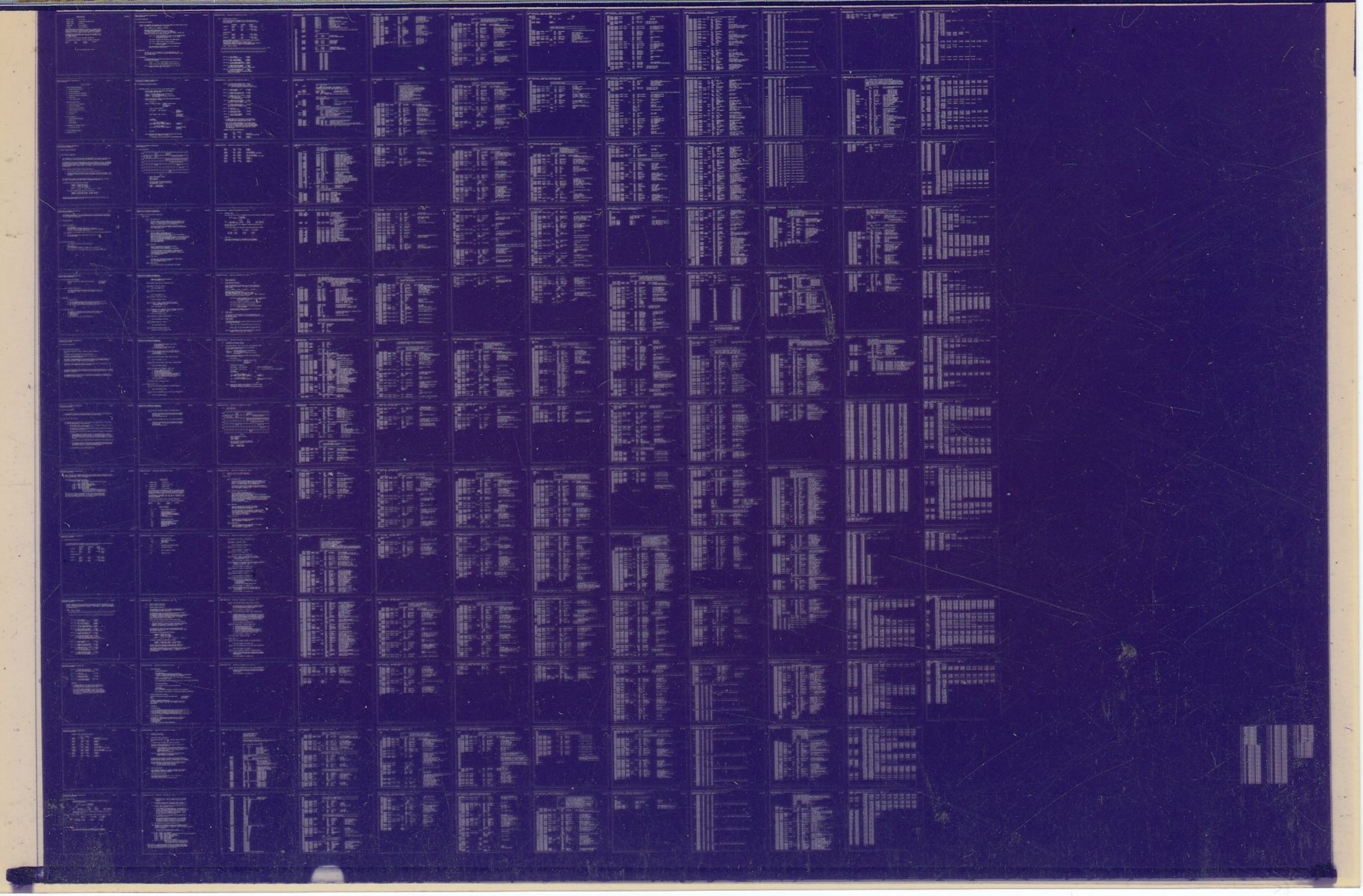
MxVM-B Hfunct Diag AH-T552A-HC
CVMxBAO Juli 1983



IDENTIFICATION

SEQ 0001

PRODUCT CODE: AC-T551A-MC

PRODUCT NAME: CVMXBA0 MXV11-B MFUNCT DIAG

PRODUCT DATE: 02-MAR-83

MAINTAINER: E.S.D. METHODS

AUTHOR: DICE SYSTEMS,INC.

MXV11-B DIAGNOSTIC USER'S MANUAL

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

THIS DIAGNOSTIC IS A LOGIC TEST TO VERIFY THE OPERATION OF THE 2 SERIAL LINE UNITS, ROM AND CLOCK OPTIONS, THE PCR REGISTER AND THE DDR REGISTER, AND RAM ON THE MXV11-B.

THE PROGRAM WILL TEST TO WHATEVER OPTIONS THE DEVICE MAP ($DEVM) IS SET TO. SEE PROGRAM OPTIONS AND DEFAULTS SEC 2.4. THE PROGRAM WILL PRINT THE CONTENTS OF $DEVM FOR OPERATOR VERIFICATION AND A SUMMARY OF SIGNIFICANT DIFFERENCES FROM THE DEFAULT CONDITIONS I.E.: CHANNEL(S) DROPPED FROM TESTING, CHANNEL 1 AS CONSOLE. ROM/RAM TESTING BYPASSED AND CLOCK ENABLED.

SERIAL LINE UNIT TESTING IS DONE IN TWO DISTINCT PHASES:

1. EACH OF THE 2 CHANNELS OF THE MXV11-B IS TESTED INDIVIDUALLY.
2. THE MXV11-B MODULE IS TESTED AS A WHOLE FOR CHANNEL INTERACTION PROBLEMS. THIS DIAGNOSTIC IS DESIGNED TO TEST AND DETECT ERRORS TO THE LOGIC LEVEL AND NOT THE CHIP LEVEL.

THE OPERATOR MUST INSTALL DATA WRAP AROUND CONNECTORS TO DO DATA TESTING. TO BYPASS DATA TESTS, THE OPERATOR MUST MODIFY $DEVM. SEE PROGRAM OPTIONS SEC. 2.4

THE ADDRESSES AND VECTOR RANGES ARE AS FOLLOWS:

```
CHANNEL 0: 776500 THRU 777570
CHANNEL 1: 776510 THRU 77760C
           777560   ( AS CONSOLE )
VECTORS  :      4 THRU    376
PCR      : 777520       DDR:  777524    LTC:   777546
```

THE DEFAULT ADDRESSES AND VECTORS ARE AS FOLLOWS:

```
CHANNEL 0: 776500 THRU 776506   VECTORS: 300/304
CHANNEL 1: 177560 THRU 177566   VECTORS:  60/64
```

FOR ANY OTHER DEVICE ADDRESSES THE OPERATOR MUST CHANGE THE DEFAULT LOCATIONS. SEE PROGRAM OPTIONS AND DEFAULTS SEC. 2.4 .

THIS PROGRAM IS DESIGNED TO RUN ON ANY Q-BUS/Q22 PDP-11, WITH 64K MEMORY, AND WITH AN MXV11-B (Q-BUS/Q22) MODULE. IT CAN RUN UNDER XXDP AND APT MONITORS, AND ON PROCESSORS WITH NO HARDWARE SWITCH REGISTER.

1.2 SYSTEM REQUIREMENTS.

1.2.1 HARDWARE REQUIREMENTS: -

KDF11 PROCESSOR
64K WORDS MEMORY - MINIMUM (CONTAINED ON THE MXV11-B)
A SPECIAL DATA WRAP AROUND CONNECTOR (PN M3270-A) OR EQUIVALENT
(REQ'D IF DATA WRAP AROUND TESTS DESIRED)

IF CHAN. 1 IS THE CONSOLE, TESTS 11-14, 16-21, 23-24 ARE BYPASSED (BYPASS FOR CHAN.1 ONLY).

IF DATA WRAP AROUND TESTS ARE BYPASSED, TESTS 12-14, 16-2'
23-24 ARE BYPASSED.

1.2.2 SOFTWARE REQUIREMENTS: -

THIS DIAGNOSTIC CAN RUN IN 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 DIAGNOSTICS ARE REQUIRED TO RUN BEFURE THIS, BUT THE PROCESSOR, MEMORY, AND BUS ARE ASSUMED TO BE FULLY OPERATIONAL.

1.5 ASSUMPTIONS.

THE OPERATOR MUST:

1. SET THE SOFTWARE SWITCH REGISTER (SWR) IF NOT DEFAULTED.(SEC. 2.3)
2. SET THE DEVICE MAP ($DEVM) IF NOT DEFAULTED.(SEC. 2.4)
3. SET THE SERIAL LINE ADDRESSES AND VECTORS IF NOT DEFAULTED.(SEC 2.4)
4. SET THE RUM HI AND LO ADDR IF NOT DEFAULTED.(SEC 2.4)

1.6 VT-100 CONSOLE SETUP

IF CHANNEL 1 IS CONFIGURED AS THE CONSOLE AND THE VT-100 IS THE CONSOLE DEVICE, "SETUP B" MUST BE SET IN FOR .HE FOLLOWING:

A. DISABLE XON/XOFF
B. JUMP SCROLL ON
C. NEW LINE OFF
D. ALL OTHER OPTIONS PER SYSTEM REQUIREMENTS.

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES.

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

ALL NORMAL STARTS AND RESTARTS ARE FROM LOCATION 200.

THERE ARE 2 STARTING ADDRESSES TO BE USED OFF LINE ONLY FOR INTERRUPT VECTOR TROUBLE SHOOTING:

1334 START: LOADS ADDRESSES 0 TO 1000 WITH THE ADDRESS OF AN INTERRUPT ROUTINE THAT JUST DOES RTI'S ALLOWING LOOPING IN THAT PART OF THE TEST WHERE INTERRUPT VECTOR PROBLEMS ARE OCCURING. NORMAL TESTING WILL THEN BEGIN.

1356 START: LOADS ADDRESSES 0 TO 1000 WITH TRAP CATCHER CODE. ANY INTERRUPT TO THIS REGION WILL HALT. NORMAL TESTING WILL THEN BEGIN.

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

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 UNDER 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:

   1. 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.

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

   3. IF THE INPUT CHARACTER IS NOT ONE OF THE CHARACTERS MENTIONED ABOVE THEN A QUESTION MARK (?) WILL BE TYPED CARRIAGE RETURN AND A LINE FEED SEQUENCE THEN PROCEED FROM STEP 2 (ERASING ALL PREVIOUS INPUT).

4. THE DIAGNOSTIC WILL CONTINUE ON RUNNING <CR>.

NOTE: BECAUSE OF THE FREQUENT BUS RESETS IN THE PROGRAM, MULTIPLE CONTROL-G'S MAY BE REQ'D. IF NECESSARY, 'BREAK' INTO THE PROGRAM AND LOAD LOCATION 176 (SWREG) BY 'ODT' TO THE DESIRED CONTENTS. DO A 'P' TO PROCEED.

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)

NOTE: IF BIT 14, 9 OR 8 IS SELECTED THE LEDS ON THE MXV11-B MODULE MAY GIVE FALSE TEST NUMBER INDICATIONS. IF A TEST HAS A UNIBUS RESET COMMAND THE LEDS WILL BE FULLY LIGHTED EVERY PASS THROUGH THE RESET INSTRUCTION. THUS, TEST 7 WILL APPEAR AS TEST 17 (ALL LEDS ON) ETC.

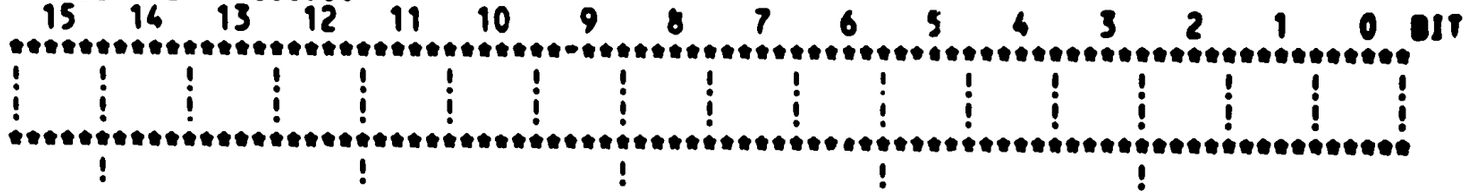
2.4 PROGRAM OPTIONS AND DEFAULTS.

THIS PROGRAM REQUIRES THE ADDRESS OF THE FIRST RCSR OF EACH SERIAL LINE UNIT AND ITS INTERRUPT VECTORS TO BE PREVIOUSLY STORED IF NOT DEFAULTED:

	REGISTER	LOCATION	DEFAULT	
CHANNEL 0	BASE0	1254	776500	RCSR
	VECT0	1256	300	VECTOR
CHANNEL 1	BASE1	1260	777560	RCSR
	VECT1	1262	60	VECTOR
SECOND MXV11-B				
CHANNEL 2	BASE2	1264	776510	RCSR
	VECT2	1266	310	VECTOR
CHANNEL 3	BASE3	1270	776520	RCSR
	VECT3	1272	320	VECTOR

LOCATION '$DEVM' IS USED AS A BIT MAP TO INDICATE WHICH OPTIONS ARE PRESENT AND TO BE TESTED. THE OPERATOR IS PROMPTED ON INITIAL PROGRAM STARTUP. '$DEVM' CAN BE CHANGED ANY TIME BY TYPING 'CONTROL-G' AND 'CONTROL-C'. THE PROGAM WILL RE-SIZE AND RESTART AT THE BEGINNING AGAIN.

$DEVM DEFAULT = 000000



BIT 15: 0 = TEST CHANNEL 0 = DEFAULT
1 = BYPASS CHANNEL 0 TEST = 100000

BIT 13: 0 = TEST 1 MXV11B MODULE = DEFAULT
1 = TEST 2 MXV11B MODULES = 020000

BIT 12: 0 = CPU HAS MEMORY MANAGEMENT = DEFAULT
1 = CPU HAS NO MEM MNGT (LSI/2) = 10000

BIT 11: 0 = BREAK DETECTION DISABLED = DEFAULT
1 = BREAK DETECTION ENABLED = 4000
NOTE: BREAK DETECTION IS TESTED ON CHANNEL 0 ONLY

BIT 10: 0 = DO DATA WRAP AROUND TESTS = DEFAULT
1 = BYPASS DATA WRAP TESTS = 2000

BIT 9: 0 = DO DATA WRAP EXTERNAL TESTS = DEFAULT
1 = DO DATA WRAP INTERNAL TESTS = 1000
NOTE: THIS BIT SELECTION IS ONLY USED IF DATA
WRAP HAS ALREADY BEEN SELECTED VIA BIT 10 OR 3.

BIT 8: 0 = BYPASS CHANNEL 1 TEST = DEFAULT
1 = TEST CHANNEL 1 = 400

BIT 7: 0 = ENABLE PCR REGISTER TEST = DEFAULT
1 = BYPASS PCR REGISTER TEST = 200

BIT 6: 0 = ENABLE LEDS TEST DISPLAY = DEFAULT
1 = BYPASS USE OF LEDS = 100
NOTE: THE LED REGISTER IS THE ALSO KNOWN
AS THE DDR REGISTER.

BIT 4: 0 = BREAK DETECTION DISABLED = DEFAULT
1 = BREAK DETECTION ENABLED = 20
NOTE: BIT 4 AFFECTS CHANNEL 1 ONLY

BIT 3: 0 = BYPASS DATA WRAP TESTS = DEFAULT
1 = DO DATA WRAP AROUND TESTS = 10
NOTE: BIT 3 AFFECTS CHANNEL 1 ONLY

BIT 2: 0 = TEST RAM = DEFAULT
1 = BYPASS RAM TESTS = 4

BIT 1: 0 = BYPASS ROM TESTING = DEFAULT
1 = ROM PRESENT (TEST) = 2

BIT 0: 0 = CLOCK OPTION DISABLED = DEFAULT
1 = CLOCK OPTION ENABLED = 1

IMPORTANT:

1. IF RUNNING UNDER APT THE CONSOLE IS ON THE MXV11-B, THE CONSOLE MUST NOT BE TESTED SINCE APT SENDS 'BREAKS' TO THE CONSOLE WHICH INTERFERES WITH THE TESTS.

2. CPU'S WITHOUT MEMORY MANAGEMENT (LSI OR LSI/2) CANNOT TEST ALL OF THE MXV11-B'S MEMORY. THESE SYSTEMS CAN ONLY VERIFY/TEST THE LOWER 32K WORDS (ACTUALLY 32K WORDS MINUS THE I/O PAGE I.E. LOWER 28K). CPU'S WITH MEMORY MANAGEMENT CAN CHECK UP TO TWO MXV11-B MODULES MINUS THE I/O PAGE I.E. 124K WORDS.

SUMMARY OF USER LOCATIONS AND DEFAULTS.

	LOC	DEFAULT	
SDEVM	1252	0	DEVICE MAP
SWREG	176	0	SOFTWARE SWITCH REGISTER
BASE0	1254	776500	CHANNEL 0
VECT0	1256	300	CHANNEL 0
BASE1	1260	777560	CHANNEL 1
VECT1	1262	60	CHANNEL 1
LOROM	1274	773000	LOW ROM ADDRESS
HIROM	1276	773776	HIGH ROM ADDRESS (256 WORDS)
LOROM2	1300	765000	2ND LOW ROM ADDRESS
HIROM2	1302	765776	2ND HIGH ROM ADDRESS (256 WORDS)
BASE2	1264	776510	CHANNEL 2
VECT2	1266	310	CHANNEL 2
BASE3	1270	776520	CHANNEL 3
VECT3	1272	320	CHANNEL 3

2.5 EXECUTION TIMES.

EXECUTION TIMES FOR AN LSI-11 PROCESSOR WITH THE MXV11-B MODULE AT SHIPMENT CONFIGURATION:

CH. 0 AT 500 BAUD.
CH. 1 (CONSOLE) AT 9600 BAUD.

		LSI-11	F-11	
ARE:	FIRST PASS-	17 SEC	08 SEC	WITH 1 MXV11-B
	ADDITIONAL PASSES	45 SEC	23 SEC	WITH 1 MXV11-B

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

THE RAM TESTS REQUIRE THE ADDITIONAL TIMES SHOWN BELOW FOR ALL PASSES:

	LSI-11	F-11 (64KW)	F-11 (124KW)
1ST PASS	4 SEC	10 SEC	19 SEC
2ND PASS	16 SEC	10 SEC	33 SEC

2.6 POWER FAIL

AUTO START FROM POWER FAIL IS IMPLEMENTED IN THIS PROGRAM.
UPON POWER UP, THE PROGRAM WILL RESTART FROM THE BEGINNING.

3.0 ERROR INFORMATION

3.1 ERROR REPORTING PROCEDURE.

SINCE THIS DIAGNOSTIC WAS DESIGNED TO FIT INTO 16K 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

3.3 ERROR NUMBERS

THE ERROR NUMBERS HAVE BEEN UP DATED TO REFLECT THE TEST THEY ARE CALLED FROM. FOR EXAMPLE:

ERROR 3 = TEST 3, ERROR NUMBER WITHIN THE TEST IS 5

ERROR 237= TEST 23, ERROR NUMBER WITHIN THE TEST IS 7

NOTE: ALL TESTS CONFORM TO THIS FORMAT EXCEPT TEST 21. THAT TEST NUMBERS ITS TESTS FROM 1 TO 13 (OCTAL).

4.0 PERFORMANCE AND PROGRESS REPORTS.

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

AS EACH CHANNEL 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 AND CHANNELS TO BE TESTED HAVE BEEN EXERCISED,
AN END PASS STATEMENT IS TYPED:

'END PASS __.'

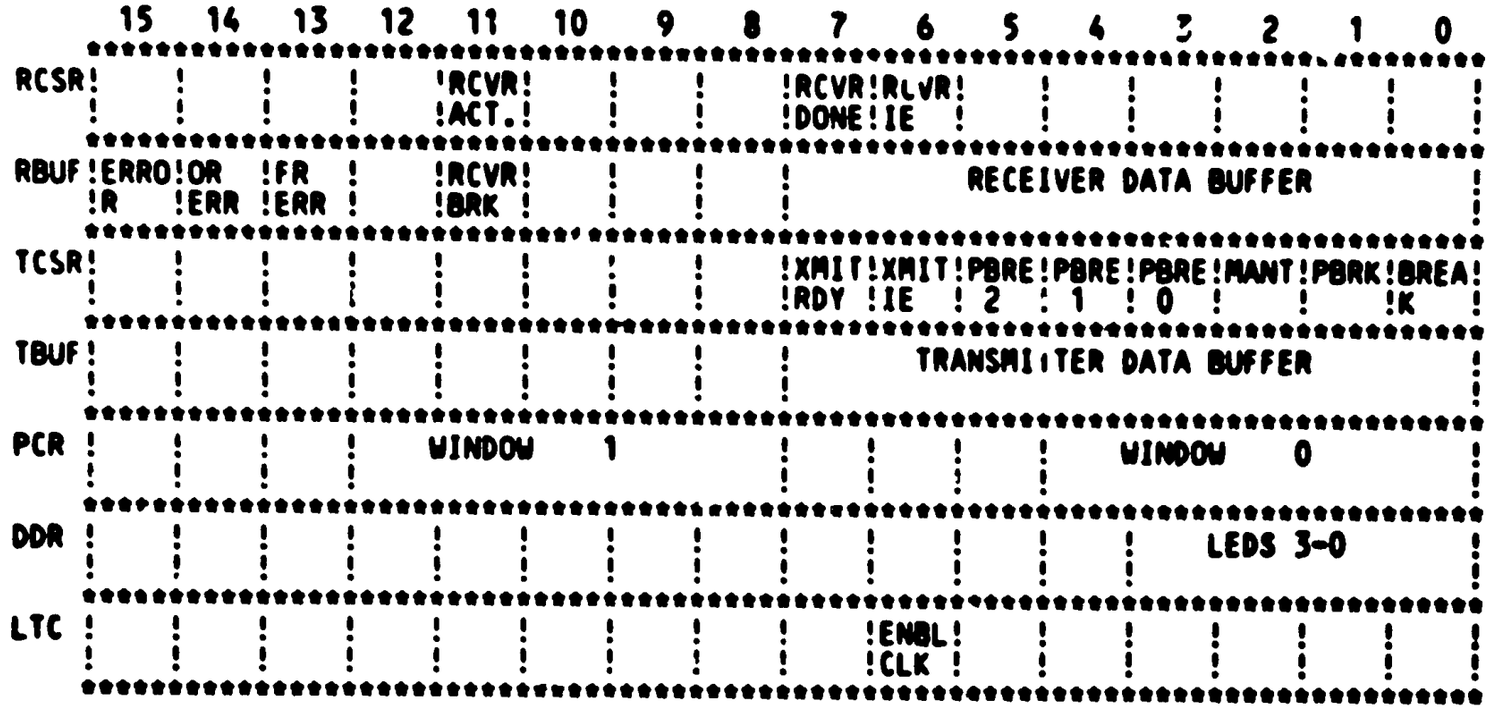
EXAMPLE OF PRINTOUT ASSUMING: SEPERATE CONSOLE DEVICE
64K RAM
CLOCK ENABLED
NO ERRORS

```
CVMXBA, MXV11B DIAGNOSTIC
SWR= 000000     NEW= 10000<CR>                  (ENABLE
                                                PERFORMANCE
                                                REPORTS)

DEVM = 000000   NEW = 201<CR>                   (8 BITS WORD
                                                [NO PARITY],
                                                CLOCK ENABLED.)


64K MEMORY
CLOCK ENABLED

        ** PHASE 1 SUMMARY **
CSR: 776500, VECTOR: 000760, ERRORS: 0          (CHANNEL 0)
CSR: 777500, VECTOR: 000770, ERRORS: 0          (CHANNEL 1)

        ** PHASE 2 SUMMARY **
CSR: 776500, VECTOR: 000760, ERRORS: 0          (INTERACTION TESTS
                                                STARTING
                                                WITH CH 0)

END PASS        1
```

NOTE: THE DEVICE MAP '$DEVM' CAN BE CHANGED AT ANY TIME BY TYPING
'CONTROL-G' AND 'CONTROL-C'. SEE SEC. 2.4.

THE PROGRAM WILL RE-SIZE AND RESTART AT THE BEGINNING AGAIN.

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

3. ORERR = OVERRUN ERROR
   FRERR = FRAMING ERROR

6.0 SUMMARY OF TESTS AND SPECIAL SUBROUTINES.

PHASE 1 TESTS

TEST 1 RAM ADDRESS TEST

THIS TEST WRITES THE ENTIRE FIRST 32K WITH THE ADDRESS OF THE
LOCATION. IT THEN CHECKS THE MEMORY TO BE SURE THE WRITE WAS
CORRECT. IF AN ERROR OCCURS, AND THE PROGRAM HAS BEEN SET UP
TO HALT, THE FAILING ADDRESS IS IN CPU REGISTER 0.

THIS TEST ALSO CHECKS FOR THE PRESENCE OF THE DDR REGISTER. IF
THE REGISTER "READ" CREATES A TIMEOUT AN ERROR WILL BE GENERATED.
TEST 1 HAS ERRORS 11-12

TEST 2 RAM DATA VOLITILITY TEST

WRITE ALL MEMORY TO BACKGROUND OF ALL 1'S.
TEST LOCATION FOR CORRECT BACKGROUND.
FLOAT 0'S COMPLEMENT THRU WORD. RESET LOCATION TO BACKGROUND.
REPEAT ABOVE 3 STEPS FOR EACH LOCATION.
WHEN ALL LOCATIONS TESTED, CHECK ENTIRE MEMORY FOR BACKGROUND PATTERN.
REPEAT ALL THE ABOVE FOR BACKGROUND PATTERN OF ALL 0'S FLOATING 1'S.
THIS TEST USES THE MEMORY MANAGEMENT OPTION. IF A FAILURE OCCURS
EXAMINE CPU REGISTER R2. THIS REGISTER WILL CONTAIN THE FAILING
4K (OCTAL) PAGE NUMBER.
TEST 2 HAS ERRORS 21-24

TEST 3 ROM TESTS

THIS IS A TWO PART TEST. THE FIRST PART CHECKS
ADRRESS X73000-X73776 FOR PRESENCE.
THE SECOND PART CHECKS X65000-X65776 FOR PRESENCE.

THIS TEST ALSO CHECKS FOR THE PRESENCE, IF SDEVM SELECTED,
OF THE PCR REGISTER. IF THE REGISTER IS PRESENT THEN IT
CHECKS FOR AN ALL ONES WRITE TO THE REGISTER. IT THEN FLOATS
A ONE THROUGH A FIELD OF ZEROS.
TEST 3 HAS ERRORS 31-37

TEST 4 CLOCK TESTS

THE CLOCK TEST, IF SELECTED VIA THE SDEVM, SIMPLY ENABLES
THE CLOCK AND CHECKS FOR AN INTERRUPT.
TEST 4 HAS ERRORS 41 AND 42

TEST 5 ADDRESSABILITY

THIS TEST VERIFIES THAT ALL 8 REGISTERS OF THE CHANNE

UNDER TEST RESPOND TO THEIR ADDRESSES.
TEST 5 HAS ERRORS 51-55

THE FOLLOWING 3 TESTS TEST ALL 'READ WRITE' BITS

TEST 6 BREAK - TCSR 0 SET, CLEAR,

TEST 6 HAS ERRORS 61-64

TEST 7 XMITIE - TCSR 6 SET, CLEAR,

TEST 7 HAS ERRORS 71-74

TEST 10 RCVRIE - RCSR 6 SET, CLEAR,

TEST 10 HAS ERRORS 101-105

TEST 11 XMIT RDY - TCSR 7 - CLEARS WHEN TBUF IS LOADED WITH A CHARACTER AND THAT IT SETS WITHIN A REASONABLE AMOUNT OF TIME.

TEST 11 HAS ERRORS 111-113

TEST 12 OUTPUTTING A CHAR FROM TBUF (WITH WRAP AROUND CONNECTED) RESULTS IN RCVRDONE SETTING WITHIN A REASONABLE AMOUNT OF TIME.

TEST 12 HAS ERRORS 121-122

TEST 3 RCVRDONE IS CLEARED BY READING RBUF

TEST 13 HAS ERRORS 131 AND 132

TEST 14 OVERRUN ERROR BIT - RBUF 14

TEST 14 HAS ERRORS 141-147

TEST 15 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 15 HAS ERRORS 151-154

TEST 16 RECEIVER INTERRUPT LOGIC TEST THIS TEST COVERS ALL
---- -- OF THE RECEIVER SIDE OF THE INTERRUPT LOGIC IN
CHARACTER MODE.

TEST 16 HAS ERRORS 161-164

TEST 17 TEST DATA WRAP AROUND BINARY COUNT: FLAG MODE.
---- --

TEST 17 HAS ERRORS 171-176

TEST 20 TEST DATA WRAP AROUND BINARY COUNT: INTERRUPT MODE.
---- --

TEST 20 HAS ERRORS 201-205

TEST 21 TEST BREAK 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 21 HAS ERRORS 1-13

TEST 22 NOT A TEST - SEND BACK TO LOOP

****** PHASE 2 TESTS *****

TEST 23 TEST THAT CHANNELS INTERRUPT AT ASSIGNED PRIORITY
---- --

TEST 23 HAS ERRORS 231-237

TEST 24 TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING.
---- --

TEST 24 HAS ERRORS 241-245

TEST 25 TEST THAT CHANNELS INTERRUPT AT ASSIGNED PRIORITY

THIS TEST IS EXECUTED ONLY ON THE SECOND MXV11-B MODULE.
CHANNELS 2 AND 3. ANY FAILURE IS RELATED TO THE SECOND
MXV11-B BOARD.
TEST 25 HAS ERRORS 251-256

TEST 26 TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING.

THIS TEST IS EXECUTED ONLY ON THE SECOND MXV11-B MODULE.
CHANNELS 2 AND 3. ANY FAILURE IS RELATED TO THE SECOND
MXV11-B BOARD.
TEST 26 HAS ERRORS 261-264

```
        .REM %
```

IDENTIFICATION

PRODUCT CODE: AC-T551A-MC

PRODUCT NAME: CVMXBA0 MXV11B DIAG

PRODUCT DATE: FEBRUARY 1983

MAINTAINER: E.S.D. METHODS

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

THIS DIAGNOSTIC IS A LOGIC TEST TO VERIFY THE OPERATION OF THE
2 SERIAL LINE UNITS, ROM & CLOCK OPTIONS, THE PCR REGISTER,
THE DDR REGISTER & RAM ON THE MXV11-B.

THE PROGRAM WILL TEST TO WHATEVER OPTIONS THE DEVICE MAP ($DEVM)
IS SET TO. SEE PROGRAM OPTIONS & DEFAULTS SEC 2.4 .
THE PROGRAM WILL PRINT THE CONTENTS OF $DEVM FOR OPERATOR VERIFICATION
& A SUMMARY OF SIGNIFICANT DIFFERENCES FROM THE DEFAULT CONDITIONS
I.E: CHANNEL(S) DROPPED FROM TESTING, CHANNEL 1 AS CONSOLE,
ROM/RAM TESTING BYPASSED & CLOCK ENABLED.

SERIAL LINE UNIT TESTING IS DONE IN TWO DISTINCT PHASES:

1. EACH OF THE 2 CHANNELS OF THE MXV11-B IS TESTED INDIVIDUALLY.

2. THE MXV11-B MODULE IS TESTED AS A WHOLE FOR CHANNEL INTERACTION
   PROBLEMS. THIS DIAGNOSTIC IS DESIGNED TO TEST AND DETECT
   ERRORS TO THE LOGIC LEVEL & NOT THE CHIP LEVEL.

THE OPERATOR MUST INSTALL DATA WRAP AROUND CONNECTORS TO DO
DATA TESTING. TO BYPASS DATA TESTS, THE OPERATOR MUST MODIFY $DEVM.
SEE PROGRAM OPTIONS SEC. 2.4

THE ADDRESSES & VECTOR RANGES ARE AS FOLLOWS:

```
CHANNEL 0: 776500 THRU 777570
CHANNEL 1: 776510 THRU 777600
           777560   ( AS CONSOLE )
VECTORS  :      4 THRU    376
PCR      :777520        DDR     :777524 LTC     :777546
```

THE DEFAULT ADDRESSES & VECTORS ARE AS FOLLOWS:

```
CHANNEL 0: 776500 THRU 776506   VECTORS: 300/304
CHANNEL 1: 777560 THRU 777566   VECTORS:  60/64
```

FOR ANY OTHER DEVICE ADDRESSES THE OPERATOR MUST CHANGE THE
DEFAULT LOCATIONS. SEE PROGRAM OPTIONS & DEFAULTS SEC. 2.4 .

THIS PROGRAM IS DESIGNED TO RUN ON ANY Q-BUS/Q22 PDP-11 WITH 64K OF
MEMORY AND AN MXV11-B (Q-BUS/Q22) MODULE. IT CAN RUN UNDER XXDP &
APT MONITORS, AND ON PROCESSORS WITH NO HARDWARE
SWITCH REGISTER.

1.2 SYSTEM REQUIREMENTS.

HARDWARE REQUIREMENTS:

KDF11 PROCESSOR
64K MEMORY - MINIMUM (CONTAINED ON THE MXV11-B)
A SPECIAL DATA WRAP AROUND CONNECTOR (PN # H3270-A) OR EQUIVALENT
(REQ'D IF EXTERNAL DATA WRAP AROUND TESTS DESIRED)

IF CHAN. 1 IS THE CONSOLE, TESTS 11-14, 16-21, 23-24 ARE BYPASSED.
(BYPASS IS FOR CHANNEL 1 ONLY)

IF DATA WRAP AROUND TESTS ARE BYPASSED, TESTS 12-14, 16-21, 23-24 ARE BYPASSED.

SOFTWARE REQUIREMENTS:

THIS DIAGNOSTIC CAN RUN IN 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 DIAGNOSTICS ARE REQUIRED TO RUN BEFORE THIS, BUT THE PROCESSOR, MEMORY, AND BUS ARE ASSUMED TO BE FULLY OPERATIONAL.

1.5 ASSUMPTIONS.

THE OPERATOR MUST:

1. SET THE SOFTWARE SWITCH REGISTER (SWR) IF NOT DEFAULTED.(SEC. 2.3)
2. SET THE DEVICE MAP ($DEVM) IF NOT DEFAULTED.(SEC. 2.4)
3. SET THE SERIAL LINE ADDRESSES & VECTORS IF NOT DEFAULTED.(SEC 2.4)
4. SET THE ROM HI & LO ADDR IF NOT DEFAULTED.(SEC 2.4)
5. SET BIT 15 ($DEVM) IF CHANNELS ON A SECOND MXV11-B ARE TO BE TESTED.

1.6 VT-100 CONSOLE SETUP.

IF CHANNEL 1 IS CONFIGURED AS THE CONSOLE & THE VT-100 IS THE CONSOLE DEVICE, "SETUP B" MUST BE SET IN FOR THE FOLLOWING.
A. DISABLE XON/XOFF
B. JUMP SCROLL ON
C. NEW LINE OFF
D. ALL OTHER OPTIONS PER SYSTEM REQUIREMENTS.

2.0 OPERATING INSTRUCTIONS.

2.1 LOADING AND STARTING PROCEDURES.

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

ALL NORMAL STARTS & RESTARTS ARE FROM LOCATION 200.

THERE ARE 2 STARTING ADDRESSES TO BE USED OFF LINE ONLY FOR INTERRUPT VECTOR TROUBLE SHOOTING:

1334 START: LOADS ADDRESSES 0 TO 1000 WITH THE ADDRESS OF AN INTERRUPT ROUTINE THAT JUST DOES RTI'S ALLOWING LOOPING IN THAT PART OF THE TEST WHERE INTERRUPT VECTOR PROBLEMS ARE OCCURING.
NORMAL TESTING WILL THEN BEGIN.

1356 START: LOADS ADDRESSES 0 TO 1000 WITH TRAP CATCHER CODE.
ANY INTERRUPT TO THIS REGION WILL HALT.
NORMAL TESTING WILL THEN BEGIN.

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.

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 OCTAL 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 2 (ERASING ALL PREVIOUS INPUT).

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

NOTE: BECAUSE OF THE FREQUENT BUS RESETS IN THE PROGRAM, MULTIPLE CONTROL-G'S MAY BE REQ'D. IF NECESSARY, 'BREAK' INTO THE PROGRAM & LOAD LOCATION 176 (SWREG) BY 'ODT' TO THE DESIRED CONTENTS. DO A 'P' TO PROCEED.

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 PREVIOUS TO THE SELECTED TEST
                      ARE EXECUTED FIRST WITH 1 ITERATION ONLY)
```

NOTE: IF BIT 14, 9 OR 8 IS SELECTED THE LEDS ON THE MXV11-B MODULE MAY GIVE FALSE TEST NUMBER INDICATIONS. IF A TEST HAS A UNIBUS RESET COMMAND THE LEDS WILL BE FULLY LIGHTED EVERY PASS THROUGH THE RESET INSTRUCTION. THUS, TEST 7 WILL APPEAR AS TEST 17 (ALL LEDS ON) ETC.

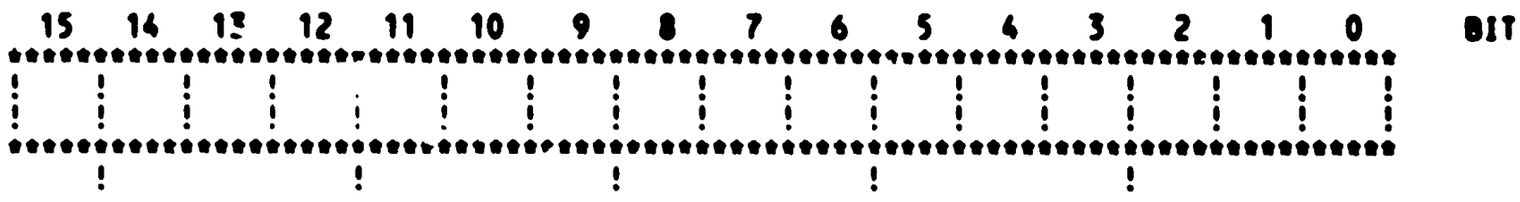
2.4 PROGRAM OPTIONS & DEFAULTS.

THIS PROGRAM REQUIRES THE ADDRESS OF THE FIRST RCSR OF EACH
SERIAL LINE UNIT & ITS INTERRUPT VECTORS TO BE PREVIOUSLY STORED
IF NOT DEFAULTED:

	REGISTER	LOCATION	DEFAULT	
CHANNEL 0	BASE0	1254	776500	RCSR
	VECT0	1256	300	VECTOR
CHANNEL 1	BASE1	1260	777560	RCSR
	VECT1	1262	60	VECTOR
SECOND MXV11-B				
CHANNEL 2	BASE2	1264	776510	RCSR
	VECT2	1266	310	VECTOR
CHANNEL 3	BASE3	1270	776520	RCSR
	VECT3	1272	320	VECTOR

LOCATION '$DEVM' IS USED AS A BIT MAP TO INDICATE WHICH OPTIONS
ARE PRESENT AND TO BE TESTED.
THE OPERATOR IS PROMPTED ON INITIAL PROGRAM STARTUP.
'$DEVM' CAN BE CHANGED ANY TIME BY TYPING 'CONTROL-G' AND 'CONTROL-C'.
THE PROGAM WILL RE-SIZE & RESTART AT THE BEGINNING AGAIN.

$DEVM DEFAULT = 000000



BIT 15: 0 = TEST CHANNEL 0 = DEFAULT
1 = BYPASS CHANNEL 0 TEST = 100000

BIT 13: 0 = TEST 1 MXV11B MODULE = DEFAULT
1 = TEST 2 MXV11B MODULES = 020000

BIT 12: 0 = CPU HAS MEMORY MANAGEMENT = DEFAULT
1 = CPU HAS NO MEM MNGT (LSI/2) = 10000

BIT 11: 0 = BREAK DETECTION DISABLED = DEFAULT
1 = BREAK DETECTION ENABLED = 4000
NOTE: BREAK DETECTION IS TESTED ON CHANNEL 0 ONLY

BIT 10: 0 = DO DATA WRAP AROUND TESTS = DEFAULT
1 = BYPASS DATA WRAP TESTS = 2000
NOTE: THIS BIT CONTROLS DATA WRAP ON CHANNEL
0 AND 2/3 IF SELECTED.

```
          BIT 9:  0 = DO DATA WRAP EXTERNAL TESTS = DEFAULT
                  1 = DO DATA WRAP INTERNAL TESTS =   1000
                  NOTE: THIS BIT SELECTION IS ONLY USED IF DATA
                        WRAP HAS ALREADY BEEN SELECTED VIA BIT 10 OR 3.


          BIT 8:  0 = BYPASS CHANNEL 1 TEST       = DEFAULT
                  1 = TEST CHANNEL 1              =    400

          BIT 7:  0 = ENABLE PCR REGISTER TEST    = DEFAULT
                  1 = BYPASS PCR REGISTER TEST    =    200

          BIT 6:  0 = ENABLE LEDS TEST # DISPLAY  = DEFAULT
                  1 = BYPASS USE OF LEDS          =    100
                  NOTE: THE LED REGISTER IS THE ALSO KNOWN
                        AS THE DDR REGISTER.

          BIT 4:  0 = BREAK DETECTION DISABLED    = DEFAULT
                  1 = BREAK DETECTION ENABLED     =     20
                  NOTE: BIT 4 AFFECTS CHANNEL 1 ONLY

          BIT 3:  0 = BYPASS DATA WRAP TESTS      = DEFAULT
                  1 = DO DATA WRAP AROUND TESTS   =     10
                  NOTE: BIT3 AFFECTS CHANNEL 1 ONLY

          BIT 2:  0 = TEST RAM                    = DEFAULT
                  1 = BYPASS RAM TESTS            =      4

          BIT 1:  0 = BYPASS ROM TESTING          = DEFAULT
                  1 = ROM PRESENT (TEST)          =      2

          BIT 0:  0 = CLOCK OPTION DISABLED       = DEFAULT
                  1 = CLOCK OPTION ENABLED        =      1

          IMPORTANT:

          1. IF RUNNING UNDER APT & THE CONSOLE IS ON THE MXV11-B,
             THE CONSOLE MUST NOT BE TESTED SINCE APT SENDS 'BREAKS'
             TO THE CONSOLE WHICH INTERFERES WITH THE TESTS.

          2. CPU'S WITHOUT MEMORY MANAGEMENT (LSI OR LSI/2) CANNOT
             TEST ALL OF THE MXV11-B'S MEMORY. THESE SYSTEMS CAN ONLY
             VERIFY/TEST THE LOWER 32K WORDS (ACTUALLY 32K WORDS MINUS
             THE I/O PAGE I.E. LOWER 28K). CPU'S WITH MEMORY MANAGEMENT
             CAN CHECK UP TO TWO MXV11-B MODULES MINUS THE I/O PAGE
             I.E. 124K WORDS.

          SUMMARY OF USER LOCATIONS & DEFAULTS.

