;INIT MOD #
 3954  013332  012737  000060  013500      MOV     #60,CHNUM+2             ;INIT CHANNEL #
 3955
 3956  013340  012037  013516      7$:     MOV     (R0)+,MSK
 3957  013344  001001                      BNE     8$                      ;BR IF ANY THERE
 3958  013346  000441                      BR      11$                     ;ELSE TRY NEXT MODULE
 3959
 3960  013350                      8$:     TYPTXT  <<CRLF>.MODULE .>
 3961  013370  104401  013474              TYPE    ,MODNUM
 3962  013374                              TYPTXT  <.  CHANNEL .>
 3963
 3964  013416  032737  000001  013516  9$: BIT     #BIT0,MSK
 3965  013424  001402                      BEQ     10$                     ;BR IF NOT TESTING THAT CHANNEL
 3966  013426  104401  013476              TYPE    ,CHNUM                  ;ELSE TYPE IT
 3967
 3968  013432  005237  013500      10$:    INC     CHNUM+2                 ;BUMP CHANNEL CTR
 3969  013436  006237  013516              ASR     MSK
 3970  013442  001365                      BNE     9$                      ;BR IF MORE TO CHECK
 3971  013444  012737  000060  013500      MOV     #60,CHNUM+2             ;ELSE RESET CH #
 3972
 3973  013452  005237  013474      11$:    INC     MODNUM                  ;BUMP MOD #
 3974  013456  023727  013474  000065      CMP     MODNUM,#65              ;ALL DONE?
 3975  013464  001325                      BNE     7$                      ;BR IF NO
 3976  013466  104401  001171              TYPE    ,$CRLF
 3977  013472  000207                      RTS     PC                      ;ELSE EXIT
 3978
 3979  013474  000061              MODNUM: .ASCIZ  /1/
 3980  013476  020040  000060      CHNUM:  .ASCIZ  /  0/
 3981
 3982  013502  000000              MASK1:  0                       ;CONTAINS MODULE 1 CHANNELS
 3983  013504  000000              MASK2:  0                       ;                       2
 3984  013506  000000              MASK3:  0                       ;                       3
 3985  013510  000000              MASK4:  0                       ;                       4
 3986  013512  000000              SFTREG: 0
 3987  013514  000000              TMP6:   0
 3988  013516  000000              MSK:    0
 3989  013520  000010              TABL1:  10                      ;SET $DEVM IF CONSOLE ON MODULE 1
 3990  013522  000200                      200                     ;                               2
 3991  013524  004000                      4000                    ;                               3
 3992  013526  100000                      100000                  ;                               4
 3993
```

```
3994
3995
3996
3997                                    .SBTTL  ROUTINE TO TEST FOR CONSOLE DEVICE
3998
3999  013530  113737  001221  013606  TSTCON: MOVB    $USWR+1,TMP7            ;GET CONSOLE INFO ONLY
4000  013536  006237  013606                  ASR     TMP7                    ;RIGHT JUSTIFY
4001  013542  023737  013606  001206          CMP     TMP7,$UNIT              ;SEE IF UNIT UNDER TEST HAS CONSOLE
4002  013550  001403                          BEQ     1$                      ;BR IF YES
4003  013552  005037  013604                  CLR     CONSOLE                 ;ELSE CLEAR STALE INFO
4004  013556  000207                          RTS     PC
4005
4006  013560  013737  001144  014156  1$:     MOV     $TKS,ADDRESS            ;FORCE TTY ADDRESS
4007  013566  012737  000060  014160          MOV     #TKVEC,VECTOR           ;       VECTOR
4008  013574  012737  000001  013604          MOV     #TRUE,CONSOLE           ;SET FLAG
4009  013602  000207                          RTS     PC
4010
4011  013604  000000                  CONSOLE:        0                       ;FLAG, 1= TESTING CONSOLE DEVICE
4012  013606  000000                  TMP7:           0                       ;HOLD CONSOLE #
4013
4014
4015
4016                                    .SBTTL  INTSRV  INTERRUPT SERVICE ROUTINE
4017
4018                                    ;;*****************************************************************
4019                                    ;*      THIS GLOBAL ROUTINE DOES NOTHING BUT INCREMENT
4020                                    ;*      'INTFLAG'  EACH TIME IT IS CALLED.  IT ASSUMES
4021                                    ;*      THAT THE MAIN CALLING ROUTINE WILL  KNOW  WHAT
4022                                    ;*      TO LOOK FOR.
4023                                    ;;*****************************************************************
4024  013610                            BGNSRV INTSRV                   ;GLOBAL INTERRUPT SERVICE ROUTINE
4025  013610                            INTSRV:
4026                                                    ;ADD 1 TO 'INTERRUPT OCCURED' FLAG
4027  013610                                                          LET INTFLAG := INTFLAG + #1
4028  013610  005237  013616                  INC     INTFLAG
4029  013614                                  ENDSRV                          ;THAT'S ALL
4030  013614  000002                          RTI
4031  013616  000000                  INTFLAG:  0
4032
```

```
4033
4034                                    .SBTTL  ROUTINE TO SET UP FOR NEXT MODULE/CHANNEL ADDRESS
4035
4036
4037                                    ;;*****************************************************************
4038                                    ;*      THIS  ROUTINE  CAUSES  ADRS   TO POINT TO  THE
4039                                    ;*      ADDRESS  OF  DLV11-J  UNDER  TEST,  ADRS +2 TO
4040                                    ;*      POINT TO THE VECTOR OF THE DLV11-J UNDER TEST.
4041                                    ;;*****************************************************************
4042
4043  013620  013737  001250  014156  CYCLE:  MOV     $BASE,ADDRESS
4044  013626  013737  001244  014160          MOV     $VECT1,VECTOR
4045
4046  013634  023727  001206  000005          CMP     $UNIT,#5                ;DONE ALL MODULES?
4047  013642  001013                          BNE     1$                      ;BR IF NO
4048  013644  005037  014166                  CLR     PHASE2                  ;RE-INIT
4049  013650  005037  014164                  CLR     CHAN
4050  013654  012737  000001  001206          MOV     #1,$UNIT
4051  013662  005037  014170                  CLR     P1CNT
4052  013666  000137  014362                  JMP     $EOP                    ;& EXIT BACK TO 'LOOP'
4053                                                                          ;STACK WILL GET RESET THERE.
4054  013672  013700  001206          1$:     MOV     $UNIT,R0
4055  013676  006300                          ASL     R0                      ;MULT BY 2 TO INDEX INTO TABLES
4056  013700  066037  014114  014156          ADD     TABL2-2(R0),ADDRESS     ;ADD MODULE OFFSET
4057  013706  066037  014114  014160          ADD     TABL2-2(R0),VECTOR      ;ADD VECTOR OFFSET
4058  013714  017037  014124  014162          MOV     @TABL3-2(R0),MASK       ;STORE CHANNELS IN 'MASK'
4059
4060  013722  005037  013604                  CLR     CONSOLE
4061  013726  005737  014162                  TST     MASK                    ;ANY CHANNELS TO BE TESTED?
4062  013732  001004                          BNE     2$                      ;BR IF YES
4063  013734  005237  001206                  INC     $UNIT                   ;ELSE BUMP UNIT #
4064  013740  000137  013620                  JMP     CYCLE                   ;TRY AGAIN
4065
4066  013744  023727  014166  000001  2$:     CMP     PHASE2,#TRUE            ;DOING PHASE 2?
4067  013752  001452                          BEQ     6$                      ;BR IF YES
4068  013754  023727  014164  000004          CMP     CHAN,#4                 ;ELSE, DONE ALL CHANNELS?
4069  013762  001021                          BNE     4$                      ;BR IF NO
4070
4071  013764  005037  014164                  CLR     CHAN                    ;ELSE BACK TO 0
4072  013770  032737  000020  001220          BIT     #WRAP,$USWR             ;DOING DATA WRAP TESTS?
4073  013776  001004                          BNE     3$                      ;BR IF YES
4074  014000  005237  001206                  INC     $UNIT                   ;ELSE GO TO NEXT UNIT, NO PHASE 2
4075  014004  000137  013620                  JMP     CYCLE                   ;& DO IT
4076
4077  014010  012737  000001  014166  3$:     MOV     #TRUE,PHASE2
4078  014016  005037  014170                  CLR     P1CNT
4079  014022  000137  013620                  JMP     CYCLE                   ;GO TO PHASE 2 TESTING
4080
4081  014026  013700  014164          4$:     MOV     CHAN,R0
4082  014032  006300                          ASL     R0                      ;MULT BY 2 TO INDEX TABLE
4083  014034  066037  014136  014156          ADD     TABL4(R0),ADDRESS       ;ADD CHANNEL OFFSET
4084  014042  066037  014136  014160          ADD     TABL4(R0),VECTOR        ;ADD VECTOR OFFSET
4085  014050  036037  014146  014162          BIT     TABL5(R0),MASK          ;IS THAT CHANNEL TO BE TESTED?
4086  014056  001413                          BEQ     7$                      ;BR IF NO
4087
4088  014060  023727  014164  000003          CMP     CHAN,#3                 ;IS IT CHAN 3?
```

```
                                        BNE     5$                      ;BR IF NO
                                        JSR     PC,TSTCON               ;ELSE SEE IF ITS THE CONSOLE

                                5$:     INC     CHAN                    ;BUMP CHANNEL CTR FOR NEXT TIME
                                6$:     MOV     #ADDRESS,ADRS
                                        RTS     PC                      ;GO TO PHASE 1 TESTS

                                7$:     INC     CHAN
                                        JMP     CYCLE                   ;TRY AGAIN

                                TABL2:  0               ;OFFSET FOR MODULE 1
                                        40              ;                  2
                                        100             ;                  3
                                        140             ;                  4

                                TABL3:  MASK1           ;MODULE 1 CHANNEL DATA
                                        MASK2           ;        2
                                        MASK3           ;        3
                                        MASK4           ;        4

                                TABL4:  0               ;CHANNEL 0 OFFSET
                                        10              ;         1
                                        20              ;         2
                                        30              ;         3

                                TABL5:  BIT0            ;BIT TO TEST CHANNEL 0 PRESENT
                                        BIT1            ;                    1
                                        BIT2            ;                    2
                                        BIT3            ;                    3

                                ADDRESS: 0
                                VECTOR: 0
                                MASK:   0
                                CHAN:   0
                                PHASE2: 0
                                P1CNT:  0               ;CTR TO PRINT 'PHASE 1' ONLY ONCE
```

```
 4127
 4128  014172                                                    ROUTINE MYTYPE
 4129  014172                          MYTYPE:
 4130  014172                                  TYPTXT  <<CRLF>*TEST # *>
 4131  014212                                  TYPOCT  $TESTN
 4132  014220                                  TYPTXT  <*,ERROR # *>
 4133  014240  113737  001114  001176          MOVB    $ITEMB,$FATAL   ; APT FATAL ERROR NUMBER
 4134  014246                                  TYPOCS  $FATAL
 4135  014256                                  TYPTXT  <*,PC = *>
 4136  014274                                  TYPOCT  $ERRPC
 4137  014302                                  TYPTXT  <*,CSR: *>
 4138  014320                                  TYPOCT  DLADD
 4139  014326                                  TYPTXT  <*,VECTOR: *>
 4140  014346                                  TYPOCT  DLVEC
 4141  014354  104401  001171                  TYPE    ,$CRLF
 4142  014360                                                    ENDRTN
 4143  014360                          $255:
 4144  014360                          $256:
 4145  014360  000207                          RTS     PC
 4146
```