                          LOC     DEFAULT
                          -----   ---------
          $DEVM           1252          0       DEVICE MAP
          SWREG            176          0       SOFTWARE SWITCH REGISTER

          BASE0           1254     776500       CHANNEL 0
          VECT0           1256        300       CHANNEL 0
```

BASE1	1260	777560	CHANNEL 1
VECT1	1262	60	CHANNEL 1
LOROM	1274	773000	LOW ROM ADDRESS
HIROM	1276	773776	HIGH ROM ADDRESS (256 WORDS)
LOROM2	1300	765000	2ND LOW ROM ADDRESS
HIROM2	1302	765776	2ND HIGH ROM ADDRESS (256 WORDS)
BASE2	1264	776510	CHANNEL 2
VECT2	1266	310	CHANNEL 2
BASE3	1270	776520	CHANNEL 3
VECT3	1272	320	CHANNEL 3

2.5 EXECUTION TIMES.

EXECUTION TIMES FOR AN LSI-11 PROCESSOR WITH THE MXV11-B MODULE AT SHIPMENT CONFIGURATION:

CH. 0 AT 300 BAUD.
CH. 1 (CONSOLE) AT 9600 BAUD.

		LSI-11	F-11	
ARE:	FIRST PASS-	17 SEC	08 SEC	WITH 1 MXV11-B
	ADDITIONAL PASSES	45 SEC	23 SEC	WITH 1 MXV11-B

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

THE RAM TESTS REQUIRE THE ADDITIONAL TIMES SHOWN BELOW FOR ALL PASSES:

	LSI-11	F-11 (64KW)	F-11 (124KW)
1ST PASS	4 SEC	10 SEC	19 SEC
2ND PASS	16 SEC	10 SEC	33 SEC

2.6 POWER FAIL.

AUTO START FROM POWER FAIL IS IMPLEMENTED IN THIS PROGRAM.
UPON POWER UP, THE PROGRAM WILL RESTART FROM THE BEGINNING.

3.0 ERROR INFORMATION.

3.1 ERROR REPORTING PROCEDURE.

SINCE THIS DIAGNOSTIC WAS DESIGNED TO FIT IN 16K 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.

3.3 ERROR NUMBERS

THE ERROR NUMBER HAVE BEEN UP DATED TO REFLECT THE TEST THEY ARE CALLED FROM. FOR EXAMPLE:

ERROR # 35 = TEST 3, ERROR NUMBER WITHIN THE TEST IS 5

ERROR # 237= TEST 23, ERROR NUMBER WITHIN THE TEST IS 7

NOTE: ALL TEST CONFORM TO THIS FORMAT EXCEPT TEST 21. THAT TEST NUMBERS ITS TESTS FROM 1 TO 13 (OCTAL).

4.0 PERFORMANCE AND PROGRESS REPORTS.

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

AS EACH CHANNEL 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: SEPERATE CONSOLE DEVICE
64K RAM
CLOCK ENABLED
NO ERRORS

```
CVMXBA0 MXV11B DIAGNOSTIC

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

DEVM = 000000   NEW = 201<CR>               (CH.1 =  8 BITS/WORD (NO PARITY), CLOCK ENABLED)

64K MEMORY
CLOCK ENABLED

        ** PHASE 1 SUMMARY **
CSR: 776500, VECTOR: 000760, ERRORS: 0      (CHANNEL 0)
CSR: 777500, VECTOR: 000770, ERRORS: 0      (CHANNEL 1)

        ** PHASE 2 SUMMARY **
CSR: 776500, VECTOR: 000760, ERRORS: 0      (INTERACTION TESTS STARTING
                                            WITH CH 0)

END PASS #      1
```

NOTE: THE DEVICE MAP 'SDEVM' CAN BE CHANGED AT ANY TIME BY TYPING
'CONTROL-G' & 'CONTROL-C'. SEE SEC. 2.4.
THE PROGRAM WILL RE-SIZE & RESTART AT THE BEGINNING AGAIN.

5.0 DEVICE REGISTERS.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCSR					RCVR ACT.				RCVR DONE	RLVR IE						
RBUF	ERRO R	OR ERR	FR ERR		RCVR BRK				RECEIVER DATA BUFFER							
TCSR									XMIT RDY	XMIT IE	PBRE 2	PBRE 1	PBRE 0	MANT	PBRK	BREAK
TBUF									TRANSMITTER DATA BUFFER							
PCR					WINDOW #1								WINDOW #0			
DDR													LEDS 3-0			
L ^										ENBL CLK						

NOTES:

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

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

3. ORERR = OVERRUN ERROR
   FRERR = FRAMING ERROR

6.0 SUMMARY OF TESTS AND SPECIAL SUBROUTINES.

PHASE 1 TESTS

TEST 1 RAM ADDRESS TEST

THIS TEST WRITES THE ENTIRE FIRST 32K WITH THE ADDRESS OF THE
LOCATION. IT THEN CHECKS THE MEMORY TO BE SURE THE WRITE WAS
CORRECT. IF AN ERROR OCCURS, AND THE PROGRAM HAS BEEN SET UP
TO HALT, THE FAILING ADDRESS IS IN CPU REGISTER 0.

THIS TEST ALSO CHECKS FOR THE PRESENCE OF THE DDR REGISTER. IF
THE REGISTER "READ" CREATES A TIMEOUT AN ERROR WILL BE GENERATED.
TEST 1 HAS ERRORS 11 THROUGH 12

TEST 2 RAM DATA & VOLITILITY TEST

WRITE ALL MEMORY TO BACKGROUND OF ALL 1'S.
TEST LOCATION FOR CORRECT BACKGROUND.
FLOAT 0'S & COMPLEMENT THRU WORD. RESET LOCATION TO BACKGROUND.
REPEAT ABOVE 3 STEPS FOR EACH LOCATION.
WHEN ALL LOCATIONS TESTED, CHECK ENTIRE MEMORY FOR BACKGROUND PATTERN.
REPEAT ALL THE ABOVE FOR BACKGROUND PATTERN OF ALL 0'S & FLOATING 1'S.
THIS TEST USES THE MEMORY MANAGEMENT OPTION. IF A FAILURE OCCURS
EXAMINE CPU REGISTER R2. THIS REGISTER WILL CONTAIN THE FAILING
4K (OCTAL) PAGE NUMBER.
TEST 2 HAS ERRORS 21 THROUGH 24

TEST 3 ROM TESTS

THIS IS A TWO PART TEST. THE FIRST PART CHECKS
ADRRESS X73000-X73776 FOR PRESENCE. THE SECOND
PART CHECKS X65000-X65776 FOR PRESENCE.

THIS TEST ALSO CHECKS FOR THE PRESENCE, IF $DEVM SELECTED,
OF THE PCR REGISTER. IF THE REGISTER IS PRESENT THEN IT
CHECKS FOR AN ALL ONES WRITE TO THE REGISTER. IT THEN FLOATS
A ONE THROUGH A FIELD OF ZEROS.
TEST 3 HAS ERRORS 31 THROUGH 37

TEST 4 CLOCK TESTS

THE CLOCK TEST, IF SELECTED VIA THE $DEVM, SIMPLY ENABLES
THE CLOCK AND CHECKS FOR AN INTERRUPT.
TEST 4 HAS ERRORS 41 THROUGH 42

TEST 5 ADDRESSABILITY

THIS TEST VERIFIES THAT ALL 8 REGISTERS OF THE CHANNEL
UNDER TEST RESPOND TO THEIR ADDRESSES.
TEST 5 HAS ERRORS 51 THROUGH 55

THE FOLLOWING 3 TESTS TEST ALL 'READ WRITE' BITS

TEST 6 BREAK - TCSR 0 SET, CLEAR

TEST 6 HAS ERRORS 61 THROUGH 64

TEST 7 XMITIE - TCSR 6 SET, CLEAR

TEST 7 HAS ERRORS 71 THROUGH 74

TEST 10 RCVRIE - RCSR 6 SET, CLEAR

TEST 10 HAS ERRORS 101 THROUGH 105

TEST 11 XMIT RDY - TCSR 7 - CLEARS WHEN TBUF IS LOADED
    WITH A CHARACTER AND THAT IT SETS WITHIN A
    REASONABLE AMOUNT OF TIME.

TEST 11 HAS ERRORS 111 THROUGH 113

TEST 12 OUTPUTTING A CHAR FROM TBUF (WITH WRAP AROUND CONNECTED)
    RESULTS IN RCVRDONE SETTING WITHIN A
    REASONABLE AMOUNT OF TIME.

TEST 12 HAS ERRORS 121 THROUGH 122

TEST 13 RCVRDONE IS CLEARED BY READING RBUF

TEST 13 HAS ERRORS 131 THROUGH 132

TEST 14 OVERRUN & ERROR BIT - RBUF 14

TEST 14 HAS ERRORS 141 THROUGH 147

TEST 15 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 15 HAS ERRORS 151 THROUGH 154

TEST 16 RECEIVER INTERRUPT LOGIC TEST THIS TEST COVERS ALL
    OF THE RECEIVER SIDE OF THE INTERRUPT LOGIC IN
    CHARACTER MODE.

TEST 16 HAS ERRORS 161 THROUGH 164

TEST 17 TEST DATA WRAP AROUND BINARY COUNT: FLAG MODE.

FLAG MODE DENOTES THAT THE PROGRAM IS WAITING (CHECKING) FOR THE DONE BIT TO BE SET IN THE CSR REGISTER. THUS, IT CAN BE CONSIDERED NON-INTERRUPT MODE.

TEST 17 HAS ERRORS 171 THROUGH 176

TEST 20 TEST DATA WRAP AROUND BINARY COUNT: INTERRUPT MODE.

INTERRUPT MODE DENOTES THAT THE PROGRAM HAS SET THE "IE" (INTERRUPT ENABLE) BIT IN THE CSR REGISTER. THE PROGRAM ALSO MUST HAVE AN ADDRESS AND NEW PSW VALUE PLACED AT THE INTERRUPT VECTOR LOCATIONS. AT THE SPECIFIED ADDRESS THE PROGRAM MUST ALSO HAVE AN "RTI" INSTRUCTION AT THE END OF THE INTERRUPT ROUTINE. INTERRUPT MODE CAN ALSO BE CONSIDERED AS NON-FLAG MODE.

TEST 20 HAS ERRORS 201 THROUGH 205

TEST 21 TEST BREAK 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 21 HAS ERRORS 1 THROUGH 13

TEST 22 NOT A TEST - SEND BACK TO LOOP

****** PHASE 2 TESTS *****

TEST 23 TEST THAT CHANNELS INTERRUPT AT ASSIGNED PRIORITY

TEST 23 HAS ERRORS 231 THROUGH 237

TEST 24 TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING.

TEST 24 HAS ERRORS 241 THROUGH 2..

TEST 25 TEST THAT CHANNELS INTERRUPT AT ASSIGNED PRIORITY

THIS TEST IS EXECUTED ONLY ON THE SECOND MXV11-B MODULE. CHANNELS 2 AND 3. ANY FAILURE IS RELATED TO THE SECOND MXV11-B BOARD.

TEST 25 HAS ERRORS 251 THROUGH 256

TEST 26 TEST DATA TRANSFERS WITH ALL ACTIVE LINES INTERRUPTING.

THIS TEST IS EXECUTED ONLY ON THE SECOND MXV11-B MODULE.
CHANNELS 2 AND 3. ANY FAILURE IS RELATED TO THE SECOND
MXV11-B BOARD.

TEST 26 HAS ERRORS 261 THROUGH 264

%

```
                        .TITLE  CVMXBA0 MXV11B DIAG
                        ;*COPYRIGHT (C) 1982
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DICE SYSTEMS, INC.
                        ;*
                        ;*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
104000                  ERROR=EMT       ;;BASIC DEFINITION OF ERROR CALL
000004                  SCOPE=IOT       ;;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
177776                  PSW=PS
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE 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
001000                  SW9=SW09
000400                  SW8=SW08
000200                  SW7=SW07
000100                  SW6=SW06
000040                  SW5=SW05
000020                  SW4=SW04
000010                  SW3=SW03
000004                  SW2=SW02
000002                  SW1=SW01
000001                  SW0=SW00
                        ;*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
001000                  BIT9=BIT09
000400                  BIT8=BIT08
000200                  BIT7=BIT07
000100                  BIT6=BIT06
000040                  BIT5=BIT05
000020                  BIT4=BIT04
000010                  BIT3=BIT03
000004                  BIT2=BIT02
000002                  BIT1=BIT01
000001                  BIT0=BIT00
                        ;*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
    828
    830         177777          SET= -1                 ;; THE FOLLOWING DEFINITIONS APPLY TO THE GLOBAL SUBS
    831         000000          CLR=  0
    832         000001          TRUE= 1
    833
    834                         ; RCSR REGISTER BIT NAMES
    835
    836         000200          DONE=       BIT07       ; RECEIVER DONE
    837         000100          IE=         BIT06       ; RECEIVER INTERRUPT ENABLE
    838
    839                         ; RBUF REGISTER BIT NAMES
    840
    841         100000          ERR15=      BIT15       ; ERROR INDICATOR
    842         040000          ORERR=      BIT14       ; OVERRUN ERROR
    843         020000          FRERR=      BIT13       ; FRAMING ERROR
    844         010000          PERR=       BIT12       ; PARITY ERROR
    845
    846                         ; TCSR REGISTER BIT NAMES
    847
    848         000200          RDY=        BIT07       ; TRANSMITTER READY
    849         000100          IE=         BIT06       ; TRANSMITTER INTERRUPT ENABLE
    850         000001          BREAK=      BIT00       ; SEND BREAK (CONTINUOUS SPACE)
    851
    853                         ;MEMORY MENAGEMENT DEFINITIONS
    855
    856         172340                  PAR0=   172340
    857         172342                  PAR1=   172342
    858         172344                  PAR2=   172344
    859         172346                  PAR3=   172346
    860         172350                  PAR4=   172350
    861         172352                  PAR5=   172352
    862         172354                  PAR6=   172354
    863         172356                  PAR7=   172356
    866         172300                  PDR0=   172300
    867         172302                  PDR1=   172302
    868         172304                  PDR2=   172304
    869         172306                  PDR3=   172306
    870         172310                  PDR4=   172310
    871         172312                  PDR5=   172312
    872         172314                  PDR6=   172314
    873         172316                  PDR7=   172316
    874         177572                  SR0=    177572
    875         172516                  SR3=    172516
```

```
                                ;;*****************************************************************
                                .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  001414              JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

                                .SBTTL  ACT11 HOOKS
                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
            000204                      $SVPC=.                 ;SAVE PC
            000046                      .=46
000046      024752                      $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
            000052                      .=52
000052      000000                      .WORD   0               ;;2)SET LOC.52 TO ZERO
            000204                      .=$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      000202              $TSTM:  .WORD   130.    ;;RUN TIM OF LONGEST TEST
001006      000214              $PASTM: .WORD   140.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010      000175              $UNITM: .WORD   125.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012      000030                      .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
888                                     .SBTTL  COMMON TAGS
                                        ;;*****************************************************************
                                        ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                        ;*USED IN THE PROGRAM.
                 001100                         .=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     207     377     377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001167     000
001170     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172     012     000          $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
001212  000000                  $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001214                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
001214     000                  $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
```

```
001215     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001216  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001220  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001222  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001224     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. B  E
001225     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOL R=002
                                ;*                              500 NSEC MOS=003
001226  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001230     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001231     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001232  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001234     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001235     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001236  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001240     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001241     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001242  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001244  000000                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001246  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001250  000000                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001252  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001254                          $ETEND:
                                .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:
889
890 001254  176500              BASE0:  176500                  ;DEFAULT CH 0 BASE ADDR
891 001256  000300              VECT0:  300                     ;DEFAULT CH 0 VECTOR
892 001260  177560              BASE1:  177560                  ;DEFAULT CH 1 BASE ADDR (CONSOLE)
893 001262  000060              VECT1:  60                      ;DEFAULT CH 1 VECTOR
894 001264  176510              BASE2:  176510                  ;DEFAULT CH 2 BASE ADDR
895 001266  000310              VECT2:  310                     ;DEFAULT CH 2 VECTOR
896 001270  176520              BASE3:  176520                  ;DEFAULT CH 3 BASE ADDR
897 001272  000320              VECT3:  320                     ;DEFAULT CH 3 VECTOR
898 001274  173000              LOROM:  173000                  ;DEFAULT LO ROM ADDR
899 001276  173776              HIROM:  173776                  ;DEFAULT HI ROM ADDR (256 WORDS)
900 001300  165000              LOROM2: 165000                  ;DEFAULT LO ROM ADDR SECOND ROM
901 001302  165776              HIROM2: 165776                  ;DEFAULT HI ROM ADDR 2ND ROM (256 WORDS)
902 001304  177524              LEDREG: 177524                  ;ADDRESS REGISTER FOR LEDS
903                                                             ;OPTION 1: 4K   000000 - 017776
904                                                             ;OPTION 2: 4K   020000 - 037776
905
906 001306  177546              LKSREG: 177546                  ;LINE CLOCK REGISTER ADDRESS
907 001310  177520              PCRREG: 177520                  ;PCR REGISTER CONSTANT
908 001312  000000              DLADD:  0                       ;CH ADDR UNDER TEST...LOADED BY 'CYCLE'
909 001314  000000              RCSR:   0                       ;CH REGS UNDER TEST...LOADED BY 'LOOP'
910 001316  000000              RBUF:   0
911 001320  000000              TCSR:   0
912 001322  000000              TBUF:   0
913 001324  000000              DLVEC:  0                       ;CH RECVR VECTOR UNDER TEST...LOADED BY 'CYCLE'
914         000100              CLKVEC= 100                     ;LINE CLOCK INTR VECTOR
915 001326  000000              TMP1:   0                       ;TEMPORARY REGS
916 001330  000000              TMP2:   0
917 001332  000000              LEDHLD: .WORD                   ;THIS HOLD VALUE IN LED
918 001334      015     000     CARR:   .ASCIZ  <15>
919                             .EVEN
920
921                             ;;**************************************************************
922                             ;TRAP CATCHER FOR OFF LINE USE ONLY. BEGIN PROGRAM WITH FRESH LOAD
923                             ;;**************************************************************
924
925 001336  005000              TRPCAT: CLR R0
926 001340  012720  021522      1$:     MOV     #INTSRV,(R0)+
927 001344  012720  000340              MOV     #PR7,(R0)+
928 001350  020027  001000              CMP     R0,#1000
929 001354  001371                      BNE     1$
930 001356  000412                      BR      3$
931
932 001360  005000                      CLR     R0
933 001362  012701  000002              MOV     #2,R1
934 001366  010120              2$:     MOV     R1,(R0)+
```

```
    935 001370  005020                          CLR     (R0)+
    936 001372  062701  000004                  ADD     #4,R1
    937 001376  020027  001000                  CMP     R0,#1000
    938 001402  001371                          BNE     2$
    939 001404  005067  176566          3$:     CLR     176
    940 001410  000167  000000                  JMP     START
    941
    943 001414  000005                  START:  RESET
    944 001416  012700  177777                  MOV     #-1, R0
    945 001422  005300                  3$:     DEC     R0
    946 001424  001376                          BNE     3$
    947                                 .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
        001426  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
        001432  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
        001434  022706  001140                  CMP     #$WR,R6 ;;DONE?
        001440  001374                          BNE     .-6             ;;LOOP BACK IF NO
        001442  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
        001446  012737  026762  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
        001454  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
        001462  012737  026562  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
        001470  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
        001476  012737  027714  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
        001504  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
        001512  012737  025006  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
        001520  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
        001526  016767  023166  023156          MOV     $ENDCT,$LOOPCT  ;;SETUP END-OF-PROGRAM COUNTER
        001534  005067  177420                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
        001540  005067  177416                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
        001544  112767  000001  177343          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
        001552  012767  001552  177326          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
        001560  012767  001560  177322          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
        001566  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
        001572  012737  001626  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
        001600  012767  177570  177332          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
        001606  012767  177570  177326          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
        001614  022777  177777  177316          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
        001622  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
        001624  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
        001626  012716  001634          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
        001632  000002                          RTI
        001634  012767  000176  177276  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
        001642  012767  000174  177272          MOV     #DISPREG,DISPLAY
        001650  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
        001654  005067  177322                  CLR     $PASS           ;;CLEAR PASS COUNT
        001660  132767  000200  177327          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
        001666  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
        001670  012767  001216  177242          MOV     #$SWREG SWR     ;;NO,USE APT SWITCH REGISTER
        001676                          67$:
    948                                 .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
        001676  005227  177777                  INC     #-1             ;;FIRST TIME?
        001702  001045                          BNE     68$             ;;BRANCH IF NO
```

```
        001704  022737  024752  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
        001712  001441                          BEQ     68$             ;;BRANCH IF YES
        001714  104401  001762                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
        001720  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
        001724  001012                          BNE     70$             ;;BRANCH IF YES
        001726  126727  177262  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
        001734  001406                          BEQ     70$             ;;BRANCH IF YES
        001736  026727  177176  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
        001744  001005                          BNE     71$             ;;BRANCH IF NO
        001746  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
        001750  000403                          BR      71$
        001752  112767  000001  177154  70$:   MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
        001760                          71$:
        001760  000416                          BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <CRLF>*CVMXBA0 MXV11B DIAGNOSTIC*<CRLF>
        002016                          68$:
    949 002016  012767  177524  177260          MOV     #177524,LEDREG          ;SET UP LED REGISTER
    950 002024  005227  177777                  INC     #-1                     ;1'ST TIME?
    951 002030  001014                          BNE     1$                      ;BR IF NO
    952 002032  126727  177156  000001          CMPB    $ENV,#1                 ;APT?
    953 002040  001410                          BEQ     1$                      ;BR IF YES
    954 002042  005737  000042                  TST     @#42                    ;CHECK FOR CHAIN MODE
    955 002046  001404                          BEQ     2$                      ;0=NOT CHAINED NON-ZERO= CHAINED
    956 002050  052767  000004  177174          BIS     #BIT2,$DEVM             ;SET UP VALUE FOR TEST
    957 002056  000401                          BR      1$                      ;SKIP OVER QUESTION
    958 002060  104412                  2$:     GTDEVM                          ;ELSE SHOW CURRENT DEVM & GET NEW
    959
    960 002062  004767  015726          1$:     JSR     PC,SIZE                 ;SIZE THE MXV11-B
    961
    962 002066  012706  001100          RAMROM: MOV     #STACK,SP               ;RESET STACK PTR
    963
    964
    965
    976                                 ;;*******************************************************************
                                        ;*TEST 1        RAM ADDRESS TEST
                                        ;*
                                        ;*      WRITE ENTIRE MEMORY WITH ADDRESS AS DATA.
                                        ;*      READ ENTIRE MEMORY TO VERIFY.
                                        ;*
                                        ;;*******************************************************************
        002072  000004                  TST1:   SCOPE
        002074  012767  000001  177056          MOV     #1,$TIMES       ;;DO 1 ITERATION
        002102  012767  000001  177070          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
    981 002110  000240                          NOP                     ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
    982
    983 002112  032767  000004  177132          BIT     #BIT2,$DEVM     ;BYPASS RAM TESTS?
    984 002120  001022                          BNE     5$              ;BR, IF NO RAM TESTS
    985
    986 002122  012700  037776                  MOV     #37776,R0       ;ADDRESS REG
    987
    988 002126  010010                  1$:     MOV     R0,(R0)         ;LOAD ADDRESS AS DATA
    989 002130  020027  137776                  CMP     R0,#137776      ;LAST ADRR (LEAVE ROOM FOR XXDP)
    990 002134  001402                          BEQ     2$              ;BR IF YES
    991 002136  005720                          TST     (R0)+           ;ELSE BUMP ADDR REG
    992 002140  000772                          BR      1$
    993
```

```
 002142  012700  037776          2$:   MOV    #37776,R0        ;VERIFY
 002146  020010                  3$:   CMP    R0,(R0)          ;READ & CHECK
 002150  001401                        BEQ    4$
 002152                                ERROR  11               ;DATA NOT = ADDRESS
 002152  104011                        EMT 11

 002154  020027  137776          4$:   CMP    R0,#137776       ;LAST ADRR (LEAVE ROOM FOR XXDP)
 002160  001402                        BEQ    5$               ;BR, IF ALL DONE RAM TESTS
 002162  005720                        TST    (R0)+            ;ELSE BUMP ADDR
 002164  000770                        BR     3$
 002166  032767  000100  177056  5$:   BIT    #BIT6,$DEVM      ;SHOULD WE TEST/USE THE LEDS
 002174  001023                        BNE    TST2             ;BR, IF WE SHOULDN'T
 002176  005067  017326                CLR    INTFLG           ;RESET INT FLAG
 002202  005777  177076                TST    @LEDREG          ;ARE LEDS THERE
 002206  005767  017316                TST    INTFLG           ;DID WE GET A TRAP
 002212  001012                        BNE    6$               ;BR, IF THEY WEREN'T THERE

 002214  016700  176662                MOV    $TSTNM,R0        ;GET TEST NUMBER
 002220  005100                        COM    R0               ;1'S COMP TEST NUMBER
 002222  032767  000100  177022        BIT    #BIT6,$DEVM      ;CAN THE LEDS BE USED
 002230  001002                        BNE    501$             ;BR, IF THEY CAN'T
 002232  110077  177046                MOVB   R0,@LEDREG       ;DISPLAY TEST NUMBER IN LEDS
 002236                          501$:
 002236  000402                        BR     TST2             ;ALL DONE
 002240                          6$:   ERROR  12               ;LEDS NOT THERE
 002240  104012                        EMT 12
 002242  000400                        BR     TST2             ;GO ON
```

```
 1020
 1037                                   ;;*****************************************************************
                                        ;*TEST 2        RAM DATA & VOLITILITY TEST
                                        ;*
                                        ;* WRITE ALL MEMORY TO BACKGROUND OF ALL 1'S.
                                        ;* TEST LOCATION FOR CORRECT BACKGROUND.
                                        ;* FLOAT 0'S & COMPLEMENT THRU WORD.
                                        ;* RESET LOCATION TO BACKGROUND.
                                        ;* REPEAT ABOVE 3 STEPS FOR EACH LOCATION.
                                        ;* WHEN ALL LOCATIONS TESTED, CHECK ENTIRE MEMORY FOR BACKGROUND PATTERN.
                                        ;* CHECK VOLITILITY BY WAITING MORE THAN 5 SEC & RECHECKING BACKGROUND PATT.
                                        ;* REPEAT ALL THE ABOVE FOR BACKGROUND PATTERN OF ALL 0'S & FLOATING 1'S.
                                        ;*
                                        ;;*****************************************************************
      002244  000004                    TST2:   SCOPE
      002246  012767  000001  176704            MOV     #1,$TIMES       ;;DO 1 ITERATION
      002254  012767  000002  176716            MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 1042 002262  000240                            NOP                     ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 1043
 1044 002264  016700  176612                    MOV     $TSTNM,R0       ;GET TEST NUMBER
 1045 002270  005100                            COM     R0              ;1'S COMP TEST NUMBER
 1046 002272  032767  000100  176752            BIT     #BIT6,$DEVM     ;CAN THE LEDS BE USED
 1047 002300  001002                            BNE     50$             ;BR, IF THEY CAN'T
 1048 002302  110077  176776                    MOVB    R0,@LEDREG      ;DISPLAY TEST NUMBER IN LEDS
 1049 002306                            50$:
 1050 002306  032767  010000  176736            BIT     #BIT12,$DEVM    ;BYPASS MEM MNGT CODE
 1051 002314  001004                            BNE     51$             ;BR, IF BY PASS IS SELECTED
 1052 002316  032767  000004  176726            BIT     #BIT2,$DEVM     ;BYPASS RAM TESTS?
 1053 002324  001402                            BEQ     TEST5A          ;BR, IF TESTS ARE TO BE RUN
 1054 002326  000167  000456            51$:    JMP     TST3            ;JMP IF RAM TESTS ARE NOT TO BE RUN
 1055
 1056                                           .SBTTL  EXTENDED MEMORY TEST
 1057
 1058 002332                            TEST5A:
 1059 002332  012701  172340                    MOV     #PAR0,R1        ;START OF PAR REGISTERS
 1060 002336  012705  172300                    MOV     #PDR0,R5        ;START OF THE PDR REGISTERS
 1061 002342  005000                            CLR     R0              ;FIRST PAGE BASE ADDRESS
 1062 002344  010021                    20$:    MOV     R0,(R1)+        ;SET BASE FOR NEXT MAP
 1063 002346  012725  077406                    MOV     #77406,(R5)+    ;4K READ/WRITE EACH PAGE
 1064 002352  062700  000200                    ADD     #200,R0         ;BASE FOR THE NEXT PAGE
 1065 002356  020027  001600                    CMP     R0,#1600        ;DONE ALL PAGES ?
 1066 002362  003770                            BLE     20$             ;SET UP ALL MEMORY MANAGEMENT PAGES
 1067 002364  012741  007600                    MOV     #7600,-(R1)     ;SET UP I/O PAGE
 1068 002370  052737  000001  177572            BIS     #1,@#SR0        ;ENABLE MEMORY MANAGEMENT
 1069 002376  012704  000200                    MOV     #200,R4         ;4K INC VALUE FOR PAR6
 1070 002402  012737  002434  000004            MOV     #16$,@#4        ;LOAD UP VECTOR AREA
 1071 002410  012737  000340  000006            MOV     #340,@#6        ;PSW = NO INTERRUPTS
 1072 002416  010437  172354            15$:    MOV     R4,@#PAR6       ;START CHECKING FOR TOP
 1073 002422  005737  140000                    TST     @#140000        ;LOOK AT MEMORY
 1074 002426  062704  000200                    ADD     #200,R4         ;NEXT 4K VALUE
 1075 002432  000771                            BR      15$             ;LOOP UNTIL TOP OF MEMORY
 1076 002434                            16$:
 1077 002434  012737  000400  172354            MOV     #400,@#PAR6     ;START MEMORY PAGE aaaaa
 1078 002442  005005                            CLR     R5              ;INITIAL VALUE
 1079 002444  012701  140000            17$:    MOV     #140000,R1      ;START AT  LOC 0, RELATIVE TO PAR6
 1080 002450  010521                    10$:    MOV     R5,(R1)+        ;STORE 0 MEMORY
 1081 002452  020127  157776                    CMP     R1,#157776      ;END OF MEMORY PAGE YET ?
```

```
1082 002456 101774                          BLOS    10$             ;LOOP TILL WHOLE PAGE WRITTEN
1083 002460 062737 000200 172354            ADD     #200,@#PAR6     ;MAP INTO NEXT PAGE
1084 002466 023704 172354                   CMP     @#PAR6,R4       ;UP TO THE I/O PAGE YET ?
1085 002472 002764                          BLT     17$             ;  JP UNTIL ALL MEMORY WRITTEN
1086 002474 162737 000200 172354 27$:       SUB     #200,@#PAR6     ;MOVE DOWN TO THE NEXT PAGE
1087 002502 003443                          BLE     33$             ;ALL DONE WHEN BELOW START
1088 002504 012701 160000                   MOV     #160000,R1      ;TOP ADDRESS OF PAGE+2
1089 002510 012705 177777           11$:    MOV     #-1,R5          ;SET TO ALL ONES
1090 002514 005741                  18$:    TST     -(R1)           ;CHECK FOR 0 AND DEC ADDRESS
1091 002516 001012                          BNE     2$              ;BR, IF NOT = 0 (ERROR)
1092 002520 074511                          XOR     R5,(R1)         ;LOAD MEMORY SORT OF
1093 002522 005111                          COM     (R1)            ;CHECK FOR CORRECT DATA
1094 002524 001001                          BNE     12$             ;BR, IF NOT 0 (ERROR)
1095 002526 000414                          BR      13$             ;KEEP GOING
1096 002530 042737 000001 177572 12$:       BIC     #1,@#SR0        ;TURN OFF MEMORY MNGT
1097 002536 013702 172354                   MOV     @#PAR6,R2       ;CONTENTS OF CURRENT MAPPING REGISTER
1098 002542                                 ERROR   21              ;PRINT "MEM BAD"
     002542 104021                          EMT 21
1099 002544 042737 000001 177572 2$:        BIC     #1,@#SR0        ;TURN OFF MEMORY MANGEMENT
1100 002552 013702 172354                   MOV     @#PAR6,R2       ;CONTENTS OF CURRENT MAPPING REGISTER
1101 002556                                 ERROR   22              ;PRINT "MEM BAD"
     002556 104022                          EMT 22
1102 002560 005011                  13$:    CLR     (R1)            ;RETURN MEMORY TO ORIG STATE
1103 002562 020127 140000                   CMP     R1,#140000      ;TESTED ALL OF PAGE ?
1104 002566 101352                          BHI     18$             ;BR IF NOT
1105 002570 062701 020000                   ADD     #20000,R1       ;SET TO START VALUE
1106 002574 162737 000200 172354            SUB     #200,@#PAR6     ;RESET THE MAP REGISTER
1107 002602 023727 172354 000400            CMP     @#PAR6,#400     ;CHECK FOR END
1108 002610 101341                          BHI     18$             ;BR, IF MORE TO TEST
1109 002612                         33$:
1110 002612 012737 000400 172354            MOV     #400,@#PAR6     ;START MEMORY PAGE aaaaa
1111 002620 012705 177777                   MOV     #-1,R5          ;ALL ONES THIS TIME
1112 002624 012701 140000           117$:   MOV     #140000,R1      ;START AT  LOC 0, RELATIVE TO PAR6
1113 002630 010521                  110$:   MOV     R5,(R1)+        ;STORE 0 MEMORY
1114 002632 020127 157776                   CMP     R1,#157776      ;END OF MEMORY PAGE YET ?
1115 002636 101774                          BLOS    110$            ;LOOP TILL WHOLE PAGE WRITTEN
1116 002640 062737 000200 172354            ADD     #200,@#PAR6     ;MAP INTO NEXT PAGE
1117 002646 023704 172354                   CMP     @#PAR6,R4       ;UP TO THE I/O PAGE YET ?
1118 002652 002764                          BLT     117$            ;LOOP UNTIL ALL MEMORY WRITTEN
1119 002654 162737 000200 172354 127$:      SUB     #200,@#PAR6     ;MOVE DOWN TO THE NEXT PAGE
1120 002662 003443                          BLE     133$            ;ALL DONE WHEN BELOW ZERO aaaaa
1121 002664 012701 160000                   MOV     #160000,R1      ;TOP ADDRESS OF THE PAGE +2
1122 002670 005005                  111$:   CLR     R5              ;SET R5 TO 000000'S
1123 002672 005741                  116$:   TST     -(R1)           ;CHECK FOR 0 AND DEC ADDRESS
1124 002674 001412                          BEQ     212$            ;BR, IF = 0 (ERROR)
1125 002676 010511                          MOV     R5,(R1)         ;LOAD MEMORY SORT OF
1126 002700 005111                          COM     (R1)            ;CHECK FOR CORRECT DATA
1127 002702 001401                          BEQ     112$            ;BR, IF 0 (ERROR)
1128 002704 000414                          BR      113$            ;KEEP GOING
1129 002706 042737 000001 177572 112$:      BIC     #1,@#SR0        ;TURN OFF MEMORY MANAGEMENT
1130 002714 013702 172354                   MOV     @#PAR6,R2       ;CONTENTS OF CURRENT MAPPING REGISTER
1131 002720                                 ERROR   23              ;PRINT "MEM BAD"
     002720 104023                          EMT 23
1132 002722 042737 000001 177572 212$:      BIC     #1,@#SR0        ;TURN OFF MEMORY MANGEMENT
1133 002730 013702 172354                   MOV     @#PAR6,R2       ;CONTENTS OF CURRENT MAPPING REGISTER
1134 002734                                 ERROR   24              ;PRINT "MEM BAD"
     002734 104024                          EMT 24
```

SEQ 0050

```
1135 002736  012711  177777          113$:   MOV     #-1,(R1)        ;RETURN MEMORY TO ORIG STATE
1136 002742  020127  140000                  CMP     R1,#140000      ;TESTED ALL OF PAGE ?
1137 002746  101351                          BHI     116$            ;BR IF NOT
1138 002750  062701  020000                  ADD     #20000,R1       ;SET TO START VALUE
1139 002754  162737  000200  172354          SUB     #200,@#PAR6     ;RESET THE MAP REGISTER
1140 002762  023727  172354  000400          CMP     @#PAR6,#400     ;CHECK FOR END
1141 002770  101340                          BHI     116$            ;BR, IF MORE TO TEST
1142 002772                          133$:
1143 002772  042737  000001  177572          BIC     #1,@#SR0        ;TURN OFF MEMORY MANAGEMENT
1144                                  ;
1145                                  ;
1146                                  ;
1147 003000  000403                          BR      TST3            ;ALL DONE
1148
1149 003002  000000                  BKGND:  0                       ;BACKGROUND
1150 003004  000000                  RDWRD:  0                       ;STORE READ WORD HERE
1151 003006  000000                  SAVBK:  0                       ;ROTATE BACKGROUND HERE
1152
1153
```

```
   1159                                 ;;*****************************************************************
                                        ;*TEST 3         ROM TESTS
                                        ;;*****************************************************************
        003010  000004                  TST3:   SCOPE
        003012  012767  000010  176140          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
        003020  012767  000003  176152          MOV     #3,$TSTN        ;;SET TEST NUMBER IN APT MAIL BOX
   1164 003026  000240                          NOP                     ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
   1165
   1166
   1167 003030  016700  176046          MOV     $TSTNM,R0       ;GET TEST NUMBER
   1168 003034  005100                  COM     R0              ;1'S COMP TEST NUMBER
   1169 003036  032767  000100  176206  BIT     #BIT6,$DEVM     ;CAN THE LEDS BE USED
   1170 003044  001002                  BNE     5018            ;BR, IF THEY CAN'T
   1171 003046  110077  176232          MOVB    R0,@LEDREG      ;DISPLAY TEST NUMBER IN LEDS
   1172 003052                  5018:
   1173 003052  032767  000002  176172  BIT     #BIT1,$DEVM     ;BYPASS ROM TEST?
   1174 003060  001521                  BEQ     TST3S           ;EXIT IF YES
   1175 003062  012767  021522  174714  MOV     #INTSRV,ERRVEC  ;SETUP TIMEOUT TRAP ADDR
   1176 003070  012767  000200  174710  MOV     #200,ERRVEC+2
   1181
   1182                                 ;;*****************************************************************
   1183                                 ;TEST 'LOROM' TO 'HIROM' PRESENT
   1184                                 ;;*****************************************************************
   1189
   1190 003076  016703  176172          MOV     LOROM,R3
   1191 003102  005067  016422  38:     CLR     INTFLG
   1192 003106  005713                  TST     (R3)            ;ACCESS
   1193 003110  005767  016414          TST     INTFLG          ;TIMEOUT?
   1194 003114  001403                  BEQ     48              ;BR IF NO
   1195 003116  010367  000200          MOV     R3,ADDR         ;FOR ERROR ANALYSIS
   1196 003122                          ERROR   31
        003122  104031                  EMT 31
   1197
   1198 003124  020367  176146  48:     CMP     R3,HIROM        ;ALL DONE?
   1199 003130  001403                  BEQ     58              ;BR IF YES
   1200 003132  062703  000002          ADD     #2,R3           ;ELSE BUMP ADDR
   1201 003136  000761                  BR      38
   1202
   1207                                 ;;*****************************************************************
   1208                                 ;TEST THAT ROM CANNOT BE WRITTEN INTO
   1209                                 ;;*****************************************************************
   1214
   1215 003140  016703  176130  58:     MOV     LOROM,R3
   1216 003144  012700  101010  68:     MOV     #101010,R0      ;PATTERN FOR WRITE
   1217 003150  010013                  MOV     R0,(R3)         ;TRY TO WRITE INTO ROM
   1218 003152  011301                  MOV     (R3),R1         ;READ ROM BACK
   1219 003154  020100                  CMP     R1,R0           ;CHECK IN VS OUT
   1220 003156  001003                  BNE     78              ;BR IF YES
   1221 003160  010367  000136          MOV     R3,ADDR         ;FOR ERROR ANALYSIS
   1222 003164                          ERROR   32
        003164  104032                  EMT 32
   1223
   1224 003166  020367  176104  78:     CMP     R3,HIROM        ;ALL DONE?
   1225 003172  001403                  BEQ     88              ;BR IF YES
   1226 003174  062703  000002          ADD     #2,R3           ;ELSE BUMP ADDR
   1227 003200  000761                  BR      68
   1232
```

```
                                ;;*****************************************************************
                                ;TEST 'LOROM2' TO 'HIROM2' PRESENT
                                ;;*****************************************************************

003202  016703  176072          8$:     MOV     LOROM2,R3
003206  005067  016316          18$:    CLR     INTFLG
003212  005713                          TST     (R3)            ;ACCESS
003214  005767  016310                  TST     INTFLG          ;TIMEOUT?
003220  001403                          BEQ     9$              ;BR IF NO
003222  010367  000074                  MOV     R3,ADDR         ;FOR ERROR ANALYSIS
003226                                  ERROR   33
003226  104033                          EMT 33

003230  020367  176046          9$:     CMP     R3,HIROM2       ;;ALL DONE?
003234  001403                          BEQ     10$             ;BR IF YES
003236  062703  000002                  ADD     #2,R3           ;ELSE BUMP ADDR
003242  000761                          BR      18$

                                ;;*****************************************************************
                                ;TEST THAT ROM CANNOT BE WRITTEN INTO
                                ;;*****************************************************************

003244  016703  176030          10$:    MOV     LOROM2,R3
003250  012700  101010          11$:    MOV     #101010,R0              ;PATTERN FOR WRITE
003254  010013                          MOV     R0,(R3)                 ;TRY TO WRITE INTO ROM
003256  011301                          MOV     (R3),R1                 ;RE'D ROM BACK
003260  020100                          CMP     R1,R0                   ;CHECK IN VS OUT
003262  001003                          BNE     12$                     ;BR IF YES
003264  010367  000032                  MOV     R3,ADDR                 ;FOR ERROR ANALYSIS
003270                                  ERROR   34
003270  104034                          EMT 34

003272  020367  176004          12$:    CMP     R3,HIROM2               ;ALL DONE?
003276  001403                          BEQ     13$                     ;BR IF YES
003300  062703  000002                  ADD     #2,R3                   ;ELSE BUMP ADDR
003304  000761                          BR      11$
003306  012767  000006  174470  13$:    MOV     #ERRVEC+2,ERRVEC        ;RESTORE TIMEOUT VECTOR
003314  005067  174466                  CLR     ERRVEC+2
003320  000401                          BR      TST35                   ;EXIT TEST

003322  000000                  ADDR:   .WORD 0                         ;ROM ADDRESS WHERE ERROR OCCURED
                                ;+
                                ;*********************************
                                ;       TEST OF PCR REGISTER
                                ;*********************************
                                ;-
003324  032767  000200  175720  TST35:  BIT     #BIT7,$DEVM             ;IS THE PCR REGISTER THERE
003332  001067                          BNE     TST4                    ;BR, IF NO TESTING TO BE DONE
003334  012767  021522  174442          MOV     #INTSRV,ERRVEC          ;SET UP VECTOR
003342  012767  000200  174436          MOV     #200,ERRVEC+2           ;SET UP BR
003350  005067  016154                  CLR     INTFLG                  ;CLEAR INTERRUPT INDICATOR
003354  005777  175730                  TST     @PCRREG                 ;LOOK AT REGISTER
003360  005767  016144                  TST     INTFLG                  ;CHECK FOR TRAP
003364  001401                          BEQ     10$                     ;BR, IF NO TRAP (OK)
003366                                  ERROR   35
003366  104035                          EMT 35
003370  012701  177777          10$:    MOV     #177777,R1              ;ALL ONES
```

```
1299 003374  042701  160340              BIC     #160340,R1          ;PCR REGISTER MASK
1300 003400  010177  175704              MOV     R1,@PCRREG          ;LOAD REGISTER WITH "ALL" ONES
1301 003404  017702  175700              MOV     @PCRREG,R2          ;READ REGISTER BACK
1302 003410  042702  160340              BIC     #160340,R2          ;PCR REGISTER MASK
1303 003414  020201                      CMP     R2,R1               ;CAN PCR HOLD "ALL" ONES
1304 003416  001401                      BEQ     12$                 ;BR, IF PCR CAN (GOOD)
1305 003420                              ERROR   36                  ;ERROR IT CAN'T
     003420  104036                      EMT 36
1306 003422                      12$:
1307 003422  005000                      CLR     R0                  ;CLEAR REGISTER
1308 003424  000241                      CLC                         ;CLEAR THE CARRY BIT
1309 003426  012701  000001              MOV     #1,R1               ;SET UP COUNTER
1310 003432  012703  000016              MOV     #16,R3              ;SET ANOTHER COUNTER
1311 003436  010100              15$:    MOV     R1,R0               ;START MASK
1312 003440  042700  160340              BIC     #160340,R0          ;REAL MASK
1313 003444  010077  175640              MOV     R0,@PCRREG          ;LOAD THE PCR REGISTER
1314 003450  017702  175634              MOV     @PCRREG,R2          ;READ IT BACK
1315 003454  042702  160340              BIC     #160340,R2          ;REAL MASK (ONLY 10 BITS USED)
1316 003460  020002                      CMP     R0,R2               ;RIGHT BITS COME BACK
1317 003462  001401                      BEQ     20$                 ;BR, IF ALL OK (EQUAL)
1318 003464                              ERROR   37                  ;
     003464  104037                      EMT 37
1319 003466  006301              20$:    ASL     R1                  ;SHIFT BIT
1320 003470  005701                      TST     R1                  ;CHECK FOR END
1321 003472  001401                      BEQ     30$                 ;BR, IF AT END
1322 003474  000760                      BR      15$                 ;KEEP GOING NOT DONE
1323 003476  012767  000006  174300  30$:    MOV     #ERRVEC+2,ERRVEC    ;RESET VECTORS
1324 003504  005067  174276              CLR     ERRVEC+2            ;RESET VECTORS
1325 003510  000400                      BR      TST4                ;NEXT TEST
1326                             ;
1327                             ;       END OF PCR REGISTER TEST
1328                             ;
```

```
 1350
 1355                                ;;*****************************************************************
                                     ;*TEST 4          CLOCK TESTS
                                     ;*
                                     ;*       THE CLOCK TESTS NOW WORK AS FOLLOWS:
                                     ;*       1.THE BR LEVEL IS SET TO 6
                                     ;*       2.THE CLOCK VECTOR IS SET WITH INTSRV ADDRESS
                                     ;*         THE PRIORTY IS ALSO SET TO ZERO (ALLOW ANYONE)
                                     ;*       3.THE BR LEVEL IS DROPPED TO ZERO AND INTERRUPT
                                     ;*         ENABLE IS SET IN THE CLOCK'S CSR (BIT 6)
                                     ;*       4.PROGRAM DELAYS ONE REGISTER COUNT DOWN
                                     ;*       5.IF NO INTERRUPT OCCURRED
                                     ;*         THEN PRINT ERROR NUMBER 41
                                     ;*       6.DISABLE CLOCK INTERRUPTS (BIT 6 = 0)
                                     ;*       7.PROGRAM DELAYS ONE REGISTER COUNT DOWN
                                     ;*       8.IF AN INTERRUPT OCCURRED
                                     ;*         THEN PRINT ERROR 42
                                     ;*       9.SET INTERRUPT PRIORITY AT MAXIMUM (BR = 7)
                                     ;*      10.RESTORE THE CLOCK'S INTERRUPT VECTOR
                                     ;*      11.DROP BR LEVEL BACK TO 0
                                     ;*
                                     ;;*****************************************************************
 003512  000004                      TST4:   SCOPE
 003514  012767  000010  175436              MOV     #10.,$TIMES     ;;DO 10 ITERATIONS
 003522  012767  000004  175450              MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 1360 003530  000240                         NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 1361
 1362 003532  016700  175344                 MOV     $TSTNM,R0       ;GET TEST NUMBER
 1363 003536  005100                         COM     R0              ;1'S COMP TEST NUMBER
 1364 003540  032767  000100  175504         BIT     #BIT6,$DEVM     ;CAN THE LEDS BE USED
 1365 003546  001002                         BNE     50$             ;BR, IF THEY CAN'T
 1366 003550  110077  175530                 MOVB    R0,@LEDREG      ;DISPLAY TEST NUMBER IN LEDS
 1367 003554                          50$:
 1368 003554  032767  000001  175470         BIT     #BIT0,$DEVM     ;SHOULD WE TEST THE CLOCK
 1369 003562  001002                         BNE     52$             ;BR, IF YES WE SHOULD
 1370 003564  000167  000136                 JMP     SLU             ;NO DON'T TEST THE CLOCK
 1371 003570                          52$:
 1372 003570  106427                         .WORD   106427          ;MTPS
 1373 003572  000300                         .WORD   300             ;SET THE BR LEVEL = 6
 1374 003574  012767  021522  174276         MOV     #INTSRV,CLKVEC  ;SETUP VECTOR AREA
 1375 003602  012767  000340  174272         MOV     #340,CLKVEC+2   ;LET ANYONE IN AFTER CLOCK'S INTR
 1376 003610  005067  015714                 CLR     INTFLG
 1377 003614  106427                         .WORD   106427          ;MTPS
 1378 003616  000000                         .WORD   0               ;ALLOW CLOCK TO INTERR THRU INTSRV
 1379 003620  112777  000100  175460         MOVB    #100,@LKSREG    ;SET BIT6 (IE)
 1380
 1381
 1382 003626  012700  177777                 MOV     #-1,R0          ;TIMER
 1383 003632  005300                  1$:    DEC     R0
 1384 003634  001376                         BNE     1$
 1385 003636  042777  000100  175442         BIC     #BIT6,@LKSREG   ;CLEAR IE SO NO MORE CLOCK INTERRUPTS
 1386 003644  005767  015660                 TST     INTFLG          ;GOT INTERRUPT?
 1387 003650  001002                         BNE     2$              ;BR IF YES
 1388 003652                                 ERROR   41              ;NO CLOCK INTERRUPT
      003652  104041                        EMT 41
 1389 003654  000412                         BR      4$
 1390
```

```
 .391 003656                            2$:
1392 003656  005067  015646                    CLR     INTFLG          ;CLEAR OUT THE INTERRUPT FLAG
1393 003662  012700  177777                    MOV     #-1,R0          ;TIMER
1394 003666  005300                    3$:     DEC     R0
1395 003670  001376                            BNE     3$
1396 003672  005767  015632                    TST     INTFLG          ;GOT INTERRUPT?
1397 003676  001401                            BEQ     4$              ;BR, IF NO INTERRUPT (GOOD)
1398 003700                                    ERROR   42              ;DID NOT EXPECT 2ND INTERRUPT
     003700  104042                            EMT 42
1399
1400 003702                            4$:
1401
1402 003702  106427                            .WORD   106427          ;MTPS
1403 003704  000340                            .WORD   340             ;DISABLE INTERR
1404
1405 003706  012767  022560  174164            MOV     #ILLCLK,CLKVEC  ;RESTORE CLOCK VECTOR AREA
1406 003714  012767  000340  174160            MOV     #340,CLKVEC+2
1407
1408 003722  106427                            .WORD   106427          ;MTPS
1409 003724  000000                            .WORD   0               ;ENABLE INTERRUPTS
```

```
1411
1412 003726  005067  016424          SLU:    CLR     PHASE2                  ;INIT FOR TESTS
1413 003732  005067  175250                  CLR     $UNIT                   ;CONTAINS CHAN # UNDER TEST ( 0 OR 1 )
1414 003736  005067  016416                  CLR     P1CNT
1415
1416 003742  012706  001100          LOOP:   MOV     #STACK,SP               ;RESET STACK POINTER
1417 003746  005767  015442                  TST     NOCHAN                  ;ANY CHANNELS TO TEST?
1418 003752  001404                          BEQ     1$                      ;BR IF YES
1419 003754  104401  001334                  TYPE    ,CARR
1420 003760  000167  020700                  JMP     $EOP                    ;ELSE ALL DONE
1421
1422 003764  004767  015542          1$:     JSR     PC,CYCLE                ;ELSE SETUP MODULE AND CHANNEL
1423
1424 003770  016767  175316  175316          MOV     DLADD,RCSR
1425
1426 003776  016767  175310  175312          MOV     DLADD,RBUF
1427 004004  062767  000002  175304          ADD     #2,RBUF
1428
1429 004012  016767  175274  175300          MOV     DLADD,TCSR
1430 004020  062767  000004  175272          ADD     #4,TCSR
1431
1432 004026  016767  175260  175266          MOV     DLADD,TBUF
1433 004034  062767  000006  175260          ADD     #6,TBUF
1434
1435 004042  012700  177777                  MOV     #-1,R0                  ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
1436 004046  077001                          SOB     R0,.                    ;FINISH BEFORE RESET
1437 004050  004567  016512                  JSR     R5,TIMER                ;GO OUT TO TIMER
1438 004054  001320                          TCSR
1439 004056  000200                          DONE
1440 004060  000240                          NOP
1441 004062  000240                          NOP
1442 004064  000005                          RESET
1443
1444 004066  005267  175112                  INC     $DEVCT
1445 004072  126727  016200  000001          CMPB    PHASE2,#TRUE            ;DOING PHASE 2?
1446 004100  001010                          BNE     2$                      ;BR IF NO
1447 004102  042777  000004  175210          BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
1448 004110  112767  000022  174764          MOVB    #22,$TSTNM              ;GET TEST NUMBER READY
1449 004116  000167  005362                  JMP     MODTST
1450 004122                          2$:
1451
```