```
 4147                                   .SBTTL  END OF PASS ROUTINE
 4148
 4149                                   ;;**********************************************************************
 4150                                   ;*INCREMENT THE PASS NUMBER ($PASS)
 4151                                   ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
 4152                                   ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
 4153                                   ;*IF THERES A MONITOR GO TO IT
 4154                                   ;*IF THERE ISN'T JUMP TO LOOP
 4155
 4156  014362                           $EOP:
 4157  014362  000004                          SCOPE
 4158  014364  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 4159  014370  005037  001160                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 4160  014374  005237  001202                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
 4161  014400  042737  100000  001202        BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 4162  014406  005327                          DEC     (PC)+           ;;LOOP?
 4163  014410  000001                   $EOPCT: .WORD   1
 4164  014412  003022                          BGT     $DOAGN          ;;YES
 4165  014414  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 4166  014416  000001                   $ENDCT: .WORD   1
 4167  014420  014410                          $EOPCT
 4168  014422  104401  014467                  TYPE    ,$ENDMG         ;;TYPE 'END PASS #''
 4169  014426  013746  001202                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 4170  014432  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 4171  014434  104401  014464                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 4172  014440  013700  000042           $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 4173  014444  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 4174  014446  000005                          RESET                   ;;CLEAR THE WORLD
 4175  014450  004710                   $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 4176  014452  000240                          NOP                     ;;SAVE ROOM
 4177  014454  000240                          NOP                     ;;FOR
 4178  014456  000240                          NOP                     ;;ACT11
 4179  014460                           $DOAGN:
 4180  014460  000137                          JMP     @(PC)+          ;;RETURN
 4181  014462  001732                   $RTNAD: .WORD   LOOP
 4182  014464     377     377     000   $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 4183  014467     015  042412  042116   $ENDMG: .ASCIZ  <15><12>/END PASS #/
 4184  014474  050040  051501  020123
 4185  014502  000043
```

```
 4186                                   .SBTTL  TYPE ROUTINE
 4187
 4188                                   ;;*********************************************************************
 4189                                   ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 4190                                   ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 4191                                   ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 4192                                   ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 4193                                   ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 4194                                   ;*
 4195                                   ;*CALL:
 4196                                   ;*1) USING A TRAP INSTRUCTION
 4197                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 4198                                   ;*OR
 4199                                   ;*      TYPE
 4200                                   ;*      MESADR
 4201                                   ;*
 4202
 4203  014504  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 4204  014510  100002                          BPL     1$              ;;BR IF YES
 4205  014512  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
 4206  014514  000430                          BR      3$              ;;LEAVE
 4207  014516  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
 4208  014520  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 4209  014524  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 4210  014532  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
 4211  014534  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
 4212  014542  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
 4213  014544  010037  014554                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
 4214  014550  004737  015550                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
 4215  014554  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
 4216  014556  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
 4217  014564  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
 4218  014566  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 4219  014570  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 4220  014572  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 4221  014574  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
 4222  014576  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 4223  014602  000002                          RTI                     ;;RETURN
 4224  014604  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 4225  014610  001430                          BEQ     8$
 4226  014612  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 4227  014616  001006                          BNE     5$
 4228  014620  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
 4229  014622  104401                          TYPE                    ;;TYPE A CR AND LF
 4230  014624  001171                          $CRLF
 4231  014626  105037  014762                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 4232  014632  000755                          BR      2$              ;;GET NEXT CHARACTER
 4233  014634  004737  014716          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 4234  014640  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 4235  014644  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
 4236  014646  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 4237                                                                  ;;AND THE NULL CHAR.
 4238  014652  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 4239  014656  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 4240  014660  004737  014716                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
 4241  014664  105337  014762                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
 4242  014670  000770                          BR      7$              ;;LOOP
 4243
 4244                                ;HORIZONTAL TAB PROCESSOR
 4245
 4246  014672  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 4247  014676  004737  014716          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 4248  014702  132737  000007  014762          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 4249  014710  001372                          BNE     9$              ;;TAB STOP
 4250  014712  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
 4251  014714  000724                          BR      2$              ;;GET NEXT CHARACTER
 4252  014716  105777  164226          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 4253  014722  100375                          BPL     $TYPEC
 4254  014724  116677  000002  164220        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 4255  014732  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 4256  014740  001003                          BNE     1$              ;;BRANCH IF NO
 4257  014742  105037  014762                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 4258  014746  000406                          BR      $TYPEX          ;;EXIT
 4259  014750  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 4260  014756  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 4261  014760  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 4262  014762  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 4263  014764  000207          $TYPEX: RTS     PC
 4264
```

```
                                        .SBTTL  TTY INPUT ROUTINE

                                        ;;*******************************************************************
                                        .ENABL  LSB

                                        ;;*******************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                        ;*WHEN OPERATING IN TTY FLAG MODE.
014766  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
014774  001074                        BNE     15$             ;;BRANCH IF NO
014776  105777  164142                TSTB    @$TKS           ;;CHAR THERE?
015002  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
015004  117746  164136                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
015010  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
015014  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
015020  001062                        BNE     15$             ;;NO, RETURN TO USER
015022  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
015030  001456                        BEQ     15$             ;;BRANCH IF YES

015032  104401  015513                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
015036  104401  015520        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
015042  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
015046  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
015050  104401  015531                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
015054  005046          19$:    CLR     -(SP)           ;;CLEAR COUNTER
015056  005046                  CLR     -(SP)           ;;THE NEW SWR
015060  105777  164060          7$:     TSTB    @$TKS           ;;CHAR THERE?
015064  100375                  BPL     7$              ;;IF NOT TRY AGAIN