```
 1464                                   ;;*****************************************************************
                                        ;*TEST 5        ADDRESSABILITY
                                        ;*
                                        ;*      THIS TEST VERIFIES THAT ALL 4 REGISTERS OF THE CHANNEL UNDER TEST
                                        ;*      RESPOND TO THEIR ADDRESSES.
                                        ;*
                                        ;;*****************************************************************
      004122  000004                    TST5:   SCOPE
      004124  012767  000002  175026            MOV     #2,$TIMES       ;;DO 2 ITERATIONS
      004132  012767  000005  175040            MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 1469 004140  000240                            NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 1472 004142  042777  000004  175150            BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
 1473 004150  016700  174726                    MOV     $TSTNM,R0               ;GET TEST NUMBER
 1474 004154  005100                            COM     R0                      ;1'S COMP TEST NUMBER
 1475 004156  032767  000100  175066            BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 1476 004164  001002                            BNE     50$                     ;BR, IF THEY CAN'T
 1477 004166  110077  175112                    MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 1478 004172                            50$:
 1479 004172  012767  021522  173604            MOV     #INTSRV,ERRVEC          ;SETUP TIMEOUT VECTOR
 1480 004200  012767  000340  173600            MOV     #PR7,ERRVEC+2
 1482 004206  005000                            CLR     R0
 1483 004210  016701  175100                    MOV     RCSR,R1
 1485 004214  005067  015310            1$:     CLR     INTFLG
 1486 004220  005721                            TST     (R1)+                   ;DARE TO TIMEOUT
 1487 004222  005767  015302                    TST     INTFLG                  ;DID IT?
 1488 004226  001401                            BEQ     2$                      ;BR IF NO
 1489 004230                                    ERROR   51                      ;TIMEOUT ON CHANNEL ADDRESS
      004230  104051                            EMT 51
 1491 004232  005200                    2$:     INC     R0
 1492 004234  020027  000004                    CMP     R0,#4                   ;ALL DONE?
 1493 004240  001365                            BNE     1$                      ;BR IF NO
 1495 004242  012767  000006  173534            MOV     #6,ERRVEC               ;RESTORE TIMEOUT VECTOR
 1496 004250  005067  173532                    CLR     ERRVEC+2
 1498 004254  032777  177477  175032            BIT     #177477,@RCSR           ;CHECK THAT ALL UNUSED BITS ARE 0
 1499 004262  001401                            BEQ     3$
 1500 004264                                    ERROR   52                      ;RCSR HAS UNUSED BITS SET
      004264  104052                            EMT 52
 1502 004266  032777  007400  175022    3$:     BIT     #7400,@RBUF
 1503 004274  001401                            BEQ     4$
 1504 004276                                    ERROR   53                      ;RBUF HAS UNUSED BITS SET
      004276  104053                            EMT 53
 1506 004300  032777  177476  175012    4$:     BIT     #177476,@TCSR
 1507 004306  001401                            BEQ     5$
 1508 004310                                    ERROR   54                      ;TCSR HAS UNUSED BITS SET
      004310  104054                            EMT 54
 1510 004312  032777  177400  175002    5$:     BIT     #177400,@TBUF
 1511 004320  001401                            BEQ     .+4                     ;EXIT IF OK
 1512 004322                                    ERROR   55                      ;TBUF HAS ;UNUSED BITS SET
      004322  104055                            EMT 55
```

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


                                ;;*****************************************************************
                                ;*TEST 6        BREAK - TCSR 0 SET, CLEAR, RESET
                                ;*               THIS BIT IS THE ONLY ONE IN THIS POSITION
                                ;*               THAT IS READ AND WRITE.
                                ;;*****************************************************************
004324  000004                  TST6:   SCOPE
004326  012767  000010  174624          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
004334  012767  000006  174636          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1537 004342  000240                    NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
1538 004344  042777  000004  174746    BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
1539 004352  016700  174524            MOV     $TSTNM,R0               ;GET TEST NUMBER
1540 004356  005100                    COM     R0                      ;1'S COMP TEST NUMBER
1541 004360  032767  000100  174664    BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
1542 004366  001002                    BNE     50$                     ;BR, IF THEY CAN'T
1543 004370  110077  174710            MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
1544 004374                    501$:
1545 004374  005767  015750            TST     BRK                     ;BREAK DETECTION ENABLED?
1546 004400  001403                    BEQ     1$                      ;BR IF NO
1547 004402  005767  015746            TST     CONSOLE                 ;ELSE ARE WE ON CONSOLE?
1548 004406                            SKIP    NE,<EXIT IF YES>
1553 004410                    1$:
1554
1555 004410                                                            ; SEE IF IT IS CLEAR
                                                                       LPADR
004410  012767  004416  174472          MOV     #64$,$LPERR
004416                          64$:
1556
1557
1558 004416  032777  000001  174674    BIT     #BREAK,@TCSR            ;IF #BREAK SETIN @TCSR THEN
1559 004424  001401                    BEQ     10$                     ;DID BREAK RESET TCSR
1560 004426                            ERROR 61                        ;BR, IF NO ERROR
004426  104061                        EMT 61
1561 004430                    10$:                                    ;ENDIF
1562                                                                   ; TRY TO SET BREAK BIT
1563 004430                                                            LPADR
004430  012767  004436  174452          MOV     #65$,$LPERR
004436                          65$:
1564
1565 004436  052777  000001  174654    BIS     #BREAK,@TCSR            ;LET @TCSR := @TCSR SET.BY #BREAK
1566                                                                   ;SET BREAK TO A ONE
1567                                                                   ; STUCK TO 0
1568 004444  032777  000001  174646    BIT     #BREAK,@TCSR            ;IF #BREAK NOTSETIN @TCSR THEN
1569                                                                   ;DID BIT GET SET
1570 004452  001001                    BNE     20$                     ; BREAK DID NOT SET IN TCSR
1571 004454                            ERROR 62                        ;BR, IF NO ERROR
004454  104062                        EMT 62
1572 004456                    20$:                                    ;ENDIF
1573
1574                                                                   ; TRY TO CLEAR A SET BIT
1575 004456                                                            LPADR
004456  012767  004464  174424          MOV     #66$,$LPERR
004464                          66$:
1576 004464  042777  000001  174626    BIC     #BREAK,@TCSR            ;LET @TCSR := @TCSR CLR.BY #BREAK
```

```
                                                                ; SHOULD HAVE CLEARED
004472  032777  000001  174620          BIT     #BREAK,@TCSR    ;IF #BREAK SETIN @TCSR THEN
004500  001401                          BEQ     30$             ;BR, IF OK (NO ERROR)
                                                                ; BREAK  DID NOT CLEAR IN TCSR
004502                                  ERROR 63
004502  104063                          EMT 63
004504                          30$:                            ;ENDIF

                                                                ; NOW SEE IF RESET CLEARS IT
                                                                LPADR
004504
004504  012767  004512  174376          MOV     #67$,$LPERR
004512
004512                          67$:
004512  052777  000001  174600          BIS     #BREAK,@TCSR    ;LET @TCSR := @TCSR SET.BY #BREAK
                                                                ; ISSUE BUS RESET
004520  012700  177777                  MOV     #-1,R0          ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
004524  077001                          SOB     R0,.            ;FINISH BEFORE RESET
004526  000005                          RESET
004530  032777  000001  174562          BIT     #BREAK,@TCSR    ;IF #BREAK SETIN @TCSR THEN
004536  001401                          BEQ     40$             ;BR, IF NO ERROR BIT NOT SET
                                                                ; BREAK DID NOT RESET IN TCSR
004540                                  ERROR 64
004540  104064                          EMT 64
004542                          40$:                            ;ENDIF
```

```
 1597
 1602                                   ;;*****************************************************************
                                        ;*TEST 7        IE - TCSR 6    SET, CLEAR, RESET
                                        ;;*****************************************************************
     004542  000004                     TST7:   SCOPE
     004544  012767  000010  174406             MOV     #10,$TIMES      ;;DO 10 ITERATIONS
     004552  012767  000007  174420             MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 1603                                                                            ; USE PRIORITY OF 7
 1608 004560  000240                        NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 1609
 1610 004562  042777  000004  174530        BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
 1611 004570  016700  174306                MOV     $TSTNM,R0               ;GET TEST NUMBER
 1612 004574  005100                        COM     R0                      ;1'S COMP TEST NUMBER
 1613 004576  032767  000100  174446        BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 1614 004604  001002                        BNE     50I$                    ;BR, IF THEY CAN'T
 1615 004606  110077  174472                MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 1616 004612                            50I$:
 1621 004612  012746  000340                MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     004616  012746  004624                 MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
     004622  000002                        RTI                     ;;POP NEW PC AND PS
     004624                             64$:
 1626
 1627                                                                   ; SEE IF IT IS CLEAR
 1628 004624                                                            LPADR
     004624  012767  004632  174256             MOV     #65$,$LPERR
     004632                             65$:
 1629
 1630 004632  032777  000100  174460        BIT     #IE,@TCSR               ;IF #IE SETIN @TCSR THEN
 1631                                                                       ; IE DID NOT RESET IN TCSR
 1632 004640  001401                        BEQ     50$                     ;BR, IF NO ERROR
 1633 004642                                ERROR 71
     004642  104071                         EMT 71
 1634 004644                            50$:                                ;ENDIF
 1635
 1636                                                                   ; TRY TO SET IE BIT
 1637 004644                                                            LPADR
     004644  012767  004652  174236             MOV     #66$,$LPERR
     004652                             66$:
 1638 004652  052777  000100  174440        BIS     #IE,@TCSR               ;LET @TCSR := @TCSR SET.BY #IE
 1639                                                                       ; STUCK TO 0
 1640 004660  032777  000100  174432        BIT     #IE,@TCSR               ;IF #IE NOTSETIN @TCSR THEN
 1641                                                                       ; XMIT DID NOT SET IN TCSR
 1642 004666  001001                        BNE     60$                     ;BR, IF NO ERROR
 1643 004670                                ERROR 72
     004670  104072                         EMT 72
 1644 004672                            60$:                                ;ENDIF
 1645
 1646                                                                   ; TRY TO CLEAR A SET BIT
 1647 004672                                                            LPADR
     004672  012767  004700  174210             MOV     #67$,$LPERR
     004700                             67$:
 1648 004700  042777  000100  174412        BIC     #IE,@TCSR               ;LET @TCSR := @TCSR CLR.BY #IE
 1649                                                                       ; SHOULD HAVE CLEARED
 1650 004706  032777  000100  174404        BIT     #IE,@TCSR               ;IF #IE SETIN @TCSR THEN
 1651                                                                       ;   XMIT DID NOT CLEAR IN TCSR
 1652 004714  001401                        BEQ     70$                     ;BR, IF NO ERROR
 1653 004716                                ERROR 73
```

```
        004716  104073                          EMT 73
1654    004720                          70$:                            ;ENDIF
1655
1656                                                                    ; NOW SEE IF RESET CLEARS IT
1657    004720                                                          LPADR
        004720  012767  000426  174162          MOV     #68$,$LPERR
        004726                          68$:
1658    004726  052777  000100  174364          BIS     #IE,@TCSR       ;LET @TCSR := @TCSR SET.BY #IE
1659                                                                    ; ISSUE BUS RESET
1660    004734  000005                          RESET
1661    004736  016700  174140                  MOV     $TSTNM,R0       ;GET TEST NUMBER
1662    004742  005100                          COM     R0              ;1'S COMP TEST NUMBER
1663    004744  032767  000100  174300          BIT     #BIT6,$DEVM     ;CAN THE LEDS BE USED
1664    004752  001002                          BNE     50$             ;BR, IF THEY CAN'T
1665    004754  110077  174324                  MOVB    R0,@LEDREG      ;DISPLAY TEST NUMBER IN LEDS
1666    004760                          50$:
1667    004760  032777  000100  174332          BIT     #IE,@TCSR       ;IF #IE SETIN @TCSR THEN
1668                                                                    ; XMIT DID NOT RESET IN TCSR
1669    004766  001401                          BEQ     80$             ;BR, IF NO ERROR
1670    004770                                  ERROR 74
        004770  104074                          EMT 74
1671    004772                          80$:                            ;ENDIF
```

```
 1673
 1682                                   ;;*****************************************************************
                                        ;*TEST 10        IE - RCSR 6    SET, CLEAR, RESET
                                        ;*              THIS BIT IS THE ONLY ONE IN THIS POSITION
                                        ;*              THAT IS READ AND WRITE.
                                        ;;*****************************************************************
      004772  000004                    TST10:  SCOPE
      004774  012767  000010  174156            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
      005002  012767  000010  174170            MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1687 005010  000240                            NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 1688
 1689 005012  042777  000004  174300            BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
 1690 005020  016700  174056                    MOV     $TSTNM,R0               ;GET TEST NUMBER
 1691 005024  005100                            COM     R0                      ;1'S COMP TEST NUMBER
 1692 005026  032767  000100  174216            BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 1693 005034  001002                            BNE     501$                    ;BR, IF THEY CAN'T
 1694 005036  110077  174242                    MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 1695 005042                            501$:
 1696                                                                           ; SEE IF IT IS CLEAR
 1697 005042                                                                    LPADR
      005042  012767  005050  174040            MOV     #64$,$LPERR
      005050                            64$:
 1698 005050  032777  000100  174236            BIT     #IE,@RCSR               ;IF #IE SETIN @RCSR THEN
 1699                                                                           ; IE DID NOT RESET IN RCSR
 1700 005056  001401                            BEQ     110$
 1701 005060                                    ERROR 101
      005060  104101                            EMT 101
 1702 005062                            110$:                                   ;ENDIF
 1703                                                                           ; TRY TO SET IE BIT
 1704 005062                                                                    LPADR
      005062  012767  005070  174020            MOV     #65$,$LPERR
      005070                            65$:
 1705 005070  052777  000100  174216            BIS     #IE,@RCSR               ;LET @RCSR := @RCSR SET.BY #IE
 1706                                                                           ; STUCK TO 0
 1707 005076  032777  000100  174210            BIT     #IE,@RCSR               ;IF #IE NOTSETIN @RCSR THEN
 1708                                                                           ; IE DID NOT SET IN RCSR
 1709 005104  001001                            BNE     120$                    ;BR, IF NO ERROR
 1710 005106                                    ERROR 102
      005106  104102                            EMT 102
 1711 005110                            120$:                                   ;ENDIF
 1712                                                                           ; TRY TO CLEAR A SET BIT
 1713 005110                                                                    LPADR
      005110  012767  005116  173772            MOV     #66$,$LPERR
      005116                            66$:
 1714 005116  042777  000100  174170            BIC     #IE,@RCSR               ;LET @RCSR := @RCSR CLR.BY #IE
 1715                                                                           ; SHOULD HAVE CLEARED
 1716 005124  032777  000100  174162            BIT     #IE,@RCSR               ;IF #IE SETIN @RCSR THEN
 1717                                                                           ; IE DID NOT CLEAR IN RCSR
 1718 005132  001401                            BEQ     130$                    ;BR, IF NO ERROR
 1719 005134                                    ERROR 103
      005134  104103                            EMT 103
 1720 005136                            130$:                                   ;ENDIF
 1721                                                                           ; NOW SEE IF RESET CLEARS IT
 1722 005136                                                                    LPADR
      005136  012767  005144  173744            MOV     #67$,$LPERR
      005144                            67$:
 1723 005144  052777  000100  174142            BIS     #IE,@RCSR               ;LET @RCSR := @RCSR SET.BY #IE
```

```
1724
1725 005152  000005                        RESET                           ; ISSUE BUS RESET
1726 005154  016700  173722                MOV     $TSTNM,R0               ;GET TEST NUMBER
1727 005160  005100                        COM     R0                      ;1'S COMP TEST NUMBER
1728 005162  032767  000100  174062        BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
1729 005170  001002                        BNE     502$                    ;BR, IF THEY CAN'T
1730 005172  110077  174106                MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
1731 005176                        502$:
1732 005176  032777  000100  174110        BIT     #IE,@RCSR               ;IF #IE SETIN @RCSR THEN
1733                                                                        ; IE DID NOT RESET IN RCSR
1734 005204  001401                        BEQ     140$                    ;BR, IF NO ERROR
1735 005206                                ERROR 104
     005206  104104                        EMT 104
1736 005210                        140$:                                   ;ENDIF
1737 005210  012767  005216  173672        MOV     #1$,$LPERR
1738
1739 005216  032777  000200  174074  1$:   BIT     #RDY,@TCSR              ;XMIT RDY SHOULD BE SET BY RESET ABOVE
1744 005224  001013                        BNE     TST11           ;;EXIT IF SET
1749 005226                                ERROR   105
     005226  104105                        EMT 105
1750
1751 005230  000005                        RESET                           ;ALLOW LOOPING ON ERROR
1752 005232  016700  173644                MOV     $TSTNM,R0               ;GET TEST NUMBER
1753 005236  005100                        COM     R0                      ;1'S COMP TEST NUMBER
1754 005240  032767  000100  174004        BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
1755 005246  001002                        BNE     503$                    ;BR, IF THEY CAN'T
1756 005250  110077  174030                MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
1757 005254                        503$:
1758
```

```
                                ;;*****************************************************************
                                ;*TEST 11        TEST THAT XMIT RDY - TCSR 7 - CLEARS
                                ;*               WHEN TBUF IS LOADED WITH A CHARACTER
                                ;*               AND THAT IT SETS WITHIN A REASONABLE AMOUNT OF TIME.
                                ;;*****************************************************************
005254  000004                  TST11:  SCOPE
005256  012767  000010  173674          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
005264  012767  000011  173706          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
005272  000240                          NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
005274  042777  000004  174016          BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
005302  016700  173574                  MOV     $TSTNM,R0               ;GET TEST NUMBER
005306  005100                          COM     R0                      ;1'S COMP TEST NUMBER
005310  032767  000100  173734          BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
005316  001002                          BNE     501$                    ;BR, IF THEY CAN'T
005320  110077  173760                  MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
005324                          501$:
                                ;       TST     WRAP                    ;CAN WE TEST DATA WRAP
                                ;       SKIP    NE,<EXIT IF NO>
005324  005767  015024                  TST     CONSOLE                 ;ARE WE ON CONSOLE?
005330                                  SKIP    NE,<EXIT IF YES>

005332  005067  000164                  CLR     PASS                    ;LET PASS := #0 ;INIT COUNT OF TIMES THRU
005336                          150$:                                   ;LOOP START OF LOOP
                                                                        ;  MAX OF 2 TIMES THRU
005336  005067  000162                  CLR     ERRFLG                  ;LET ERRFLG := #0
005342  005067  000160                  CLR     EXFLG                   ;LET EXFLG := #0
                                                                        ; LOAD TBUF WITH ONE CHARACTER
                                                                        ; WAIT FOR READY TO SET
                                                                        ; (SHOULD BE VERY SHORT WAIT
                                                                        ; SINCE UART DOUBLE BUFFERS ITS INPUT)
005346  105077  173750                  CLRB    @TBUF                   ;SEND A CHARACTER
                                                                        ;LET @TBUF :B= #0
005352  004567  015210                  JSR     R5,TIMER                ;WAIT FOR XMIT READY
005356  001320                              TCSR
005360  000200                              RDY
005362                                  ERROR   111                     ;XMIT RDY DID NOT SET IN TCSR
005362  104111                          EMT 111
005364  000461                          BR      TST12           ;;EXIT TEST

005366  105077  173730                  CLRB    @TBUF                   ;SHIP 1'ST CHAR
005372  105777  173722          1$:     TSTB    @TCSR                   ;WAIT FOR RDY
005376  100375                          BPL     1$
                                                                        ; LOAD TBUF WITH A SECOND CHARACTER (TO DOUBLE BUFFE
                                                                        ; CHECK IMMEDIATELY THAT RDY IS CLEAR
                                                                        ; AND THEN WAIT FOR IT TO SET
                                                                        ;SEND SECOND CHARACTER
005400  105077  173716                  CLRB    @TBUF                   ;LET @TBUF :B= #0
005404  000240                                          NOP             ; GIVE IT TIME TO CLEAR
                                                                        ; RDY SHOULD HAVE CLEARED UPON
                                                                        ; RECEIPT OF A CHARACTER
005406  032777  000200  173704          BIT     #RDY,@TCSR              ;IF #RDY SETIN @TCSR THEN
005414  001404                          BEQ     170$                    ;BR, IF READY DID CLEAR
                                                                        ; RDY DID NOT CLEAR IN TCSR
                                                                        ;WILL RESULT IN ERR 67 IF FAILS 2X
005416  012767  177777  000100          MOV     #SET,ERRFLG             ;LET ERRFLG := #SET
```

```
1833
1834 005424  000406                    BR      200$                    ; DEFER ERROR TYPEOUT
1835 005426                      170$:                                 ;ELSE
1836 005426  004567  015134            JSR     R5,TIMER                ;WAIT FOR XMIT READY
1837 005432  001320                        TCSR
1838 005434  000200                        RDY
1839 005436                            ERROR   112                     ;XMIT RDY DID NOT SET IN TCSR
     005436  104112                    EMT 112
1844 005440  000433                    BR      TST12           ;;EXIT TEST
1849 005442                      200$:                                 ;ENDIF OF DEFERED ERROR CALL
1850 005442  026727  000056  177777    CMP     ERRFLG,#SET             ;IF ERRFLG EQ #SET THEN
1851 005450  001013                    BNE     210$                    ;BR, IF NO ERROR
1852 005452  005267  000044            INC     PASS                    ;LET PASS := PASS + #1
1853 005456  026727  000040  000001    CMP     PASS,#1                 ;IF PASS GT #1 THEN
1854                                                                   ;CALL ERROR IF 2ND TRY
1855                                                                   ;ON XMIT RDY NOT CLEARING
1856 005464  003404                    BLE     220$                    ;BR, NO ERROR
1857 005466                            ERROR 113
     005466  104113                    EMT 113
1858 005470  012767  177777  000030    MOV     #SET,EXFLG              ;LET EXFLG := #SET
1859 005476                      220$:                                 ;ENDIF
1860 005476  000403                    BR      230$                    ;ELSE NO ERROR
1861 005500                      210$:
1862 005500  012767  177777  000020    MOV     #SET,EXFLG              ;LET EXFLG := #SET
1863 005506                      230$:                                 ;ENDIF
1864 005506  026727  000014  177777    CMP     EXFLG,#SET              ;EXIF EXFLG EQ #SET
1865 005514  001401                    BEQ     160$                    ;BR, IF NO ERROR
1866                                                                   ;ENDLOOP
1867 005516  000707                    BR      150$                    ;LOOP
1868 005520                      160$:
1869 005520  000403                    BR      .+10                    ;EXIT TEST
1870
1871 005522  000000            PASS:   0
1872 005524  000000            ERRFLG: 0
1873 005526  000000            EXFLG:  0
```

```
 1875
 1884
 1885                                   ;;*****************************************************************
                                        ;*TEST 12        TEST OUTPUT CHAR FROM TBUF (WRAP CONNECTED)
                                        ;*              RESULTS IN DONE SETTING WITHIN A REASONABLE AMOUNT OF TIME
                                        ;*              AND THAT RESET CLEARS THE BIT.
                                        ;;*****************************************************************
      005530  000004                    TST12:  SCOPE
      005532  012767  000010  173420            MOV     #10.,$TIMES     ;;DO 10 ITERATIONS
      005540  012767  000012  173432            MOV     #12.,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 1890 005546  000240                            NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 1891
 1892 005550  042777  000004  173542            BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
 1893 005556  016700  173320                    MOV     $TSTNM,R0               ;GET TEST NUMBER
 1894 005562  005100                            COM     R0                      ;1'S COMP TEST NUMBER
 1895 005564  032767  000100  173460            BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 1896 005572  001002                            BNE     501$                    ;BR, IF THEY CAN'T
 1897 005574  110077  173504                    MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 1898 005600                            501$:
 1899 005600  005767  014546                    TST     WRAP                    ;DOING DATA WRAP TESTS?
 1900 005604                                    SKIP    NE,<EXIT IF NO>
 1901 005606  005767  014542                    TST     CONSOLE                 ;ARE WE ON CONSOLE?
 1902 005612                                    SKIP    NE,<EXIT IF YES>
 1907 005614                                                                    LPADR
      005614  012767  005622  173266            MOV     #64$,$LPERR
      005622                            64$:
 1908
 1909 005622  032767  001000  173422            BIT     #BIT9,$DEVM             ; SEND A CHARACTER AND LET IT WRAP AROUND
 1910 005630  001403                            BEQ     235$                    ;INTERNAL OR EXTERNAL
 1911 005632  052777  000004  173460            BIS     #BIT2,@TCSR             ;BR, IF EXTERNAL
 1912 005640  105077  173456            235$:   CLRB    @TBUF                   ;SET MAINTENANCE BIT INT LOOP
 1913 005644  004567  014716                    JSR     R5,TIMER                ;LET @TBUF :B= #0
 1914 005650  001314                                RCSR                        ;WAIT FOR RECV DONE
 1915 005652  000200                                DONE
 1916 005654                                    ERROR   121                     ;RECV DONE DID NOT SET IN RCSR
      005654  104121                            EMT 121
 1921 005656  000415                            BR      TST13           ;;EXIT TEST
 1926
 1927 005660                                                                    LPADR
      005660  012767  005666  173222            MOV     #65$,$LPERR
      005666                            65$:
 1928
 1929 005666  016702  173420                    MOV     @LADD,R2                ; NOW THAT IT IS SET SEE IF IT CAN BE RESET
 1930 005672  062702  000002                    ADD     #2,R2                   ;LOAD R2 WITH BASE ADDR
 1931 005676  011203                            MOV     (R2),R3                 ;BUMP R2 TO POINT TO RBUF
 1932 005700  032777  000200  173406            BIT     #DONE,@RCSR             ;DUMMY READ OF BUFFER
 1933                                                                           ;IF #DONE SETIN @RCSR THEN
 1934 005706  001401                                                            ; DONE DID NOT RESET IN RCSR.
 1935 005710                                    BEQ     240$                    ;BR, IF NO ERROR
      005710  104122                            ERROR 122
 1936 005712                                    EMT 122
                                        240$:                                   ;ENDIF
```

```
 1938
 1943                                   ;;*****************************************************************
                                        ;*TEST 13        TEST THAT DONE IS CLEARED BY READING RBUF
                                        ;;*****************************************************************
 005712  000004                        TST13:  SCOPE
 005714  012767  000010  173236                MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 005722  012767  000013  173250                MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 1948
 1949 005730  000240                           NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 1950 005732  042777  000004  173360           BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
 1951 005740  016700  173136                   MOV     $TSTNM,R0               ;GET TEST NUMBER
 1952 005744  005100                           COM     R0                      ;1'S COMP TEST NUMBER
 1953 005746  032767  000100  173276           BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 1954 005754  001002                           BNE     501$                    ;BR, IF THEY CAN'T
 1955 005756  110077  173322                   MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 1956 005762                            501$:
 1957 005762  005767  014364                   TST     WRAP                    ,DOING DATA WRAP TESTS?
 1958 005766                                    SKIP    NE,<EXIT IF NO>
 1959 005770  005767  014360                   TST     CONSOLE                 ;ARE WE ON CONSOLE?
 1960 005774                                    SKIP    NE,<EXIT IF YES>
 1965
 1966 005776                                                                    LPADR
      005776  012767  006004  173104           MOV     #64$,$LPERR
      006004                            64$:
 1967
 1968                                                                           ; OUTPUT A CHARACTER AND WAIT FOR DONE TO SET.
                                                                                ; OUTPUT A CHARACTER
 1969 006004  032767  001000  173240           BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
 1970 006012  001403                           BEQ     245$                    ;BR, IF EXTERNAL
 1971 006014  052777  000004  173276           BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
 1972 006022  105077  173274            245$:  CLRB    @TBUF                   ;LET @TBUF :B= #0
 1973
 1974 006026  004567  014534                   JSR     R5,TIMER                ;WAIT FOR RECV DONE
 1975 006032  001314                               RCSR
 1976 006034  000200                               DONE
 1977 006036                                    ERROR   131                     ;RECV DONE DID NOT SET IN RCSR
      006036  104131                           EMT 131
 1982 006040  000407                           BR      TST14           ;;EXIT TEST
 1987
 1988                                                                           ; NOW THAT IT IS SET LETS SEE IF READING THE
 1989                                                                           ; BUFFER CLEARS DONE.
 1990                                                                           ;READ BUFFER
 1991 006042  117700  173250                   MOVB    @RBUF,R0                ;LET R0 :B= @RBUF
 1992 006046  032777  000200  173240           BIT     #DONE,@RCSR             ;IF #DONE SETIN @RCSR THEN
 1993                                                                           ;DONE DID NOT CLEAR IN RCSR
 1994 006054  001401                           BEQ     250$                    ;BR, IF NO ERROR
 1995 006056                                    ERROR 132
      006056  104132                           EMT 132
 1996
 1997 006060                            250$:                                   ; SET IT BACK TO CONTINUE
                                                                                ;ENDIF
```

```
 1999
 2004                                   ;;*****************************************************************
                                        ;*TEST 14       TEST THE OVERRUN & ERROR BITS - RBUF 14
                                        ;;*****************************************************************
      006060  000004                    TST14:  SCOPE
      006062  012767  000010  173070            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
      006070  012767  000014  173102            MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2005 006076  000240                            NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 2006 006100  052777  000004  173212            BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
 2007 006106  042777  000004  173204            BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
 2008 006114  005767  014232                    TST     WRAP                    ;DOING DATA WRAP TESTS?
 2009 006120  001136                            BNE     TST15           ;;EXIT IF NO
 2010 006122  005767  014226                    TST     CONSOLE                 ;ARE WE ON CONSOLE?
 2011 006126  001133                            BNE     TST15           ;;EXIT IF YES
 2016
 2017 006130  016700  172746                    MOV     $TSTNM,R0               ;GET TEST NUMBER
 2018 006134  005100                            COM     R0                      ;1'S COMP TEST NUMBER
 2019 006136  032767  000100  173106            BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 2020 006144  001002                            BNE     50I$                    ;BR, IF THEY CAN'T
 2021 006146  110077  173132                    MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 2022 006152                            50I$:
 2023
 2024 006152                                                                    LPADR
      006152  012767  006160  172730            MOV     #64$,$LPERR
      006160                            64$:
 2025                                                                           ;OUTPUT 2 CHARACTERS
 2026                                                                           ;THIS SHOULD AN CAUSE OVERRUN ERROR.
 2027
 2028                                                                           ;OUTPUT 1 CHARACTER
 2029 006160  032767  001000  173064            BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
 2030 006166  001403                            BEQ     255$                    ;BR, IF EXTERNAL
 2031 006170  052777  000004  173122            BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
 2032 006176  105077  173120            255$:   CLRB    @TBUF                   ;LET @TBUF :B= #0
 2033
 2034 006202  004567  014360                    JSR     R5,TIMER                ;WAIT FOR RECV DONE
 2035 006206  001314                                RCSR
 2036 006210  000200                                DONE
 2037 006212                                    ERROR   141                     ;RECV DONE DID NOT SET IN RCSR
      006212  104141                            EMT 141
 2042 006214  000500                            BR      TST15           ;;EXIT TEST
 2047
 2048                                                                           ;OUTPUT 2ND CHARACTER
 2049 006216  032767  001000  173026            BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
 2050 006224  001403                            BEQ     257$                    ;BR, IF EXTERNAL
 2051 006226  052777  000004  173064            BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
 2052 006234  105077  173062            257$:   CLRB    @TBUF                   ;LET @TBUF :B= #0
 2053 006240  004767  014440                    JSR     PC,WAIT                 ;LET OVERRUN HAPPEN
 2054                                                                           ;READ BUFFER AND ERROR BITS
 2055 006244  017704  173046                    MOV     @RBUF,R4                ;LET R4 := @RBUF
 2056                                                                           ;IT DIDN'T SET
 2057 006250  032704  040000                    BIT     #ORERR,R4               ;IF #ORERR NOTSETIN R4 THEN
 2058                                                                           ;ORERR DID NOT SET IN RBUF
 2059 006254  001002                            BNE     260$                    ;BR, IF NO ERROR
 2060 006256                                    ERROR 142
      006256  104142                            EMT 142
 2061                                                                           ;NO USE COMPOUNDING ERRORS
 2066 006260  000456                            BR      TST15           ;;EXIT TEST
```

```
 2071 006262                        260$:                                ;ENDIF
 2072
 2073
 2074 006262                                                             ;NOW SEE IF ERROR BIT SET WITH OVERRUN  ERROR:
      006262  012767  006270  172620        MOV     #65$,$LPERR          LPADR
      006270                        65$:
 2075 006270  032704  100000                BIT     #ERR15,R4            ;IF #ERROR NOTSETIN R4 THEN
 2076                                                                    ;ERROR DID NOT SET IN RBUF
 2077 006274  001002                        BNE     270$                 ;BR, IF NO ERROR
 2078 006276                                ERROR 143
      006276  104143                        EMT 143
 2079
 2080
 2081                                                                    ;-WHEN ORERR SET.
 2086 006300  000446                        BR      TST15          ;;EXIT TEST  ;GET OUT NOW.
 2091 006302                        270$:                                ;ENDIF
 2092
 2093 006302                                                             LPADR
      006302  012767  006310  172600        MOV     #66$,$LPERR
      006310                        66$:
 2094
 2095                                                                    ;CHECK REAL RBUF TO SEE IF ORERR IS STILL SET.
 2096 006310  032777  040000  173000        BIT     #ORERR,@RBUF         ;IF #ORERR NOTSETIN @RBUF THEN
 2097                                                                    ;READING RBUF CLEARED ORERR.
 2098 006316  001002                        BNE     300$                 ;BR, IF NO ERROR
 2099 006320                                ERROR 144
      006320  104144                        EMT 144
 2100                                                                    ;SKIP REST OF TEST
 2105 006322  000435                        BR      TST15          ;;EXIT TEST
 2110 006324                        300$:                                ;ENDIF
 2111
 2112 006324                                                             LPADR
      006324  012767  006332  172556        MOV     #67$,$LPERR
      006332                        67$:
 2113                                                                    ;READING RBUF ABOVE SHOULD ENABLE ERROR TO BE CLEARE
 2114                                                                    ;BY NEXT TRANSFER.
 2115                                                                    ;NOW SEE IF THEY CLEAR WHEN ANOTHER CHAR. IS RECEIVE
 2116
 2117                                                                    ;SEND A CHARACTER AROUND.
 2118 006332  032767  001000  172712        BIT     #BIT9,$DEVM          ;INTERNAL OR EXTERNAL
 2119 006340  001403                        BEQ     305$                 ;BR  IF EXTERNAL
 2120 006342  052777  000004  172750        BIS     #BIT2,@TCSR          ;SET MAINTENANCE BIT INT LOOP
 2121 006350  105077  172746        305$:   CLRB    @TBUF                ;LET @TBUF :B= #0
 2122 006354  004567  014206                JSR     R5,TIMER             ;WAIT FOR RECV DONE
 2123 006360  001314                           RCSR
 2124 006362  000200                           DONE
 2125 006364                                ERROR   145                  ;RECV DONE DID NOT SET IN RCSR
      006364  104145                        EMT 145
 2130 006366  000413                        BR      TST15          ;;EXIT TEST
 2135
 2136 006370  032777  040000  172720        BIT     #ORERR,@RBUF         ;IF #ORERR SETIN @RBUF THEN
 2137                                                                    ;ORERR DID NOT CLEAR IN RBUF
 2138 006376  001402                        BEQ     310$                 ;BR, IF NO ERROR
 2139 006400                                ERROR 146
      006400  104146                        EMT 146
 2140
 2141                                                                    ;-AFTER RECEIVING ANOTHER CHAR
```

```
 2142                                                                   ;SKIP AROUND REST
 2147 006402  000405                    BR      TST15          ;;EXIT TEST
 2152 006404                    3108:                                   ;ENDIF
 2153
 2154 006404  032777  100000  172704     BIT     #ERR15,@RBUF           ;IF #ERROR SETIN @RBUF THEN
 2155                                                                   ;ERROR DID NOT CLEAR IN RBUF
 2156 006412  001401                    BEQ     3208                   ;BR, IF NO ERROR
 2157 006414                            ERROR 147
      006414  104147                    EMT 147
 2158
 2159 006416                    3208:                                   ;ENDIF
```

```
                                ;;*****************************************************************
                                ;*TEST 15        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
                                ;;*****************************************************************
006416  000004
006420  012767  000010  172532  TST15:  SCOPE
006426  012767  000015  172544          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
2178 006434  000240                    MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2179                                    NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
2180 006436  042777  000004  172654    BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
2181 006444  016700  172432            MOV     $TSTNM,R0               ;GET TEST NUMBER
2182 006450  005100                    COM     R0                      ;1'S COMP TEST NUMBER
2183 006452  032767  000100  172572    BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
2184 006460  001002                    BNE     501$                    ;BR, IF THEY CAN'T
2185 006462  110077  172616            MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
2186 006466                     501$:
2187
2188 006466  005067  013036            CLR     INTFLG
2189
2190 006472  016703  172626            MOV     DLVEC,R3
2191 006476  062703  000004            ADD     #4,R3
2192 006502  012723  021522            MOV     #INTSRV,(R3)+           ;GET XMIT VECT ADDR
2193 006506  012713  000340            MOV     #PR7,(R3)
2194 006512                                                             LPADR
006512  012767  006520  172370        MOV     #64$,$LPERR
006520                          64$:
2195 006520  004567  014042            JSR     R5,TIMER                ;WAIT FOR XMIT READY
2196 006524  001320                        TCSR
2197 006526  000200                        RDY
2198 006530                             ERROR   151                     ;XMIT RDY DID NOT SET IN TCSR
006530  104151                         EMT 151
2203 006532  000453                    BR      TST16           ;;EXIT TEST
2208                                                                    ;CLEAR INTERRUPT ENABLE
2209 006534  042777  000100  172556    BIC     #IE,@TCSR               ;LET @TCSR := @TCSR CLR.BY #IE
221' 006542  012746  0C0000            MOV     #PR0,-(SP)      ;;PUT NEW PS ON STACK
006546  012746  006554                MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
006552  000002                        RTI                     ;;POP NEW PC AND PS
006554                          65$:
2219
2220 006554  052777  000100  172536    BIS     #IE,@TCSR               ;NOW SET I.E. BIT
2221 006562  004567  014054            JSR     R5,TIME$1               ;LET @TCSR := @TCSR SET.BY #IE
2222 006566  021530                        INTFLG                      ;SFE IF FLAG SETS
2223 006570                             ERROR   152                     ;INTERR FLAG DID NOT SET
006570  104152                         EMT 152
2228 006572  000433                    BR      TST16           ;;EXIT TEST
2233                                                                    ;BYPASS THIS PART IF CONSOLE
2234 006574  026727  813554  000000    CMP     CONSOLE,#0              ;IF CONSOLE EQ 0 THEN ;;;aaa IS 0 CORRECT???
2235 006602  001007                    BNE     330$                    ;BR, IF NO ERROR
2236 006604  004767  014074            JSR     PC,WAIT                 ;LET POSSIBLE 2'ND INTERR OCCUR
2237                                                                    ;DID EXACTLY 1 INTERRUPT OCCUR
2238 006610  026727  012714  000001    CMP     INTFLG,#1               ;IF INTFLG GT #1 THEN
2239                                                                    ;TRANSMITTER INTERRUPTED TWICE
```

```
 2240 006616  003401                          BLE     340$                ;BR, IF NO ERROR
 2241 006620                                  ERROR   153
      006620  104153                          EMT 153
 2242 006622                          340$:                               ;ENDIF
 2243 006622                          330$:                               ;ENDIF
 2244
 2245                                                                     ;INTERRUPT WITHOUT INTERRUPT ENABLE SET
 2246 006622                                                              LPADR
      006622  012767  006630  172260          MOV     #66$,$LPERR
      006630                          66$:
 2247
 2248 006630  042777  000100  172462          BIC     #IE,@TCSR           ;CLEAR INTERRUPT ENABLE
 2249                                                                     ;LET @TCSR := @TCSR CLR.BY #IE
 2250 006636  005067  012666                  CLR     INTFLG              ;CLEAR 'INTERRUPT OCCURED' FLAG
 2251                                                                     ;LET INTFLG := #0
 2252                                                                     ;NO INTERRUPTS SHOULD OCCUR, PSW STILL AT 0.
 2253 006642  005077  172454                  CLR     @TBUF               ;DARE IT TO HAPPEN
 2254                                                                     ;LET @TBUF := #0
 2255 006646  004767  014032                  JSR     PC,WAIT             ;SEE IF INTERR FLAG EVER SETS
 2256 006652  005767  012652                  TST     INTFLG
 2257 006656  001401                          BEQ     .+4                 ;BR IF NO
 2258 006660                                  ERROR   154                 ;INTERR FLAG OCCURED WITH IE DISABLED
      006660  104154                          EMT 154
 2259
 2260
```

```
                                ;;*****************************************************************
                                ;*TEST 16       RECEIVER INTERRUPT LOGIC TEST
                                ;*             THIS TEST COVERS ALL OF THE RECEIVER
                                ;*             SIDE OF THE INTERRUPT LOGIC IN
                                ;*             CHARACTER MODE.
                                ;;*****************************************************************
006662  000004                  TST16:  SCOPE
006664  012767  000010  172266          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
006672  012767  000016  172300          MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2273 006700  000240                    NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
2274 006702  042777  000004  172410    BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
2275 006706  005767  013436            TST     WRAP                    ;DOING DATA WRAP TESTS?
2276 006714  001135                    BNE     TST17           ;;EXIT IF NO
2277 006716  005767  013432            TST     CONSOLE                 ;ARE WE ON CONSOLE?
2278 006722  001132                    BNE     TST17           ;;EXIT IF YES
2283
2284 006724  042777  000004  172366    BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
2285 006732  016700  172144            MOV     $TSTNM,R0               ;GET TEST NUMBER
2286 006736  005100                    COM     R0                      ;1'S COMP TEST NUMBER
2287 006740  032767  000100  172304    BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
2288 006746  001002                    BNE     501$                    ;BR, IF THEY CAN'T
2289 006750  110077  172330            MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
2290 006754                     501$:
2291                                                                    ;CLEAR INTERRUPT OCCURED FLAG
2292                                                                    ;SET UP RECEIVER INTER.VECTOR
2293 006754  016701  172344            MOV     DLVEC,R1
2294 006760  012721  021522            MOV     #INTSRV,(R1)+
2295 006764  012711  000340            MOV     #PR7,(R1)
2296
2297                                                                    ;PRIORITY 0 AND MULTIPLE INTERRUPT TEST.-IE
2298 006770                                                             LPADR
006770  012767  006776  172112          MOV     #64$,$LPERR
006776                          64$:
2299 006776  005067  012526            CLR     INTFLG                  ;LET INTFLG := #0
2300                                                                    ;CLEAR INTERRUPTS
2301 007002  042777  000100  172304    BIC     #IE,@RCSR               ;LET @RCSR := @RCSR CLR.BY #IE
2302                                                                    ;CHANGE PRIORITY
2303                                                                    ;..TO 0
2308 007010  012746  000000            MOV     #PR0,-(SP)      ;;PUT NEW PS ON STACK
007014  012746  007022                  MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
007020  000002                         RTI                     ;;POP NEW PC AND PS
007022                          65$:
2313
2314 007022  032767  001000  172222    BIT     #BIT9,$DEVM             ;SEND A CHARACTER
2315 007030  001403                    BEQ     345$                    ;INTERNAL OR EXTERNAL
2316 007032  052777  000004  172260    BIS     #BIT2,@TCSR             ;BR, IF EXTERNAL
2317 007040  105077  172256    345$:   CLRB    @TBUF                   ;SET MAINTENANCE BIT INT LOOP
2318 007044  004567  013516            JSR     R5,TIMER                ;LET @TBUF :B= #0
2319 007050  001320                       TCSR                         ;WAIT FOR XMIT READY
2320 007052  000200                       RDY
2321 007054                             ERROR   161                     ;XMIT RDY DID NOT SET IN TCSR
007054  104161                         EMT 161
2326 007056  000454                    BR      TST17           ;;EXIT TEST
2331
2332                                                                    ;SET INTERRUPT ENABLE
2333 007060  052777  000100  172226    BIS     #IE,@RCSR               ;LET @RCSR := @RCSR SET.BY #IE
```

```
2334
2335 007066  004567  013550                   JSR     R5,TIMER1               ;SEE IF FLAG SETS
2336 007072  021530                                 INTFLG
2337 007074                                   ERROR   162                     ;INTERR FLAG DID NOT SET
     007074  104162                           EMT 162
2342 007076  000444                           BR      TST17           ;;EXIT TEST
2347
2348 007100  004767  013600                   JSR     PC,WAIT
2349                                                  ;LET POSSIBLE 2'ND INTERR OCCUR
2350                                                                          ;EXACTLY 1 INTERRUPT?
2351 007104  026727  012420  000001           CMP     INTFLG,#1               ;IF INTFLG GT #1 THEN
2352                                                                          ;RECEIVER INTERRUPTED TWICE
2353 007112  003401                           BLE     350$                    ;BR, IF NO ERROR
2354 007114                                   ERROR   163
     007114  104163                           EMT 163
2355 007116                           350$:                                   ;ENDIF
2356
2357                                                                          ;INTERRUPT WITHOUT IE SET.
2358 007116                                                                   LPADR
     007116  012767  007124  171764           MOV     #66$,$LPERR
     007124                           66$:
2359                                                                          ;CLEAR INTERRUPT
2360 007124  042777  000100  172162           BIC     #IE,@RCSR               ;LET @RCSR := @RCSR CLR.BY #IE
2361                                                                          ;CLEAR INTERRUPT FLAG
2362 007132  005067  012372                   CLR     INTFLG                  ;LET INTFLG := #0
2363                                                                          ; SEND A CHARACTER
2364 007136  032767  001000  172106           BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
2365 007144  001403                           BEQ     355$                    ;BR, IF EXTERNAL
2366 007146  052777  000004  172144           BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
2367 007154  105077  172142           355$:   CLRB    @TBUF                   ;LET @TBUF :B= #0
2368                                                                          ; DARE IT
2369 007160  004767  013520                   JSR     PC,WAIT                 ;SEE IF INTERR FLAG EVER SETS
2370 007164  005767  012340                   TST     INTFLG
2371 007170  001401                           BEQ     .+4                     ;BR IF NO
2372 007172                                   ERROR   164                     ;INTERR FLAG OCCURED WITH IE DISABLED
     007172  104164                           EMT 164
2373
2378 007174  012746  000340                   MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     007200  012746  007206                   MOV     #67$,-(SP)      ;;PUT NEW PC ON  STACK
     007204  000002                           RTI                     ;;POP NEW PC AND PS
     007206                           67$:
2383 007206  000005                           RESET                           ;CLEAR THE WORLD
2384
```

```
 2386
 2391                                   ;;*****************************************************************
                                        ;*TEST 17       TEST DATA  WRAP AROUND  BINARY COUNT: FLAG MODE
                                        ;;*****************************************************************
      007210  000004                    TST17:  SCOPE
      007212  012767  000001  171740            MOV     #1,@TIMES       ;;DO 1 ITERATION
      007220  012767  000017  171752            MOV     #17,@TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2392 007226  000240                            NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 2393 007230  042777  000004  172062            BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
 2394 007236  005767  013110                    TST     WRAP                    ;DOING DATA WRAP TESTS?
 2395 007242  001064                            BNE     TST20           ;;EXIT IF NO
 2396 007244  005767  013104                    TST     CONSOLE                 ;ARE WE ON CONSOLE?
 2397 007250  001061                            BNE     TST20           ;;EXIT IF YES
 2402
 2403 007252  016700  171624                    MOV     $TSTNM,R0               ;GET TEST NUMBER
 2404 007256  005100                            COM     R0                      ;1'S COMP TEST NUMBER
 2405 007260  032767  000100  171764            BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 2406 007266  001002                            BNE     501$                    ;BR, IF THEY CAN'T
 2407 007270  110077  172010                    MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 2408 007274                            501$:
 2409 007274  017704  172016                    MOV     @RBUF R4                ;READ REGISTER TO ASSURE DONE CLR
 2410                                                                           ;BINARY COUNT PATTERN
 2411 007300  005002                            CLR     R2                      ;INCR R2 FROM #0 TO #377 BY #1
 2412 007302  000401                            BR      360$
 2413 007304  005202                    370$:   INC     R2
 2414 007306  020227  000377            360$:   CMP     R2,#377                 ;CMP FOR END
 2415 007312  003040                            BGT     400$                    ;BR, IF GREATER THAN
 2416 007314  004567  013246                    JSR     R5,TIMER                ;WAIT FOR XMIT READY
 2417 007320  001320                               TCSR
 2418 007322  000200                               RDY
 2419 007324                                    ERROR   171                     ;XMIT RDY DID NOT SET IN TCSR
      007324  104171                           EMT 171
 2424 007326  000432                            BR      TST20           ;;EXIT TEST
 2429
 2430                                                                           ;START IT ON ITS WAY
 2431 007330  032767  001000  171714            BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
 2432 007336  001403                            BEQ     365$                    ;BR, IF EXTERNAL
 2433 007340  052777  000004  171752            BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
 2434 007346  110277  171750            365$:   MOVB    R2,@TBUF                ;LET @TBUF :B= R2
 2435
 2436 007352  004567  013210                    JSR     R5,TIMER                ;WAIT FOR RECV DONE
 2437 007356  001314                               RCSR
 2438 007360  000200                               DONE
 2439 007362                                    ERROR   172                     ;RECV DONE DID NOT SET IN RCSR
      007362  104172                           EMT 172
 2444 007364  000413                            BR      TST20           ;;EXIT TEST
 2449
 2450                                                                           ;RETRIEVE
 2451 007366  017703  171724                    MOV     @RBUF,R3                ;LET R3 := @RBUF
 2452                                                                           ;CHECK FOR ERROR DURING TRANSFER
 2453 007372  032703  100000                    BIT     #ERR15,R3               ;IF #ERROR SETIN R3 THEN
 2454                                                                           ;ERROR BIT SET IN HIGH BYTE OF RBUF
 2455 007376  001401                            BEQ     410$                    ;BR, IF NO ERROR
 2456 007400                                    ERROR   173
      007400  104173                           EMT 173
 2457 007402                            410$:                                   ;ENDIF
 2458 007402  020302                            CMP     R3,R2                   ;COMPARE DATA
```

```
 2459 007404  001402                    BEQ     4$                ;BR IF OK
 2460 007406                            ERROR   174               ;DATA COMPARE ERR IN 8 BIT WORD
      007406  104174                    EMT 174
 2465 007410  000401                    BR      TST20        ;:EXIT TEST
 2466
 2467
 2468 007412                    4$:                               ;ENDINC ;R2
 2469 007412  000734                    BR      370$              ;LOOP
 2470 007414                    400$:
```

```
                                ;;*****************************************************************
                                ;*TEST 20       TEST DATA WRAP AROUND  BINARY COUNT: INTERRUPT MODE
                                ;;*****************************************************************
007414  000004                  TST20:  SCOPE
007416  012767  000001  171534          MOV     #1,$TIMES       ;;DO 1 ITERATION
007424  012767  000020  171546          MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2482 007432  000240                     NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
2483
2484 007434  042777  000004  171656     BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
2485 007442  016700  171434             MOV     $TSTNM,R0               ;GET TEST NUMBER
2486 007446  005100                     COM     R0                      ;1'S COMP TEST NUMBER
2487 007450  032767  000100  171574     BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
2488 007456  001002                     BNE     501$                    ;BR, IF THEY CAN'T
2489 007460  110077  171620             MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
2490 007464                     501$:
2491 007466  005767  012662             TST     WRAP                    ;DOING DATA WRAP TESTS?
2492 007470  001003                     BNE     1$                      ;BR IF NO
2493 007472  005767  012656             TST     CONSOLE                 ;ARE WE ON CONSOLE?
2494 007476  001402                     BEQ     2$                      ;BR IF NO
2495 007500  000167  000232     1$:     JMP     EX1                     ;EXIT TEST
2496 007504                     2$:
2497                                                                    ; THIS TEST WILL RUN BOTH TRANSMITTER AND
2498                                                                    ; RECIEVER AT FULL SPEED TESTING
2499                                                                    ; THE ABILITY OF THE MODULE
2500                                                                    ; TO HANDLE INTERRUPTS FROM BOTH SIDES AT ONCE.
2501                                                                    ;
2502                                                                    ;DUUBLE BUFFERING IS NOT FULLY TESTED BECAUSE OF
2503                                                                    ;APT CONSIDERATIONS. I.E. 'BREAK' FROM APT
2504                                                                    ;CAUSES OVERRUN ERRORS. THEREFORE TRANSMIT INTR IS
2505                                                                    ;ENABLED ONLY AFTER THE RECVR HAS OBTAINED THE LAST
2506
2507                                                                    ;  THIS TEST WILL TRANSFER A MAXIMUM OF 400(8)
2508                                                                    ; CHARACTERS THROUGH THE MODULE, BUT IF AN ERROR
2509                                                                    ; IS DETECTED BY THE TEST A PREMATURE SHUTDOWN OCCUR
2510
2511                                                                    ;CHANGE PRIORITY
2512                                                                    ;..TO 0
2517 007504  012746  000000             MOV     #PR0,-(SP)      ;;PUT NEW PS ON STACK
007510  012746  007516             MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
007514  000002                     RTI                     ;;POP NEW PC AND PS
007516                          64$:
2522
2523 007516  016701  171602             MOV     DLVEC,R1                ;GET VECTOR ADDRESS
2524                                                                    ;LET R1 := DLVEC
                                                                        ;RCVR VECTOR
2525 007522  012721  010020             MOV     #REC,(R1)+              ;LET (R1)+ := #REC
2526 007526  012721  000340             MOV     #PR7,(R1)+              ;LET (R1)+ := #PR7
2527                                                                    ;POINT TO TRANSMITTER VECTOR
2528                                                                    ;AND SET IT UP ALSO
2529 007532  012721  007744             MOV     #TRAN,(R1)+             ;LET (R1)+ := #TRAN
2530 007536  012711  000340             MOV     #PR7,(R1)               ;LET (R1)  := #PR7
2531                                                                    ; CLEAR ERROR COUNTER
2532 007542  005067  000172             CLR     ERRCNT                  ;LET ERRCNT := #0
2533 007546  005067  000244             CLR     DATA                    ;LET DATA := #0 XMIT DATA
2534
2535 007552  012767  000400  000162     MOV     #400,NUMBER
2536
```

```
2537 007560                          4$:
2538 007560  052777  000100  171532          BIS     #IE,@TCSR       ;SET I.E. IN TRANSMITTER
2539                                                                 ;LET @TCSR := @TCSR SET.BY #IE
2540 007566  052777  000100  171520          BIS     #IE,@RCSR       ;AND RECEIVER
2541                                                                 ;LET @RCSR := @RCSR SET.BY #IE
2542
2543                                                                 ;NOW WE WAIT
2544 007574  012767  000050  171526          MOV     #50,TMP2
2545 007602  012767  177777  171516  8$:     MOV     #-1,TMP1
2546 007610  026767  000202  000124  5$:     CMP     DATA,NUMBER     ;ALL DONE?
2547 007616  001413                          BEQ     7$              ;BR IF YES
2548 007620  005367  171502                  DEC     TMP1            ;ELSE TIMER DONE?
2549 007624  001005                          BNE     6$              ;BR IF NO
2550 007626  005367  171476                  DEC     TMP2
2551 007632  001363                          BNE     8$
2552 007634                                  ERROR   201             ;HUNG, NO DATA XFERS
     007634  104201                          EMT 201
2553 007636  000403                          BR      7$
2554
2555 007640  005767  000074          6$:     TST     ERRCNT          ;ANY ERRORS?
2556 007644  001761                          BEQ     5$              ;BR IF NO & TRY AGAIN
2557 007646                          7$:                             ;NOW LETS CHECK.
2558                                                                 ;TURN OFF ALL INTR ENABLE
2559 007646  042777  000100  171444          BIC     #IE,@TCSR       ;LET @TCSR := @TCSR CLR.BY #IE
2560 007654  042777  000100  171432          BIC     #IE,@RCSR       ;LET @RCSR := @RCSR CLR.BY #IE
2561 007662  005767  000052                  TST     ERRCNT          ;IF ERRCNT NE #0 THEN
2562 007666  001423                          BEQ     42C$            ;BR, IF NO ERROR
2563 007670  032767  104000  000226          BIT     #ERROR,RHLD     ;IF #ERROR SETIN RHLD THEN
2564 007676  001416                          BEQ     430$            ;BR, IF NO ERROR
2565 007700  032767  040000  000216          BIT     #ORERR,RHLD     ;IF #ORERR SETIN RHLD THEN
2566 007706  001402                          BEQ     440$            ;BR, IF NO ERROR
2567                                                                 ;OVERRUN ERROR
2568 007710                                  ERROR 202
     007710  104202                          EMT 202
2569 007712  000407                          BR      450$            ;ELSE IF #FRERR SETIN RHLD THEN
2570 007714                          440$:
2571                                                                 ;FRAMING ERROR
2572 007714  032767  020000  000202          BIT     #FRERR,RHLD
2573 007722  001402                          BEQ     460$            ;BR, IF NO ERROR
2574 007724                                  ERROR 203
     007724  104203                          EMT 203
2575 007726  000403                          BR 520$                 ;EXIT THE TEST
2576
2577 007730                          460$:
2578 007730                          500$:
2579                                                                 ;FRAMING ERROR
2580 007730                                  ERROR 204
     007730  104204                          EMT 204
2581 007732                          510$:
2582 007732                          470$:
2583 007732                          450$:
2584                                                                 ;ENDIF
2585 007732  000401                          BR      520$            ;ELSE
2586                                                                 ;DATA COMPARE ERROR
2587 007734                          430$:
2588 007734                                  ERROR   205
     007734  104205                          EMT 205
```

```
                                520$:                                   ;ENDIF
                                420$:                                   ;ENDIF
                                EX1:
007736  000473                            BR      TST21          ;;EXIT TEST

007740  000000                  ERRCNT: 0
007742  000000                  NUMBER: 0

                                ;;*****************************************************************
                                ;TRANSMIT INTERRUPT HANDLER
                                ;;*****************************************************************

007744                          TRAN:                                   ;IF DATA NE NUMBER AND ERRCNT FQ #0 THEN
007744  026767  000046  177770          CMP     DATA,NUMBER
007752  001415                        BEQ     530$                    ;BR, IF THEY ARE =
007754  005767  177760                TST     ERRCNT                  ;CHECK ERROR COUNTER
007760  001012                        BNE     530$                    ;BR, IF ERRORS
                                                                        ;SHIP OUT WORD
007762  032767  001000  171262        BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
007770  001403                        BEQ     525$                    ;BR, IF EXTERNAL
007772  052777  000004  171320        BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
010000  016777  000012  171314  525$:  MOV     DATA,@TBUF              ;LET @TBUF := DATA
010006                          530$:                                   ;ENDIF
                                                                        ;STOP INTERR, NOT EXER DOUBL BUFFER
010006  042777  000100  171304        BIC     #IE,@TCSR               ;LET @TCSR := @TCSR CLR.BY #IE
010014  000002                        RTI

010016  000000                  DATA:   0
```

```
                                ;;*****************************************************************
                                ;RECEIVER INTERRUPT HANDLER
                                ;;*****************************************************************

010020                          REC:
010020  017767  171272  000076          MOV     @RBUF,RHLD              ;GET CHAR
                                                                        ;LET RHLD := @RBUF
                                                                        ;CHECK ERROR
010026  036727  000072  100000          BIT     RHLD,#ERR15             ;IF #ERROR SETIN RHLD OR RHLD NE  DATA THEN
010034  001004                          BNE     540$                    ;BR, IF NO ERROR
010036  026767  000062  177752          CMP     RHLD,DATA
010044  001411                          BEQ     550$                    ;BR, IF EQUAL
010046                          540$:
                                                                        ;STOP ALL INTERR PROC & GET OUT
010046  016767  177670  177742          MOV     NUMBER,DATA             ;LET DATA := NUMBER
010054  042777  000100  171232          BIC     #IE,@RCSR               ;LET @RCSR := @RCSR CLR.BY #IE
010062  005267  177652                  INC     ERRCNT                  ;LET ERRCNT := ERRCNT + #1
010066  000415                          BR      560$                    ;ELSE       ;LET DATA := DATA + #1
010070                          550$:
010070  005267  177722                  INC     DATA
010074  026767  177716  177640          CMP     DATA,NUMBER             ;IF DATA EQ NUMBER THEN
010102  001004                          BNE     570$                    ;BR, IF NO ERROR
010104  042777  000100  171202          BIC     #IE,@RCSR               ;LET @RCSR := @RCSR CLR.BY #IE
010112  000403                          BR      600$                    ;ELSE
010114                          570$:
                                                                        ;ALLOW NEXT XMIT INTERR
010114  052777  000100  171176          BIS     #IE,@TCSR               ;LET @TCSR := @TCSR SET.BY #IE
010122                          600$:                                   ;ENDIF
010122                          560$:                                   ;ENDIF
010122  000002                          RTI
                                
010124  000000                  RHLD:   0
                                
```

```
                        ;;*****************************************************************
                        ;*TEST 21       TEST BREAK LOGIC
                        ;*              TRANSMIT KNOWN CHAR WITH BREAK SET
                        ;*              AND COMPARE RECEIVED WITH 0.
                        ;*              FRAMING ERROR WILL ALSO BE CHECKED
                        ;*              IF ERROR BITS ARE ENABLED.
                        ;;*****************************************************************
010126  000004          TST21:  SCOPE
010130  012767  000010  171022          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
010136  012767  000021  171034          MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2694 010144  000240                     NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
2695
2696 010146  042777  000004  171144     BIC     #BIT2,@TCSR             ;ALWAYS TURN OFF INTERNAL MAINT. BIT
2697 010154  016700  170722             MOV     $TSTNM,R0               ;GET TEST NUMBER
2698 010160  005100                     COM     R0                      ;1'S COMP TEST NUMBER
2699 010162  032767  000100  171062     BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
2700 010170  001002                     BNE     501$                    ;BR, IF THEY CAN'T
2701 010172  110077  171106             MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
2702 010176                     501$:
2703
2704 010176  005767  012150             TST     WRAP                    ;WRAP TESTS?
2705 010202  001006                     BNE     1$                      ;BR IF NO
2706 010204  005767  012144             TST     CONSOLE                 ;ELSE ARE WE ON CONSOLE?
2707 010210  001003                     BNE     1$                      ;BR IF YES...DONT TEST
2708 010212  005767  012132             TST     BRK                     ;ELSE IS BREAK DET ENABLED?
2709 010216  001402                     BEQ     2$                      ;BR IF NO
2710 010220  000167  000350     1$:     JMP     EX2
2711 010224                     2$:
2712 010224                                                             LPADR
010224  012767  010232  170656          MOV     #64$,$LPERR
010232                          64$:
2713
2714 010232  005067  000340             CLR     ERRCHK                  ;LET ERRCHK := #0 CLEAR ERROR WORD
2715                                                                    ;SET BREAK BIT
2716                                                                    ;NON-ZERO CHAR. '*'
2717 010236  032767  001000  171006     BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
2718 010244  001403                     BEQ     605$                    ;BR, IF EXTERNAL
2719 010246  052777  000004  171044     BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
2720 010254  052777  000001  171036     BIS     #BREAK,@TCSR    605$:   ;LET @TCSR := @TCSR SET.BY #BREAK
2721                                     ;MOVB   #125,@TBUF              ;LET @TBUF := #125
2722
2723 010262  004567  012300             JSR     R5,TIMER                ;WAIT FOR RECV DONE
2724 010266  001314                        RCSR
2725 010270  000200                        DONE
2726 010272                             ERROR   1                       ;RECV DONE DID NOT SET IN RCSR
010272  104001                          EMI 1
2731 010274  000541                     BR      TST22           ;;EXIT TEST
2736
2737 010276  017700  171014             MOV     @RBUF,R0                ;LET R0 := @RBUF
2738 010302  105700                     TSTB    R0                      ;IFB R0 NE #0 THEN
2739                                                                    ; BREAK DID NOT EQUAL 0
2740 010304  001403                     BEQ     610$                    ;BR, IF NOT = 0
2741 010306  052767  000001  000262     BIS     #BIT0,ERRCHK            ;LET ERRCHK := ERRCHK SET.BY #BIT0
2742 010314                     610$:                                   ;ENDIF
2743 010314                     $61:
2744 010314  032700  020000             BIT     #FRERR,R0               ;IF #FRERR NOTSETIN R0 THEN
```

```
 2745 010320  001003                        BNE     620$
 2746 010322  052767  000002  000246        BIS     #BIT1,ERRCHK            ;LET ERRCHK := ERRCHK SET.BY #BIT1
 2747 010330                          620$:                                 ;ENDIF
 2748
 2749 010330                          4$:
 2750 010330  042777  000001  170762        BIC     #BREAK,@TCSR            ;LET @TCSR := @TCSR CLR.BY #BREAK
 2751 010336  004767  012342                JSR     PC,WAIT                 ;WAIT FOR REC BREAK BIT TO CLEAR
 2752 010342  032777  004000  170746        BIT     #BIT11,@RBUF            ;NO CHECK BIT FOR CLEAR
 2753 010350  001402                        BEQ     5$                      ;BR, IF CLEAR (NO ERROR)
 2754 010352                                ERROR   2                       ;BREAK FAILED TO CLEAR IN RBUF
      010352  104002                        EMT 2
 2755 010354  000511                        BR      TST22                   ;EXIT TEST
 2756 010356                          5$:
 2757 010356  032767  001000  170666        BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
 2758 010364  001403                        BEQ     625$                    ;BR, IF EXTERNAL
 2759 010366  052777  000004  170724        BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
 2760 010374  112777  000125  170720  625$: MOVB    #125,@TBUF              ;SEND OUT CHARACTER
 2761 010402  004567  012160                JSR     R5,TIMER                ;WAIT FOR READY
 2762 010406  001314                        RCSR
 2763 010410  000200                        DONE
 2764 010412                                ERROR   3                       ;DONE FAILED TO SET
      010412  104003                        EMT 3
 2765 010414  000471                        BR      TST22                   ;EXIT
 2766 010416  032777  170000  170672        BIT     #170000,@RBUF
 2767 010424  001401                        BEQ     1$
 2768 010426                                ERROR   4                       ;RESET DID NOT CLEAR ERROR BITS
      010426  104004                        EMT 4
 2769                                                                       ;SENDING ANOTHER CHARACTER
 2770 010430                          1$:
 2771
 2772 010430  032767  000001  000140        BIT     #BIT0,ERRCHK            ;IF #BIT0 SETIN ERRCHK THEN
 2773 010436  001401                        BEQ     630$
 2774 010440                                ERROR   5                       ;BREAK ERROR
      010440  104005                        EMT 5
 2775 010442                          630$:                                 ;ENDIF
 2776 010442  032767  000002  000126        BIT     #BIT1,ERRCHK            ;IF #BIT1 SETIN ERRCHK THEN
 2777 010450  001401                        BEQ     660$
 2778 010452                                ERROR   6                       ; FRAMING ERROR
      010452  104006                        EMT 6
 2779                                                                       ;OCCURED WITH EVEN PARITY
 2780                                                                       ;ENABLED AND BREAK SET
 2781 010454                          660$:                                 ;ENDIF
 2782 010454                                                                LPADR
      010454  012767  010462  170426        MOV     #64$,$LPERR
      010462                          64$:
 2783                                                                       ;SET BREAK BIT
 2784 010462  052777  000001  170630        BIS     #BREAK,@TCSR            ;LET @TCSR := @TCSR SET.BY #BREAK
 2785
 2786 010470  004567  012072                JSR     R5,TIMER                ;WAIT FOR BREAK BIT
 2787 010474  001320                            TCSR
 2788 010476  000001                            BREAK
 2789 010500                                ERROR   11                      ;BREAK DID NOT SET IN TCSR
      010500  104011                        EMT 11
 2794 010502  000436                        BR      TST22           ;;EXIT TEST
 2799
 2800                                                                       ;CLEAR  BREAK BIT
 2801 010504  042777  000001  170606        BIC     #BREAK,@TCSR            ;LET @TCSR := @TCSR CLR.BY #BREAK
```

```
 2802
 2803 010512  004767  012166                    JSR     PC,WAIT
 2804
 2805 010516  017700  170574                    MOV     @RBUF,R0                ;READ RBUF TO CLEAR REC DONE
 2806                                                                            ;LET R0 := @RBUF
 2807 010522  032767  001000  170522            BIT     #BIT9,$DEVM             ;SEND CHAR
 2808 010530  001403                            BEQ     665$                    ;INTERNAL OR EXTERNAL
 2809 010532  052777  000004  170560            BIS     #BIT2,@TCSR             ;BR, IF EXTERNAL
 2810 010540  112777  000125  170554  665$:     MOVB    #125,@TBUF              ;SET MAINTENANCE BIT INT LOOP
 2811                                                                            ;LET @TBUF := #125
 2812 010546  004567  012014                    JSR     R5,TIMER                ;WAIT FOR RECV DONE
 2813 010552  001314                                RCSR
 2814 010554  000200                                DONE
 2815 010556                                    ERROR   12                      ;RECV DONE DID NOT SET IN RCSR
      010556  104012                            EMT 12
 2820 010560  000407                            BR      TST22           ;;EXIT TEST
 2825
 2826                                                                            ;WAS CHAR AFTER BREAK RECEIVED
 2827 010562  027727  170530  000125            CMP     @RBUF,#125              ;IFB @RBUF NE #125 THEN
 2828 010570  001401                            BEQ     670$                    ;BR, IF NO ERROR
 2829                                                                            ;CHAR AFTER BREAK NOT RECEIVED CORRECTLY
 2830 010572                                    ERROR 13
      010572  104013                            EMT 13
 2831 010574                            670$:                                   ;ENDIF
 2836 010574                            EX2:
      010574  000401                            BR      TST22           ;;EXIT TEST
 2841
 2842 010576  000000                    ERRCHK: .WORD 0
```

```
2844
2849 010600                          NOTST:
2850                                 ;;*****************************************************************
                                     ;*TEST 22       NOT A TEST - SEND BACK TO LOOP
                                     ;;*****************************************************************
     010600  000004                  TST22:  SCOPE
     010602  012767  000001  170350          MOV     #1,$TIMES       ;;DO 1 ITERATION
2855 010610  000240                          NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
2856
2857
2858 010612  012700  000022                  MOV     #22,R0                  ;GET TEST NUMBER
2859 010616  005100                          COM     R0                      ;1'S COMP TEST NUMBER
2860 010620  032767  000100  170424          BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
2861 010626  001002                          BNE     501$                    ;BR, IF THEY CAN'T
2862 010630  110077  170450                  MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
2863 010634                          501$:
2864 010634  032777  010000  170276          BIT     #BIT12,@SWR             ;WANT SUMMARY?
2865 010642  001002                          BNE     303$                    ;BR, IF YES
2866 010644  000167  000560                  JMP     3$                      ;JUMP IF NO
2867 010650  026727  011502  000001  303$:   CMP     PHASE2,#TRUE            ;ELSE IN PHASE 2?
2868 010656  001176                          BNE     1$                      ;BR IF NO
2869 010660  032767  100000  170364          BIT     #BIT15,$DEVM            ;IS CHANNEL 0 BEING TESTED
2870 010666  001061                          BNE     34$                     ;BR, IF IT NOT BEING TESTED
2871 010670                                  TYPTXT  <<CRLF>/                ** PHASE 2 SUMMARY **/<CRLF>>
2872 010730                                  TYPTXT  <*CSR: *>
2873 010744                                  TYPOCT  BASE0
2874 010752                                  TYPTXT  <*,VECTOR: *>
2875 010772                                  TYPOCT  VECT0
2876 011000                                  TYPTXT  <*,ERRORS: *>
2877 011020                                  TYPDEC  $ERTTL
2878 011026  104401  001171                  TYPE    ,$CRLF
2879 011032  032767  020000  170212  34$:    BIT     #BIT13,$DEVM            ;CHECK FOR 2ND MXV
2880 011040  001002                          BNE     33$                     ;BR IF PRESENT
2881 011042  000167  000362                  JMP     3$                      ;JUMP IF NOT PRESENT
2882 011046                          33$:    TYPTXT  <*CSR: *>
2883 011062                                  TYPOCT  BASE2
2884 011070                                  TYPTXT  <*,VECTOR: *>
2885 011110                                  TYPOCT  VECT2
2886 011116                                  TYPTXT  <*,ERRORS: *>
2887 011136                                  TYPDEC  $ERTTL
2888 011144  104401  001171                  TYPE    ,$CRLF
2889 011150                                  TYPTXT  <*CSR: *>
2890 011164                                  TYPOCT  BASE3
2891 011172                                  TYPTXT  <*,VECTOR: *>
2892 011212                                  TYPOCT  VECT3
2893 011220                                  TYPTXT  <*,ERRORS: *>
2894 011240                                  TYPDEC  $ERTTL
2895 011246  104401  001171                  TYPE    ,$CRLF
2896 011252  000466                          BR      3$
2897
2898 011254  005767  011100          1$:     TST     P1CNT
2899 011260  001022                          BNE     2$
2900 011262                                  TYPTXT  <<CRLF>/                ** PHASE 1 SUMMARY **/<CRLF>>
2901 011322  005267  011032                  INC     P1CNT
2902
2903 011326                          2$:     TYPTXT  <*CSR: *>
2904 011342                                  TYPOCT  DCADD
```

```
 2905 011350                                    TYPTXT  <*,VECTOR: *>
 2906 011370                                    TYPOCT  DLVEC
 2907 011376                                    TYPTXT  <*,ERRORS: *>
 2908 011416                                    TYPDEC  $ERTTL
 2909 011424  104401  001171                    TYPE    ,$CRLF
 2910
 2911 011430  005067  167456          3$:       CLR     $ERTTL          ; RESET FOR NEXT DEVICE/PASS
 2912 011434  026727  010716  000001            CMP     PHASE2,#TRUE    ;IN PHASE 2?
 2913 011442  001010                            BNE     4$              ;BR IF NO
 2914 011444  005067  010706                    CLR     PHASE2
 2915 011450  005067  167532                    CLR     $UNIT
 2916 011454  104401  001334                    TYPE    ,CARR
 2917 011460  000167  013200                    JMP     $EOP
 2918
 2919 011464  042777  000004  167626  4$:       BIC     #BIT2,@TCSR     ;ALWAYS TURN OFF INT MAINT WRAP
 2920 011472  112767  000005  167402            MOVB    #5,$TSTNM       ;FAKE OUT FOR 2'ND CHANNEL
 2921 011500  000167  172236                    JMP     LOOP            ; BACK UP TO THE BEGINNING
 2926
```

```
 2928
 2929 011504                            MODTST:
 2930
 2936                                   ;;*****************************************************************
                                        ;*TEST 23       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.
                                        ;;*****************************************************************
      011504  000004                   TST23:  SCOPE
      011506  012767  000001  167444            MOV     #1,$TIMES       ;;DO 1 ITERATION
      011514  012767  000023  167456            MOV     #23,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2941 011522  000240                            NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
 2942
 2943
 2944 011524  042777  000004  167566            BIC     #BIT2,@TCSR             ;CLEAR MAINTENANCE BIT
 2945 011532  012700  000023                    MOV     #23,R0                  ;GET TEST NUMBER
 2946 011536  005100                            COM     R0                      ;1'S COMP TEST NUMBER
 2947 011540  032767  000100  167504            BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
 2948 011546  001002                            BNE     501$                    ;BR, IF THEY CAN'T
 2949 011550  110077  167530                    MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
 2950 011554                            501$:
 2951 011554  036727  167472  100000            BIT     $DEVM,#BIT15            ;CHANNEL 0 SELECTED
 2952 011562  001406                            BEQ     504$                    ;BR, IF CHANNEL 0 SELECTED
 2953 011564  036727  167462  000400            BIT     $DEVM,#BIT8             ;CHECK FOR CHANNEL 1
 2954 011572  001002                            BNE     504$                    ;BR, IF CHANNEL 1 SELECTED
 2955 011574  000167  000756                    JMP     TST24                   ;NEITHER CHANNEL SELECTED GET OUT
 2956 011600                            504$:
 2957 011600  005067  000742                    CLR     INTABL                  ;CLEAR OUT INTERRUPT TABLE
 2958 011604  005067  000740                    CLR     INTABL+2
 2959 011610  005067  000736                    CLR     INTABL+4
 2960 011614  005067  000734                    CLR     INTABL+6
 2961
 2966 011620  012746  000340                    MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
      011624  012746  011632                    MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
      011630  000002                            RTI                     ;;POP NEW PC AND PS
      011632                            64$:
 2971
 2972 011632  032767  002000  167412            BIT     #<BIT10>,$DEVM          ;CHECK FOR TEST SKIP
 2973 011640  001406                            BEQ     691$                    ;BR, IF NOT TO BE SKIPPED
 2974 011642  032767  000010  167402            BIT     #<BIT3>,$DEVM           ;CHECK OTHER CHANNEL
 2975 011650  001002                            BNE     691$                    ;BR, IF NOT TO BE SKIPPED
 2976 011652  000167  003040                    JMP     TST25                   ;YEP SKIP THIS TEST
 2977 011656                            691$:
 2978 011656  032767  100000  167366            BIT     #<BIT15>,$DEVM          ;CH 0 DROPPED OR NO WRAP?
 2979 011664  001052                            BNE     1$                      ;BR IF EITHER SET
 2980 011666  032767  002000  167356            BIT     #<BIT10>,$DEVM          ;CH 0 DROPPED OR NO WRAP?
 2981 011674  001046                            BNE     1$                      ;BR IF EITHER SET
 2982
 2983 011676  016700  167354                    MOV     VECT0,R0                ;SETUP CH 0 VECTOR AREA
 2984 011702  012720  012450                    MOV     #CHOR,(R0)+             ;RECV
 2985 011706  012720  000340                    MOV     #PR7,(R0)+
 2986 011712  012720  012464                    MOV     #CHOX,(R0)+             ;XMIT
 2987 011716  012710  000340                    MOV     #PR7,(R0)
 2988
 2989 011722  016700  167326                    MOV     BASE0,R0                ;ENABLE INTERRUPTS
 2990 011726  052710  000100                    BIS     #IE,(R0)                ;RECV
```

```
2991 011732 062700 000004            ADD     #4,R0
2992 011736 032767 001000 167306     BIT     #BIT9,$DEVM     ;INTERNAL OR EXTERNAL
2993 011744 001402                   BEQ     680$            ;BR, IF EXTERNAL
2994 011746 052710 000004            BIS     #BIT2,(R0)      ;SET MAINTENANCE BIT INT LOOP
2995 011752 052720 000100    680$:   BIS     #IE,(R0)+       ;XMIT
2996
2997 011756 012710 000125            MOV     #125,(R0)       ;XMIT CHAR TO PRIME RECEIVERS
2998 011762 162700 000006            SUB     #6,R0           ;GO BACK TO RCSR
2999 011766 012767 100000 167332     MOV     #100000,TMP1
3000 011774 032710 000200    7$:     BIT     #DONE,(R0)      ;DONE?
3001 012000 001004                   BNE     1$              ;BR IF YES
3002 012002 005367 167320            DEC     TMP1            ;TIMED OUT?
3003 012006 001372                   BNE     7$              ;BR IF NO
3004 012010                          ERROR   231             ;NO RCVR DONE
     012010 104231                   EMT 231
3005 012012 032767 000400 167232 1$: BIT     #<BIT8>,$DEVM   ;CH 1 DROPPED OR NO WRAP?
3006
3007 012020 001456                   BEQ     2$              ;BR IF EITHER SET
3008 012022 032767 000010 167222     BIT     #<BIT3>,$DEVM   ;CH 1 DROPPED OR NO WRAP?
3009 012030 001452                   BEQ     2$              ;BR IF EITHER SET
3010 012032 026767 167222 167104     CMP     BASE1,$TKS      ;CH 1 CONSOLE?
3011 012040 001446                   BEQ     2$              ;BR IF YES
3012
3013 012042 016700 167214            MOV     VECT1,R0        ;SETUP CH 1 VECTOR AREA
3014 012046 012720 012506            MOV     #CH1R,(R0)+     ;RECV
3015 012052 012720 000340            MOV     #PR7,(R0)+
3016 012056 012720 012524            MOV     #CH1X,(R0)+     ;XMIT
3017 012062 012710 000340            MOV     #PR7,(R0)
3018
3019 012066 016700 167166            MOV     BASE1,R0        ;ENABLE INTERRUPTS
3020 012072 052710 000100            BIS     #IE,(R0)        ;RECV
3021 012076 062700 000004            ADD     #4,R0
3022 012102 032767 001000 167142     BIT     #BIT9,$DEVM     ;IS INTERNAL WRAP SELECTED
3023 012110 001402                   BEQ     690$            ;BR, IF IT IS NOT
3024 012112 052710 000004            BIS     #BIT2,(R0)      ;SET THE MAINTENANCE BIT
3025 012116                  690$:
3026 012116 052720 000100            BIS     #IE,(R0)+       ;XMIT
3027
3028 012122 012710 000125            MOV     #125,(R0)       ;XMIT CHAR TO PRIME RECEIVERS
3029 012126 162700 000006            SUB     #6,R0           ;GO BACK TO RCSR
3030 012132 012767 100000 167166     MOV     #100000,TMP1
3031 012140 032710 000200    8$:     BIT     #DONE,(R0)      ;DONE?
3032 012144 001004                   BNE     2$              ;BR IF YES
3033 012146 005367 167154            DEC     TMP1            ;TIMED OUT?
3034 012152 001372                   BNE     8$              ;BR IF NO
3035 012154                          ERROR   232             ;NO RCVR DONE
     012154 104232                   EMT 232
3036
3037                                                         ;ALL XMIT & REC INTERRUPTS SHOULD BE
3038                                                         ;IN CONTENTION.
3039                                                         ;CHAN 0 HAS PRIORITY OVER CHAN 1.
3040                                                         ;RECEIVE HAS PRIORITY OVER XMIT INTERRUPTS.
3041                                                         ;THEREFORE, ONCE CPU PRIORITY IS LOWERED,
3042                                                         ;INTERRUPTS S/B IN THE ORDER SHOWN IN THE
3043                                                         ;CHANNEL IDENTIFIER TABLE (RCH0:)
3044
3045 012156 032767 100000 167066 2$:  BIT     #<BIT15>,$DEVM  ;CH0 DROPPED OR NO WRAP?
```

```
3046 012164 001007                      BNE     5$                      ;BR IF EITHER SET
3047 012166 032767 002000 167056        BIT     #<BIT10>,$DEVM          ;CH0 DROPPED OR NO WRAP?
3048 012174 001003                      BNE     5$                      ;BR IF EITHER SET
3049
3050 012176 012704 012546               MOV     #INTABL,R4              ;SETUP FOR SERVICE ROUTINES
3051 012202 000402                      BR      6$
3052
3053 012204 012704 012552       5$:     MOV     #INTABL+4,R4
3054 012210                     6$:
3055 012210 012700 177777               MOV     #-1,R0                  ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
3056 012214 077001                      SOB     R0,.                    ;FINISH BEFORE RESET
3057 012216 012700 177777               MOV     #-1,R0                  ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
3058 012222 077001                      SOB     R0,.                    ;FINISH BEFORE RESET
3059
3064 012224 012746 000000               MOV     #PP0,-(SP)      ;;PUT NEW PS ON STACK
     012230 012746 012256               MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
     012234 000002                      RTI                     ;;POP NEW PC AND PS
     012236                     65$:
3069 012236 012700 177777               MOV     #-1,R0                  ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
3070 012242 077001                      SOB     R0,.                    ;FINISH BEFORE RESET
3071 012244 000005                      RESET                           ;DISABLE ALL INTERRUPTS
3076 012246 012746 000340               MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     012252 012746 012260               MOV     #66$,-(SP)      ;;PUT NEW PC ON  STACK
     012256 000002                      RTI                     ;;POP NEW PC AND PS
     012260                     66$:
3081
3082                                                                    ;NOW LETS SEE IF INTABL HAS ENTRIES IN THE CORRECT O
3083
3084 012260 032767 100000 166764        BIT     #<BIT15>,$DEVM          ;CH 0 DROPPED OR NO WRAP?
3085 012266 001015                      BNE     3$                      ;BR IF EITHER SET
3086 012270 032767 002000 166754        BIT     #<BIT10>,$DEVM          ;CH 0 DROPPED OR NO WRAP?
3087 012276 001011                      BNE     3$                      ;BR IF EITHER SET
3088
3089 012300 005767 000242               TST     INTABL                  ;1'ST ENTRY 0?
3090 012304 001401                      BEQ     .+4                     ;BR IF YES
3091 012306                             ERROR   233                     ;DID NOT INTERR AT ASSIGNED PRIOR
     012306 104233                      EMT 233
3092
3093 012310 026727 000234 000001        CMP     INTABL+2,#1             ;2'ND ENTRY 1?
3094 012316 001401                      BEQ     .+4                     ;BR IF YES
3095 012320                             ERROR   234
     012320 104234                      EMT 234
3096
3097 012322 032767 000400 166722 3$:    BIT     #<BIT8>,$DEVM           ;CH 1 DROPPED OR NO WRAP?
3098 012330 001422                      BEQ     4$                      ;BR IF EITHER SET
3099 012332 032767 000010 166712        BIT     #<BIT3>,$DEVM           ;CH 1 DROPPED OR NO WRAP?
3100 012340 001416                      BEQ     4$                      ;BR IF EITHER SET
3101 012342 026767 166712 166574        CMP     BASE1,$TKS              ;CH 1 CONSOLE?
3102 012350 001412                      BEQ     4$                      ;BR IF YES
3103
3104 012352 026727 000174 000002        CMP     INTABL+4,#2             ;3'RD ENTRY 2?
3105 012360 001401                      BEQ     .+4                     ;BR IF YES
3106 012362                             ERROR   235
     012362 104235                      EMT 235
3107
3108 012364 026727 000164 000003        CMP     INTABL+6,#3             ;4'TH ENTRY 3?
3109 012372 001401                      BEQ     .+4                     ;BR IF ;YES
```

```
3110 012374                                    ERROR   236
     012374  104236                            EMT 236
3111 012376  112777  000000  166716  4$:       MOVB    #0,@TBUF            ;SEND OUT DUMMY CHARACTER
3112 012404  004567  010156                    JSR     R5,TIMER            ;CALL TIMER ROUTINE
3113 012410  001314                            RCSR
3114 012412  000200                            DONE
3115 012414                                    ERROR   237                 ;DONE FAILED TO SET AFTER TRANSMIT
     012414  104237                            EMT 237
3116 012416  000401                            BR      502$                ;EXIT TEST BRANCH
3117 012420  000402                            BR      503$                ;GOOD BRANCH ETC
3118 012422  000167  177512          502$:     JMP     8$                  ;REALLY EXIT TEST
3119 012426  117700  166664          503$:     MOVB    @RBUF,R0            ;CLEAR OUT REC BUFFER
3120 012432  012737  026762  000020            MOV     #$SCOPE,@#IOTVEC    ;RESET VECTOR 20 IN CASE USED
3121 012440  012737  000340  000022            MOV     #340,@#IOTVEC+2     ;SET TO LEVEL 7 AGAIN
3122
3127 012446  000443                            BR      TST24           ;;EXIT TEST
3132
```

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

012450  005024          CH0R:   CLR     (R4)+           ;PUT IDENTIFIER IN TABLE
012452  016700  166576          MOV     BASE0,R0
012456  042710  000100          BIC     #IE,(R0)        ; 1 INTERR IS ALL WE WANT
012462  000002                  RTI

012464  012724  000001  CH0X:   MOV     #1,(R4)+        ;PUT IDENTIFIED IN TABLE
012470  016700  166560          MOV     BASE0,R0
012474  062700  000004          ADD     #4,R0
012500  042710  000100          BIC     #IE,(R0)        ; 1 INTERR IS ALL WE WANT
012504  000002                  RTI

012506  012724  000002  CH1R:   MOV     #2,(R4)+        ;PUT IDENTIFIER IN TABLE
012512  016700  166542          MOV     BASE1,R0
012516  042710  000100          BIC     #IE,(R0)        ; 1 INTERR IS ALL WE WANT
012522  000002                  RTI

012524  012724  000003  CH1X:   MOV     #3,(R4)+        ;PUT IDENTIFIED IN TABLE
012530  016700  166524          MOV     BASE1,R0
012534  062700  000004          ADD     #4,R0
012540  042710  000100          BIC     #IE,(R0)        ; 1 INTERR IS ALL WE WANT
012544  000002                  RTI


                                                        ;THIS TABLE WILL CONTAIN ENTRIES
                                                        ;REPRESENTING EACH INTR IN THE ORDER
                                                        ;THAT IT OCCURED
                                                        ; S/B IN THE FOLLOWING ORDER: 0,1,2,3

012546                  INTABL: .BLKW   4
```

```
3170
3186                                    ;;*****************************************************
                                        ;*TEST 24        TEST DATA XFERS, ALL ACTIVE LINES INTERRUPTING
                                        ;*              IN THIS WE'LL ENABLE INTERRUPTS ON ALL CHANNELS.
                                        ;*              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.
                                        ;;*****************************************************
     012556  000004                     TST24:  SCOPE
     012560  012767  000001  166372             MOV     #1,$TIMES       ;;DO 1 ITERATION
     012566  012767  000024  166404             MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3191 012574  000240                             NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
3192
3193
3194 012576  042777  000004  166514             BIC     #BIT2,@TCSR             ;CLEAR MAINTENANCE BIT
3195 012604  012700  000024                     MOV     #24,R0                  ;GET TEST NUMBER
3196 012610  005100                             COM     R0                      ;1'S COMP TEST NUMBER
3197 012612  032767  000100  166432             BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
3198 012620  001002                             BNE     50$                     ;BR, IF THEY CAN'T
3199 012622  110077  166456                     MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
3200 012626                             50$:
3201 012626  036727  166420  100000             BIT     $DEVM,#BIT15            ;CHANNEL 0 SELECTED
3202 012634  001406                             BEQ     50$                     ;BR, IF CHANNEL 0 SELECTED
3203 012636  036727  166410  000400             BIT     $DEVM,#BIT8             ;CHECK FOR CHANNEL 1
3204 012644  001002                             BNE     50$                     ;BR, IF CHANNEL 1 SELECTED
3205 012646  000167  002044                     JMP     TST25                   ;NEITHER CHANNEL SELECTED GET OUT
3206 012652                             50$:
3207 012652  032767  001000  166372             BIT     #BIT9,$DEVM             ;CHECK FOR MAINTENANCE BIT (LOOP)
3208 012660  001403                             BEQ     9$                      ;BR, IF NOT SET
3209 012662  052777  000004  166430             BIS     #BIT2,@TCSR             ;SET EXTERNAL LOOP BACK BIT
3210 012670  112777  000000  166424  9$:        MOVB    #0,@TBUF                ;SEND OUT DUMMY CHARACTER
3211 012676  004567  007664                     JSR     R5,TIMER                ;CALL TIMER ROUTINE
3212 012702  001314                             RCSR
3213 012704  000200                             DONE
3214 012706                                     ERROR   241                     ;DONE FAILED TO SET AFTER TRANSMIT
     012706  104241                             ER: 241
3215 012710  000401                             BR      50Z$                    ;EXIT TEST BRANCH
3216 012712  000402                             BR      503$                    ;GOOD BRANCH ETC
3217 012714  000167  000534             502$:   JMP     8$                      ;REALLY EXIT TEST
3218 012720  117700  166372             503$:   MOVB    @RBUF,R0                ;CLEAR OUT REC BUFFER
3219 012724  032767  002000  166320             BIT     #BIT10,$DEVM            ;CHECK FOR DATA WRAP
3220 012732  001402                             BEQ     10$                     ;BR, IF DATA WRAP SELECTED
3221 012734  000167  000514                     JMP     8$                      ;BR, IF WRAP NOT SELECTED
3222 012740  004567  007622             10$.    JSR     R5,TIMER                ;CALL TIMER ROUTINE
3223 012744  001314                             RCSR                            ;LOOK AT THE THE RECEIVER REGISTER
3224 012746  000200                             DONE                            ;CHECK FOR DONE
3225 012750  000240                             NOP
3226 012752  000240                             NOP
3227 012754  000240                             NOP
3228 012756  117700  166334                     MOVB    @RBUF,R0                ;CLEAR OUT REC BUFFER ONCE MORE
3229 012762  005067  001716                     CLR     C0XMIT                  ;1'ST WORD OF CH 0 TO XMIT
3230 012766  012767  000100  001714             MOV     #100,C1XMIT             ;1'ST WORD OF CH 1 TO XMIT
```

```
3231
3232 012774 012700 013704          MOV   #CHOTAB,R0          ;CLEAR OUT RECEIVER TABLES
3233 013000 005020                 CLR   (R0)+
3234 013002 020027 014704          CMP   R0,#STATEND         ;ALL DONE?
3235 013006 001374                 BNE   .-6                 ;BR IF NO
3236
3241 013010 012746 000340          MOV   #PR7,-(SP)     ;;PUT NEW PS ON STACK
     013014 012746 013022          MOV   #64$,-(SP)     ;;PUT NEW PC ON  STACK
     013020 000002                 RTI                  ;;POP NEW PC AND PS
     013022                  64$:
3246
3247 013022 032767 100000 166222   BIT   #<BIT15>,$DEVM       ;CH 0 DROPPED OR NO WRAP?
3248 013030 001031                 BNE   1$                  ;BR IF EITHER SET
3249 013032 032767 002000 166212   BIT   #<BIT10>,$DEVM       ;CH 0 DROPPED OR NO WRAP?
3250 013040 001025                 BNE   1$                  ;BR IF EITHER SET
3251
3252 013042 016700 166210          MOV   VECT0,R0            ;SETUP CH 0 VECTOR AREA
3253 013046 012720 013474          MOV   #R0SRV,(R0)+        ;RCVR
3254 013052 012720 000340          MOV   #PR7,(R0)+
3255 013056 012720 013534          MOV   #X0SRV,(R0)+        ;XMIT
3256 013062 012710 000340          MOV   #PR7,(R0)
3257
3258 013066 016700 166162          MOV   BASE0,R0            ;ENABLE INTERRUPTS
3259 013072 052710 000100          BIS   #IE,(R0)            ;RCVR
3260 013076 062700 000004          ADD   #4,R0
3261 013102 052710 000100          BIS   #IE,(R0)            ;XMIT
3262
3263 013106 012767 000177 001572   MOV   #177,C0END          ;8 BIT CHARACTERS
3264
3265 013114 032767 000400 166130 1$:  BIT   #<BIT8>,$DEVM       ;CH 1 DROPPED OR NO WRAP?
3266 013122 001435                 BEQ   2$                  ;BR IF EITHER SET
3267 013124 032767 000010 166120   BIT   #<BIT3>,$DEVM       ;CH 1 DROPPED OR NO WRAP?
3268 013132 001431                 BEQ   2$                  ;BR IF EITHER SET
3269 013134 026767 166120 166002   CMP   BASE1,$TKS          ;CH 1 CONSOLE?
3270 013142 001425                 BEQ   2$                  ;BR IF YES
3271
3272 013144 016700 166112          MOV   VECT1,R0            ;SETUP CH 1 VECTOR AREA
3273 013150 012720 013600          MOV   #R1SRV,(R0)+        ;RCVR
3274 013154 012720 000340          MOV   #PR7,(R0)+
3275 013160 012720 013640          MOV   #X1SRV,(R0)+        ;XMIT
3276 013164 012710 000340          MOV   #PR7,(R0)
3277
3278 013170 016700 166064          MOV   BASE1,R0            ;ENABLE INTERRUPTS
3279 013174 052710 000100          BIS   #IE,(R0)            ;RCVR
3280 013200 062700 000004          ADD   #4,R0
3281 013204 052710 000100          BIS   #IE,(R0)            ;XMIT
3282
3283
3284 013210 012767 000177 001474   MOV   #177,C1END          ;8 BIT CHARACTERS
3285 013216 012700 013704       2$:  MOV   #CHOTAB,R0          ;INIT BUFFER PTRS FOR SERV ROUTINES
3286 013222 012701 014304          MOV   #CH1TAB,R1
3287
3292 013226 012746 000000          MOV   #PR0,-(SP)     ;;PUT NEW PS ON STACK
     013232 012746 013240          MOV   #65$,-(SP)     ;;PUT NEW PC ON  STACK
     013236 000002                 RTI                  ;;POP NEW PC AND PS
     013240                  65$:
3297
```

```
3298 013240  012767  000040  001446          MOV     #32.,SPTMR              ;SET UP SPECIAL TIMER
3299 013246  012703  177777          650$:   MOV     #-1,R3                  ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
3300 013252  077301                          SOB     R3,.                    ;FINISH BEFORE RESET
3301 013254  005367  001434                  DEC     SPTMR                   ;BUMP BIG COUNTER
3302 013260  001372                          BNE     650$                    ;BR, IF MORE TIME TO GO
3303 013262  000005                          RESET                           ;DISABLE ALL INTERR
3304
3309 013264  012746  000340                  MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     013270  012746  013276                  MOV     #66$,-(SP)      ;;PUT NEW PC ON  STACK
     013274  000002                          RTI                     ;;POP NEW PC AND PS
     013276                          66$:
3314
3315                                                                 ;NOW LETS CHECK TO SEE IF TABLE(S) ARE IN CORRECT OR
3316
3317 013276  032767  100000  165746          BIT     #<BIT15>,$DEVM          ;CH 0 DROPPED OR NO WRAP
3318 013304  001025                          BNE     5$                      ;BR IF EITHER SET
3319 013306  032767  002000  165736          BIT     #<BIT10>,$DEVM          ;CH 0 DROPPED OR NO WRAP
3320 013314  001021                          BNE     5$                      ;BR IF EITHER SET
3321
3322 013316  005000                          CLR     R0                      ;EXPECTED WORD
3323 013320  012701  013704                  MOV     #CHOTAB,R1              ;TABLE PTR
3324
3325 013324  020011                  3$:     CMP     R0,(R1)                 ;EXPECT = ACTUAL?
3326 013326  001405                          BEQ     4$                      ;BR IF YES
3327
3328 013330  005711                          TST     (R1)                    ;ERROR SET?
3329 013332  100002                          BPL     .+6                     ;BR IF NO
3330 013334                                  ERROR   242                     ;ERROR FLAG AFTER XFER
     013334  104242                          EMT 242
3331 013336  000401                          BR      .+4
3332 013340                                  ERROR   243                     ;DATA COMPARE ERROR
     013340  104243                          EMT 243
3333
3334 013342  020067  001340          4$:     CMP     R0,COEND                ;ALL  DONE?
3335 013346  001404                          BEQ     5$                      ;BR IF YES
3336 013350  005200                          INC     R0
3337 013352  062701  000002                  ADD     #2,R1
3338 013356  000762                          BR      3$                      ;ELSE DO AGAIN
3339
3340 013360  032767  000400  165664  5$:     BIT     #<BIT8>,$DEVM           ;CH 1 DROPPED OR NO WRAP
3341 013366  001432                          BEQ     8$                      ;BR IF EITHER SET
3342 013370  032767  000010  165654          BIT     #<BIT3>,$DEVM           ;CH 1 DROPPED OR NO WRAP
3343 013376  001426                          BEQ     8$                      ;BR IF EITHER SET
3344 013400  026767  165654  165536          CMP     BASE1,$TKS              ;CH 1 CONSOLE?
3345 013406  001422                          BEQ     8$                      ;BR IF YES
3346
3347 013410  012700  000100                  MOV     #100,R0                 ;EXPECTED WORD
3348 013414  012701  014304                  MOV     #CH1TAB,R1              ;TABLE PTR
3349
3350 013420  020011                  6$:     CMP     R0,(R1)                 ;EXPECT = ACTUAL?
3351 013422  001405                          BEQ     7$                      ;BR IF YES
3352
3353 013424  005711                          TST     (R1)                    ;ERROR SET?
3354 013426  100002                          BPL     .+6                     ;BR IF NO
3355 013430                                  ERROR   244                     ;ERROR FLAG UP AFTER XFER
     013430  104244                          EMT 244
3356 013432  000401                          BR      .+4
```

```
3357 013434                                    ERROR   245                     ;DATA COMPARE ERROR
     013434  104245                            EMT 245
3358
3359 013436  020067  001250          7$:       CMP     R0,C1END                ;ALL DONE?
3360 013442  001404                            BEQ     8$                      ;BR IF YES
3361 013444  005200                            INC     R0
3362 013446  062701  000002                    ADD     #2,R1
3363 013452  000762                            BR      6$                      ;ELSE DO AGAIN
3364
3365 013454  012737  026762  000020  8$:       MOV     #$SCOPE,@#IOTVEC        ;RESET VECTOR 20 IN CASE USED
3366 013462  012737  000340  000022            MOV     #340,@#IOTVEC+2         ;SET TO LEVEL 7 AGAIN
3367 013470  000167  001222                    JMP     TST25                   ;BR COULDN'T MAKE IT
3368
3373                                    ;;*****************************************************************
3374                                    ;*               START OF SERVICE ROUTINES
3375                                    ;;*****************************************************************
3376
3377 013474  016702  165554          ROSRV:    MOV     BASE0,R2
3378 013500  062702  000002                    ADD     #2,R2                   ;POINT TO RBUF
3379 013504  011220                            MOV     (R2),(R0)+              ;PUT IN TABLE
3380 013506  026767  001172  001172            CMP     COXMIT,COEND            ;LAST CHAR?
3381 013514  001406                            BEQ     1$                      ;BR IF YES
3382 013516  005267  001162                    INC     COXMIT                  ;ELSE BUMP CHAR TO XMIT NEXT
3383 013522  062702  000002                    ADD     #2,R2                   ;POINT TO XCSR
3384 013526  052712  000100                    BIS     #IE,(R2)                ;RE-ENABLE
3385 013532  000002                    1$:     RTI
3386
3387 013534  016702  165514          XOSRV:    MOV     BASE0,R2
3388 013540  062702  000006                    ADD     #6,R2                   ;POINT TO XBUF
3389 013544  032767  001000  165500            BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
3390 013552  001403                            BEQ     1$                      ;BR, IF EXTERNAL
3391 013554  052777  000004  165536            BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
3392 013562  016712  001116          1$:       MOV     COXMIT,(R2)             ;SHIP WORD
3393 013566  162702  000002                    SUB     #2,R2                   ;POINT TO XCSR
3394 013572  042712  000100                    BIC     #IE,(R2)                ;DISABLE XMIT INTERR
3395                                                                            ;RE-ENABLE IN REC HANDLER
3396 013576  000002                            RTI
3397
3398
3399 013600  016702  165454          R1SRV:    MOV     BASE1,R2
3400 013604  062702  000002                    ADD     #2,R2                   ;POINT TO RBUF
3401 013610  011221                            MOV     (R2),(R1)+              ;PUT IN TABLE
3402 013612  026767  001072  001072            CMP     C1XMIT,C1END            ;LAST CHAR?
3403 013620  001406                            BEQ     1$                      ;BR IF YES
3404 013622  005267  001062                    INC     C1XMIT                  ;ELSE BUMP CHAR TO XMIT NEXT
3405 013626  062702  000002                    ADD     #2,R2                   ;POINT TO XCSR
3406 013632  052712  000100                    BIS     #IE,(R2)                ;RE-ENABLE
3407 013636  000002                    1$:     RTI
3408
3409 013640  016702  165414          X1SRV:    MOV     BASE1,R2
3410 013644  062702  000006                    ADD     #6,R2                   ;POINT TO XBUF
3411 013650  032767  001000  165374            BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
3412 013656  001403                            BEQ     1$                      ;BR, IF EXTERNAL
3413 013660  052777  000004  165432            BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
3414 013666  016712  001016          1$:       MOV     C1XMIT,(R2)             ;SHIP WORD
3415 013672  162702  000002                    SUB     #2,R2                   ;POINT TO XCSR
3416 013676  042712  000100                    BIC     #IE,(R2)                ;DISABLE XMIT INTERR
```

```
3417
3418 013702  000002                    RTI                          ;RE-ENABLE IN REC HANDLER
3419
3420
3421
3422                                ;** RECEIVER STATUS TABLES **
3423 013704                         CH0TAB: .BLKW 200               ; 0- 77
3424 014304                         CH1TAB: .BLKW 200               ;100-177
3425 014704                         STATEND:
3426
3427 014704  000000                 C0XMIT: 0                       ;START WORD FOR CH 0: 0
3428 014706  000000                 C0END:  0                       ;END   WORD FOR CH 0:137 OR 177
3429 014710  000000                 C1XMIT: 0                       ;START WORD FOR CH 1: 100
3430 014712  000000                 C1END:  0                       ;END   WORD FOR CH 1: 137 OR 177
3431 014714  000000                 SPTMR:  0                       ;SPECIAL TIMER AREA
3432
```

```
3439                                    ;;*****************************************************************
                                        ;*TEST 25       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.
                                        ;*****************************************************************
     014716  000004                     TST25:  SCOPE
     014720  012767  000001  164232             MOV     #1,$TIMES       ;;DO 1 ITERATION
     014726  012767  000025  164244             MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3444 014734  000240                             NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
3445
3446
3447 014736  042777  000004  164354             BIC     #BIT2,@TCSR             ;CLEAR MAINTENANCE BIT
3448 014744  012700  000025                     MOV     #25,R0                  ;GET TEST NUMBER
3449 014750  005100                             COM     R0                      ;1'S COMP TEST NUMBER
3450 014752  032767  000100  164272             BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
3451 014760  001002                             BNE     501$                    ;BR, IF THEY CAN'T
3452 014762  110077  164316                     MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
3453 014766                             501$:
3454 014766  032767  020000  164256             BIT     #BIT13,$DEVM            ;SECOND MXV11-B SELECTED?
3455 014774  001002                             BNE     522$                    ;BR, IF SELECTED IN DEVM
3456 014776  000167  000654                     JMP     TST26                   ;EXIT TEST
3457 015002                             522$:
3458 015002  032767  002000  164242             BIT     #BIT10,$DEVM            ;CHECK FOR DATA WRAP
3459 015010  001402                             BEQ     10$                     ;BR, IF DATA WRAP SELECTED
3460 015012  000167  000640                     JMP     TST26                   ;BR, IF WRAP NOT SELECTED
3461 015016  005067  000624             10$:    CLR     INTABM                  ;CLEAR OUT INTERRUPT TABLE
3462 015022  005067  000622                     CLR     INTABM+2
3463 015026  005067  000620                     CLR     INTABM+4
3464 015032  005067  000616                     CLR     INTABM+6
3465
3470 015036  012746  000340                     MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     015042  012746  015050                     MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
     015046  000002                             RTI                     ;;POP NEW PC AND PS
     015050                             64$:
3475
3476
3477 015050  016700  164212                     MOV     VECT2,R0                ;SETUP CH 0 VECTOR AREA
3478 015054  016767  164204  164230             MOV     BASE2,DLADD             ;SET ERROR CALL UP
3479 015062  016767  164200  164234             MOV     VECT2,DLVEC             ;SET ERROR CALL UP
3480 015070  012720  015540                     MOV     #CH2R,(R0)+             ;RECV
3481 015074  012720  000340                     MOV     #PR7,(R0)+
3482 015100  012720  015560                     MOV     #CH2X,(R0)+             ;XMIT
3483 015104  012710  000340                     MOV     #PR7,(R0)
3484
3485 015110  016700  164150                     MOV     BASE2,R0                ;ENABLE INTERRUPTS
3486 015114  052710  000100                     BIS     #IE,(R0)                ;RECV
3487 015120  062700  000004                     ADD     #4,R0
3488 015124  032767  001000  164120             BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
3489 015132  001402                             BEQ     680$                    ;BR, IF EXTERNAL
3490 015134  052710  000004                     BIS     #BIT2,(R0)              ;SET MAINTENANCE BIT INT LOOP
3491 015140  052720  000100             680$:   BIS     #IE,(R0)+               ;XMIT
3492
3493 015144  012710  000125                     MOV     #125,(R0)               ;XMIT CHAR TO PRIME RECEIVERS
3494 015150  162700  000006                     SUB     #6,R0                   ;GO BACK TO RCSR
3495 015154  012767  100000  164144             MOV     #100000,TMP1
3496 015162  032710  000200             7$:     BIT     #DONE,(R0)              ;DONE?
```

```
3497 015166  001004                        BNE     1$              ;BR IF YES
3498 015170  005367  164132                DEC     TMP1            ;TIMED OUT?
3499 015174  001372                        BNE     7$              ;BR IF NO
3500 015176                                ERROR   251             ;NO RCVR DONE
     015176  104251                        EMT 251
3501
3502 015200  016700  164066          1$:   MOV     VECT3,R0        ;SETUP CH 1 VECTOR AREA
3503 015204  016767  164060  164100        MOV     BASE3,DLADD     ;SET ERROR CALL UP
3504 015212  016767  164054  164104        MOV     VECT3,DLVEC     ;SET ERROR CALL UP
3505 015220  012720  015602                MOV     #CH3R,(R0)+     ;RECV
3506 015224  012720  000340                MOV     #PR7,(R0)+
3507 015230  012720  015624                MOV     #CH3X,(R0)+     ;XMIT
3508 015234  012710  000340                MOV     #PR7,(R0)
3509
3510 015240  016700  164024                MOV     BASE3,R0        ;ENABLE INTERRUPTS
3511 015244  052710  000100                BIS     #IE,(R0)        ;RECV
3512 015250  062700  000004                ADD     #4,R0
3513 015254  032767  001000  163770        BIT     #BIT9,$DEVM     ;INTERNAL OR EXTERNAL
3514 015262  001402                        BEQ     69$             ;BR, IF EXTERNAL
3515 015264  052710  000004                BIS     #BIT2,(R0)      ;SET MAINTENANCE BIT INT LOOP
3516 015270  052720  000100          69$:  BIS     #IE,(R0)+       ;XMIT
3517
3518 015274  012710  000125                MOV     #125,(R0)       ;XMIT CHAR TO PRIME RECEIVERS
3519 015300  162700  000006                SUB     #6,R0           ;GO BACK TO RCSR
3520 015304  012767  100000  164014        MOV     #100000,TMP1
3521 015312  032710  000200          8$:   BIT     #DONE,(R0)      ;DONE?
3522 015316  001004                        BNE     2$              ;BR IF YES
3523 015320  005367  164002                DEC     TMP1            ;TIMED OUT?
3524 015324  001372                        BNE     8$              ;BR IF NO
3525 015326                                ERROR   252             ;NO RCVR DONE
     015326  104252                        EMT 252
3526
3527                                                               ;ALL XMIT & REC INTERRUPTS SHOULD BE
3528                                                               ;IN CONTENTION.
3529                                                               ;CHAN 0 HAS PRIORITY OVER CHAN 1.
3530                                                               ;RECEIVE HAS PRIORITY OVER XMIT INTERRUPTS.
3531                                                               ;THEREFORE, ONCE CPU PRIORITY IS LOWERED,
3532                                                               ;INTERRUPTS S/B IN THE ORDER SHOWN IN THE
3533                                                               ;CHANNEL IDENTIFIER TABLE (RCH2:)
3534
3535 015330  012704  015646          2$:   MOV     #INTABM,R4      ;SETUP FOR SERVICE ROUTINES
3536 015334  000402                        BR      6$
3537
3538 015336  012704  015652          5$:   MOV     #INTABM+4,R4
3539
3540 015342  012700  177777          6$:   MOV     #-1,R0          ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
3541 015346  077001                        SOB     R0,.            ;FINISH BEFORE RESET
3542 015350  012700  177777                MOV     #-1,R0          ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
3543 015354  077001                        SOB     R0,.            ;FINISH BEFORE RESET
3544
3549 015356  012746  000000                MOV     #PR0,-(SP)      ;;PUT NEW PS ON STACK
     015362  012746  015370                MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
     015366  000002                        RTI                     ;;POP NEW PC AND PS
     015370                        65$:
3554 015370  012700  177777                MOV     #-1,R0          ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
3555 015374  077001                        SOB     R0,.            ;FINISH BEFORE RESET
3556 015376  000005                        RESET                   ;DISABLE ALL INTERRUPTS
```

```
3561 015400  012746  000340                MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     015404  012746  015412                MOV     #66$,-(SP)      ;;PUT NEW PC ON  STACK
     015410  000002                        RTI                     ;;POP NEW PC AND PS
     015412                        66$:
3566
3567                                                                 ;NOW LETS SEE IF INTABM HAS ENTRIES IN THE CORRECT O
3568
3569 015412  016767  163646  163672        MOV     BASE2,DLADD     ;SET ERROR CALL UP
3570 015420  016767  163642  163676        MOV     VECT2,DLVEC     ;SET ERROR CALL UP
3571
3572 015426  005767  000214                TST     INTABM          ;1'ST ENTRY 0?
3573 015432  001401                        BEQ     .+4             ;BR IF YES
3574 015434                                ERROR   253             ;DID NOT INTERR AT ASSIGNED PRIOR
     015434  104253                        EMT 253
3575
3576 015436  026727  000206  000001        CMP     INTABM+2,#1     ;2'ND ENTRY 1?
3577 015444  001401                        BEQ     .+4             ;BR IF YES
3578 015446                                ERROR   254
     015446  104254                        EMT 254
3579
3580 015450                        38:
3581 015450  016767  163614  163634        MOV     BASE3,DLADD     ;SET ERROR CALL UP
3582 015456  016767  163610  163640        MOV     VECT3,DLVEC     ;SET ERROR CALL UP
3583
3584 015464  026727  000162  000002        CMP     INTABM+4,#2     ;3'RD ENTRY 2?
3585 015472  001401                        BEQ     .+4             ;BR IF YES
3586 015474                                ERROR   255
     015474  104255                        EMT 255
3587
3588 015476  026727  000152  000003        CMP     INTABM+6,#3     ;4'TH ENTRY 3?
3589 015504  001401                        BEQ     .+4             ;BR IF ;YES
3590 015506                                ERROR   256
     015506  104256                        EMT 256
3591 015510                        48:
3592 015510  000402                        BR      503$            ;GOOD BRANCH ETC
3593 015512  000167  177574        502$:   JMP     8$              ;REALLY EXIT TEST
3594 015516  117700  163574        503$:   MOVB    @RBUF,R0        ;CLEAR OUT REC BUFFER
3595 015522  012737  026762  000020        MOV     #$SCOPE,@#IOTVEC ;RESET VECTOR 20 IN CASE USED
3596 015530  012737  000340  000022        MOV     #340,@#IOTVEC+2 ;SET TO LEVEL 7 AGAIN
3597
3602 015536  000447                        BR      TST26           ;;EXIT TEST
3607
3608
3609                               ;*****************************************
3610                               ;*              START OF SERVICE ROUTINES
3611                               ;*****************************************
3612
3613 015540  005024                CH2R:   CLR     (R4)+           ;PUT IDENTIFIER IN TABLE
3614 015542  016700  163516                MOV     BASE2,R0
3615 015546  042710  000100                BIC     #IE,(R0)        ; 1 INTERR IS ALL WE WANT
3616 015552  116701  163510                MOVB    BASE2+2,R1      ;CLEAN OUT THE REC BUFFER
3617 015556  000002                        RTI
3618
3619 015560  012724  000001        CH2X:   MOV     #1,(R4)+        ;PUT IDENTIFIED IN TABLE
3620 015564  016700  163474                MOV     BASE2,R0
3621 015570  062700  000004                ADD     #4,R0
3622 015574  042710  000100                BIC     #IE,(R0)        ; 1 INTERR IS ALL WE WANT
```

```
3623 015600  000002                          RTI
3624
3625 015602  012724  000002        CH3R:     MOV     #2,(R4)+          ;PUT IDENTIFIER IN TABLE
3626 015606  016700  163456                  MOV     BASE3,R0
3627 015612  042710  000100                  BIC     #1E,(R0)          ; 1 INTERR IS ALL WE WANT
3628 015616  116701  163450                  MOVB    BASE3+2,R1        ;CLEAN OUT THE REC BUFFER
3629 015622  000002                          RTI
3630
3631 015624  012724  000003        CH3X:     MOV     #3,(R4)+          ;PUT IDENTIFIED IN TABLE
3632 015630  016700  163434                  MOV     BASE3,R0
3633 015634  062700  000004                  ADD     #4,R0
3634 015640  042710  000100                  BIC     #1E,(R0)          ; 1 INTERR IS ALL WE WANT
3635 015644  000002                          RTI
3636
3637
3638                                                                    ;THIS TABLE WILL CONTAIN ENTRIES
3639                                                                    ;REPRESENTING EACH INTR IN THE ORDER
3640                                                                    ;THAT IT OCCURED
3641                                                                    ; S/B IN THE FOLLOWING ORDER: 0,1,2,3
3642
3643 015646                  INTABM: .BLKW   4
3644
3645
```

```
3662                                    ;;*****************************************************************
                                        ;*TEST 26       TEST DATA XFERS, ALL ACTIVE LINES INTERRUPTING
                                        ;*             IN THIS WE'LL ENABLE INTERRUPTS ON ALL CHANNELS.
                                        ;*             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.
                                        ;*****************************************************************
 015656  000004                  TST26:  SCOPE
 015660  012767  000001  163272          MOV     #1,$TIMES       ;;DO 1 ITERATION
 015666  012767  000026  163304          MOV     #26,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3667 015674  000240                     NOP                             ;SPECIAL DEBUG FEATURE (GOOD HALT LOC)
3668
3669
3670 015676  042777  000004  163414      BIC     #BIT2,@TCSR             ;CLEAR MAINTENANCE BIT
3671 015704  012700  000026              MOV     #26,R0                  ;GET 1ST NUMBER
3672 015710  005100                      COM     R0                      ;1'S COMP TEST NUMBER
3673 015712  032767  000100  163332      BIT     #BIT6,$DEVM             ;CAN THE LEDS BE USED
3674 015720  001002                      BNE     5018                    ;BR, IF THEY CAN'T
3675 015722  110077  163356              MOVB    R0,@LEDREG              ;DISPLAY TEST NUMBER IN LEDS
3676 015726                      5018:
3677 015726  032767  020000  163316      BIT     #BIT13,$DEVM            ;SECOND MXV11-B SELECTED
3678 015734  001002                      BNE     5028                    ;BR, IF SELECTED IN DEVM
3679 015736  000167  000612              JMP     8$                      ;EXIT TEST (NOT SELECTED)
3680 015742                      5028:
3681 015742  032767  002000  163302      BIT     #BIT10,$DEVM            ;CHECK FOR DATA WRAP
3682 015750  001402                      BEQ     5038                    ;BR, IF DATA WRAP SELECTED
3683 015752  000157  000576              JMP     8$                      ;BR, IF WRAP NOT SELECTED
3684 015756  032767  001000  163266  5038: BIT     #BIT9,$DEVM             ;CHECK FOR MAINTENANCE BIT (LOOP)
3685 015764  001414                      BEQ     9$                      ;BR, IF NOT SET
3686 015766  016700  163272              MOV     BASE2,R0                ;GET ADDRESS OF 2ND MX'S 1ST LINE
3687 015772  062700  000004              ADD     #4,R0                   ;POINT AT THE TRANSMITTER'S CSR
3688 015776  052710  000004              BIS     #BIT2,(R0)              ;SET INTERNAL LOOP BACK BIT
3689 016002  016700  163262              MOV     BASE3,R0                ;GET ADDRESS OF 2ND MX'S 2ND LINE
3690 016006  062700  000004              ADD     #4,R0                   ;POINT AT THE TRANSMITTER'S CSR
3691 016012  052710  000004              BIS     #BIT2,(R0)              ;SET INTERNAL LOOP BACK BIT
3692 016016  016767  163242  163272  9$:  MOV     BASE2,RBUF              ;GET BASE ADDRESS
3693 016024  062767  000002  163264      ADD     #2,RBUF                 ;NOW POINTS TO REC BUFFER
3694 016032  117701  163260      11$:    MOVB    @RBUF,R1                ;READ REC BUFFER
3695 016036  004567  004524              JSR     R5,TIMER                ;CALL TIMER ROUTINE
3696 016042  001264                      BASE2                           ;ADDRESS OF CSR
3697 016044  000200                      DONE                            ;DONE CHECK BITS
3698 016046  000401                      BR      12$                     ;BR  IF DONE IS NOT SET
3699 016050  000770                      BR      11$                     ;BR, TO TRY AND CLEAR REC DONE
3700 016052                      12$:
3701 016052  016767  163212  163236  19$:  MOV     BASE3,RBUF              ;GET BASE ADDRESS
3702 016060  062767  000002  163230      ADD     #2,RBUF                 ;NOW POINTS TO REC BUFFER
3703 016066  117701  163224      111$:   MOVB    @RBUF,R1                ;READ REC BUFFER
3704 016072  004567  004470              JSR     R5,TIMER                ;CALL TIMER ROUTINE
3705 016076  001270                      BASE3                           ;ADDRESS OF CSR
3706 016100  000200                      DONE                            ;DONE CHECK BITS
3707 016102  000401                      BR      112$                    ;BR, IF DONE IS NOT SET
3708 016104  000770                      BR      111$                    ;BR, TO TRY AND CLEAR REC DONE
```

```
3709 016106                                  112$:
3710 016106  032767  002000  163136                BIT     #BIT10,$DEVM            ;CHECK FOR DATA WRAP
3711 016114  001402                                BEQ     10$                     ;BR, IF DATA WRAP SELECTED
3712 016116  000167  000432                        JMP     8$                      ;BR, IF WRAP NOT SELECTED
3713 016122  005067  001656              10$:      CLR     C2XMIT                  ;1'ST WORD OF CH 0 TO XMIT
3714 016126  012767  000100  001654                MOV     #100,C3XMIT             ;1'ST WORD OF CH 1 TO XMIT
3715
3716 016134  012700  017004                        MOV     #CH2TAB,R0              ;CLEAR OUT RECEIVER TABLES
3717 016140  005020                                CLR     (R0)+
3718 016142  020027  020004                        CMP     R0,#STTEND              ;ALL DONE?
3719 016146  001374                                BNE     .-6                     ;BR IF NO
3720
3725 016150  012746  000340                        MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     016154  012746  016162                        MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
     016160  000002                                RTI                     ;;POP NEW PC AND PS
     016162                              64$:
3730
3731 016162  016700  163100                        MOV     VECT2,R0                ;SETUP CH 0 VECTOR AREA
3732 016166  016767  163072  163116                MOV     BASE2,DLADD             ;SET ERROR CALL UP
3733 016174  016767  163066  163122                MOV     VECT2,DLVEC             ;SET ERROR CALL UP
3734 016202  012720  016574                        MOV     #R2SRV,(R0)+            ;RCVR
3735 016206  012720  000340                        MOV     #PR7,(R0)+
3736 016212  012720  016634                        MOV     #X2SRV,(R0)+            ;XMIT
3737 016216  012710  000340                        MOV     #PR7,(R0)
3738
3739 016222  016700  163036                        MOV     BASE2,R0                ;ENABLE INTERRUPTS
3740 016226  052710  000100                        BIS     #IE,(R0)                ;RCVR
3741 016232  062700  000004                        ADD     #4,R0
3742 016236  052710  000100                        BIS     #IE,(R0)                ;XMIT
3743
3744 016242  012767  000177  001536                MOV     #177,C2END              ;8 BIT CHARACTER
3745
3746 016250  016700  163016              1$:       MOV     VECT3,R0                ;SETUP CH 1 VECTOR AREA
3747 016254  016767  163010  163030                MOV     BASE3,DLADD             ;SET ERROR CALL UP
3748 016262  016767  163004  163034                MOV     VECT3,DLVEC             ;SET ERROR CALL UP
3749 016270  012720  015700                        MOV     #R3SRV,(R0)+            ;RCVR
3750 016274  012720  000340                        MOV     #PR7,(R0)+
3751 016300  012720  016740                        MOV     #X3SRV,(R0)+            ;XMIT
3752 016304  012710  000340                        MOV     #PR7,(R0)
3753
3754 016310  016700  162754                        MOV     BASE3,R0                ;ENABLE INTERRUPTS
3755 016314  052710  000100                        BIS     #IE,(R0)                ;RCVR
3756 016320  062700  000004                        ADD     #4,R0
3757 016324  052710  000100                        BIS     #IE,(R0)                ;XMIT
3758
3759
3760 016330  012767  000177  001454                MOV     #177,C3END              ;8 BIT CHARACTERS
3761 016336  012700  017004              2$:       MOV     #CH2TAB,R0              ;INIT BUFFER PTRS FOR SERV ROUTINES
3762 016342  012701  017404                        MOV     #CH3TAB,R1
3763
3768 016346  012746  000000                        MOV     #PR0,-(SP)      ;;PUT NEW PS ON STACK
     016352  012746  016360                        MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
     016356  000002                                RTI                     ;;POP NEW PC AND PS
     016360                              65$:
3773
3774 016360  012767  000040  176326                MOV     #32.,SPTMR              ;SET UP SPECIAL TIMER
3775 016366  012703  177777              650$:     MOV     #-1,R3                  ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
```

```
3776 016372  077301                    SOB     R3,.                    ;FINISH BEFORE RESET
3777 016374  005367  176314            DEC     SPTMR                   ;BUMP BIG COUNTER
3778 016400  001372                    BNE     65O$                    ;BR, IF MORE TIME TO GO
3779 016402  000005                    RESET                           ;DISABLE ALL INTERR
3780
3785 016404  012746  000340            MOV     #PR7,-(SP)      ;;PUT NEW PS ON STACK
     016410  012746  016416            MOV     #66$,-(SP)      ;;PUT NEW PC ON  STACK
     016414  000002                    RTI                     ;;POP NEW PC AND PS
     016416                    66$:
3790
3791                                                            ;NOW LETS CHECK TO SEE IF TABLE(S) ARE IN CORRECT OR
3792
3793
3794 016416  005000                    CLR     R0                      ;EXPECTED WORD
3795 016420  012701  017004            MOV     #CH2TAB,R1              ;TABLE PTR
3796 016424  016767  162634  162660    MOV     BASE2,DLADD             ;SET ERROR CALL UP
3797 016432  016767  162630  162664    MOV     VECT2,DLVEC             ;SET ERROR CALL UP
3798
3799 016440  020011                3$:  CMP     R0,(R1)                 ;EXPECT = ACTUAL?
3800 016442  001405                    BEQ     4$                      ;BR IF YES
3801
3802 016444  005711                    TST     (R1)                    ;ERROR SET?
3803 016446  100002                    BPL     .+6                     ;BR IF NO
3804 016450                            ERROR   261                     ;ERROR FLAG AFTER XFER
     016450  104261                    EMT 261
3805 016452  000401                    BR      .+4
3806 016454                            ERROR   262                     ;DATA COMPARE ERROR
     016454  104262                    EMT 262
3807
3808 016456  020067  001374        4$: CMP     R0,C2END                ;ALL  DONE?
3809 016462  001404                    BEQ     5$                      ;BR IF YES
3810 016464  005200                    INC     R0
3811 016466  062701  000002            ADD     #2,R1
3812 016472  000762                    BR      3$                      ;ELSE DO AGAIN
3813
3814 016474  012700  000100        5$: MOV     #100,R0                 ;EXPECTED WORD
3815 016500  012701  017404            MOV     #CH3TAB,R1              ;TABLE PTR
3816 016504  016767  162560  162600    MOV     BASE3,DLADD             ;SET ERROR CALL UP
3817 016512  016767  162554  162604    MOV     VECT3,DLVEC             ;SET ERROR CALL UP
3818
3819 016520  020011                6$:  CMP     R0,(R1)                 ;EXPECT = ACTUAL?
3820 016522  001405                    BEQ     7$                      ;BR IF YES
3821
3822 016524  005711                    TST     (R1)                    ;ERROR SET?
3823 016526  100002                    BPL     .+6                     ;BR IF NO
3824 016530                            ERROR   263                     ;ERROR FLAG UP AFTER XFER
     016530  104263                    EMT 263
3825 016532  000401                    BR      .+4
3826 016534                            ERROR   264                     ;DATA COMPARE ERROR
     016534  104264                    EMT 264
3827
3828 016536  020067  001250        7$: CMP     R0,C3END                ;ALL DONE?
3829 016542  001404                    BEQ     8$                      ;BR IF YES
3830 016544  005200                    INC     R0
3831 016546  062701  000002            ADD     #2,R1
3832 016552  000762                    BR      6$                      ;ELSE DO AGAIN
3835
```

```
3834 016554  012737  026762  000020  8$:     MOV     #$SCOPE,@#IOTVEC        ;RESET VECTOR 20 IN CASE USED
3835 016562  012737  000340  000022          MOV     #340,@#IOTVEC+2         ;SET TO LEVEL 7 AGAIN
3836 016570  000167  172004                  JMP     NOTST
3837
3842                                 ;;*************************************************** ****************
3843                                 ;*              START OF SERVICE ROUTINES
3844                                 ;;*************************************************************
3845
3846 016574  016702  162464          R2SRV:  MOV     BASE2,R2
3847 016600  062702  000002                  ADD     #2,R2                   ;POINT TO RBUF
3848 016604  011220                          MOV     (R2),(R0)+              ;PUT IN TABLE
3849 016606  026767  001172  001172          CMP     C2XMIT,C2END            ;LAST CHAR?
3850 016614  001406                          BEQ     1$                      ;BR IF YES
3851 016616  005267  001162                  INC     C2XMIT                  ;ELSE BUMP CHAR TO XMIT NEXT
3852 016622  062702  000002                  ADD     #2,R2                   ;POINT TC XCSR
3853 016626  052712  000100                  BIS     #IE,(R2)                ;RE-ENABLE
3854 016632  000006                  1$:     RTI
3855
3856 016634  016702  002424          X2SRV:  MOV     BASE2,R2
3857 016640  062702  000006                  ADD     #6,R2                   ;POINT TO XBUF
3858 016644  032767  001000  162400          BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
3859 016652  001403                          BEQ     1$                      ;BR, IF EXTERNAL
3860 016654  052777  000004  162436          BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
3861 016662  016712  001116          1$:     MOV     C2XMIT,(R2)             ;SHIP WORD
3862 016666  162702  000002                  SUB     #2,R2                   ;POINT TO XCSR
3863 016672  042712  000100                  BIC     #IE,(R2)                ;DISABLE XMIT INTERR
3864                                                                         ;RE-ENABLE IN REC HANDLER
3865 016676  000002                          RTI
3866
3867
3868 016700  016702  162364          R3SRV:  MOV     BASE3,R2
3869 016704  062702  000002                  ADD     #2,R2                   ;POINT TO RBUF
3870 016710  011221                          MOV     (R2),(R1)+              ;PUT IN TABLE
3871 016712  026767  001072  001072          CMP     C3XMIT,C3END            ;LAST CHAR?
3872 016720  001406                          BEQ     1$                      ;BR IF YES
3873 016722  005267  001062                  INC     C3XMIT                  ;ELSE BUMP CHAR TO XMIT NEXT
3874 016726  062702  000002                  ADD     #2,R2                   ;POINT TO XCSR
3875 016732  052712  000100                  BIS     #IE,(R2)                ;RE-ENABLE
3876 016736  000002                  1$:     RTI
3877
3878 016740  016702  162324          X3SRV:  MOV     BASE3,R2
3879 016744  062702  000006                  ADD     #6,R2                   ;POINT TO XBUF
3880 016750  032767  001000  162274          BIT     #BIT9,$DEVM             ;INTERNAL OR EXTERNAL
3881 016756  001403                          BEQ     1$                      ;BR, IF EXTERNAL
3882 016760  052777  000004  162332          BIS     #BIT2,@TCSR             ;SET MAINTENANCE BIT INT LOOP
3883 016766  016712  001016          1$:     MOV     C3XMIT,(R2)             ;SHIP WORD
3884 016772  162702  000002                  SUB     #2 R2                   ;POINT TO XCSR
3885 016776  042712  000100                  BIC     #IE,(R2)                ;DISABLE XMIT INTERR
3886                                                                         ;RE-ENABLE IN REC HANDLER
3887 017002  000002                          RTI
3888
```

```
                                ;** RECEIVER STATUS TABLES **

                                                                ;8 BIT WDS
017004                          CH2TAB: .BLKW 200               ; 0- 77
017404                          CH3TAB: .BLKW 200               ;100-177
020004                          STTEND:

020004  000000                  C2XMIT: 0                       ;START WORD FOR CH 0: 0
020006  000000                  C2END:  0                       ;END   WORD FOR CH 0: 137 OR 177
020010  000000                  C3XMIT: 0                       ;START WORD FOR CH 1: 100
020012  000000                  C3END:  0                       ;END   WORD FOR CH 1: 137 OR 177
```