015066  117746  164054                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
015072  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII


015076  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
015102  001005                  BNE     10$             ;;BRANCH IF NOT
015104  104401  015506                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
015110  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
015114  000757                  BR      19$             ;;LET'S TRY IT AGAIN


015116  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
015122  001022                  BNE     16$             ;;BRANCH IF NO
015124  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
015130  001403                  BEQ     11$             ;;BRANCH IF YES
015132  016677  000002  164000        MOV     2(SP),@SWR      ;;SAVE NEW SWR
015140  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
015144  104401  001171          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
015150  123727  001135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
015156  001003                  BNE     15$             ;;BRANCH IF NOT
015160  012777  000100  163756        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
015166  000002          15$:    RTI                     ;;RETURN
015170  004737  014716          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
015174  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
```

```
 4321  015200  002420                          BLT     18$             ;;BRANCH IF YES
 4322  015202  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 4323  015206  003015                          BGT     18$             ;;BRANCH IF YES
 4324  015210  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 4325  015214  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 4326  015220  001403                          BEQ     17$             ;;BRANCH IF YES
 4327  015222  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 4328  015224  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
 4329  015226  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
 4330  015230  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 4331  015234  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 4332  015240  000707                          BR      7$              ;;GET THE NEXT ONE
 4333  015242  104401  001170          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 4334  015246  000720                          BR      20$             ;;SIMULATE CONTROL-U
 4335                                  .DSABL  LSB
 4336
 4337
 4338                                  ;;*****************************************************************
 4339                                  ;*THIS ROUTINE WIL' INPUT A SINGLE CHARACTER FROM THE TTY
 4340                                  ;*CALL:
 4341                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 4342                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 4343                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
 4344                                  ;
 4345
 4346  015250  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
 4347  015252  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
 4348  015260  105777  163660          1$:     TSTB    @$TKS           ;;WAIT FOR
 4349  015264  100375                          BPL     1$              ;;A CHARACTER
 4350  015266  117766  163654  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
 4351  015274  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 4352  015302  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 4353  015310  001013                          BNE     3$              ;;BRANCH IF NO
 4354  015312  105777  163626          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 4355  015316  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
 4356  015320  117746  163622                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 4357  015324  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4358  015330  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 4359  015334  001366                          BNE     2$              ;;IF NOT DISCARD IT
 4360  015336  000750                          BR      1$              ;;YES, RESUME
 4361  015340  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 4362  015346  002407                          BLT     4$              ;;BRANCH IF YES
 4363  015350  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 4364  015356  003003                          BGT     4$              ;;BRANCH IF YES
 4365  015360  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 4366  015366  000002                  4$:     RTI                     ;;GO BACK TO USER
 4367                                  ;;*****************************************************************
 4368                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 4369                                  ;*CALL:
 4370                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 4371                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 4372                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 4373
 4374  015370  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 4375  015372  012703  015476          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 4376  015376  022703  015506          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
```

```
 4377  015402  101405                          BLOS    4$              ;;BR IF YES
 4378  015404  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 4379  015406  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
 4380  015410  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 4381  015414  001003                          BNE     3$              ;;SKIP IF NOT
 4382  015416  104401  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
 4383  015422  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
 4384  015424  111337  015474          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 4385  015430  104401  015474                  TYPE    ,9$
 4386  015434  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 4387  015440  001356                          BNE     2$              ;;LOOP IF NOT RETURN
 4388  015442  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 4389  015446  104401  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
 4390  015452  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 4391  015454  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 4392  015456  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 4393  015464  012766  015476  000004          MOV     #$TTYIN,4(SP)
 4394  015472  000002                          RTI                     ;;RETURN
 4395  015474     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 4396  015475     000                          .BYTE   0               ;;TERMINATOR
 4397  015476  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
 4398  015506  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
 4399  015513     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
 4400  015520  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR   /
 4401  015526  020075     000
 4402  015531     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
 4403  015536  036440  000040
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE
                                ;;*****************************************************************
015542  112737  000001  016006  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
015550  112737  000001  016004  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
015556  000403                         BR      $ATYC
015560  112737  000001  016006  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
015566                         $ATYC:
015566  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
015570  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015572  105737  016004                 TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
015576  001450                         BEQ     5$              ;;IF NOT:  BR
015600  122737  000001  001214         CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
015606  001031                         BNE     3$              ;;IF NOT:  BR
015610  132737  000100  001215         BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
015616  001425                         BEQ     3$              ;;IF NOT:  BR
015620  017600  000004                 MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
015624  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
015632  005737  001174         1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
015636  001375                         BNE     1$              ;;IF NOT:  WAIT
015640  010037  001210                 MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
015644  105720                 2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
015646  001376                         BNE     2$
015650  163700  001210                 SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
015654  006200                         ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
015656  010037  001212                 MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
015662  012737  000004  001174         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
015670  000413                         BR      5$
015672  017637  000004  015716  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
015700  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
015706  013746  177776                 MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
015712  004737  014504                 JSR     PC,$TYPE        ;;CALL TYPE MACRO
015716  000000                 4$:     .WORD   0
015720                         5$:
015720  105737  016006         10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
015724  001416                         BEQ     12$             ;;IF NOT:  BR
015726  005737  001214                 TST     $ENV            ;;RUNNING UNDER APT?
015732  001413                         BEQ     12$             ;;IF NOT:  BR
015734  005737  001174         11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
015740  001375                         BNE     11$             ;;IF NOT:  WAIT
015742  017637  000004  001176         MOV     @4(SP),$FATAL   ;;GET ERROR #
015750  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
015756  005237  001174                 INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
015762  105037  016006         12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
015766  105037  016005                 CLRB    $LFLG           ;;CLEAR LOG FLAG
015772  105037  016004                 CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
015776  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
016000  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
016002  000207                         RTS     PC              ;;RETURN
016004     000                 $MFLG:  .BYTE   0               ;;MESSG. FLAG
016005     000                 $LFLG:  .BYTE   0               ;;LOG FLAG
016006     000                 $FFLG:  .BYTE   0               ;;FATAL FLAG
            016010                         .EVEN
            000200                 APTSIZE=200
            000001                 APTENV=001
            000100                 APTSPOOL=100
```