```
                                        .SBTTL  ROUTINE TO SIZE THE DEVICE MAP ($DEVM) & MEMORY

020014  032767  100000  161230  SIZE:   BIT     #BIT15,$DEVM            ;TEST CH 0?
020022  001424                          BEQ     1$                      ;BR IF YES
020024  104401  023116                  TYPE    ,CODROP
020030  032767  000400  161214          BIT     #BIT8,$DEVM             ;TEST CH 1?
020036  001055                          BNE     2$                      ;BR IF YES
020040  104401  023146                  TYPE    ,C1DROP
020044  032767  020000  161200          BIT     #BIT13,$DEVM            ;TWO MXV11-B BIT
020052  001016                          BNE     111$                    ;BR, IF THERE ARE TWO (TO BE TESTED)
020054  104401  024226                  TYPE    ,TWMXN                  ;"SECOND MXV11-B TEST BYPASSED"
020060  012767  000002  002274          MOV     #2,HICHAN               ;SET TO TWO POSSIBLE CHANNELS
020066  005267  001322                  INC     NOCHAN                  ;SET FLAG
020072  000471                          BR      4$

020074  032767  000400  161150  1$:     BIT     #BIT8,$DEVM             ;TEST CH 1?
020102  001033                          BNE     2$                      ;BR IF YES
020104  104401  023146                  TYPE    ,C1DROP
020110  032767  020000  161134  111$:   BIT     #BIT13,$DEVM            ;TWO MXV11-B BIT
020116  001025                          BNE     2$                      ;BR, IF THERE ARE TWO (TO BE TESTED)
020120  104401  024226                  TYPE    ,TWMXN                  ;"SECOND MXV11-B TEST BYPASSED"
020124  012767  000002  002230          MOV     #2,HICHAN               ;SET TO TWO POSSIBLE CHANNELS
020132  026767  161122  161004          CMP     BASE1,$TKS              ;CONSOLE?
020140  001103                          BNE     41$                     ;NO, KEEP GOING
020142  104401  024115                  TYPE    ,C1CON
020146  032767  000400  161076          BIT     #BIT8,$DEVM             ;CHECK FOR CHANNEL 1 USE
020154  001405                          BEQ     401$                    ;BR, IF NOT BEING TESTED
020156  104401  023227                  TYPE    ,NOCH1                  ;CAN'T TEST CH1 WHEN CONSOLE
020162  104412                          GTDEVM                          ;NEW DEVM
020164  000167  161224                  JMP     START                   ;REALLY START OVER
020170                          401$:

020170  000425                          BR      3$

020172  026767  161062  160744  2$:     CMP     BASE1,$YKS              ;IS IT CONSOLE?
020200  001021                          BNE     3$                      ;BR IF NO
020202  104401  024115                  TYPE    ,C1CON
020206  032767  000400  161036          BIT     #BIT8,$DEVM             ;CHECK FOR CHANNEL 1 USE
020214  001405                          BEQ     402$                    ;BR, IF NOT BEING TESTED
020216  104401  023227                  TYPE    ,NOCH1                  ;CAN'T TEST CH1 WHEN CONSOLE
020222  104412                          GTDEVM                          ;NEW DEVM
020224  000167  161164                  JMP     START                   ;REALLY START OVER
020230                          402$:
020230  032767  020000  161014          BIT     #BIT13,$DEVM            ;TWO MXV11-B BIT
020236  001002                          BNE     3$                      ;BR, IF THERE ARE TWO (TO BE TESTED)
020240  104401  024226                  TYPE    ,TWMXN                  ;"SECOND MXV11-B TEST BYPASSED"
020244  012767  000002  002110  3$:     MOV     #2,HICHAN               ;HIGHEST CHANNEL NUMBER
020252  005067  001136                  CLR     NOCHAN

020256  032767  020000  160766  4$:     BIT     #BIT13,$DEVM            ;CHECK TWO MXV11-B BIT
020264  001431                          BEQ     41$                     ;BR, NO SECOND MX TESTING
020266  104401  024265                  TYPE    ,TWOMX                  ;"TESTING SECOND MXV11-B"
020272  013746  000004                  MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
020276  012737  020312  000004          MOV     #100$,@#ERRVEC          ;NEW GUY IN PLACE
020304  005777  160754                  TST     @BASE2                  ;TRY READ
```

```
                                                BR      120$                    ;NO PROBLEM KEEP GOING
                                        100$:   MOV     #105$,(SP)              ;HAD PROBLEM
                                                RTI                             ;RETURN
                                        105$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
                                                TYPE    ,NOSMX                  ;SECOND MXV11-B WAS SELECTED NOT PRESENT
                                                GTDEVM                          ;TRY AGAIN
                                                JMP     START                   ;REALLY START OVER
                                        120$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
                                                MOV     #4,HICHAN               ;HIGHEST CHANNEL NUMBER
                                        41$:
                                                BIT     #BIT2,$DEVM             ;DO RAM TEST?
                                                BEQ     10$                     ;BR IF YES
                                                TYPE    ,NORAM

                                        10$:    BIT     #BIT1,$DEVM             ;DO ROM TEST?
                                                BNE     5$                      ;BR IF YES
                                                TYPE    ,NOROM

                                        5$:     BIT     #BIT0,$DEVM             ;CLOCK DISABLED?
                                                BEQ     50$                     ;BR IF YES
                                                TYPE    ,CLKEN
                                                MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
                                                MOV     #200$,@#ERRVEC          ;NEW GUY IN PLACE
                                                TST     @LKSREG                 ;TRY READ
                                                BR      220$                    ;NO PROBLEM KEEP GOING
                                        200$:   MOV     #205$,(SP)              ;HAD PROBLEM
                                                RTI                             ;RETURN
                                        205$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
                                                TYPE    ,NOCLK                  ;SECOND MXV11-B WAS SELECTED NOT PRESENT
                                                GTDEVM                          ;TRY AGAIN
                                                JMP     START                   ;REALLY START OVER
                                        220$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR

                                                BR      51$                     ;SKIP OVER MESSAGE
                                        50$:    TYPE    ,CLKDS                  ;CLOCK DISABLED MESSAGE

                                        51$:    BIT     #BIT7,$DEVM             ;PCR REGISTER TESTING?
                                                BEQ     52$                     ;BR IF YES
                                                TYPE    ,NOPCR                  ;"PCR REGISTER TEST BYPASSED"
                                                BR      53$                     ;SKIP OVER CHECK AND MESSAGE
                                        52$:
                                                MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
                                                MOV     #500$,@#ERRVEC          ;NEW GUY IN PLACE
                                                TST     @PCRREG                 ;TRY READ
                                                BR      520$                    ;NO PROBLEM KEEP GOING
                                        500$:   MOV     #505$,(SP)              ;HAD PROBLEM
                                                RTI                             ;RETURN
                                        505$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
                                                TYPE    ,NOPCR2                 ;PCR REGISTER NOT PRESENT
                                                GTDEVM                          ;TRY AGAIN
                                                JMP     START                   ;REALLY START OVER
                                        520$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR

                                        53$:    BIT     #BIT6,$DEVM             ;LEDS TO BE USED/TESTED?
                                                BEQ     66$                     ;BR IF YES
                                                TYPE    ,NOLED                  ;"LEDS NOT USED OR TESTED"
                                                BR      6$                      ;SKIP OTHER STUFF
```

```
                                                66$:
020576                                          
020576  013746  000004                          MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
020602  012737  020616  000004                  MOV     #300$,@#ERRVEC          ;NEW GUY IN PLACE
020610  005777  160470                          TST     @LEDREG                 ;TRY READ
020614  000412                                  BR      320$                    ;NO PROBLEM KEEP GOING
020616  012716  020624                  300$:   MOV     #305$,(SP)              ;HAD PROBLEM
020622  000002                                  RTI                             ;RETURN
020624  012637  000004                  305$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
020630  104401  024012                          TYPE    ,NODDR                  ;NO DDR REGISTER FOUND
020634  104412                                  GTDEVM                          ;TRY AGAIN
020636  000167  160552                          JMP     START                   ;REALLY START OVER
020642  012637  000004                  320$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR

020646                                          6$:
020646  032767  010000  160376                  BIT     #<BIT12>,$DEVM          ;MEM MNGT SELECTED
020654  001024                                  BNE     625$                    ;BR, IF NOT SELECTED (I BELIEVE)
020656  013746  000004                          MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
020662  012737  020676  000004                  MOV     #600$,@#ERRVEC          ;NEW GUY IN PLACE
020670  005767  151444                          TST     PAR0                    ;TRY READ MEMORY MANAGEMENT REGISTER
020674  000412                                  BR      620$                    ;NO PROBLEM KEEP GOING
020676  012716  020704                  600$:   MOV     #605$,(SP)              ;HAD PROBLEM
020702  000002                                  RTI                             ;RETURN
020704  012637  000004                  605$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
020710  104401  023630                          TYPE    ,NOMMNGT                ;NO MEMORY MANAGEMENT PRESENT
020714  104412                                  GTDEVM                          ;TRY AGAIN
020716  000167  160472                          JMP     START                   ;REALLY START OVER
020722  012637  000004                  620$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
020726                                  625$:
020726  032767  000001  160316                  BIT     #<BIT0>,$DEVM           ;IS LINE CLOCK SELECTED
020734  001021                                  BNE     825$                    ;BR, IF SELECTED
020736  013746  000004                          MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
020742  012737  020766  000004                  MOV     #700$,@#ERRVEC          ;NEW GUY IN PLACE
020750  005777  160332                          TST     @LKSREG                 ;TRY AND LOOK AT CLOCK REGISTER
020754  012637  000004                          MOV     (SP)+,@#ERRVEC          ;RESTORE THE VECTOR
020760  104401  023375                          TYPE    ,LNKMSG                 ;"LNK AVAIL BUT NOT TESTED"
020764  000405                                  BR      825$                    ;NO PROBLEM KEEP GOING
020766  012716  020774                  700$:   MOV     #705$,(SP)              ;NO PROBLEM IT IS NOT THERE
020772  000002                                  RTI                             ;RETURN
020774  012637  000004                  705$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR

021000  032767  000100  160244          825$:   BIT     #<BIT6>,$DEVM           ;IS DDR BEING TESTED
021006  001421                                  BEQ     925$                    ;BR, IF SELECTED
021010  013746  000004                          MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
021014  012737  021040  000004                  MOV     #800$,@#ERRVEC          ;NEW GUY IN PLACE
021022  005777  160256                          TST     @LEDREG                 ;TRY AND LOOK AT DDR REGISTER
021026  012637  000004                          MOV     (SP)+,@#ERRVEC          ;RESTORE THE VECTOR
021032  104401  023442                          TYPE    ,DDRMSG                 ;"DDR AVAIL BUT NOT TESTED"
021036  000405                                  BR      925$                    ;NO PROBLEM KEEP GOING
021040  012716  021046                  800$:   MOV     #805$,(SP)              ;NO PROBLEM IT IS NOT THERE
021044  000002                                  RTI                             ;RETURN
021046  012637  000004                  805$:   MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
021052  032767  000200  160172          925$:   BIT     #<BIT7>,$DEVM           ;IS PCR REGISTER SELECTED
021060  001421                                  BEQ     975$                    ;BR, IF SELECTED
021062  013746  000004                          MOV     @#ERRVEC,-(SP)          ;HOLD PRESENT VECTOR
021066  012737  021112  000004                  MOV     #900$,@#ERRVEC          ;NEW GUY IN PLACE
021074  005777  160210                          TST     @PCRREG                 ;TRY AND LOOK AT PCR REGISTER
```

```
4074 021100  012637  000004              MOV     (SP)+,@#ERRVEC          ;RESTORE THE VECTOR
4075 021104  104401  023561              TYPE    ,PCRMSG                 ;"PCR AVAIL BUT NOT TESTED"
4076 021110  000405                      BR      97$                     ;NO PROBLEM KEEP GOING
4077 021112  012716  021120    900$:     MOV     #905$,(SP)              ;NO PROBLEM IT IS NOT THERE
4078 021116  000002                      RTI                             ;RETURN
4079 021120  012637  000004    905$:     MOV     (SP)+,@#ERRVEC          ;RESET VECTOR
4080 021124                    975$:
4081 021124  012767  021522  156652      MOV     #INTSRV,ERRVEC          ;SETUP TIMEOUT TRAP ADDR
4082 021132  012767  000200  156646      MOV     #200,ERRVEC+2           ;LOCKOUT INTERR
4083 021140  005067  000364              CLR     INTFLG
4084 021144  012700  020000              MOV     #20000,R0               ;GO TO BOT OF NEXT 4K
4085
4086 021150  005001                      CLR     R1                      ;MEM MAP.  0=4K, 1=8K, ETC
4087
4088 021152  011010              7$:     MOV     (R0),(R0)               ;DO MOV INSTEAD OF 'TST'
4089 021154  005767  000350              TST     INTFLG                  ;GOT TIMEOUT INTERR?
4090 021160  001006                      BNE     8$                      ;BR IF YES
4091 021162  005201                      INC     R1
4092 021164  062700  020000              ADD     #20000,R0               ;GO TO BOT OF NEXT 4K
4093 021170  020027  160000              CMP     R0,#160000              ;BOT OF 32K?
4094 021174  001366                      BNE     7$                      ;BR IF NO
4095
4096 021176  010067  000316      8$:     MOV     R0,MAXMEM               ;SAVE
4097 021202  162767  000002  000310      SUB     #2,MAXMEM               ;GO BACK TO TOP OF LAST VALID BLOCK
4098 021210  032767  010000  160034      BIT     #BIT12,$DEVM            ;CHECK FOR MEMORY MNGT OPTION
4099 021216  001405                      BEQ     85$                     ;BR, IF MEMORY MNGT IS SELECTED
4100 021220  006301                      ASL     R1                      ;MULT BY 2
4101 021222  016167  021416  000142      MOV     MEMTBL(R1),9$
4102 021230  000457                      BR      19$                     ;PRINT SIZE
4103 021232  012701  172340      85$:    MOV     #PAR0,R1                ;START OF PAR REGISTERS
4104 021236  012705  172300              MOV     #PDR0,R5                ;START OF THE PDR REGISTERS
4105 021242  005000                      CLR     R0                      ;FIRST PAGE BASE ADDRESS
4106 021244  010021              20$:    MOV     R0,(R1)+                ;SET BASE FOR NEXT MAP
4107 021246  012725  077406              MOV     #77406,(R5)+            ;4K READ/WRITE EACH PAGE
4108 021252  062700  000200              ADD     #200,R0                 ;BASE FOR THE NEXT PAGE
4109 021256  020027  001600              CMP     R0,#160000              ;DONE ALL PAGES ?
4110 021262  003770                      BLE     20$                     ;SET UP ALL MEMORY MANAGEMENT PAGES
4111 021264  012741  007600              MOV     #7600,-(R1)             ;SET UP I/O PAGE
4112 021270  052737  000001  177572      BIS     #1,@#SR0                ;ENABLE MEMORY MANAGEMENT
4113 021276  012704  000200              MOV     #200,R4                 ;4K INC VALUE FOR PAR6
4114 021302  010437  172354      15$:    MOV     R4,@#PAR6               ;START CHECKING FOR TOP
4115 021306  005767  000216              TST     INTFLG                  ;DID WE GET TRAP
4116 021312  001005                      BNE     11$                     ;BR, IF WE DID (END OF MEMORY)
4117 021314  005737  140000              TST     @#140000                ;LOOK AT MEMORY
4118 021320  062704  000200              ADD     #200,R4                 ;NEXT 4K VALUE
4119 021324  000766                      BR      15$                     ;LOOP UNTIL TOP OF MEMORY
4120 021326  042737  000001  177572  11$:BIC     #1,@#SR0                ;DISABLE MEMORY MANAGEMENT
4121 021334  162704  000200              SUB     #200,R4                 ;SET TO 4K PAGE THAT DIDN'T TRAP
4122         000006                      .REPT   6
4124 021340  006204                      ASR     R4                      ;REDUCE TO NUMBER OF 4K PAGES
     021342  006204                      ASR     R4                      ;REDUCE TO NUMBER OF 4K PAGES
     021344  006204                      ASR     R4                      ;REDUCE TO NUMBER OF 4K PAGES
     021346  006204                      ASR     R4                      ;REDUCE TO NUMBER OF 4K PAGES
     021350  006204                      ASR     R4                      ;REDUCE TO NUMBER OF 4K PAGES
     021352  006204                      ASR     R4                      ;REDUCE TO NUMBER OF 4K PAGES
4125 021354  010401                      MOV     R4,R1                   ;GET SET TO PRINT MESSAGE
4126 021356  162701  000001              SUB     #1,R1                   ;POINT TO LAST 4K PAGE
```

```
4127 021362  016167  021416  000002          MOV     MEMTBL(R1),9$
4128 021370  104401                  19$:    TYPE
4129 021372  000000                  9$:     .WORD   0                       ;MEM SIZE
4130 021374  104401  024644                  TYPE    ,MEM
4131
4132 021400  012767  000006  156376          MOV     #ERRVEC+2,ERRVEC        ;RESET TMO VECTOR
4133 021406  005067  156374                  CLR     ERRVEC+2
4134
4135 021412  000207                          RTS     PC
4136
4137 021414  000000                  NOCHAN: 0                               ;0 = CHANNEL(S) TO TEST
4138                                                                          ;1 = NO CHANNELS TO TEST
4139
4140                                                                          ;MSG ADDRESSES
4141 021416  024327                  MEMTBL: M4K                             ;       0 -  17776
4142 021420  024334                          M8K                             ;   20000 -  37776
4143 021422  024341                          M12K                            ;   40000 -  57776
4144 021424  024347                          M16K                            ;   60000 -  77776
4145 021426  024355                          M20K                            ;  100000 - 117776
4146 021430  024363                          M24K                            ;  120000 - 137776
4147 021432  024371                          M28K                            ;  140000 - 157776
4148 021434  024377                          M32K                            ;  160000 - 177776
4149 021436  024405                          M36K                            ;  200000 - 217776
4150 021440  024413                          M40K                            ;  220000 - 237776
4151 021442  024421                          M44K                            ;  240000 - 257776
4152 021444  024427                          M48K                            ;  240000 - 257776
4153 021446  024435                          M52K                            ;  260000 - 277776
4154 021450  024443                          M56K                            ;  300000 - 317776
4155 021452  024451                          M60K                            ;  320000 - 337776
4156 021454  024457                          M64K                            ;  340000 - 357776
4157 021456  024465                          M68K                            ;  360000 - 377776
4158 021460  024473                          M72K                            ;  400000 - 417776
4159 021462  024501                          M76K                            ;  420000 - 437776
4160 021464  024507                          M80K                            ;  440000 - 457776
4161 021466  024515                          M84K                            ;  460000 - 477776
4162 021470  024523                          M88K                            ;  500000 - 517776
4163 021472  024531                          M92K                            ;  520000 - 537776
4164 021474  024537                          M96K                            ;  540000 - 557776
4165 021476  024545                          M100K                           ;  560000 - 607776
4166 021500  024554                          M104K                           ;  620000 - 637776
4167 021502  024563                          M108K                           ;  V0000 - 657776
4168 021504  024572                          M112K                           ;  660000 - 677776
4169 021506  024601                          M116K                           ;  700000 - 717776
4170 021510  024610                          M120K                           ;  720000 - 737776
4171 021512  024617                          M124K                           ;  740000 - 757776
4172 021514  024626                          M128K                           ;  760000 - 777776
4173 021516  024635                          M132K                           ;  10000000 - AND UP
4174
4175 021520  000000                  MAXMEM: 0                               ;MAX MEMORY
4176
4177
4178
4179                                  .SBTTL  INTSRV  INTERRUPT SERVICE ROUTINE
4180
4181                                  ;*      THIS GLOBAL ROUTINE DOES NOTHING BUT INCREMENT
4182                                  ;*      'INTFLG'  EACH TIME IT IS CALLED.  IT ASSUMES
4183                                  ;*      THAT THE MAIN CALLING ROUTINE WILL  KNOW  WHAT
```

```
4184                                    ;*       TO LOOK FOR.
4185
4186 021522  005267  000002            INTSRV: INC     INTFLG                  ;ADD 1 TO 'INTERRUP' OCCURED' FLAG
4187 021526  000002                            RTI                             ;THAT'S ALL
4188
4189 021530  000000                    INTFLG:  0
4190
4191
4192                                    .SBTTL  ROUTINE TO SET UP FOR NEXT CHANNEL ADDRESS
4193
4194                                    ;;*****************************************************************
4195                                    ;*       THIS  ROUTINE  CAUSES  ADRS   TO POINT TO  THE
4196                                    ;*       ADDRESS  OF  CHANNEL  UNDER  TEST.  ADRS +2 TO
4197                                    ;*       POINT TO THE VECTOR OF THE CHANNEL UNDER TEST.
4198                                    ;;*****************************************************************
4199
4200 021532  005767  177656            CYCLE:  TST     NOCHAN                  ;ANY CHANNELS TO BE TESTED?
4201 021536  001401                            BEQ     1$                      ;BR IF YES
4202 021540  000207                            RTS     PC                      ;ELSE RETURN
4203
4204 021542  026767  157440  000612    1$:     CMP     SUNIT,HICHAN            ;DONE ALL AVAIL CHANNELS?
4205 021550  001032                            BNE     3$                      ;BR IF NO
4206 021552  005067  157430                    CLR     SUNIT
4207 021556  016767  157472  157526            MOV     BASE0,DLADD             ;SET CHAN 0 RCSR ADDR
4208 021564  016767  157466  157532            MOV     VECT0,DLVEC             ; & VECTOR
4209 021572  032767  002000  157452            BIT     #BIT10,SDEVM            ;CHAN 0 DATA WRAP?
4210 021600  001410                            BEQ     2$                      ;BR IF YES
4211 021602  032767  000010  157442            BIT     #BIT3,SDEVM             ;CHAN 1 DATA WRAP?
4212 021610  001404                            BEQ     2$                      ;BR IF YES
4213 021612  104401  001334                    TYPE    ,CARR
4214 021616  000167  003042                    JMP     $EOP                    ;ELSE ALL DONE
4215
4216 021622  012767  000001  000526  2$:     MOV     #TRUE,PHASE2
4217 021630  005067  000524                    CLR     P1CNT
4218 021634  000207                            RTS     PC
4219
4220 021636  005767  157344          3$:     TST     SUNIT                   ;CHAN 0 TO BE TESTED?
4221 021642  001050                            BNE     12$                     ;BR IF NO
4222
4223 021644  032767  100000  157400            BIT     #BIT15,SDEVM            ;CHAN 0 TO BE TESTED?
4224 021652  001403                            BEQ     4$                      ;BR IF YES
4225 021654  005267  157326                    INC     SUNIT
4226 021660  000724                            BR      CYCLE                   ;TRY OVER AGAIN FOR CHAN 1
4227
4228 021662  016767  157366  157422  4$:     MOV     BASE0,DLADD             ;SET CHAN 0 RCSR ADDR
4229 021670  016767  157362  157426            MOV     VECT0,DLVEC             ; & VECTOR
4230
4231 021676  005067  000442                    CLR     WORDL                   ;ELSE CLEAR TO DEFAULT & NO PARITY
4232
4233 021702  005067  000440                    CLR     ODD                     ;ELSE CLEAR TO DEFAULT
4234
4235 021706  032767  004000  157336            BIT     #BIT11,SDEVM            ;BREAK DETECTION ENABLED?
4236 021714  001003                            BNE     8$                      ;BR IF YES
4237 021716  005067  000426                    CLR     BRK                     ;ELSE CLEAR TO DEFAULT
4238 021722  000402                            BR      .+6
4239 021724  005267  000420          8$:     INC     BRK
4240
```

```
4241 021730  032767  002000  157314          BIT     #BIT10,$DEVM            ;DATA WRAP AROUND?
4242 021736  001003                          BNE     9$                      ;BR IF NO
4243 021740  005067  000406                  CLR     WRAP                    ;ELSE CLEAR TO DEFAULT
4244 021744  000402                          BR      .+6
4245 021746  005267  000400          9$:     INC     WRAP
4246
4247 021752  005067  000376          10$:    CLR     CONSOLE                 ;CHAN 0 IS NOT CONSOLE
4248 021756  005267  157224          11$:    INC     $UNIT                   ;SETUP FOR NEXT TIME
4249 021762  000207                          RTS     PC
4250
4251 021764  022767  000001  157214  12$:    CMP     #1,$UNIT                ;CHECK FOR UNIT NUMBER ONE
4252 021772  001417                          BEQ     13$                     ;BR, IF PRESENT UNIT = 1
4253 021774  032767  020000  157250          BIT     #BIT13,$DEVM            ;CHECK FOR 2 MXV11-B TESTING
4254 022002  001410                          BEQ     19$                     ;BR, IF NO 2 MXV11-B TESTING
4255 022004  022767  000002  157174          CMP     #2,$UNIT                ;CHECK FOR UNIT NUMBER 2
4256 022012  001456                          BEQ     20$                     ;BR, IF WE SHOULD SET UP UNIT 2
4257 022014  022767  000003  157164          CMP     #3,$UNIT                ;CHECK FOR UNIT NUMBER 3
4258 022022  001511                          BEQ     30$                     ;BR, IF WE SHOULD SET UP UNIT 3
4259 022024  005267  157156          19$:    INC     $UNIT
4260 022030  000640                          BR      CYCLE                   ;TRY OVER AGAIN FOR CHAN 1
4261
4262 022032  032767  000400  157212  13$:    BIT     #BIT8,$DEVM             ;TEST CHANNEL 1?
4263 022040  001771                          BEQ     19$                     ;BR, IF NOT CHANNEL 1 TESTING
4264 022042  016767  157212  157242          MOV     BASE1,DLADD             ;SET CHAN 1 RCSR ADDR
4265 022050  016767  157206  157246          MOV     VECT1,DLVEC             ; & VECTOR
4266
4267 022056  005067  000262                  CLR     WORDL                   ;ELSE CLEAR TO DEFAULT & NO PARITY
4268 022062  005067  000260                  CLR     ODD                     ;ELSE CLEAR TO DEFAULT
4269
4270 022066  032767  000020  157156          BIT     #BIT4,$DEVM             ;BREAK DETECTION ENABLED?
4271 022074  001003                          BNE     17$                     ;BR IF YES
4272 022076  005067  000246                  CLR     BRK                     ;ELSE CLEAR TO DEFAULT
4273 022102  000402                          BR      .+6
4274 022104  005267  000240          17$:    INC     BRK
4275
4276 022110  032767  000010  157134          BIT     #BIT3,$DEVM             ;DATA WRAP AROUND?
4277 022116  001003                          BNE     18$                     ;BR IF NO
4278 022120  005067  000226                  CLR     WRAP                    ;ELSE CLEAR TO DEFAULT
4279 022124  000402                          BR      .+6
4280 022126  005267  000220          18$:    INC     WRAP
4281
4282 022132  026767  157122  157004          CMP     BASE1,$TKS              ;CHAN 1 CONSOLE?
4283 022140  001304                          BNE     10$                     ;BR IF NO
4284 022142  005267  000206                  INC     CONSOLE
4285 022146  000207                          RTS     PC                      ;HAVE VECTORS RETURN
4286                         ;*****************************************
4287                         ;       CHANNEL 2 SETUP
4288                         ;*****************************************
4289 022150  016767  157110  157134  20$:    MOV     BASE2,DLADD             ;SET CHAN 2 RCSR ADDR
4290 022156  016767  157104  157140          MOV     VECT2,DLVEC             ; & VECTOR
4291 022164  005067  000154                  CLR     WORDL                   ;ELSE CLEAR TO DEFAULT & NO PARITY
4292 022170  005067  000152                  CLR     ODD                     ;ELSE CLEAR TO DEFAULT
4293 022174  032767  000020  157050          BIT     #BIT4,$DEVM             ;BREAK DETECTION ENABLED?
4294 022202  001003                          BNE     21$                     ;BR IF YES
4295 022204  005067  000140                  CLR     BRK                     ;ELSE CLEAR TO DEFAULT
4296 022210  000402                          BR      .+6
4297 022212  005267  000132          21$:    INC     BRK
```

```
 4298 022216  032767  002000  157026          BIT     #BIT10,$DEVM            ;DATA WRAP AROUND?
 4299 022224  001003                          BNE     22$                     ;BR IF NO
 4300 022226  005067  000120                  CLR     WRAP                    ;ELSE CLEAR TO DEFAULT
 4301 022232  000402                          BR      .+6
 4302 022234  005267  000112          22$:    INC     WRAP
 4303 022240  005267  156742                  INC     $UNIT                   ;BUMP UNIT NUMBER
 4304 022244  000207                          RTS     PC                      ;HAVE VECTORS RETURN
 4305
 4306                                 ;*******************************************************
 4307                                 ;       SET UP FOR CHANNEL 3
 4308                                 ;*******************************************************
 4309 022246  016767  157016  157036  30$:    MOV     BASE3,DLADD             ;SET CHAN 3 RCSR ADDR
 4310 022254  016767  157012  157042          MOV     VECT3,DLVEC             ; & VECTOR
 4311 022262  005067  000056                  CLR     WORDL                   ;ELSE CLEAR TO DEFAULT & NO PARITY
 4312 022266  005067  000054                  CLR     ODD                     ;ELSE CLEAR TO DEFAULT
 4313 022272  032767  000020  156752          BIT     #BIT4,$DEVM             ;BREAK DETECTION ENABLED?
 4314 022300  001003                          BNE     31$                     ;BR IF YES
 4315 022302  005067  000042                  CLR     BRK                     ;ELSE CLEAR TO DEFAULT
 4316 022306  000402                          BR      .+6
 4317 022310  005267  000034          31$:    INC     BRK
 4318 022314  032767  002000  156730          BIT     #BIT10,$DEVM            ;DATA WRAP AROUND?
 4319 022322  001003                          BNE     32$                     ;BR IF NO
 4320 022324  005067  000022                  CLR     WRAP                    ;ELSE CLEAR TO DEFAULT
 4321 022330  000402                          BR      .+6
 4322 022332  005267  000014          32$:    INC     WRAP
 4323 022336  005267  156644                  INC     $UNIT                   ;BUMP UNIT NUMBER
 4324 022342  000207                          RTS     PC                      ;HAVE VECTORS RETURN
 4325
 4326
 4327
 4328 022344  000000          WORDL:  0                                       ;WORD LENGTH    0 = 8 BITS (NO PARITY), 1 = 7 BITS (
 4329 022346  000000          ODD:    0                                       ;ODD PARITY     0 = ODD,    1 = EVEN
 4330 022350  000000          BRK:    0                                       ;BREAK DET      0 = DISABL, 1 = ENABL
 4331 022352  000000          WRAP:   0                                       ;DATA WRAP      0 = YES,    1 = NO
 4332 022354  000000          CONSOLE: 0                                      ;CONSOLE        0 = CH UNDER TEST NOT CONSOLE
 4333                                                                         ;               1 = CH UNDER TEST IS  CONSOLE
 4334 022356  000000          PHASE2: 0
 4335 022360  000000          P1CNT:  0                                       ;CTR TO PRINT 'PHASE 1' ONLY ONCE
 4336 022362  000000          HICHAN: .WORD   0                               ;STORE HIGHEST CHANNEL NUMBER
 4337
 4338
 4339                                 ;;*******************************************************************
 4340                                 ;ROUTINE TO DISPLAY CURRENT DEVM & ALLOW OPERATOR TO INPUT NEW VALUE.
 4341                                 ;*******************************************************************
 4342
 4343 022364  104401  024316  GTSDEV: TYPE    ,DEVM                           ;SHOW CURRENT
 4344 022370  016746  156656          MOV     $DEVM,-(SP)     ;;SAVE $DEVM FOR TYPEOUT
      022374  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4345 022376  104401  026302          TYPE    ,$MNEW                          ;GET NEW
 4346
 4347 022402  005046          1$:     CLR     -(SP)                           ;CLR CTR
 4348 022404  005046                  CLR     -(SP)                           ;CLR NEW DEVM
 4349
 4350 022406  105777  156532  2$:     TSTB    @$TKS                           ;CHAR THERE?
 4351 022412  100375                  BPL     2$                              ;BR IF NO
 4352 022414  117746  156526          MOVB    @$TKB,-(SP)                     ;ELSE GET CHAR
 4353 022420  042716  177600          BIC     #^C177,(SP)                     ;MAKE IT 7 BIT ASCII
```

```
4354
4355 022424  021627  000015          3$:   CMP   (SP),#15          ;<CR>?
4356 022430  001017                       BNE   7$                ;BR IF NO
4357 022432  005766  000004               TST   4(SP)             ;ELSE IS IT 1'ST CHAR?
4358 022436  001403                       BEQ   4$                ;BR IF YES
4359 022440  016667  000002  156604       MOV   2(SP),$DEVM       ;ELSE SAVE NEW DEVM
4360 022446  104401  001171          4$:   TYPE  ,$CRLF
4361 022452  004767  175336          6$:   JSR   PC,SIZE           ;RE-SIZE
4362 022456  062706  000006               ADD   #6,SP             ;RESET STACK
4363 022462  012716  002066               MOV   #RAMROM,(SP)
4364 022466  000002                       RTI
4365
4366 022470  004767  002706          7$:   JSR   PC,$TYPEC         ;ECHO CHAR
4367 022474  021627  000060               CMP   (SP),#60          ;CHAR < 0?
4368 022500  002420                       BLT   9$                ;BR IF YES
4369 022502  021627  000067               CMP   (SP),#67          ;CHAR > 7?
4370 022506  003015                       BGT   9$                ;BR IF YES
4371 022510  042726  000060               BIC   #60,(SP)+         ;STRIP OFF ASCII
4372 022514  005766  000002               TST   2(SP)             ;1'ST CHAR?
4373 022520  001403                       BEQ   8$                ;BR IF YES
4374 022522  006316                       ASL   (SP)              ;ELSE SHIFT PRESENT CHAR
4375 022524  006316                       ASL   (SP)              ;TO MAKE ROOM
4376 022526  006316                       ASL   (SP)              ;FOR NEW ONE
4377
4378 022530  005266  000002          8$:   INC   2(SP)             ;KEEP CHAR CT
4379 022534  056616  177776               BIS   -2(SP),(SP)       ;SET IN NEW CHAR
4380 022540  000722                       BR    2$                ;GET NEW ONE
4381
4382 022542  104401  001170          9$:   TYPE  ,$QUES
4383 022546  104401  001171                TYPE  ,$CRLF
4384 022552  005726                       TST   (SP)+             ;RESET PTR
4385 022554  005016                       CLR   (SP)              ;CLR DEVM
4386 022556  000713                       BR    2$                ;TRY AGAIN
4387
4388 022560  104401  023511  ILLCLK: TYPE,ILLMSG                   ;ILLEGAL CLOCK INTERRUPT HANDLER
4389 022564  000002                  RTI
```

```
                                ;;*****************************************************************
                                ;
                                ; PROGRAM TIMERS
                                ;
                                ;;*****************************************************************

022566  013567  000044          TIMER:  MOV     @(R5)+,REGHLD
022572  012567  000042                  MOV     (R5)+,BITHLD

022576  012701  000010                  MOV     #10,R1                  ;DOUBLE LOOP
022602  012700  177777          1$:     MOV     #-1,R0
022606  036777  000026  000022  2$:     BIT     BITHLD,@REGHLD          ;BIT THERE?
022614  001005                          BNE     3$                      ;BR IF YES
022616  005300                          DEC     R0
022620  001372                          BNE     2$
022622  005301                          DEC     R1
022624  001366                          BNE     1$
022626  000402                          BR      4$                      ;ALL TIMED OUT

022630  062705  000004          3$:     ADD     #4,R5                   ;JUMP OVER ERR ON RETURN
022634  000205                  4$:     RTS     R5

022636  000000                  REGHLD: 0                               ;REGISTER TO BE TESTED
022640  000000                  BITHLD: 0                               ;BIT TO BE TESTED IN REGISTER


022642  012567  177770          TIMER1: MOV     (R5)+,REGHLD

022646  012701  000010                  MOV     #10,R1
022652  012700  177777          1$:     MOV     #-1,R0
022656  005777  177754          2$:     TST     @REGHLD                 ;ZERO?
022662  001005                          BNE     3$                      ;BR IF NO
022664  005300                          DEC     R0
022666  001373                          BNE     2$
022670  005301                          DEC     R1
022672  001367                          BNE     1$
022674  000402                          BR      4$                      ;ALL TIMED OUT

022676  062705  000004          3$:     ADD     #4,R5                   ;JUMP OVER ERR ON RETURN
022702  000205                  4$:     RTS     R5


022704  012700  177777          WAIT:   MOV     #-1,R0
022710  005300                          DEC     R0
022712  001376                          BNE     .-2
022714  000207                          RTS     PC
```

```
                                                L 9
CVMXBA0 MXV11B DIAG     MACRO M1113  28-FEB-83 16:15  PAGE 44
ROUTINE TO SET UP FOR NEXT CHANNEL ADDRESS                                                    SEQ 0115

 4440
 4445 022716                                MYTYPE: TYPTXT  <<CRLF>*TEST # *>
 4446 022736                                        TYPOCT  $TESTN
 4447 022744                                        TYPTXT  <*,ERROR # *>
 4448 022764  116767  156124  156204               MOVB    $!TEMB,$FATAL          ; APT FATAL ERROR NUMBER
 4449 022772                                        TYPOCS  $FATAL
 4450 023002                                        TYPTXT  <*,PC = *>
 4451 023020                                        TYPOCT  $ERRPC
 4452
 4453 023026  026727  156146  000005               CMP     $TESTN,#5              ;DOING SLU TESTS?
 4454 023034  100425                                BMI     1$                     ;BR IF NO
 4455
 4456 023036                                        TYPTXT  <*,CSR: *>             ;ELSE PROCEED
 4457 023054                                        TYPOCT  DLADD
 4458 023062                                        TYPTXT  <*,VECTOR: *>
 4459 023102                                        TYPOCT  DLVEC
 4460 023110  104401  001171               1$:     TYPE    ,$CRLF
 4461 023114  000207                                RTS     PC
 4466
 4467
 4468 023116     200     103     110       CODROP: .ASCIZ  <CRLF>/CHAN 0 TESTING DROPPED/
      023121     101     116     040
      023124     060     040     124
      023127     105     123     124
      023132     111     116     107
      023135     040     104     122
      023140     117     120     120
      023143     105     104     000
 4469 023146     200     103     110       C1DROP: .ASCIZ  <CRLF>/CHAN 1 TESTING DROPPED/
      023151     101     116     040
      023154     061     040     124
      023157     105     123     124
      023162     111     116     107
      023165     040     104     122
      023170     117     120     120
      023173     105     104     000
 4470 023176     200     114     105       NOLED:  .ASCIZ  <CRLF>/LEDS NOT USED OR TESTED/
      023201     104     123     040
      023204     116     117     124
      023207     040     125     123
      023212     105     104     040
      023215     117     122     040
      023220     124     105     123
      023223     124     105     104
      023226     000
 4471
 4472 023227     200     103     110       NOCH1:  .ASCIZ  <CRLF>/CHANNEL 1 CAN'T BE TESTED WHEN CONSOLE/
      023232     101     116     116
      023235     105     114     040
      023240     061     040     103
      023243     101     116     047
      023246     124     040     102
      023251     105     040     124
      023254     105     123     124
      023257     105     104     040
      023262     127     110     105
      023265     116     040     103
```

```
        023270  117     116     123
        023273  117     114     105
        023276  000
  4473  023277  200     103     114   CLKDS:  .ASCIZ  <CRLF>/CLOCK DISABLED/
        023302  117     103     113
        023305  040     104     111
        023310  123     101     102
        023313  114     105     104
        023316  000
  4474  023317  200     120     103   NOPCR2: .ASCIZ  <CRLF>/PCR REGISTER TEST SELECTED BUT NOT AVAILABLE/
        023322  122     040     122
        023325  105     107     111
        023330  123     124     105
        023333  122     040     124
        023336  105     123     124
        023341  040     123     105
        023344  114     105     103
        023347  124     105     104
        023352  040     102     125
        023355  124     040     116
        023360  117     124     040
        023363  101     126     101
        023366  111     114     101
        023371  102     114     105
        023374  000
  4475
  4476  023375  200     114     111   LNKMSG: .ASCIZ  <CRLF>/LINE CLOCK AVAILABLE BUT NOT TESTED/
        023400  116     105     040
        023403  103     114     117
        023406  103     113     040
        023411  101     126     101
        023414  111     114     101
        023417  102     114     105
        023422  040     102     122
        023425  124     040     116
        023430  117     124     040
        023433  124     105     123
        023436  124     105     104
        023441  000
  4477  023442  200     104     104   DDRMSG: .ASCIZ  <CRLF>/DDR REGISTER AVAILABLE BUT NOT TESTED/
        023445  122     040     122
        023450  105     107     111
        023453  123     124     105
        023456  122     040     101
        023461  126     101     111
        023464  114     101     102
        023467  114     105     040
        023472  102     125     124
        023475  040     116     117
        023500  124     040     124
        023503  105     123     124
        023506  105     104     000
  4478  023511  200     111     114   ILLMSG: .ASCIZ  <CRLF>/ILLEGAL INTERRUPT THROUGH CLOCK VECTOR/
        023514  114     105     107
        023517  101     114     040
        023522  111     116     124
        023525  105     122     122
```

```
                023530  125     120     124
                023533  040     124     110
                023536  122     117     125
                023541  107     110     040
                023544  103     114     117
                023547  103     113     040
                023552  126     105     103
                023555  124     117     122
                023560  000
   4479         023561  200     120     103     PCRMSG: .ASCIZ  <CRLF>/PCR REGISTER AVAILABLE BUT NOT TESTED/
                023564  122     040     122
                023567  105     107     111
                023572  123     124     105
                023575  122     040     101
                023600  126     101     111
                023603  114     101     102
                023606  114     105     040
                023611  102     125     124
                023614  040     116     117
                023617  124     040     124
                023622  105     123     124
                023625  105     104     000
   4480         023630  200     115     105     NOMMNGT:        .ASCIZ  <CRLF>/MEM MNGT SELECTED BUT NOT AVAILABLE/
                023633  115     040     115
                023636  116     107     124
                023641  040     123     105
                023644  114     105     103
                023647  124     105     104
                023652  040     102     125
                023655  124     040     116
                023660  117     124     040
                023663  101     126     101
                023666  111     114     101
                023671  102     114     105
                023674  000
   4481         023675  200     123     105     NOSMX:  .ASCIZ  <CRLF>/SECOND MXV11-B SELECTED BUT NOT AVAILABLE/
                023700  103     117     116
                023703  104     040     115
                023706  130     126     061
                023711  061     055     102
                023714  040     123     105
                023717  114     105     103
                023722  124     105     104
                023725  040     102     125
                023730  124     040     116
                023733  117     124     040
                023736  101     126     101
                023741  111     114     101
                023744  102     114     105
                023747  000
   4482         023750  200     103     114     NOCLK:  .ASCIZ  <CRLF>/CLOCK SELECTED BUT NOT AVAILABLE/
                023753  117     103     113
                023756  040     123     105
                023761  114     105     103
                023764  124     105     104
                023767  040     102     125
                023772  124     040     116
```

```
                023775  117     124     040
                024000  101     126     101
                024003  111     114     101
                024006  102     114     105
                024011  000
   4483         024012  200     104     104     NODDR:  .ASCIZ  <CRLF>/DDR REGISTER SELECTED BUT NOT PRESENT/
                024015  122     040     122
                024020  105     107     111
                024023  123     124     105
                024026  122     040     123
                024031  105     114     105
                024034  103     124     105
                024037  104     040     102
                024042  125     124     040
                024045  116     117     124
                024050  040     120     122
                024053  105     123     105
                024056  116     124     000
   4484         024061  200     120     103     NOPCR:  .ASCIZ  <CRLF>/PCR REGISTER TEST BYPASSED/
                024064  122     040     122
                024067  105     107     111
                024072  123     124     105
                024075  122     040     124
                024100  105     123     124
                024103  040     102     131
                024106  120     101     123
                024111  123     105     104
                024114  000
   4485         024115  200     103     110     C1CON:  .ASCIZ  <CRLF>/CHAN 1 IS CONSOLE/
                024120  101     116     040
                024123  061     040     111
                024126  123     040     103
                024131  117     116     123
                024134  117     114     105
                024137  000
   4486         024140  200     122     117     NOROM:  .ASCIZ  <CRLF>/ROM TEST BYPASSED/
                024143  115     040     124
                024146  105     123     124
                024151  040     102     131
                024154  120     101     123
                024157  123     105     104
                024162  000
   4487         024163  015     012     122     NORAM:  .ASCIZ  <CR><LF>/RAM TEST BYPASSED/
                024166  101     115     040
                024171  124     105     123
                024174  124     040     102
                024177  131     120     101
                024202  123     123     105
                024205  104     000
   4488         024207  200     103     114     CLKEN:  .ASCIZ  <CRLF>/CLOCK ENABLED/
                024212  117     103     113
                024215  040     105     116
                024220  101     102     104
                024223  105     104     000
   4489         024226  015     012     123     TWMXN:  .ASCIZ  <CR><LF>/SECOND MXV11-B TEST BYPASSED/
                024231  105     103     117
                024234  116     104     040
```

```
                024237  115     130     126
                024242  061     061     055
                024245  102     040     124
                024250  105     123     124
                024253  040     102     131
                024256  120     101     123
                024261  123     105     104
                024264  000
4490            024265  015     012     124     TWOMX:  .ASCIZ  <CR><LF>/TESTING SECOND MXV11-B/
                024270  105     123     124
                024273  111     116     107
                024276  040     123     105
                024301  103     117     116
                024304  104     040     115
                024307  130     126     061
                024312  061     055     102
                024315  000
4491            024316  200     104     105     DEVM:   .ASCIZ  <CRLF>/DEVM = /
                024321  126     115     040
                024324  075     040     000
4492            024327  015     012     064     M4K:    .ASCIZ  <CR><LF>/4K/
                024332  113     000
4493            024334  015     012     070     M8K:    .ASCIZ  <CR><LF>/8K/
                024337  113     000
4494            024341  015     012     061     M12K:   .ASCIZ  <CR><LF>/12K/
                024344  062     113     000
4495            024347  015     012     061     M16K:   .ASCIZ  <CR><LF>/16K/
                024352  066     113     000
4496            024355  015     012     062     M20K:   .ASCIZ  <CR><LF>/20K/
                024360  060     113     000
4497            024363  015     012     062     M24K:   .ASCIZ  <CR><LF>/24K/
                024366  064     113     000
4498            024371  015     012     062     M28K:   .ASCIZ  <CR><LF>/28K/
                024374  070     113     000
4499            024377  015     012     063     M32K:   .ASCIZ  <CR><LF>/32K/
                024402  062     113     000
4500            024405  015     012     063     M36K:   .ASCIZ  <CR><LF>/36K/
                024410  066     113     000
4501            024413  015     012     064     M40K:   .ASCIZ  <CR><LF>/40K/
                024416  060     113     000
4502            024421  015     012     064     M44K:   .ASCIZ  <CR><LF>/44K/
                024424  064     113     000
4503            024427  015     012     064     M48K:   .ASCIZ  <CR><LF>/48K/
                024432  070     113     000
4504            024435  015     012     065     M52K:   .ASCIZ  <CR><LF>/52K/
                024440  062     113     000
4505            024443  015     012     065     M56K:   .ASCIZ  <CR><LF>/56K/
                024446  066     113     000
4506            024451  015     012     066     M60K:   .ASCIZ  <CR><LF>/60K/
                024454  060     113     000
4507            024457  015     012     066     M64K:   .ASCIZ  <CR><LF>/64K/
                024462  064     113     000
4508            024465  015     012     066     M68K:   .ASCIZ  <CR><LF>/68K/
                024470  070     113     000
4509            024473  015     012     067     M72K:   .ASCIZ  <CR><LF>/72K/
                024476  062     113     000
4510            024501  015     012     067     M76K:   .ASCIZ  <CR><LF>/76K/
```

```
                024504  066  113  000
4511            024507  015  012  070  M80K:   .ASCIZ  <CR><LF>/80K/
                024512  060  113  000
4512            024515  015  012  070  M84K:   .ASCIZ  <CR><LF>/84K/
                024520  064  113  000
4513            024523  015  012  070  M88K:   .ASCIZ  <CR><LF>/88K/
                024526  070  113  000
4514            024531  015  012  071  M92K:   .ASCIZ  <CR><LF>/92K/
                024534  062  113  000
4515            024537  015  012  071  M96K:   .ASCIZ  <CR><LF>/96K/
                024542  066  113  000
4516            024545  015  012  061  M100K:  .ASCIZ  <CR><LF>/100K/
                024550  060  060  113
                024553  000
4517            024554  015  012  061  M104K:  .ASCIZ  <CR><LF>/104K/
                024557  060  064  113
                024562  000
4518            024563  015  012  061  M108K:  .ASCIZ  <CR><LF>/108K/
                024566  060  070  113
                024571  000
4519            024572  015  012  061  M112K:  .ASCIZ  <CR><LF>/112K/
                024575  061  062  113
                024600  000
4520            024601  015  012  061  M116K:  .ASCIZ  <CR><LF>/116K/
                024604  061  066  113
                024607  000
4521            024610  015  012  061  M120K:  .ASCIZ  <CR><LF>/120K/
                024613  062  060  113
                024616  000
4522            024617  015  012  061  M124K:  .ASCIZ  <CR><LF>/124K/
                024622  062  064  113
                024625  000
4523            024626  015  012  061  M128K:  .ASCIZ  <CR><LF>/128K/
                024631  062  070  113
                024634  000
4524            024635  015  012  061  M132K:  .ASCIZ  <CR><LF>/132K/
                024640  063  062  113
                024643  000
4525            024644  040  115  105  MEM:    .ASCIZ  / MEM PRESENT/<CR><LF>
                024647  115  040  120
                024652  122  105  123
                024655  105  116  124
                024660  015  012  000
4526                                      .EVEN
```

```
 4528                                   .SBTTL  END OF PASS ROUTINE
                                        ;;*****************************************************************
                                        ;*INCREMENT THE PASS NUMBER (&PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO RAMROM
 024664                                 $EOP:
 024664  000004                                SCOPE
 024666  005067  154210                         CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 024672  005067  154262                         CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 024676  005267  154300                         INC     $PASS           ;;INCREMENT THE PASS NUMBER
 024702  042767  100000  154272               BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 024710  005327                               DEC     (PC)+           ;;LOOP?
 024712  000001                 $EOPCT: .WORD   1
 024714  003022                         BGT     $DOAGN          ;;YES
 024716  012737                         MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 024720  000001                 $ENDCT: .WORD   1
 024722  024712                         $EOPCT
 024724  104401  024771                 TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 024730  016746  154246                 MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 024734  104405                         TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 024736  104401  024766                 TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 024742  013700  000042         $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 024746  001405                         BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 024750  000005                         RESET                   ;;CLEAR THE WORLD
 024752  004710                 $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 024754  000240                         NOP                     ;;SAVE ROOM
 024756  000240                         NOP                     ;;FOR
 024760  000240                         NOP                     ;;ACT11
 024762                         $DOAGN:
 024762  000137                         JMP     @(PC)+          ;;RETURN
 024764  002066                 $RTNAD: .WORD   RAMRGM
 024766     377      377      000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 024771     015      012      105  $ENDMG: .ASCIZ  <15><12>/END PASS #/
 024774     116      104      040
 024777     120      101      123
 025002     123      040      043
 025005     000
```

```
4530                                    .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
025006  012737  025152  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
025014  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
025022  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
025024  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
025026  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
025030  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
025032  010446                          MOV     R4,-(SP)                ;;PUSH R4 ON STACK
025034  010546                          MOV     R5,-(SP)                ;;PUSH R5 ON STACK
025036  017746  154076                  MOV     @$WR,-(SP)              ;;PUSH @$WR ON STACK
025042  010667  000110                  MOV     SP,$SAVR6       ;;SAVE SP
025046  012737  025060  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
025054  000000                          HALT
025056  000776                          BR      .-2             ;;HANG UP
                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
025060  012737  025152  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
025066  016706  000064                  MOV     $SAVR6,SP       ;;GET SP
025072  005067  000060                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
025076  005267  000054          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
025102  001375                          BNE     1$              ;;OF  WORD
025104  012677  154030                  MOV     (SP)+,@$WR      ;;POP STACK INTO @$WR
025110  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
025112  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
025114  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
025116  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
025120  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
025122  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
025124  012737  025006  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
025132  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
025140  104401                          TYPE                    ;REPORT THE POWER FAILURE
025142  025160                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
025144  012716                          MOV     (PC)+,(SP)      ;;RESTART AT START
025146  001414                  $PWRAD: .WORD   START           ;;RESTART ADDRESS
025150  000002                          RTI
025152  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
025154  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
025156  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
025160     015     012     120  $POWER: .ASCIZ  <15><12>"POWER"
025163     117     127     105
025166     122     000

                                        .EVEN
```

```
 4532                                   .SBTTL  TYPE ROUTINE
                                        ;;*****************************************************************
                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*      TYPE
                                        ;*      MESADR
                                        ;*
 025170  105767  153763         $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 025174  100002                         BPL     1$              ;;BR IF YES
 025176  000000                         HALT                    ;;HALT HERE IF NO TERMINAL
 025200  000430                         BR      3$              ;;LEAVE
 025202  010046                 1$:     MOV     R0,-(SP)        ;;SAVE R0
 025204  017600  000002                 MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 025210  122767  000001  153776         CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 025216  001011                         BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
 025220  132767  000100  153767         BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
 025226  001405                         BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
 025230  010067  000004                 MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
 025234  004767  001062                 JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
 025240  000000                 61$:    .WORD   0               ;;MESSAGE ADDRESS
 025242  132767  000040  153745  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
 025250  001003                         BNE     60$             ;;YES,SKIP TYPE OUT
 025252  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 025254  001005                         BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 025256  005726                         TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 025260  012600                 60$:    MOV     (SP)+,R0        ;;RESTORE R0
 025262  062716  000002         3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 025266  000002                         RTI                     ;;RETURN
 025270  122716  000011         4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 025274  001430                         BEQ     8$
 025276  122716  000200                 CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 025302  001006                         BNE     5$
 025304  005726                         TST     (SP)+           ;;POP  <CR><LF> EQUIV
 025306  104401                         TYPE                    ;;TYPE A CR AND LF
 025310  001171                         $CRLF
 025312  105067  000130                 CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 025316  000755                         BR      2$              ;;GET NEXT CHARACTER
 025320  004767  000056         5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 025324  126726  153626         6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 025330  001350                         BNE     2$              ;;IF NO GO GET NEXT CHAR.
 025332  016746  153616                 MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
 025336  105366  000001         7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 025342  002770                         BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 025344  004767  000032                 JSR     PC,$TYPEC       ;;GO TYPE A NULL
 025350  105367  000072                 DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 025354  000770                         BR      7$              ;;LOOP
                                        ;HORIZONTAL TAB PROCESSOR
 025356  112716  000040         8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
```

```
025362  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
025366  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
025374  001372                          BNE     9$              ;;TAB STOP
025376  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
025400  000724                          BR      2$              ;;GET NEXT CHARACTER
025402  105777  153542          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
025406  100375                          BPL     $TYPEC
025410  116677  000002  153534          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
025416  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
025424  001003                          BNE     1$              ;;BRANCH IF NO
025426  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
025432  000406                          BR      $TYPEX          ;;EXIT
025434  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
025442  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
025444  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
025446  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
025450  000207          $TYPEX: RTS     PC
```

```
 4534                                   .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.
025452  022767  000176  153460  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
025460  001114                          BNE     15$             ;;BRANCH IF NO
025462  105777  153456                  TSTB    @$TKS           ;;CHAR THERE?
025466  100111                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
025470  117746  153452                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
025474  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
025500  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
025504  001102                          BNE     15$             ;;NO, RETURN TO USER
025506  126727  153422  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
025514  001476                          BEQ     15$             ;;BRANCH IF YES
025516  104401  026264                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
025522  104401  026271          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
025526  016746  152444                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
025532  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
025534  104401  026302                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
025540  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
025542  005046                          CLR     -(SP)           ;;THE NEW SWR
025544  105777  153374          7$:     TSTB    @$TKS           ;;CHAR THERE?
025550  100375                          BPL     7$              ;;IF NOT TRY AGAIN
025552  117746  153370                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
025556  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
025562  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
025566  001015                          BNE     9$              ;;BRANCH IF NOT
025570  104401  026252                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
025574  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
025600  126727  153331  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
025606  001003                          BNE     8$              ;;BRANCH IF NO
025610  012777  000100  153326          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
025616  000167  174542          8$:     JMP     GT$DEV          ;;CONTROL-C RESTART
025622  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
025626  001005                          BNE     10$             ;;BRANCH IF NOT
025630  104401  026257                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
025634  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
025640  000737                          BR      19$             ;;LET'S TRY IT AGAIN
025642  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
025646  001022                          BNE     16$             ;;BRANCH IF NO
025650  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR.
025654  001403                          BEQ     11$             ;;BRANCH IF YES
025656  016677  000002  153254          MOV     2(SP),@SWR      ;;SAVE NEW SWR
025664  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
025670  104401  001171          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
025674  126727  153235  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
025702  001003                          BNE     15$             ;;BRANCH IF NOT
025704  012777  000100  153232          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
025712  000002                  15$:    RTI                     ;;RETURN
025714  004767  177462          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
025720  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
025724  002420                          BLT     18$             ;;BRANCH IF YES
025726  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
```

```
025732  003015                          BGT     18$             ;;BRANCH IF YES
025734  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
025740  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
025744  001403                          BEQ     17$             ;;BRANCH IF YES
025746  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
025750  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
025752  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
025754  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
025760  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
025764  000667                          BR      7$              ;;GET THE NEXT ONE
025766  104401  001170          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
025772  000720                          BR      20$             ;;SIMULATE CONTROL-U
                                .DSABL  LSB
                                ;;**************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;
025774  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
025776  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
026004  105777  153134          1$:     TSTB    @$TKS           ;;WAIT FOR
026010  100375                          BPL     1$              ;;A CHARACTER
026012  117766  153130  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
026020  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
026026  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
026034  001013                          BNE     3$              ;;BRANCH IF NO
026036  105777  153102          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
026042  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
026044  117746  153076                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
026050  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
026054  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
026060  001366                          BNE     2$              ;;IF NOT DISCARD IT
026062  000750                          BR      1$              ;;YES, RESUME
026064  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
026072  002407                          BLT     4$              ;;BRANCH IF YES
026074  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
026102  003003                          BGT     4$              ;;BRANCH IF YES
026104  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
026112  000002                  4$:     RTI                     ;;GO BACK TO USER
                                ;;**************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
026114  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
026116  012703  026242          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
026122  022703  026252          2$:     CMP     #$TTYIN+80.,R3  ;;BUFFER FULL?
026126  101415                          BLOS    4$              ;;BR IF YES
026130  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
026132  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
026134  122713  000003                  CMPB    #3,(R3)         ;;IS IT A CONTROL-C?
026140  001005                          BNE     10$             ;;BRANCH IF NO
026142  104401  026252                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
026146  012603                          MOV     (SP)+,R3        ;;RESTORE R3
```

```
026150  000167  174210                    JMP     GT$DEV          ;;GOTO CONTROL-C RESTART
026154  122713  000177          10$:      CMPB    #177,(R3)       ;;IS IT A RUBOUT
026160  001003                            BNE     3$              ;;SKIP IF NOT
026162  104401  001170          4$:       TYPE    ,$QUES          ;;TYPE A '?'
026166  000753                            BR      1$              ;;CLEAR THE BUFFER AND LOOP
026170  111367  000044          3$:       MOVB    (R3),9$         ;;ECHO THE CHARACTER
026174  104401  026240                    TYPE    ,9$
026200  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
026204  001346                            BNE     2$              ;;LOOP IF NOT RETURN
026206  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
026212  104401  001172                    TYPE    ,$LF            ;;TYPE A LINE FEED
026216  012603                            MOV     (SP)+,R3        ;;RESTORE R3
026220  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
026222  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
026230  012766  026242  000004            MOV     #$TTYIN,4(SP)
026236  000002                            RTI                     ;;RETURN
026240     000          9$:       .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
026241     000                    .BYTE   0                       ;;TERMINATOR
026242                  $TTYIN:   .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
026252     136     103     015    $CNTLC:   .ASCIZ  /^C/<15><12>  ;;CONTROL "C"
026255     012     000
026257     136     125     015    $CNTLU:   .ASCIZ  /^U/<15><12>  ;;CONTROL "U"
026262     012     000
026264     136     107     015    $CNTLG:   .ASCIZ  /^G/<15><12>  ;;CONTROL "G"
026267     012     000
026271     015     012     123    $MSWR:    .ASCIZ  <15><12>/SWR = /
026274     127     122     040
026277     075     040     000
026302     040     040     116    $MNEW:    .ASCIZ  /  NEW = /
026305     105     127     040
026310     075     040     000
                           .EVEN
```

```
4536                                    .SBTTL  APT COMMUNICATIONS ROUTINE
                                        ;;*****************************************************************
026314  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
026322  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
026330  000403                  $ATYC   BR      $ATYC
026332  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
026340                          $ATYC:
026340  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
026342  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
026344  105767  000206                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
026350  001450                          BEQ     5$              ;;IF NOT:  BR
026352  122767  000001  152634          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
026360  001031                          BNE     3$              ;;IF NOT:  BR
026362  132767  000100  152625          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
026370  001425                          BEQ     3$              ;;IF NOT:  BR
026372  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
026376  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
026404  005767  152564          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
026410  001375                          BNE     1$              ;;IF NOT:  WAIT
026412  010067  152572                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
026416  105720                  2$      TSTB    (R0)+           ;;FIND END OF MESSAGE
026420  001376                          BNE     2$
026422  166700  152562                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
026426  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
026430  010067  152556                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
026434  012767  000004  152532          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
026442  000413                          BR      5$
026444  017667  000004  000016  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
026452  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
026460  016746  151312                  MOV     177776,-(SP)            ;;PUSH 177776 ON STACK
026464  004767  176500                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
026470  000000                  4$:     .WORD   0
026472                          5$:
026472  105767  000062          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
026476  001416                          BEQ     12$             ;;IF NOT:  BR
026500  005767  152510                  TST     $ENV            ;;RUNNING UNDER APT?
026504  001413                          BEQ     12$             ;;IF NOT:  BR
026506  005767  152462          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
026512  001375                          BNE     11$             ;;IF NOT:  WAIT
026514  017667  000004  152454          MOV     @4(SP),$FATAL   ;;GET ERROR #
026522  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
026530  005267  152440                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
026534  105067  000020          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
026540  105067  000013                  CLRB    $LFLG           ;;CLEAR LOG FLAG
026544  105067  000006                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
026550  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
026552  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
026554  000207                          RTS     PC              ;;RETURN
026556     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
026557     000          $LFLG:  .BYTE   0               ;;LOG FLAG
026560     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
             000200     APTSIZE=200
             000001     APTENV=001
             000100     APTSPOOL=100
             000040     APTCSUP=040
```

```
4538                                    .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
026562                                  $ERROR:
026562  104407                                 CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
026564  105267  152313                  7$:    INCB    $ERFLG                  ;;SET THE ERROR FLAG
026570  001775                                 BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
026572  016777  152304  152342               MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
026600  032777  002000  152332               BIT     #BIT10,@SWR             ;;BELL ON ERROR?
026606  001402                               BEQ     1$                      ;;NO - SKIP
026610  104401  001164                       TYPE    ,$BELL                  ;;RING BELL
026614  005267  152272                  1$:    INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
026620  011667  152272                       MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
026624  162767  000002  152264               SUB     #2,$ERRPC
026632  117767  152260  152254               MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
026640  032777  020000  152272               BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
026646  001004                               BNE     20$                     ;;SKIP TYPEOUTS
026650  004767  174042                       JSR     PC,MYTYPE               ;;GO TO USER ERROR ROUTINE
026654  104401  001171                       TYPE    ,$CRLF
026660                                  20$:
026660  122767  000001  152326               CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
026666  001007                               BNE     2$                      ;;NO,SKIP APT ERROR REPORT
026670  116767  152220  000004               MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
026676  004767  177430                       JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
026702     000                          21$:   .BYTE   0
026703     000                                 .BYTE   0
026704  000777                          22$:   BR      22$                     ;;APT ERROR LOOP
026706  005777  152226                  2$:    TST     @SWR                    ;;HALT ON ERROR
026712  100002                               BPL     3$                      ;;SKIP IF CONTINUE
026714  000000                               HALT                            ;;HALT ON ERROR!
026716  104407                               CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
026720  032777  001000  152212          3$:    BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
026726  001402                               BEQ     4$                      ;;BR IF NO
026730  016716  152154                       MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
026734  005767  152222                  4$:    TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
026740  001402                               BEQ     5$                      ;;BR IF NONE
026742  016716  152214                       MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
026746                                  5$:
026746  022737  024752  000042               CMP     #$ENDAD,@#42            ;;ACT-11 AUTO-ACCEPT?
026754  001001                               BNE     6$                      ;;BRANCH IF NO
026756  000000                               HALT                            ;;YES
026760                                  6$:
026760  000002                               RTI                             ;;RETURN
```

```
4540                                    .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
026762                                  $SCOPE:
026762  104407                                 CKSWR                   ;;TEST  FOR CHANGE IN SCOPE SWR
026764  032777  040000  152146  1$:      BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
026772  001114                          BNE     $OVER           ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
026774  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
026776  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
027002  012737  027022  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
027010  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
027014  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
027020  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
027022  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
027024  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
027030  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
027032                                  6$:;#####END OF CODE FOR THE XOR TESTER#####
027032  032777  000400  152100          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
027040  001404                          BEQ     2$              ;;BR IF NO
027042  127767  152072  152032          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
027050  001465                          BEQ     $OVER           ;;BR IF YES
027052  105767  152025          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
027056  001421                          BEQ     3$              ;;BR IF NO
027060  126767  152031  152015          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
027066  101015                          BHI     3$              ;;BR IF NO
027070  032777  001000  152042          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
027076  001404                          BEQ     4$              ;;BR IF NO
027100  016767  152004  152000  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
027106  000446                          BR      $OVER
027110  105067  151767          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
027114  005067  152040                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
027120  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
027122  032777  004000  152010  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
027130  001011                          BNE     1$              ;;BR IF YES
027132  005767  152044                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
027136  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
027140  005267  151740                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
027144  026767  152010  151732          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
027152  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
027154  012767  000001  151722  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
027162  016767  000052  151770          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
027170  105267  151706          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
027174  116767  151702  151776          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
027202  011667  151700                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
027206  011667  151676                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
027212  005067  151744                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
```

```
027216  112767  000001  151671          MOVB    #1,SERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
027224  016777  151652  151710  SOVER:  MOV     STSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
027232  016716  151450                  MOV     SLPADR,(SP)     ;;FUDGE RETURN ADDRESS
027236  000002                          RTI                     ;;FIXES PS
027240  003720                  SMXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
```

```
 4542                                   .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
 027242                                 $TYPDS:
 027242  010046                                 MOV     R0,-(SP)                ;;PUSH R0 ON STACK
 027244  010146                                 MOV     R1,-(SP)                ;;PUSH R1 ON STACK
 027246  010246                                 MOV     R2,-(SP)                ;;PUSH R2 ON STACK
 027250  010346                                 MOV     R3,-(SP)                ;;PUSH R3 ON STACK
 027252  010546                                 MOV     R5,-(SP)                ;;PUSH R5 ON STACK
 027254  012746  020200                         MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 027260  016605  000020                         MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 027264  100004                                 BPL     1$              ;;BR IF INPUT IS POS.
 027266  005405                                 NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 027270  112766  000055  000001                 MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 027276  005000                         1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
 027300  012703  027456                         MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 027304  112723  000040                         MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 027310  005002                         2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 027312  016001  027446                         MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 027316  160105                         3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
 027320  002402                                 BLT     4$              ;;BR IF DONE
 027322  005202                                 INC     R2              ;;INCREASE THE BCD DIGIT BY 1
 027324  000774                                 BR      3$
 027326  060105                         4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
 027330  005702                                 TST     R2              ;;CHECK IF BCD DIGIT=0
 027332  001002                                 BNE     5$              ;;FALL THROUGH IF 0
 027334  105716                                 TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 027336  100407                                 BMI     7$              ;;BR IF YES
 027340  106316                         5$:     ASLB    (SP)            ;;MSD?
 027342  103003                                 BCC     6$              ;;BR IF NO
 027344  116663  000001  177777                 MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 027352  052702  000060                 6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 027356  052702  000040                 7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 027362  110223                                 MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 027364  005720                                 TST     (R0)+           ;;JUST INCREMENTING
 027366  020027  000010                         CMP     R0,#10          ;;CHECK THE TABLE INDEX
 027372  002746                                 BLT     2$              ;;GO DO THE NEXT DIGIT
 027374  003002                                 BGT     8$              ;;GO TO EXIT
 027376  010502                                 MOV     R5,R2           ;;GET THE LSD
 027400  000764                                 BR      6$              ;;GO CHANGE TO ASCII
 027402  105726                         8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 027404  100003                                 BPL     9$              ;;BR IF NO
 027406  116663  177777  177776                 MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 027414  105013                         9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 027416  012605                                 MOV     (SP)+,R5        ;;POP STACK INTO R5
 027420  012603                                 MOV     (SP)+,R3        ;;POP STACK INTO R3
 027422  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
 027424  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
 027426  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
 027430  104401  027456                         TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
```

```
027434  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
027442  012616                          MOV     (SP)+,(SP)
027444  000002                          RTI                     ;;RETURN TO USER
027446  023420                  $DTBL:  10000.
027450  001750                          1000.
027452  000144                          100.
027454  000012                          10.
027456                          $DBLK:  .BLKW   4
```

```
 4544                                   .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
027466  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
027472  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
027500  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
027504  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
027510  000406                          BR      $TYPON
027512  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
027520  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
027526  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
027534  010346                          MOV     R3,-(SP)        ;;SAVE R3
027536  010446                          MOV     R4,-(SP)        ;;SAVE R4
027540  010546                          MOV     R5,-(SP)        ;;SAVE R5
027542  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
027546  005404                          NEG     R4
027550  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
027554  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
027560  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
027564  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
027570  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
027572  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
027574  000404                          BR      3$              ;;GO DO MSB
027576  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
027600  006105                          ROL     R5
027602  006105                          ROL     R5
027604  010503                          MOV     R5,R3
027606  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
027610  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
027614  100016                          BPL     7$              ;;BR IF NO
027616  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
027622  001002                          BNE     4$              ;;TEST FOR 0
027624  005704                          TST     R4              ;;SUPPRESS THIS 0?
027626  001403                          BEQ     5$              ;;BR IF YES
027630  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
027632  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
027636  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
```

```
027642  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
027646  104401  027706                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
027652  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
027656  003347                          BGT     2$              ;;BR IF MORE TO DO
027660  002402                          BLT     6$              ;;BR IF DONE
027662  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
027664  000744                          BR      2$              ;;GO DO THE LAST DIGIT
027666  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
027670  012604                          MOV     (SP)+,R4        ;;RESTORE R4
027672  012603                          MOV     (SP)+,R3        ;;RESTORE R3
027674  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
027702  012616                          MOV     (SP)+,(SP)
027704  000002                          RTI                     ;;RETURN
027706     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
027707     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
027710     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
027711     000                  $0FILL: .BYTE   0               ;;ZERO FILL SWITCH
027712  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.
027714  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
027716  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
027722  005740                          TST     -(R0)           ;;BACKUP BY 2
027724  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
027726  006300                          ASL     R0              ;;POSITION FOR INDEXING
027730  016000  027750                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
027734  000200                          RTS     R0              ;;GO TO ROUTINE
                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
027736  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
027740  016666  000004  000C02          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
027746  000002                          RTI                     ;;RESTORE THE PSW
                                .SBTTL  TRAP TABLE
                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.
                                ;       ROUTINE
                                ;       -------
027750  027736                  $TRPAD: .WORD   $TRAP2
027752  025170                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
027754  027512                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
027756  027466                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
027760  027526                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
027762  027242                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
027764  025522                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
027766  025452                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
027770  025774                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
027772  026114                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
027774  022364                          GTSDEV  ;;CALL=GTDEVM   TRAP+12(104412) ^C WILL OPEN UP DEV MAP


                                        ;MAX ADDR FOR 4K ALLOWED FOR APT = 17400
                                        ;MAX ADDR FOR 4K FOR ABS LOADER ONLY = 17500
        000001                  .END
```

SYMBOL TABLE

```
ABASE = 000000        BITHLD  022640        C1DROP  023146        M104K   024554        PCRREG  001310
ACDW1 = 000000        BIT0   = 000001        C1END   014712        M108K   024563        PDR0   = 172300
ACDW2 = 000000        BIT00  = 000001        C1XMIT  014710        M112K   024572        PDR1   = 172302
ACPUOP= 000000        BIT01  = 000002        C2END   020006        M116K   024601        PDR2   = 172304
ADDR    003324        BIT02  = 000004        C2XMIT  020004        M12K    024341        PDR3   = 172306
ADDW0 = 000000        BIT03  = 000010        C3END   020012        M120K   024610        PDR4   = 172310
ADDW1 = 000000        BIT04  = 000020        C3XMIT  020010        M124K   024617        PDR5   = 172312
ADDW10= 000000        BIT05  = 000040        DATA    010016        M128K   024626        PDR6   = 172314
ADDW11= 000000        BIT06  = 000100        DDISP  = 177570        M132K   024635        PDR7   = 172316
ADDW12= 000000        BIT07  = 000200        DDRMSG  023442        M16K    024347        PERR   = 010000
ADDW13= 000000        BIT08  = 000400        DEVM    024316        M20K    024355        PHASE2  022356
ADDW14= 000000        BIT09  = 001000        DISPLA  001142        M24K    024363        PIRQ   = 177772
ADDW15= 000000        BIT1   = 000002        DISPRE  000174        M28K    024371        PIRQVE= 000240
ADDW2 = 000000        BIT10  = 002000        DLADD   001312        M32K    024377        PR0    = 000000
ADDW3 = 000000        BIT11  = 004000        DLVEC   001324        M36K    024405        PR1    = 000040
ADDW4 = 000000        BIT12  = 010000        DONE   = 000200        M4K     024327        PR2    = 000100
ADDW5 = 000000        BIT13  = 020000        DSWR   = 177570        M40K    024413        PR3    = 000140
ADDW6 = 000000        BIT14  = 040000        EMTVEC= 000030        M44K    024421        PR4    = 000200
ADDW7 = 000000        BIT15  = 100000        ERRCHK  010576        M48K    024427        PR5    = 000240
ADDW8 = 000000        BIT2   = 000004        ERRCNT  007740        M52K    024435        PR6    = 000300
ADDW9 = 000000        BIT3   = 000010        ERRFLG  005524        M56K    024443        PP7    = 000340
ADEVCT= 000000        BIT4   = 000020        ERROR  = 104000        M60K    024451        PS     = 177776
ADEVM = 000000        BIT5   = 000040        ERRVEC= 000004        M64K    024457        PSW    = 177776
AENV  = 000000        BIT6   = 000100        ERR15  = 100000        M68K    024465        PWRVEC= 000024
AENVM = 000000        BIT7   = 000200        EXFLG   005526        M72K    024473        P1CNT   022360
AFATAL= 000000        BIT8   = 000400        EX1     007736        M76K    024501        RAMROM  002066
AMADR1= 000000        BIT9   = 001000        EX2     010574        M8K     024334        RBUF    001316
AMADR2= 000000        BKGND   003002        FRERR  = 020000        M80K    024507        RCSR    001314
AMADR3= 000000        BPTVEC= 000014        GTDEVM= 104412        M84K    024515        RDCHR  = 104410
AMADR4= 000000        BREAK  = 000001        GTSWR  = 104406        M88K    024523        RDLIN  = 104411
AMAMS1= 000000        BRK     022350        GTSDEV  022364        M92K    024531        RDWRD   003004
AMAMS2= 000000        CARR    001334        HICHAN  022362        M96K    024537        RDY    = 000200
AMAMS3= 000000        CHOR    012450        HIROM   001276        NOCHAN  021414        REC     010020
AMAMS4= 000000        CHOTAB  013704        HIROM2  001302        NOCH1   023227        REGHLD  022636
AMSGAD= 000000        CHOX    012464        HT     = 000011        NOCLK   023750        RESVEC= 000010
AMSGLG= 000000        CH1R    012506        IE     = 000100        NODDR   024012        RHLD    010124
AMSGTY= 000000        CH1TAB  014304        ILLCLK  022560        NOLED   023176        ROSRV   013474
AMTYP1= 000000        CH1X    012524        ILLMSG  023511        NOMMNG  023630        R1SRV   013600
AMTYP2= 000000        CH2R    015540        INTABL  012546        NOPCR   024061        R2SRV   016574
AMTYP3= 000000        CH2TAB  017004        INTABM  015646        NOPCR2  023317        R3SRV   016700
AMTYP4= 000000        CH2X    015560        INTFLG  021530        NORAM   024163        R6     =%000006
APASS  = 000000        CH3R    015602        INTSRV  021522        NOROM   024140        R7     =%000007
APRIOR= 000000        CH3TAB  017404        IOTVEC= 000020        NOSMX   023675        SAVBK   003006
APTCSU= 000040        CH3X    015624        LEDHLD  001332        NOTST   010600        SCOPE  = 000004
APTENV= 000001        CKSWR  = 104407        LEDREG  001304        NUMBER  007742        SET    = 177777
APTSIZ= 000200        CLKDS   023277        LF     = 000012        ODD     022346        SIZE    020014
APTSPD= 000100        CLKEN   024207        LKSREG  001306        ORERR  = 040000        SLU     003726
ASWREG= 000000        CLKVEC= 000100        LNKMSG  023375        PARC   = 172340        SPTMR   014714
ATESTN= 000000        CLR    = 000000        LOOP    003742        PAR1   = 172342        SR0    = 177572
AUNIT  = 000000        CONSOL  022354        LOROM   001274        PAR2   = 172344        SR3    = 172516
AUSER  = 000000        CR     = 000015        LOROM2  001300        PAR3   = 172346        STACK  = 001100
AVECT1= 000000        CRLF   = 000200        MAXMEM  021520        PAR4   = 172350        START   001414
AVECT2= 000000        CYCLE   021532        MEM     024644        PAR5   = 172352        STATEN  014704
BASE0   001254        CODROP  023116        MEMTBL  021416        PAR6   = 172354        STKLMT= 177774
BASE1   001260        COEND   014706        MODTST  011504        PAR7   = 172356        STTEND  020004
BASE2   001264        COXMIT  014704        MYTYPE  022716        PASS    005522        SWR     001140
BASE3   001270        C1CON   024115        M100K   024545        PCRMSG  023561        SWREG   000176
```

```
SW0    = 000001        TST13   005712  $BDDAT  001126  $INTAG  001135  $RDLIN  026114
SW00   = 000001        TST14   006060  $BELL   001164  $ITEMB  001114  $RDSZ = 000010
SW01   = 000002        TST15   006416  $CHARC  025446  $LF     001172  $RTNAD  024764
SW02   = 000004        TST16   006662  $CKSWR  025452  $LFLG   026547  $SAVR6  025156
SW03   = 000010        TST17   007210  $CMTAG  001100  $LPADR  001106  $SCOPE  026762
SW04   = 000020        TST2    002244  $CM3  = 000000  $LPERR  001110  $SETUP= 000137
SW05   = 000040        TST20   007414  $CNTLC  026252  $LSTCN= 177777  $STUP = 177777
SW06   = 000100        TST21   010126  $CNTLG  026264  $LSTIN= 000000  $SVLAD  027170
SW07   = 000200        TST22   010600  $CNTLU  026257  $LSTST= 177777  $SVPC = 000204
SW08   = 000400        TST23   011504  $CPUOP  001222  $LSTTA= 000000  $SWR  = 167400
SW09   = 001000        TST24   012556  $CRLF   001171  $MADR1  001226  $SWREG  001216
SW1    = 000002        TST25   014716  $DBLK   027456  $MADR2  001232  $SWRMK= 000000
SW10   = 002000        TST26   015656  $DEVCT  001204  $MADR3  001236  $TESTN  001200
SW11   = 004000        TST3    003010  $DEVM   001252  $MADR4  001242  $TIMES  001160
SW12   = 010000        TST35   003324  $DOAGN  024762  $MAIL   001174  $TKB    001146
SW13   = 020000        TST4    003512  $DTBL   027446  $MAMS1  001224  $TKS    001144
SW14   = 040000        TST5    004122  $ENDAD  024752  $MAMS2  001230  $TN   = 000027
SW15   = 100000        TST6    004324  $ENDCT  024720  $MAMS3  001234  $TPB    001152
SW2    = 000004        TST7    004542  $ENDMG  024771  $MAMS4  001240  $TPFLG  001157
SW3    = 000010        TWMXN   024226  $ENULL  024766  $MBADR  001002  $TPS    001150
SW4    = 000020        TWOMX   024265  $ENV    001214  $MFLG   026556  $TRAP   027714
SW5    = 000040        TYPDS = 104405  $ENVM   001215  $MNEW   026302  $TRAP2  027736
SW6    = 000100        TYPE  = 104401  $EOP    024664  $MSGAD  001210  $TRP  = 000013
SW7    = 000200        TYPOC = 104402  $EOPCT  024712  $MSGLG  001212  $TRPAD  027750
SW8    = 000400        TYPON = 104404  $ERFLG  001103  $MSGTY  001174  $TSTM   001004
SW9    = 001000        TYPOS = 104403  $ERMAX  001115  $MSWR   026271  $TSTNM  001102
TBITVE= 000014         VECT0   001256  $ERROR  026562  $MTYP1  001225  $TTYIN  026242
TBUF    001322         VECT1   001262  $ERRPC  001116  $MTYP2  001231  $TYPDS  027242
TCSR    001320         VECT2   001266  $ERRTB  001254  $MTYP3  001235  $TYPE   025170
TESTSA  002332         VECT3   001272  $ERTTL  001112  $MTYP4  001241  $TYPEC  025402
TIMER   022566         WAIT    022704  $ESCAP  001162  $MXCNT  027240  $TYPEX  025450
TIMER1  022642         WORDL   022344  $ETABL  001214  $NULL   001154  $TYPOC  027512
TKVEC = 000060         WRAP    022352  $ETEMD  001254  $NUTST= 000001  $TYPON  027526
TMP1    001326         X0SRV   013534  $FATAL  001176  $OCNT   027710  $TYPOS  027466
TMP2    001330         X1SRV   013640  $FFLG   026560  $OMODE  027712  $UNIT   001206
TPVEC = 000064         X2SRV   016634  $FILLC  001156  $OVER   027224  $UNITM  001010
TRAN    007744         X3SRV   016740  $FILLS  001155  $PASS   001202  $USWR   001220
TRAPVE= 000034         $APTHD  001000  $GDADR  001120  $PASTM  001006  $VECT1  001244
TRPCAT  001336         $ATYC   026340  $GDDAT  001124  $POWER  025160  $VECT2  001246
TRTVEC= 000014         $ATY1   026314  $GET42  024742  $PWRAD  025146  $XTSTR  026774
TRUE  = 000001         $ATY3   026322  $GTSWR  025522  $PWRDN  025006  $$GET4= 000000
TST1    002072         $ATY4   026332  $HD   = 000000  $PWRMG  025142  $OFILL  027711
TST10   004772         $AUTOB  001134  $HIBTS  001000  $PWRUP  025060  $61     010314
TST11   005254         $BASE   001250  $ICNT   001104  $QUES   001170  .$X   = 001000
TST12   005530         $BDADR  001122  $ILLUP  025152  $RDCHR  025774