```
 4460           000040                  APTCSUP=040
```

```
                                        .SBTTL  ERROR HANDLER ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                        ;*AND GO TO MYTYPE ON ERROR
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW15=1         HALT ON ERROR
                                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                        ;*SW10=1         BELL ON ERROR
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR EMT AND N=ERROR ITEM NUMBER

016010                                  $ERROR:
016010  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
016012  105237  001103                  7$:     INCB    $ERFLG  ;;SET THE ERROR FLAG
016016  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
016020  013777  001102  163114          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
016026  032777  002000  163104          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
016034  001402                          BEQ     1$              ;;NO - SKIP
016036  104401  001164                  TYPE    ,$BELL          ;;RING BELL
016042  005237  001112                  1$:     INC     $ERTTL  ;;COUNT THE NUMBER OF ERRORS
016046  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
016052  162737  000002  001116          SUB     #2,$ERRPC
016060  117737  163032  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
016066  032777  020000  163044          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
016074  001004                          BNE     20$             ;;SKIP TYPEOUTS
016076  004737  014172                  JSR     PC,MYTYPE       ;;GO TO USER ERROR ROUTINE
016102  104401  001171                  TYPE    ,$CRLF
016106                                  20$:
016106  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
016114  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
016116  113737  001114  016130          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
016124  004737  015560                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
016130     000                          21$:    .BYTE   0
016131     000                          .BYTE   0
016132  000777                          22$:    BR      22$     ;;APT ERROR LOOP
016134  005777  163000                  2$:     TST     @SWR    ;;HALT ON ERROR
016140  100002                          BPL     3$              ;;SKIP IF CONTINUE
016142  000000                          HALT                    ;;HALT ON ERROR!
016144  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
016146  032777  001000  162764          3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
016154  001402                          BEQ     4$              ;;BR IF NO
016156  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
016162  005737  001162                  4$:     TST     $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
016166  001402                          BEQ     5$              ;;BR IF NONE
016170  013716  001162                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
016174                                  5$:
016174  022737  014450  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
016202  001001                          BNE     6$              ;;BRANCH IF NO
016204  000000                          HALT                    ;;YES
016206                                  6$:
016206  000002                          RTI                     ;RETURN
```

```
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;*********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09-1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE-IOT

016210                          $SCOPE:
016210  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
016212  032777  040000  162720  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
016220  001014                          BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
016222  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
016224  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
016230  012737  016250  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
016236  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
016242  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
016246  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
016250  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
016252  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
016256  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
016260                          6$:;#####END OF CODE FOR THE XOR TESTER#####
016260  032777  000400  162652          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
016266  001404                          BEQ     2$              ;;BR IF NO
016270  127737  162644  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
016276  001465                          BEQ     $OVER           ;;BR IF YES
016300  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
016304  001421                          BEQ     3$              ;;BR IF NO
016306  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
016314  101015                          BHI     3$              ;;BR IF NO
016316  032777  001000  162614          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
016324  001404                          BEQ     4$              ;;BR IF NO
016326  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
016334  000446                          BR      $OVER
016336  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
016342  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
016346  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
016350  032777  004000  162562  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
016356  001011                          BNE     1$              ;;BR IF YES
016360  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
016364  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
016366  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
016372  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
016400  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
016402  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
016410  013737  016466  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
016416  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
016422  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
```

```
 4571  016430  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 4572  016434  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 4573  016440  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 4574  016444  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 4575  016452  013777  001102  162462  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 4576  016460  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 4577  016464  000002                          RTI                     ;;FIXES PS
 4578  016466  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
```

```
                                        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                        ;;*****************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                        ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                        ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                        ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                        ;*REPLACED WITH SPACES.
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE

016470                                  $TYPDS:
016470  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016472  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016474  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016476  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
016500  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
016502  012746  020200                        MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
016506  016605  000020                        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
016512  100004                                BPL     1$              ;;BR IF INPUT IS POS.
016514  005405                                NEG     R5              ;;MAKE THE BINARY NUMBER POS.
016516  112766  000055  000001                MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
016524  005000                        1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
016526  012703  016704                        MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
016532  112723  000040                        MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
016536  005002                        2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
016540  016001  016674                        MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
016544  160105                        3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
016546  002402                                BLT     4$              ;;BR IF DONE
016550  005202                                INC     R2              ;;INCREASE THE BCD DIGIT BY 1
016552  000774                                BR      3$
016554  060105                        4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
016556  005702                                TST     R2              ;;CHECK IF BCD DIGIT 0
016560  001002                                BNE     5$              ;;FALL THROUGH IF 0
016562  105716                                TSTB    (SP)            ;;STILL DOING LEADING 0'S?
016564  100407                                BMI     7$              ;;BR IF YES
016566  106316                        5$:     ASLB    (SP)            ;;MSD?
016570  103003                                BCC     6$              ;;BR IF NO
016572  116663  000001  177777                MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
016600  052702  000060                6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
016604  052702  000040                7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
016610  110223                                MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
016612  005720                                TST     (R0)+           ;;JUST INCREMENTING
016614  020027  000010                        CMP     R0,#10          ;;CHECK THE TABLE INDEX
016620  002746                                BLT     2$              ;;GO DO THE NEXT DIGIT
016622  003002                                BGT     8$              ;;GO TO EXIT
016624  010502                                MOV     R5,R2           ;;GET THE LSD
016626  000764                                BR      6$              ;;GO CHANGE TO ASCII
016630  105726                        8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
016632  100003                                BPL     9$              ;;BR IF NO
016634  116663  177777  177776                MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
016642  105013                        9$:     CLRB    (R3)            ;;SET THE TERMINATOR
016644  012605                                MOV     (SP)+,R5        ;;POP STACK INTO R5
016646  012603                                MOV     (SP)+,R3        ;;POP STACK INTO R3
016650  012602                                MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
 4635  016652  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 4636  016654  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 4637  016656  104401  016704                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4638  016662  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4639  016670  012616                          MOV     (SP)+,(SP)
 4640  016672  000002                          RTI                     ;;RETURN TO USER
 4641  016674  023420                  $DTBL:  10000.
 4642  016676  001750                          1000.
 4643  016700  000144                          100.
 4644  016702  000012                          10.
 4645  016704  000004                  $DBLK:  .BLKW   4
```

```
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;*****************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                                      ;;1=TYPE LEADING ZEROS
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT

016714  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
016720  116637  000001  017137         MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
016726  112637  017141                 MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
016732  062716  000002                 ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
016736  000406                         BR      $TYPON
016740  112737  000001  017137 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
016746  112737  000006  017141         MOVB    #6,$OMODE+1     ;;SET FOR SIX (6) DIGITS
016754  112737  000005  017136 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
016762  010346                         MOV     R3,-(SP)        ;;SAVE R3
016764  010446                         MOV     R4,-(SP)        ;;SAVE R4
016766  010546                         MOV     R5,-(SP)        ;;SAVE R5
016770  113704  017141                 MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
016774  005404                         NEG     R4
016776  062704  000006                 ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
017002  110437  017140                 MOVB    R4,$OMODE       ;;SAVE IT FOR USE
017006  113704  017137                 MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
017012  016605  000012                 MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
017016  005003                         CLR     R3              ;;CLEAR THE OUTPUT WORD
017020  006105                 1$:     ROL     R5              ;;ROTATE MSB INTO "C"
017022  000404                         BR      3$              ;;GO DO MSB
017024  006105                 2$:     ROL     R5              ;;FORM THIS DIGIT
017026  006105                         ROL     R5
017030  006105                         ROL     R5
017032  010503                         MOV     R5,R3
017034  006103                 3$:     ROL     R3              ;;GET MSB OF THIS DIGIT
017036  105337  017140                 DECB    $OMODE          ;;TYPE THIS DIGIT?
017042  100016                         BPL     7$              ;;BR IF NO
017044  042703  177770                 BIC     #177770,R3      ;;GET RID OF JUNK
017050  001002                         BNE     4$              ;;TEST FOR 0
017052  005704                         TST     R4              ;;SUPPRESS THIS 0?
017054  001403                         BEQ     5$              ;;BR IF YES
```

```
 4702  017056  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 4703  017060  052703  000060                   BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 4704  017064  052703  000040          5$:      BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 4705  017070  110337  017134                   MOVB    R3,8$           ;;SAVE FOR TYPING
 4706  017074  104401  017134                   TYPE    ,8$             ;;GO TYPE THIS DIGIT
 4707  017100  105337  017136          7$:      DECB    $OCNT           ;;COUNT BY 1
 4708  017104  003347                           BGT     2$              ;;BR IF MORE TO DO
 4709  017106  002402                           BLT     6$              ;;BR IF DONE
 4710  017110  005204                           INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 4711  017112  000744                           BR      2$              ;;GO DO THE LAST DIGIT
 4712  017114  012605                  6$:      MOV     (SP)+,R5        ;;RESTORE R5
 4713  017116  012604                           MOV     (SP)+,R4        ;;RESTORE R4
 4714  017120  012603                           MOV     (SP)+,R3        ;;RESTORE R3
 4715  017122  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 4716  017130  012616                           MOV     (SP)+,(SP)
 4717  017132  000002                           RTI                     ;;RETURN
 4718  017134     000                  8$:      .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 4719  017135     000                           .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 4720  017136     000                  $OCNT:   .BYTE   0               ;;OCTAL DIGIT COUNTER
 4721  017137     000                  $OFILL:  .BYTE   0               ;;ZERO FILL SWITCH
 4722  017140  000000                  $OMODE:  .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .SBTTL  TRAP DECODER
                        ;
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

017142  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
017144  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
017150  005740                          TST     -(R0)           ;;BACKUP BY 2
017152  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
017154  006300                          ASL     R0              ;;POSITION FOR INDEXING
017156  016000  017176                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
017162  000200                          RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO

017164  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
017166  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
017174  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE ''TRAP'' INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
017176  017164                  $TRPAD: .WORD   $TRAP2
017200  014504                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
017202  016740                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
017204  016714                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
017206  016754                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
017210  016470                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

017212  015036                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

017214  014766                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
017216  015250                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
017220  015370                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE

                                        ;MAXMEM FOR APT = 17400
                                        ;MAXMEM NON APT   17500  (ABS LOADER)
        000001                  .END
```

```
ABASE = 176500
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ACTCH   012364
ADDRES  014156
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
ADRS  =%000001
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3  000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATESTN  000000
AUNIT = 000000
AUSWR   001031
AVECT1= 000300
AVECT2= 000000
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1    000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14   040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4    000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BPTVEC= 000014
BREAK = 000001
BRK   = 000010
CHAN    014164
CHCNT   007766
CHCTR   012362
CHMASK  007764
CHMSK   012360
CHNUM   013476
CHOTAB  011360
CH1TAB  011560
CH2TAB  011760
CH3TAB  012160
CKSWR = 104407
CLR   = 000000
CONSOL  013604
CR    = 000015
CRLF  = 000200
CYCLE   013620
DATA    005662
DDISP = 177570
DISPLA  001142
DISPRE  000174
DLADD   001254
DLADDR  176500
DLVEC   001256
DSWR  = 177570
EMTVEC= 000030
ERRCHK  006500
ERRCNT  005622
ERROR = 100000
ERRORF  003304
ERRVEC= 000004
ERWRD   012366
EVENOD= 000004
EXITFL  003306
FLAG    012612
FRERR = 020000
GTSWR = 104406
HOLDSC  012613
HOWLON= 000000
HT    = 000011
I       001270
INTFLA  013616
INTRTA  007744
INTSRV  013610
IOTVEC= 000020
LF    = 000012
LOOP    001732
MASK    014162
MASK1   013502
MASK2   013504
MASK3   013506
MASK4   013510
MODNUM  013474
MODTST  007000
MSK     013516
MYTYPE  014172
NUMBER  005624
ORERR = 040000
PARITY= 000002
PASS    003302
PERR  = 010000
PHASE2  014166
PIRQ  = 177772
PIRQVE= 000240
PRIERR  007770
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6     000300
PR7   = 000340
PS      177776
PSW   = 177776
PWRVEC= 000024
P1CNT   014170
RBUF    001262
RCH0    007724
RCH1    007726
RCH2    007730
RCH3    007732
RCSR    001260
RCVRDO= 000200
RCVRIE= 000100
RCVR0S  007424
RCVR1S  007504
RCVR2S  007564
RCVR3S  007644
RDATA0= 000001
RDATA1= 000002
RDATA2= 000004
RDATA3= 000010
RDATA4= 000020
RDATA5= 000040
RDATA6= 000100
RDATA7= 000200
RDCHR = 104410
RDLIN = 104411
RDRRUN= 000001
REC     005664
REG   = 000004
REGSAV  012606
RESVEC= 000010
RHLD    005770
R0SRV   010660
R1SRV   011000
R2SRV   011120
R3SRV   011240
R5STAC  001332
R6    =%000006
R7    =%000007
SEREND  011360
SET   = 177777
SETCLR= 000006
SFTREG  013512
SIZE    012700
SRVEND  007724
STACK = 001100
START   001334
STATEN  012360
STKLMT= 177774
SWR     001140
SWREG   000176
SW0   = 000001
SW00    000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07    000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7     000200
SW8   = 000400
SW9   = 001000
TABEND  007764
TABL1   013520
TABL2   014116
TABL3   014126
TABL4   014136
TABL5   014146
TABL6   012416
TABL7   012426
TABL8   012436
TBITVE= 000014
TBUF    001266
TCH0    007734
TCH1    007736
TCH2    007740
TCH3    007742
TCNT    007772
TCSR    001264
TDATA0= 000001
TDATA1= 000002
TDATA2= 000004
TDATA3= 000010
TDATA4= 000020
TDATA5= 000040
TDATA6= 000100
TDATA7= 000200
TEMP    007774
TIME  = 000000
TIMER   012446
TIMSAV  012610
TKVEC   000060
TMP     012370
TMP0    012372
```

```
TMPOE   012374    $APTHD  001000    $F$INC- 000210    $MXCNT  016466    $TYPEC  014716
TMP1    012376    $ATYC   015566    $F$LOO= 000200    $NESTL= 177777    $TYPEX  014764
TMP1E   012400    $ATY1   015542    $F$NAM= 000160    $NSK0 = 000300    $TYPOC  016740
TMP2    012402    $ATY3   015550    $F$NO - 000403    $NSK1 = 000210    $TYPON  016754
TMP2E   012404    $ATY4   015560    $F$OR - 000320    $NSK2 = 000210    $TYPOS  016714
TMP3    012406    $AUTOB  001134    $F$RTN= 000300    $NSK3 = 000110    $UNIT   001206
TMP3E   012410    $BASE   001250    $F$SFL= 000140    $NULL   001154    $UNITM  001010
TMP4    012414    $BDADR  001122    $F$UNT= 000130    $NWTST= 000001    $USWR   001220
TMP6    013514    $BDDAT  001126    $F$WHI= 000120    $OCNT   017136    $VECT1  001244
TMP7    013606    $BELL   001164    $F$YES= 000402    $OMODE  017140    $VECT2  001246
TPVEC = 000064    $BGNLE= 177777    $GDADR  001120    $OVER   016452    $XTSTR  016222
TRAN    005626    $CHARC  014762    $GDDAT  001124    $PASS   001202    $YESNO= 000001
TRAPVE= 000034    $CKSWR  014766    $GET42  014440    $PASTM  001006    $$ARGC= 000000
TRTVEC= 000014    $CMTAG  001100    $GTSWR  015036    $QUES   001170    $$BYTE= 000402
TRUE  = 000001    $CM3  - 000000    $HD   = 000000    $RDCHR  015250    $$FLAG= 000001
TSTCON  013530    $CNTLG  015513    $HIBTS  001000    $RDLIN  015370    $$FROM= 000000
TST1    002066    $CNTL   015506    $ICNT   001104    $RDSZ - 000010    $$GET4= 000000
TST10   003440    $CPUOP  001222    $IFLEV= 177777    $RTNAD  014462    $$LOC = 012576
TST11   003566    $CRLF   001171    $INTAG  001135    $SAVLE= 177777    $$LOCN= 000000
TST12   003712    $DBLK   016704    $ISK0 = 000001    $SCOPE  016210    $$RETU= 000000
TST13   004222    $DEVCT  001204    $ISK1 - 000001    $SETUP= 000127    $$RTN1= 000255
TST14   004512    $DEVM   001252    $ISK2 - 000001    $SSKO = 000253    $$RTN2= 000256
TST15   005046    $DOAGN  014460    $ITEMB  001114    $STUP - 177777    $$TO  = 000001
TST16   005322    $DTBL   016674    $LF     001172    $SVLAD  016416    $OFILL  017137
TST17   005772    $ENDAD  014450    $LFLG   016005    $SVPC = 000430    $1      002062
TST2    002264    $ENDCT  014416    $LOCTA= 177777    $SWR  = 167400    $10     002410
TST20   006502    $ENDMG  014467    $LPADR  001106    $SWREG  001216    $100    005162
TST21   007000    $ENULL  014464    $LPERR  001110    $SWRMK= 000000    $101    005224
TST22   007776    $ENV    001214    $LSTCN= 177777    $TAGLE= 177777    $102    005240
TST3    002440    $ENVM   001215    $LSTIN= 000000    $TAGNU= 000257    $103    005276
TST4    002614    $EOP    014362    $LSTST= 177777    $TEMP = 000300    $104    005270
TST5    002756    $EOPCT  014410    $LSTTA= 000000    $TESTN  001200    $105    005266
TST6    003016    $ERFLG  001103    $MADR1  001226    $TIMES  001160    $106    005274
TST7    003310    $ERMAX  001115    $MADR2  001232    $TKB    001146    $107    005320
TYPDS = 104405    $ERRFL= 000000    $MADR3  001236    $TKS    001144    $11     002440
TYPE  = 104401    $ERROR  016010    $MADR4  001242    $TN   = 000023    $110    005320
TYPOC = 104402    $ERRPC  001116    $MAIL   001174    $TPB    001152    $111    005360
TYPON = 104404    $ERRTB  001254    $MAMS1  001224    $TPFLG  001157    $112    005364
TYPOS - 104403    $ERTTL  001112    $MAMS2  001230    $TPS    001150    $113    005452
VECTOR  014160    $ESCAP  001162    $MAMS3  001234    $TRAP   017142    $114    005460
WAIT    012620    $ETABL  001214    $MAMS4  001240    $TRAP2  017164    $115    005476
WHICHB- 000002    $ETEND  001254    $MBADR  001002    $TRP  = 000012    $116    005514
WRAP  = 000020    $FATAL  001176    $MCALL= 000000    $TRPAD  017176    $117    005620
XMITIE  000100    $FFLG   016006    $MFLG   016004    $TSK0 = 000251    $12     002510
XMITRD= 000200    $FILLC  001156    $MNEW   015531    $TSK1 = 000250    $120    005616
XMITOS  007454    $FILLS  001155    $MSGAD  001210    $TSK2 = 000254    $121    005562
XMIT1S  007534    $F$AND= 000310    $MSGLG  001212    $TSK3 = 000253    $122    005614
XMIT2S  007614    $F$BAD  000401    $MSGTY  001174    $TSK4 = 000126    $123    005576
XMIT3S  007674    $F$BLA= 000170    $MSWR   015520    $TSTM   001004    $124    005614
XOSRV   010734    $F$CAS= 000150    $MTYP1  001225    $TSTNM  001102    $125    005612
X1SRV   011054    $F$DEC  000220    $MTYP2  001231    $TTYIN  015476    $126    005614
X2SRV   011174    $F$GOO= 000400    $MTYP3  001235    $TYPDS  016470    $127    005620
X3SRV   011314    $F$IF   000110    $MTYP4  001241    $TYPE   014504    $13     002536
```

$130	005652	$163	006646	$215	010524	$25	003300	$51	003754
$131	005712	$164	006646	$216	010534	$250	012640	$52	003750
$132	005734	$165	006774	$217	010554	$251	012670	$53	003752
$133	005766	$166	007022	$22	003016	$252	012656	$54	003756
$134	005760	$167	007054	$220	010614	$253	012654	$55	004024
$135	005766	$17	002700	$221	010614	$254	012666	$56	004062
$136	006030	$170	007112	$222	010632	$255	014360	$57	004102
$137	006036	$171	007202	$223	010644	$256	014360	$6	002334
$14	002564	$172	007202	$224	010654	$26	003126	$60	004124
$140	006052	$173	007244	$225	010726	$27	003166	$61	004172
$141	006052	$174	007244	$226	010732	$3	002130	$62	004206
$142	006134	$175	007422	$227	011046	$30	003222	$63	004220
$143	006152	$176	010024	$23	003046	$31	003222	$64	004404
$144	006166	$177	010044	$230	011052	$32	003260	$65	004432
$145	006240	$2	002066	$231	011166	$33	003256	$66	004550
$146	006224	$20	002726	$232	011172	$34	003266	$67	004554
$147	006222	$200	010132	$233	011306	$35	003346	$7	002362
$15	002614	$201	010170	$234	011312	$36	003350	$70	004724
$150	006240	$202	010174	$235	012614	$37	003416	$71	004752
$151	006240	$203	010232	$236	012616	$4	002154	$72	005032
$152	006266	$204	010232	$237	012466	$40	003440	$73	005104
$153	006300	$205	010274	$24	003052	$41	003476	$74	005106
$154	006312	$206	010316	$240	012570	$42	003500	$75	005116
$155	006324	$207	010326	$241	012504	$43	003546	$76	005114
$156	006374	$21	002756	$242	012512	$44	003566	$77	005322
$157	006464	$210	010360	$243	012530	$45	003624	.	- 017222
$16	002652	$211	010374	$244	012602	$46	003626	.$x	001000
$160	006476	$212	010450	$245	012676	$47	003674		
$161	006750	$213	010554	$246	012676	$5	002314		
$162	006574	$214	010510	$247	012644	$50	003712		

. ABS. 017222 000 (ON RO ABS LCL D

ERRORS DETECTED: 0

Z:CVDLAB/I,Z:CVDLAB.SEQ=SPMAC.SML,CVDLAB.P11
RUN-TIME: 101 93 .7 SECONDS
RUN-TIME RATIO: 312/195 1.5
CORE USED: 33K (65 PAGES)