. ABS.  027776      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  42136 WORDS  ( 165 PAGES)
DYNAMIC MEMORY:  20310 WORDS  ( 78 PAGES)
ELAPSED TIME:  00:07:51
CVMXCA,CVMXBA/CR/-SP=SYSMAC/MLB,CVMXBA
```

SYMBOL	VALUE	REFERENCES				
ABASE	= 000000	#13-794	15-888	15-888		
ACDW1	= 000000	15-888				
ACDW2	= 000000	15-888				
ACPUOP	= 000000	15-888	15-888			
ADDR	002322	*18-1193	*18-1221	*18-1246	*18-1272	#18-1283
ADDW0	= 000000	15-888				
ADDW1	= 000000	15-888				
ADDW10	= 000000	15-888				
ADDW11	= 000000	15-888				
ADDW12	= 000000	15-888				
ADDW13	= 000000	15-888				
ADDW14	= 000000	15-888				
ADDW15	= 000000	15-888				
ADDW2	= 000000	15-888				
ADDW3	= 000000	15-888				
ADDW4	= 000000	15-888				
ADDW5	= 000000	15-888				
ADDW6	= 0000C0	15-888				
ADDW7	= 000000	15-888				
ADDW8	= 000000	15-888				
ADDW9	= 000000	15-888				
ADEVCT	= 000000	15-888	15-888			
ADEVM	= 000000	#13-795	15-888	15-888		
AENV	= 000000	15-888	15-883			
AENVM	= 000000	15-888	15-888			
AFATAL	= 000000	15-888	15-888			
AMADR1	= 000000	15-888	15-888			
AMADR2	= 000000	15-888	15-838			
AMADR3	= 000000	15-888	15-888			
AMADR4	= 000000	15-888	15-888			
AMAMS1	= 000000	15-888	15-888			
AMAMS2	= 000000	15-888	15-888			
AMAMS3	= 000000	15-888	15-888			
AMAMS4	= 000000	15-888	15-888			
AMSGAD	= 000000	15-888	15-888			
AMSGLG	= 000000	15-888	15-888			
AMSGTY	= 000000	15-888	15-888			
AMTYP1	= 000000	15-888	15-888			
AMTYP2	= 000000	15-888	15-888			
AMTYP3	= 000000	15-888	15-888			
AMTYP4	= 000000	15-888	15-888			
APASS	= 000000	15-888	15-888			
APRIOR	= 000000	15-888				
APTCSU	= 000040	47-4532	#49-4536			
APTENV	= 000001	47-4532	49-4536	#49-4536	50-4538	
APTSIZ	= 000200	16-947	#49-4536			
APTSPD	= 000100	47-4532	49-4536	#49-4536		
ASWREG	= 000000	15-888	15-888			
ATESTN	= 000000	15-888	15-888			
AUNIT	= 000000	15-888	15-888			
AUSWR	= 000000	#13-797	15-888	15-888		
AVECT1	= 000000	#13-796	15-888	15-888		

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
AVECT2	= 000000	15-888	15-888							
BASE0	001254	#16-890	36-2873	37-2989	38-3140	38-3145	39-3258	39-3377	39-3387	42-4207
		42-4228								
BASE1	001260	#16-892	37-3010	37-3019	37-3101	38-3151	38-3156	39-3269	39-3278	39-3344
		39-3399	39-3409	42-3927	42-3939	42-4264	42-4282			
BASE2	001264	#16-894	36-2883	40-3478	40-3485	40-3569	40-3614	40-3616	40-3620	41-3686
		41-3692	41-3696	41-3732	41-3739	41-3796	41-3846	41-3856	42-3959	42-4289
BASE3	001270	#16-896	36-2890	40-3503	40-3510	40-3581	40-3626	40-3628	40-3632	41-3689
		41-3701	41-3705	41-3747	41-3754	41-3816	41-3868	41-3878	42-4309	
BITHLD	022640	*43-4399	43-4403	#43-4415						
BIT0	= 000001	#13-827	19-1368	35-2741	35-2772	42-3978	42-4046			
BIT00	= 000001	#13-827	13-827	13-850						
BIT01	= 000002	#13-827	13-827							
BIT02	= 000004	#13-827	13-827							
BIT03	= 000010	#13-827	13-827							
BIT04	= 000020	#13-827	13-827							
BIT05	= 000040	#13-827	13-827							
BIT06	= 000100	#13-827	13-827	13-837	13-849					
BIT07	= 000200	#13-827	13-827	13-836	13-848					
BIT08	= 000400	#13-827	13-827	51-4540						
BIT09	= 001000	#13-827	13-827	50-4538	51-4540					
BIT1	= 000002	#13-827	18-1173	35-2746	35-2776	42-3974				
BIT10	= 002000	#13-827	37-2972	37-2980	37-3047	37-3086	39-3219	39-3249	39-3319	40-3458
		41-3681	41-3710	42-4209	42-4241	42-4298	42-4318	50-4538		
BIT11	= 004000	#13-827	35-2752	42-4235	51-4540					
BIT12	= 010000	#13-827	13-844	17-1050	36-2864	42-4032	42-4098			
BIT13	= 020000	#13-827	13-843	36-2879	40-3454	41-3677	42-3913	42-3923	42-3948	42-3954
		42-4253	50-4538							
BIT14	= 040000	#13-827	13-842	51-4540						
BIT15	= 100000	#13-827	13-841	36-2869	37-2951	37-2978	37-3045	37-3084	39-3201	39-3247
		39-3317	42-3907	42-4223						
BIT2	= 000004	#13-827	16-956	16-983	17-1052	20-1447	22-1472	23-1538	24-1610	25-1689
		26-1775	27-1892	27-1911	28-1950	28-1971	29-2006	29-2007	29-2031	29-2051
		29-2120	30-2180	31-2274	31-2284	31-2316	31-2366	32-2393	32-2433	33-2484
		33-2627	35-2696	35-2719	35-2759	35-2809	36-2919	37-2944	37-2994	37-3024
		39-3194	39-3209	39-3391	39-3413	40-3447	40-3490	40-3515	41-3670	41-3688
		41-3691	41-3860	41-3882	42-3970					
BIT3	= 000010	#13-827	37-2974	37-3008	37-3099	39-3267	39-3342	42-4211	42-4276	
BIT4	= 000020	#13-827	42-4270	42-4293	42-4313					
BIT5	= 000040	#13-827								
BIT6	= 000100	#13-827	16-1003	16-1012	17-1046	18-1169	19-1364	19-1385	22-1475	23-1541
		24-1613	24-1663	25-1692	25-1728	25-1754	26-1778	27-1895	28-1953	29-2019
		30-2183	31-2287	32-2405	33-2487	35-2699	36-2860	37-2947	39-3197	40-3450
		41-3673	42-4013	42-4058						
BIT7	= 000200	#13-827	18-1289	42-3996	42-4069					
BIT8	= 000400	#13-827	37-2953	37-3005	37-3097	39-3203	39-3265	39-3340	42-3910	42-3920
		42-3930	42-3942	42-4262						
BIT9	= 001000	#13-827	27-1909	28-1969	29-2029	29-2049	29-2118	31-2314	31-2364	32-2431
		33-2625	35-2717	35-2757	35-2807	37-2992	37-3022	39-3207	39-3389	39-3411
		40-3488	40-3513	41-3684	41-3858	41-3880				
BKGND	003002	#17-1149								
BPTVEC	= 000014	#13-827								

SYMBOL	VALUE	REFERENCES								
BREAK	= 000001	#13-850	23-1558	23-1565	23-1568	23-1576	23-1578	23-1586	23-1591	35-2720
		35-2750	35-2784	35-2788	35-2801					
BRK	022350	23-1545	35-2708	*42-4237	*42-4239	*42-4272	*42-4274	*42-4295	*42-4297	*42-4315
		*42-4317	#42-4330							
CARR	001334	#16-918	20-1419	36-2916	42-4213					
CHOR	012450	37-2984	#38-3139							
CHOTAB	013704	39-3232	39-3285	39-3323	#39-3423					
CHOX	012464	37-2986	#38-3144							
CH1R	012506	37-3014	#38-3150							
CH1TAB	014304	39-3286	39-3348	#39-3424						
CH1X	012524	37-3016	#38-3155							
CH2R	015540	40-3480	#40-3613							
CH2TAB	017004	41-3716	41-3761	41-3795	#41-3893					
CH2X	015560	40-3482	#40-3619							
CH3R	015602	40-3505	#40-3625							
CH3TAB	017404	41-3762	41-3815	#41-3894						
CH3X	015624	40-3507	#40-3631							
CKSWR	= 104407	50-4538	50-4538	51-4540	#54-4546					
CLKDS	023277	42-3994	#44-4473							
CLKEN	024207	42-3980	#44-4488							
CLKVEC	= 000100	#16-914	*19-1374	*19-1375	*19-1405	*19-1406				
CLR	= 000000	#13-831								
CONSOL	022354	23-1547	26-1784	27-1901	28-1959	29-2010	30-2234	31-2277	32-2396	33-2493
		35-2706	*42-4247	*42-4284	#42-4332					
CR	= 000015	#13-827	44-4487	44-4489	44-4490	44-4492	44-4493	44-4494	44-4495	44-4496
		44-4497	44-4498	44-4499	44-4500	44-4501	44-4502	44-4503	44-4504	44-4505
		44-4506	44-4507	44-4508	44-4509	44-4510	44-4511	44-4512	44-4513	44-4514
		44-4515	44-4516	44-4517	44-4518	44-4519	44-4520	44-4521	44-4522	44-4523
		44-4524	44-4525	47-4532	47-4532					
CRLF	= 000200	#13-827	16-948	16-948	36-2871	36-2871	36-2900	36-2900	44-4445	44-4468
		44-4469	44-4470	44-4472	44-4473	44-4474	44-4476	44-4477	44-4478	44-4479
		44-4480	44-4481	44-4482	44-4483	44-4484	44-4485	44-4486	44-4488	44-4491
		47-4532	47-4532							
CYCLE	021532	20-1422	#42-4200	42-4226	42-4260					
CODROP	023116	42-3909	#44-4468							
COEND	014706	*39-3263	39-3334	39-3380	#39-3428					
COXMIT	014704	*39-3229	39-3380	*39-3382	39-3392	#39-3427				
C1CON	024115	42-3929	42-3941	#44-4485						
C1DROP	023146	42-3912	42-3922	#44-4469						
C1END	014712	*39-3284	39-3359	39-3402	#39-3430					
C1XMIT	014710	*39-3230	39-3402	*39-3404	39-3414	#39-3429				
C2END	020006	*41-3744	41-3808	41-3849	#41-3898					
C2XMIT	020004	*41-3713	41-3849	*41-3851	41-3861	#41-3897				
C3END	020012	*41-3760	41-3828	41-3871	#41-3900					
C3XMIT	020010	*41-3714	41-3871	*41-3873	41-3883	#41-3899				
DATA	010016	*33-2533	33-2546	33-2620	33-2628	#33-2634	34-2654	*34-2658	*34-2663	34-2664
DDISP	= 177570	#13-827	15-888	16-947						
DDRMSG	023442	42-4064	#44-4477							
DEVM	024316	42-4343	#44-4491							
DISPLA	001142	#15-888	*16-947	*16-947	50-4538	51-4540				
DISPRE	000174	#14-882	16-947							
DLADD	001312	#16-908	20-1424	20-1426	20-1429	20-1432	27-1929	36-2904	*40-3478	*40-3503

SYMBOL	VALUE	REFERENCES								
		*40-3569	*40-3581	*41-3732	*41-3747	*41-3796	*41-3816	*42-4207	*42-4228	*42-4264
		*42-4289	*42-4309	44-4457						
DLVEC	001324	#16-913	30-2190	31-2293	33-2523	36-2906	*40-3479	*40-3504	*40-3570	*40-3582
		*41-3737	*41-3748	*41-3797	*41-3817	*42-4208	*42-4229	*42-4265	*42-4290	*42-4310
		44-4459								
DONE	= 000200	#13-836	20-1439	27-1915	27-1932	28-1976	28-1992	29-2036	29-2124	32-2438
		35-2725	35-2763	35-2814	37-3000	37-3031	37-3114	39-3213	39-3224	40-3496
		40-3521	41-3697	41-3706						
DSWR	= 177570	#13-827	15-888	16-947						
EMTVEC	= 000030	#13-827	*16-947	*16-947						
ERRCHK	010576	*35-2714	*35-2741	*35-2746	35-2772	35-2776	#35-2842			
ERRCNT	007740	*33-2532	33-2555	33-2561	#33-2602	33-2622	*34-2660			
ERRFLG	005524	*26-1794	*26-1832	26-1850	#26-1872					
ERROR	= 104000	#13-827	33-2563							
ERRVEC	= 000004	#13-827	16-947	*16-947	*16-947	*18-1175	*18-1176	18-1279	*18-1279	*18-1280
		*18-1291	*18-1292	18-1323	*18-1323	*18-1324	*22-1479	*22-1480	*22-1495	*22-1496
		42-3957	*42-3958	*42-3963	*42-3967	42-3981	*42-3982	*42-3987	*42-3991	42-4001
		*42-4002	*42-4007	*42-4011	42-4019	*42-4020	*42-4025	*42-4029	42-4034	*42-4035
		*42-4040	*42-4044	42-4048	*42-4049	*42-4051	*42-4056	42-4060	*42-4061	*42-4063
		*42-4068	42-4071	*42-4072	*42-4074	*42-4079	*42-4081	*42-4082	42-4132	*42-4132
		*42-4133	51-4540	*51-4540	*51-4540	*51-4540				
ERR15	= 100000	#13-841	29-2075	29-2154	32-2453	34-2652				
EXFLG	005526	*26-1795	*26-1858	*26-1862	26-1864	#26-1873				
EX1	007736	33-2495	#33-2595							
EX2	010574	35-2710	#35-2836							
FRERR	= 020000	#13-843	33-2572	35-2744						
GNS	= ******	14-882	14-882	16-948	36-2871	36-2872	36-2874	36-2876	36-2882	36-2884
		36-2886	36-2889	36-2891	36-2893	36-2900	36-2903	36-2905	36-2907	44-4445
		44-4447	44-4450	44-4456	44-4458	54-4546	54-4546	54-4546	54-4546	54-4546
		54-4546	54-4546	54-4546	54-4546	54-4546	54-4546	54-4546	54-4546	54-4546
		54-4546	54-4546	54-4546	54-4546	54-4547	54-4547			
GTDEVP	= 104412	16-958	42-3933	42-3945	42-3965	42-3989	42-4009	42-4027	42-4042	#54-4547
GTSWR	= 104406	16-948	#54-4546							
GTSDEV	022364	#42-4343	48-4534	48-4534	54-4547					
HICHAN	022362	*42-3916	*42-3926	*42-3951	*42-3968	42-4204	#42-4336			
HIROM	001276	#16-899	18-1198	18-1224						
HIROM2	001302	#16-901	18-1249	18-1275						
HT	= 000011	#13-827	47-4532	47-4532						
IE	= 000100	#13-837	#13-849	24-1630	24-1638	24-1640	24-1648	24-1650	24-1658	24-1667
		25-1698	25-1705	25-1707	25-1714	25-1716	25-1723	25-1732	30-2209	30-2220
		30-2248	31-2301	31-2333	31-2360	33-2538	33-2540	33-2559	33-2560	33-2631
		34-2659	34-2666	34-2670	37-2990	37-2995	37-3020	37-3026	38-3141	38-3147
		38-3152	38-3158	39-3259	39-3261	39-3279	39-3281	39-3384	39-3394	39-3406
		39-3416	40-3496	40-3491	40-3511	40-3516	40-3615	40-3622	40-3627	40-3634
		41-3740	41-3742	41-3755	41-3757	41-3853	41-3863	41-3875	41-3885	
ILLCLK	022560	19-1405	#42-4388							
ILLMSG	023511	42-4388	#44-4478							
INTABL	012546	*37-2957	*37-2958	*37-2959	*37-2960	37-3050	37-3053	37-3089	37-3093	37-3104
		37-3108	#38-3166							
INTABM	015646	*40-3461	*40-3462	*40-3463	*40-3464	40-3535	40-3538	40-3572	40-3576	40-3584
		40-3588	#40-3643							
INTFLG	021530	*16-1005	16-1007	*18-1191	18-1193	*18-1242	18-1244	*18-1293	18-1295	*19-1376

SYMBOL	VALUE	REFERENCES								
		19-1386	*19-1392	19-1396	*22-1485	22-1487	*30-2188	30-2222	30-2238	*30-2250
		30-2256	*31-2299	31-2336	31-2351	*31-2362	31-2370	*42-4083	42-4089	42-4115
		*42-4186	#42-4189							
INTSRV	021522	16-926	18-1175	18-1291	19-1374	22-1479	30-2192	31-2294	42-4081	#42-4186
IOTVEC	= 000020	#13-827	*16-947	*16-947	*37-3120	*37-3121	*39-3365	*39-3366	*40-3595	*40-3595
		*41-3834	*41-3835							
LEDHLD	001332	#16-917								
LEDREG	001304	#16-902	*16-949	16-1006	16-1014	17-1048	18-1171	19-1366	22-1477	23-1543
		24-1615	24-1665	25-1694	25-1730	25-1756	26-1780	27-1897	28-1955	29-2021
		30-2185	31-2289	32-2407	33-2489	35-2701	36-2862	37-2949	39-3199	40-3452
		41-3675	42-4021	42-4062						
LF	= 000012	#13-827	44-4487	44-4489	44-4490	44-4492	44-4493	44-4494	44-4495	44-4496
		44-4497	44-4498	44-4499	44-4500	44-4501	44-4502	44-4503	44-4504	44-4505
		44-4506	44-4507	44-4508	44-4509	44-4510	44-4511	44-4512	44-4513	44-4514
		44-4515	44-4516	44-4517	44-4518	44-4519	44-4520	44-4521	44-4522	44-4523
		44-4524	44-4525	47-4532	47-4532					
LKSREG	001306	#16-906	19-1379	19-1385	42-3983	42-4050				
LNKMSG	023375	42-4052	#44-4476							
LOOP	003742	#20-1416	36-2921							
LOROM	001274	#16-898	18-1190	18-1215						
LOROM2	001300	#16-900	18-1241	18-1266						
MAXMEM	021520	*42-4096	*42-4097	#42-4175						
MEM	024644	42-4130	#44-4525							
MEMTBL	021416	42-4101	42-4127	#42-4141						
MODTST	011504	20-1449	#37-2929							
MYTYPE	022716	#44-4445	50-4538							
M100K	024545	42-4165	#44-4516							
M104K	024554	42-4166	#44-4517							
M108K	024563	42-4167	#44-4518							
M112K	024572	42-4168	#44-4519							
M116K	024601	42-4169	#44-4520							
M12K	024341	42-4143	#44-4494							
M120K	024610	42-4170	#44-4521							
M124K	024617	42-4171	#44-4522							
M128K	024626	42-4172	#44-4523							
M132K	024635	42-4173	#44-4524							
M16K	024347	42-4144	#44-4495							
M20K	024355	42-4145	#44-4496							
M24K	024363	42-4146	#44-4497							
M28K	024371	42-4147	#44-4498							
M32K	024377	42-4148	#44-4499							
M36K	024405	42-4149	#44-4500							
M4K	024327	42-4141	#44-4492							
M40K	024413	42-4150	#44-4501							
M44K	024421	42-4151	#44-4502							
M48K	024427	42-4152	#44-4503							
M52K	024435	42-4153	#44-4504							
M56K	024443	42-4154	#44-4505							
M60K	024451	42-4155	#44-4506							
M64K	024457	42-4156	#44-4507							
M68K	024465	42-4157	#44-4508							
M72K	024473	42-4158	#44-4509							

SYMBOL VALUE REFERENCES

```
M76K      024501   42-4159  #44-4510
M8K       024334   42-4142  #44-4493
M80K      024507   42-4160  #44-4511
M84K      024515   42-4161  #44-4512
M88K      024523   42-4162  #44-4513
M92K      024531   42-4163  #44-4514
M96K      024537   42-4164  #44-4515
NOCHAN    021414   20-1417  *42-3917  *42-3952  #42-4137  42-4200
NOCH1     023227   42-3932  42-3944  #44-4472
NOCLK     023750   42-3988  #44-4482
NODDR     024012   42-4026  #44-4483
NOLED     023176   42-4015  #44-4470
NOMMNG    023630   42-4041  #44-4480
NOPCR     024061   42-3998  #44-4484
NOPCR2    023317   42-4008  #44-4474
NORAM     024163   42-3972  #44-4487
NOROM     024140   42-3976  #44-4486
NOSMX     023675   42-3964  #44-4481
NOTST     010600  #36-2849  41-3836
NUMBER    007742  *33-2535  33-2546  #33-2603  33-2620  34-2658  34-2664
ODD       022346  *42-4233  *42-4268  *42-4292  *42-4312  #42-4329
ORERR   = 040000  #13-842  29-2057  29-2096  29-2136  33-2565
PAR0    = 172340  #13-856  17-1059  42-4036  42-4103
PAR1    = 172342  #13-857
PAR2    = 172344  #13-858
PAR3    = 172346  #13-859
PAR4    = 172350  #13-860
PAR5    = 172352  #13-861
PAR6    = 172354  #13-862  *17-1072  *17-1077  *17-1083  17-1084  *17-1086  17-1097  17-1100  *17-1106
                  17-1107  *17-1110  *17-1116  17-1117  *17-1119  17-1130  17-1133  *17-1139  17-1140
                  *42-4114
PAR7    = 172356  #13-863
PASS      005522  *26-1791  *26-1852  26-1853  #26-1871
PCRMSG    023561  42-4075  #44-4479
PCRREG    001310  #16-907  18-1294  18-1300  18-1301  18-1313  18-1314  42-4003  42-4073
PDR0    = 172300  #13-866  17-1060  42-4104
PDR1    = 172302  #13-867
PDR2    = 172304  #13-868
PDR3    = 172306  #13-869
PDR4    = 172310  #13-870
PDR5    = 172312  #13-871
PDR6    = 172314  #13-872
PDR7    = 172316  #13-873
PERR    = 010000  #13-844
PHASE2    022356  *20-1412  20-1445  36-2867  36-2912  *36-2914  *42-4216  #42-4334
PIRQ    = 177772  #13-827
PIRQVE  = 000240  #13-827
PR0     = 000000  #13-827  30-2214  31-2308  33-2517  37-3064  39-3292  40-3549  41-3768
PR1     = 000040  #13-827
PR2     = 000100  #13-827
PR3     = 000140  #13-827
PR4     = 000200  #13-827
```

SYMBOL	VALUE	REFERENCES								
PR5	= 000240	#13-827								
PR6	= 000300	#13-827								
PR7	= 000340	#13-827	16-927	22-1480	24-1621	30-2193	31-2295	31-2378	33-2526	33-2530
		37-2966	37-2985	37-2987	37-3015	37-3017	37-3076	39-3241	39-3254	39-3256
		39-3274	39-3276	39-3309	40-3470	40-3481	40-3483	40-3506	40-3508	40-3561
		41-3725	41-3735	41-3737	41-3750	41-3752	41-3785			
PS	= 177776	#13-827	13-827							
PSW	= 177776	#13-827								
PWRVEC	= 000024	#13-827	*16-947	*16-947	*46-4530	*46-4530	*46-4530	*46-4530	*46-4530	*46-4530
P1CNT	022360	*20-1414	36-2898	*36-2901	*42-4217	#42-4335				
RAMROM	002066	#16-962	42-4363	45-4528						
RBUF	001316	#16-910	*20-1426	*20-1427	22-1502	28-1991	29-2055	29-2096	29-2136	29-2154
		32-2409	32-2451	34-2650	35-2737	35-2752	35-2766	35-2805	35-2827	37-3119
		39-3218	39-3228	40-3594	*41-3692	*41-3693	41-3694	*41-3701	*41-3702	41-3703
RCSR	001314	#16-909	*20-1424	22-1483	22-1498	25-1698	25-1705	25-1707	25-1714	25-1716
		25-1723	25-1732	27-1914	27-1932	28-1975	28-1992	29-2035	29-2123	31-2301
		31-2333	31-2360	32-2437	33-2540	33-2560	34-2659	34-2656	35-2724	35-2762
		35-2813	37-3113	39-3212	39-3223					
RDCHR	= 104410	48-4534	#54-4546							
RDLIN	= 104411	#54-4546								
RDWRD	003004	#17-1150								
RDY	= 000200	#13-848	25-1739	26-1805	26-1828	26-1838	30-2197	31-2320	32-2418	
REC	010020	33-2525	#34-2649							
REGHLD	022636	*43-4398	43-4403	#43-4414	*43-4418	43-4422				
RESVEC	= 000010	#13-827								
RHLD	010124	33-2563	33-2565	33-2572	*34-2650	34-2652	34-2654	#34-2675		
ROSRV	013474	39-3253	#39-3377							
R1SRV	013600	39-3273	#39-3399							
R2SRV	016574	41-3734	#41-3846							
R3SRV	016700	41-3749	#41-3868							
R6	=%000006	#13-827	*16-947	*16-947	16-947					
R7	=%000007	#13-827								
SAVBK	003006	#17-1151								
SCOPE	= 000004	#13-827	16-976	17-1037	18-1159	19-1355	22-1464	23-1532	24-1602	25-1682
		26-1769	27-1885	28-1943	29-2004	30-2173	31-2272	32-2391	33-2477	35-2689
		36-2850	37-2936	39-3186	40-3439	41-3662	45-4528			
SET	= 177777	#13-830	26-1832	26-1850	26-1868	26-1862	26-1864			
SIZE	020014	16-960	#42-3907	42-4361						
SLU	003726	19-1370	#20-1412							
SPTMR	014714	*39-3298	*39-3301	#39-3431	*41-3774	*41-3777				
SR0	= 177572	#13-874	*17-1068	*17-1096	*17-1099	*17-1129	*17-1132	*17-1143	*42-4112	*42-4120
SR3	= 172516	#13-875								
STACK	= 001100	#13-827	16-947	16-962	20-1416					
START	001414	14-882	16-940	#16-943	42-3934	42-3946	42-3966	42-3990	42-4010	42-4028
		42-4043	46-4530							
STATEN	014704	39-3234	#39-3425							
STKLMT	= 177774	#13-827								
STTEND	020004	41-3718	#41-3895							
SWR	001140	#15-888	16-947	*16-947	16-947	*16-947	*16-947	16-948	36-2864	46-4530
		46-4530	48-4534	48-4534	50-4538	50-4538	50-4538	50-4538	51-4540	51-4540
		51-4540	51-4540	51-4540						
SWREG	000176	#14-882	16-947	16-948	48-4534	48-4534				

SYMBOL	VALUE	REFERENCES								
SW0	= 000001	#13-827								
SW00	= 000001	#13-827	13-827							
SW01	= 000002	#13-827	13-827							
SW02	= 000004	#13-827	13-827							
SW03	= 000010	#13-827	13-827							
SW04	= 000020	#13-827	13-827							
SW05	= 000040	#13-827	13-827							
SW06	= 000100	#13-827	13-827							
SW07	= 000200	#13-827	13-827							
SW08	= 000400	#13-827	13-827							
SW09	= 001000	#13-827	13-827							
SW1	= 000002	#13-827								
SW10	= 002000	#13-827								
SW11	= 004000	#13-827								
SW12	= 010000	#13-827								
SW13	= 020000	#13-827								
SW14	= 040000	#13-827								
SW15	= 100000	#13-827								
SW2	= 000004	#13-827								
SW3	= 000010	#13-827								
SW4	= 000020	#13-827								
SW5	= 000040	#13-827								
SW6	= 000100	#13-827								
SW7	= 000200	#13-827								
SW8	= 000400	#13-827								
SW9	= 001000	#13-827								
TBITVE	= 000014	#13-827								
TBUF	001322	#16-912	*20-1432	*20-1433	22-1510	26-1801	26-1817	26-1824	27-1912	28-1972
		29-2032	29-2052	29-2121	30-2253	31-2317	31-2367	32-2434	33-2628	35-2760
		35-2810	37-3111	39-3210						
TCSR	001320	#16-911	*20-1429	*20-1430	20-1438	20-1447	22-1472	22-1506	23-1538	23-1558
		23-1565	23-1568	23-1576	23-1578	23-1586	23-1591	24-1610	24-1630	24-1638
		24-1640	24-1648	24-1650	24-1658	24-1667	25-1689	25-1739	26-1775	26-1804
		26-1818	26-1828	26-1837	27-1892	27-1911	28-1950	28-1971	29-2006	29-2007
		29-2031	29-2051	29-2120	30-2180	30-2196	30-2209	30-2220	30-2248	31-2274
		31-2284	31-2316	31-2319	31-2366	32-2393	32-2417	32-2433	33-2484	33-2538
		33-2559	33-2627	33-2631	34-2670	35-2696	35-2719	35-2720	35-2750	35-2759
		35-2784	35-2787	35-2801	35-2809	36-2919	37-2944	39-3194	39-3209	39-3391
		39-3413	40-3447	41-3670	41-3860	41-3882				
TESTSA	002332	17-1053	#17-1058							
TIMER	022566	20-1437	26-1803	26-1836	27-1913	28-1974	29-2034	29-2122	30-2195	31-2318
		32-2416	32-2436	35-2723	35-2761	35-2786	35-2812	37-3112	39-3211	39-3222
		41-3695	41-3704	#43-4398						
TIMER1	022642	30-2221	31-2335	#43-4418						
TKVEC	= 000060	#13-827								
TMP1	001326	#16-915	*33-2545	*33-2548	*37-2999	*37-3002	*37-3030	*37-3033	*40-3495	*40-3498
		*40-3520	*40-3523							
TMP2	001330	#16-916	*33-2544	*33-2550						
TPVEC	= 000064	#13-827								
TRAN	007744	33-2529	#33-2619							
TRAPVE	= 000034	#13-827	*16-947	*16-947						
TRPCAT	001336	#16-925								

SYMBOL	VALUE	REFERENCES								
TRTVEC	= 000014	#13-827								
TRUE	= 000001	#13-832	20-1445	36-2867	36-2912	42-4216				
TST1	002072	#16-976								
TST10	004772	#25-1682								
TST11	005254	25-1744	#26-1769							
TST12	005530	26-1785	26-1811	26-1844	#27-1885					
TST13	005712	27-1900	27-1902	27-1921	#28-1943					
TST14	006060	28-1958	28-1960	28-1982	#29-2004					
TST15	006416	29-2009	29-2011	29-2042	29-2066	29-2086	29-2105	29-2130	29-2147	#30-2173
TST16	006662	30-2203	30-2228	#31-2272						
TST17	007210	31-2276	31-2278	31-2326	31-2342	#32-2391				
TST2	002244	16-1004	16-1016	16-1018	#17-1037					
TST20	007414	32-2395	32-2397	32-2424	32-2444	32-2465	#33-2477			
TST21	010126	33-2595	#35-2689							
TST22	010600	35-2731	35-2755	35-2765	35-2794	35-2820	35-2836	#36-2850		
TST23	011504	#37-2936								
TST24	012556	37-2955	37-3127	#39-3186						
TST25	014716	37-2976	39-3205	39-3367	#40-3439					
TST26	015656	40-3456	40-3460	40-3602	#41-3662					
TST3	003010	17-1054	17-1147	#18-1159						
TST35	003324	18-1174	18-1281	#18-1289						
TST4	003512	18-1290	18-1325	#19-1355						
TST5	004122	#22-1464								
TST6	004324	#23-1532								
TST7	004542	23-1548	#24-1602							
TWMXN	024226	42-3915	42-3925	42-3950	#44-4489					
TWOMX	024265	42-3956	#44-4490							
TYPDS	= 104405	36-2877	36-2887	36-2894	36-2908	45-4528	#54-4546			
TYPE	= 104401	16-948	20-1419	36-2871	36-2872	36-2874	36-2876	36-2878	36-2882	36-2884
		36-2886	36-2888	36-2889	36-2891	36-2893	36-2895	36-2900	36-2903	36-2905
		36-2907	36-2909	36-2916	42-3909	42-3912	42-3915	42-3922	42-3925	42-3929
		42-3932	42-3941	42-3944	42-3950	42-3956	42-3964	42-3972	42-3976	42-3980
		42-3989	42-3994	42-3998	42-4008	42-4015	42-4026	42-4041	42-4052	42-4064
		42-4075	42-4128	42-4130	42-4213	42-4343	42-4345	42-4360	42-4382	42-4383
		42-4388	44-4445	44-4447	44-4450	44-4456	44-4458	44-4460	45-4528	45-4528
		46-4530	47-4532	48-4534	48-4534	48-4534	48-4534	48-4534	48-4534	48-4534
		48-4534	48-4534	48-4534	48-4534	50-4538	50-4538	52-4542	53-4544	#54-4546
TYPOC	= 104402	36-2873	36-2875	36-2883	36-2885	36-2890	36-2892	36-2904	36-2906	42-4344
		44-4446	44-4451	44-4457	44-4459	48-4534	#54-4546			
TYPON	= 104404	#54-4546								
TYPOS	= 104403	44-4449	#54-4546							
VECT0	001256	#16-891	36-2875	37-2983	39-3252	42-4208	42-4229			
VECT1	001262	#16-893	37-3013	39-3272	42-4265					
VECT2	001266	#16-895	36-2885	40-3477	40-3479	40-3570	41-3731	41-3733	41-3797	42-4290
VECT3	001272	#16-897	36-2892	40-3502	40-3504	40-3582	41-3746	41-3748	41-3817	42-4310
WAIT	022704	29-2053	30-2236	30-2255	31-2348	31-2369	35-2751	35-2803	#43-4434	
WORDL	022344	*42-4231	*42-4267	*42-4291	*42-4311	#42-4328				
WRAP	022352	27-1899	28-1957	29-2008	31-2275	32-2394	33-2491	35-2704	*42-4243	*42-4245
		*42-4278	*42-4280	*42-4300	*42-4302	*42-4320	*42-4322	#42-4331		
XOSRV	013534	39-3255	#39-3387							
X1SRV	013640	39-3275	#39-3409							
X2SRV	016634	41-3736	#41-3856							

```
SYMBOL  VALUE           REFERENCES
X3SRV     016740         41-3751   #41-3878
$APTHD    001000         14-887    #14-887
$ASTAT  = ******         49-4536    49-4536
$ATYC     026340         49-4536   #49-4536
$ATY1     026314        #49-4536
$ATY3     026322         47-4532   #49-4536
$ATY4     026332        #49-4536    50-4538
$AUTOB    001134        #15-888    *16-948     48-4534     48-4534     48-4534
$BASE     001250        #15-888
$BDADR    001122        #15-888
$BDDAT    001126        #15-888
$BELL     001164        #15-888     50-4538     50-4538     50-4538
$CHARC    025446        *47-4532    *47-4532    47-4532    *47-4532    #47-4532
$CKSUM    025452        #48-4534    54-4546     54-4546
$CMTAG    001100        #15-888     16-947      16-947      16-947      16-947      16-947      16-947      16-947
$CM3    = 000000        #15-888     15-888
$CNTLC    026252         48-4534     48-4534     48-4534     48-4534    #48-4534
$CNTLG    026264         48-4534    #48-4534
$CNTLU    026257         48-4534    #48-4534
$CPUOP    001222        #15-888
$CRLF     001171        #15-888     36-2878     36-2888     36-2895     36-2909     42-4360     42-4383     44-4460     47-4532
                         47-4532     47-4532     48-4534     48-4534     48-4534     50-4538     50-4538     50-4538
$DBLK     027456         52-4542     52-4542    #52-4542
$DEVC*    001204        #15-888    *20-1444
$DEVM     001252        #15-888    *16-956     16-983      16-1003     16-1012     17-1046     17-1050     17-1052     18-1169
                         18-1173     18-1289     19-1364     19-1368     22-1475     23-1541     24-1613     24-1663     25-1692
                         25-1728     25-1754     26-1778     27-1895     27-1909     28-1953     28-1969     29-2019     29-2029
                         29-2049     29-2118     30-2183     31-2287     31-2314     31-2364     32-2405     32-2431     33-2487
                         33-2625     35-2699     35-2717     35-2757     35-2807     36-2860     36-2869     36-2879     37-2947
                         37-2951     37-2953     37-2972     37-2974     37-2978     37-2980     37-2992     37-30[illegible]     37-3008
                         37-3022     37-3045     37-3047     37-3084     37-3086     37-3097     37-3099     39-3197     39-3201
                         39-3203     39-3207     39-3219     39-3247     39-3249     39-3265     39-3267     39-3317     39-3319
                         39-3340     39-3342     39-3389     39-3411     40-3450     40-3454     40-3458     40-3488     40-3513
                         41-3673     41-3677     41-3681     41-3684     41-3710     41-3858     41-38d0     42-3907     42-3910
                         42-3913     42-3920     42-3923     42-3930     42-3942     42-3948     42-3954     42-3970     42-3974
                         42-3978     42-3996     42-4013     42-4032     42-4046     42-4058     42-4069     42-4098     42-4209
                         42-4211     42-4223     42-4235     42-4241     42-4253     42-4262     42-4270     42-4276     42-4293
                         42-4298     42-4313     42-4318     42-4344    *42-4359
$DOAGN    024762         45-4528     45-4528    #45-4528
$DTBL     027446         52-4542    #52-4542
$ENDAD    024752         14-885      16-948     #45-4528     50-4538
$ENDCT    024720         16-947     #45-4528
$ENDMG    024771         45-4528    #45-4528
$ENULL    024766         45-4528    #45-4528
$ENV      001214        #15-888     16-948      16-952      47-4532     49-4536     49-4536     50-4538
$ENVM     001215        #15-888     16-947      47-4532     47-4532     49-4536
$EOP      024664         20-1420     36-2917     42-4214    #45-4528
$EOPCT    024712        *16-947     #45-4528     45-4528
$ERFLG    001103        #15-888    *50-4538     50-4538     50-4538     51-4540     51-4540     51-4540    *51-4540     51-4540
                         51-4540
$ERMAX    001115        #15-888    *16-947     51-4540    *51-4540     51-4540     51-4540
$ERROR    026562         16-947    #50-4538
```

SYMBOL	VALUE	REFERENCES								
$ERRPC	001116	#15-888	44-4451	*50-4538	*50-4538	50-4538	50-4538	50-4538		
$ERRTB	001254	#16-888								
$ERTTL	001112	#15-888	36-2877	36-2887	36-2894	36-2908	*36-2911	*50-4538	50-4538	50-4538
$ESCAP	001162	#15-888	*16-947	50-4538	50-4538	50-4538	50-4538	*51-4540		
$ETABL	001214	#15-888								
$ETEND	001254	14-887	#15-888							
$FATAL	001176	#15-888	*44-4448	44-4449	*49-4536					
$FFLG	026560	*49-4536	*49-4536	49-4536	*49-4536	#49-4536				
$FILLC	001156	#15-888	47-4532	47-4532	47-4532					
$FILLS	001155	#15-888	47-4532	47-4532						
$GDADR	001120	#15-888								
$GDDAT	001124	#15-888								
$GET42	024742	#45-4528								
$GTSWR	025522	#48-4534	54-4546	54-4546						
$HD	= 000000	13-824	13-824	13-824						
$HIBTS	001000	#14-887								
$ICNT	001104	#15-888	*51-4540	51-4540	*51-4540	51-4540	51-4540			
$ILLUP	025152	46-4530	46-4530	#46-4530						
$INTAG	001135	#15-888	48-4534	48-4534	48-4534	48-4534				
$ITEMB	001114	#15-888	44-4448	*50-4538	50-4538	50-4538	50-4538			
$LF	001172	#15-888	47-4532	47-4532	48-4534	48-4534	48-4534	50-4538	50-4538	
$LFLG	026557	*49-4536	#49-4536							
$LPADR	001106	#15-888	*16-947	*51-4540	*51-4540	51-4540	51-4540	51-4540		
$LPERR	001110	#15-888	*16-947	*23-1555	*23-1563	*23-1575	*23-1585	*24-1628	*24-1637	*24-1647
		*24-1657	*25-1697	*25-1704	*25-1713	*25-1722	*25-1737	*27-1907	*27-1927	*28-1966
		*29-2024	*29-2074	*29-2093	*29-2112	*30-2194	*30-2246	*31-2298	*31-2358	*35-2712
		*35-2782	50-4538	51-4540	*51-4540	51-4540	51-4540			
$LSTCN	= 177777	#13-790								
$LSTIN	= 000000	#13-788								
$LSTST	= 177777	#13-791								
$LSTTA	= 000000	#13-789								
$MADR1	001226	#15-888								
$MADR2	001232	#15-888								
$MADR3	001236	#15-888								
$MADR4	001242	#15-888								
$MAIL	001174	14-887	14-887	#15-888	16-947	16-948	16-976	17-1037	18-1159	19-1355
		22-1464	23-1532	24-1602	25-1682	26-1769	27-1885	28-1943	29-2004	30-2173
		31-2272	32-2391	33-2477	35-2689	37-2936	39-3186	40-3439	41-3662	47-4532
		50-4538	51-4540							
$MAMS1	001224	#15-888								
$MAMS2	001230	#15-888								
$MAMS3	001234	#15-888								
$MAMS4	001240	#15-888								
$MBADR	001002	#14-887								
$MFLG	026556	*49-4536	49-4536	*49-4536	#49-4536					
$MMEM	026302	42-4345	48-4534	#48-4534						
$MSGAD	001210	#15-888	*49-4536	49-4536						
$MSGLG	001212	#15-888	*49-4536							
$MSGTY	001174	#15-888	49-4536	*49-4536	49-4536	*49-4536				
$MSWR	026271	48-4534	#48-4534							
$MTYP1	001225	#15-888								
$MTYP2	001231	#15-888								

SYMBOL	VALUE	REFERENCES
$MTYP3	001235	#15-888
$MTYP4	001241	#15-888
$MXCNT	027240	51-4540 51-4540 51-4540 #51-4540
$NULL	001154	#15-888 47-4532 47-4532 47-4532
$NWISI	= 000001	#16-976 16-976 #16-976 16-976 #17-1037 17-1037 #17-1037 17-1037 #18-1159 18-1159 #18-1159 #19-1355 19-1355 #19-1355 19-1355 #22-1464 22-1464 #22-1464 22-1464 #23-1532 23-1532 #23-1532 23-1532 #24-1602 24-1602 #24-1602 #25-1682 25-1682 #25-1682 25-1682 #26-1769 26-1769 #26-1769 26-1769 #27-1885 27-1885 #27-1885 27-1885 #28-1943 28-1943 #28-1943 #29-2004 29-2004 #29-2004 #30-2173 30-2173 #30-2173 30-2173 #31-2272 31-2272 #31-2272 31-2272 #32-2391 32-2391 #32-2391 #33-2477 33-2477 #33-2477 #35-2689 35-2689 #35-2689 35-2689 #36-2850 36-2850 #36-2850 #37-2936 37-2936 #37-2936 37-2936 #39-3186 39-3186 #39-3186 39-3186 #40-3439 40-3439 #40-3439 40-3439 #41-3662 41-3662 #41-3662 41-3662
$OCNT	027710	*53-4544 *53-4544 #53-4544
$OMODE	027712	*53-4544 *53-4544 53-4544 *53-4544 *53-4544 #53-4544
$OVER	027224	51-4540 51-4540 51-4540 51-4540 #51-4540
$PASS	001202	#15-888 *16-947 *45-4528 *45-4528 45-4528 45-4528 45-4528 51-4540 51-4540 51-4540
$PASTM	001006	#14-887
$POWER	025160	46-4530 #46-4530
$PURAD	025146	#46-4530
$PURDN	025006	16-947 #46-4530 46-4530
$PURMG	025142	#46-4530
$PURUP	025060	46-4530 #46-4530
$QUES	001170	#15-888 42-4382 47-4532 47-4532 48-4534 48-4534 48-4534 48-4534 50-4538 50-4538
$RDCHR	025774	#48-4534 54-4546 54-4546
$RDDEC	= ******	54-4546
$RDLIN	026114	#48-4534 54-4546 54-4546
$RDOCT	= ******	54-4546
$RDSZ	= 000010	#48-4534 48-4534
$RTNAD	024764	#45-4528
$R2A	= ******	54-4546
$SAVRE	= ******	54-4546
$SAVR6	025156	*46-4530 46-4530 *46-4530 *46-4530 #46-4530
$SCOPE	026762	16-947 37-3120 39-3365 40-3595 41-3834 #51-4540
$SETUP	= 000137	#16-942 16-942 #16-942 16-942 #16-942 16-942 #16-942 16-942 #16-942 16-942 #16-942 16-942 #16-942 16-947 16-947 16-947 16-947 16-947 16-947 16-947 16-947 16-947 16-947 16-947 16-947 16-948 16-948 16-948 45-4528 45-4528 48-4534 48-4534 50-4538 50-4538 50-4538 50-4538 51-4540
$STUP	= 177777	#16-942 #16-942 16-942 #16-942 #16-942 16-942 #16-942 #16-942 16-942 #16-942 #16-942 16-942 #16-942 #16-942 16-942 #16-942 #16-942 16-942
$SVLAD	027170	51-4540 #51-4540
$SVPC	= 000204	#14-885 14-885
$SWR	= 167400	#13-792 13-824 13-826 13-826 13-826 13-826 13-826 13-826 13-826 13-826 15-888 15-888 15-888 16-947 16-947 16-947 16-947 16-947 16-976 17-1037 18-1159 19-1355 22-1464 23-1532 24-1602 25-1682 26-1769 27-1885 28-1943 29-2004 30-2173 31-2272 32-2391 33-2477 35-2689 36-2850 37-2936 39-3186 40-3439 41-3662 45-4528 45-4528 45-4528 45-4528 45-4528 46-4530 50-4538 50-4538 50-4538 50-4538 50-4538 50-4538 50-4538 50-4538 50-4538 50-4538 50-4538 51-4540 51-4540 51-4540 51-4540 51-4540 51-4540

SYMBOL	VALUE	REFERENCES
		51-4540 51-4540 51-4540 51-4540 51-4540 51-4540 51-4540 51-4540 51-4540
		51-4540 51-4540 51-4540 51-4540 51-4540
$SWREG	001216	#15-888 16-947
$SWRMK	= 000000	13-826 13-826 13-826 13-826 13-826 13-826 13-826 13-826 13-826
		51-4540 51-4540 51-4540 51-4540 51-4540 51-4540 51-4540 51-4540 51-4540
		51-4540
STESTN	001200	#15-888 *16-976 *17-1037 *18-1159 *19-1355 *22-1464 *23-1532 *24-1602 *25-1682
		*26-1769 *27-1885 *28-1943 *29-2004 *30-2173 *31-2272 *32-2391 *33-2477 *35-2689
		*37-2936 *39-3186 *40-3439 *41-3662 44-4446 44-4453 *51-4540
STIMES	001160	#15-888 *16-947 *16-976 *17-1037 *18-1159 *19-1355 *22-1464 *23-1532 *24-1602
		*25-1682 *26-1769 *27-1885 *28-1943 *29-2004 *30-2173 *31-2272 *32-2391 *33-2477
		*35-2689 *36-2850 *37-2936 *39-3186 *40-3439 *41-3662 *45-4528 *51-4540 51-4540
		*51-4540 51-4540 51-4540
STKB	001146	#15-888 42-4352 48-4534 48-4534 48-4534 48-4534 48-4534 48-4534 48-4534
STKS	001144	#15-888 37-3010 37-3101 39-3269 39-3344 42-3927 42-3939 42-4282 42-4350
		48-4534 48-4534 48-4534 48-4534 48-4534 48-4534 48-4534 48-4534 48-4534
STN	= 000027	#13-793 13-824 16-976 16-976 #16-976 16-976 17-1037 17-1037 #17-1037
		17-1037 18-1159 18-1159 #18-1159 18-1159 19-1355 19-1355 #19-1355 19-1355
		22-1464 22-1464 #22-1464 22-1464 23-1532 23-1532 #23-1532 23-1532 23-1548
		24-1602 24-1602 #24-1602 24-1602 25-1682 25-1682 #25-1682 25-1682 25-1744
		26-1769 26-1769 #26-1769 26-1769 26-1785 26-1811 26-1844 27-1885 27-1885
		#27-1885 27-1885 27-1900 27-1902 27-1921 28-1943 28-1943 #28-1943 28-1943
		28-1958 28-1960 28-1982 29-2004 29-2004 #29-2004 29-2004 29-2009 29-2011
		29-2042 29-2066 29-2086 29-2105 29-2130 29-2147 30-2173 30-2173 #30-2173
		30-2173 30-2203 30-2228 31-2272 31-2272 #31-2272 31-2272 31-2276 31-2278
		31-2326 31-2342 32-2391 32-2391 #32-2391 32-2391 32-2395 32-2397 32-2424
		32-2444 32-2465 33-2477 33-2477 #33-2477 33-2477 33-2595 35-2689 35-2689
		#35-2689 35-2689 35-2731 35-2794 35-2820 35-2836 36-2850 36-2850 #36-2850
		37-2936 37-2936 #37-2936 37-2936 37-3127 39-3186 39-3186 #39-3186 39-3186
		40-3439 40-3439 #40-3439 40-3439 40-3602 41-3662 41-3662 #41-3662 41-3662
STPB	001152	#15-888 47-4532 47-4532 47-4532
STPFLG	001157	#15-888 47-4532 47-4532 47-4532
STPS	001150	#15-888 47-4532 47-4532 47-4532
STRAP	027714	16-947 #54-4546
STRAP2	027736	#54-4546 54-4546
STRP	= 000013	#54-4546 54-4546 54-4546 54-4546 54-4546 #54-4546 54-4546 54-4546 54-4546
		54-4546 #54-4546 54-4546 54-4546 54-4546 54-4546 #54-4546 54-4546 54-4546
		54-4546 54-4546 #54-4546 54-4546 54-4546 54-4546 54-4546 #54-4546 54-4546
		54-4546 54-4546 54-4546 #54-4546 54-4546 54-4546 54-4546 54-4546 #54-4546
		54-4546 54-4546 54-4546 54-4546 #54-4546 54-4546 54-4546 54-4546 54-4546
		#54-4546 54-4547 54-4547 54-4547 54-4547 #54-4547
STRPAD	027750	54-4546 #54-4546
STSTM	001004	#14-887
STSTNM	001102	#15-888 16-1010 17-1044 18-1167 19-1362 *20-1448 22-1473 23-1539 24-1611
		24-1661 25-1690 25-1726 25-1752 26-1776 27-1893 28-1951 29-2017 30-2181
		31-2285 32-2403 33-2485 35-2697 *36-2920 *45-4528 50-4538 50-4538 50-4538
		51-4540 51-4540 *51-4540 51-4540 51-4540 51-4540 51-4540
STTYIN	026242	48-4534 48-4534 48-4534 #48-4534
STYPBN	= ******	54-4546
STYPDS	027242	#52-4542 54-4546 54-4546
STYPE	025170	#47-4532 49-4536 54-4546 54-4546
STYPEC	025402	42-4366 47-4532 47-4532 47-4532 #47-4532 47-4532 48-4534

SYMBOL	VALUE	REFERENCES								
$TYPEX	025450	47-4532	47-4532	#47-4532						
$TYPOC	027512	#53-4544	54-4546	54-4546						
$TYPON	027526	53-4544	#53-4544	54-4546						
$TYPOS	027466	#53-4544	54-4546							
$UNIT	001206	#15-888	*20-1413	*36-2915	42-4204	*42-4206	42-4220	*42-4235	*42-4248	42-4251
		42-4255	42-4257	*42-4259	*42-4303	*42-4323				
$UNITM	001010	#14-887								
$USWR	001220	#15-888								
$VECT1	001244	#15-888								
$VECT2	001246	#15-888								
$XTSTR	026774	#51-4540								
$$GET4	= 000000	#45-4528	45-4528							
$OFILL	027711	*53-4544	*53-4544	53-4544	#53-4544					
$40CAT	= ******	50-4538	51-4540							
$61	010314	#35-2743								
.$ASTA	= ******	49-4536	49-4536							
.$X	= 001000	#14-887	14-887							

MACRO NAME	REFERENCES									
COMMEN	#13-827									
ENDCOM	#13-827									
ERROR	#13-814	#16-997	#16-1017	#17-1098	#17-1101	#17-1131	#17-1134	#18-1196	#18-1222	#18-1247
	#18-1273	#18-1297	#18-1305	#18-1318	#19-1388	#19-1398	#22-1489	#22-1500	#22-1504	#22-1508
	#22-1512	#23-1560	#23-1571	#23-1581	#23-1594	#24-1633	#24-1643	#24-1653	#24-1670	#25-1701
	#25-1710	#25-1719	#25-1735	#25-1749	#26-1806	#26-1839	#26-1857	#27-1916	#27-1935	#28-1977
	#28-1995	#29-2037	#29-2060	#29-2078	#29-2099	#29-2125	#29-2139	#29-2157	#30-2198	#30-2223
	#30-2241	#30-2258	#31-2321	#31-2337	#31-2354	#31-2372	#32-2419	#32-2439	#32-2456	#32-2460
	#33-2552	#33-2568	#33-2574	#33-2580	#33-2588	#35-2726	#35-2754	#35-2764	#35-2768	#35-2774
	#35-2778	#35-2789	#35-2815	#35-2830	#37-3004	#37-3035	#37-3091	#37-3095	#37-3108	#37-3110
	#37-3115	#39-3214	#39-3330	#39-3332	#39-3355	#39-3357	#40-3500	#40-3525	#40-3574	#40-3578
	#40-3586	#40-3590	#41-3804	#41-3806	#41-3824	#41-3826				
ESCAPE	#13-827									
GETPRI	#13-827									
GETSWR	#13-827	#16-948	16-948							
LPADR	#13-799	#23-1555	#23-1563	#23-1575	#23-1585	#24-1628	#24-1637	#24-1647	#24-1657	#25-1697
	#25-1704	#25-1713	#25-1722	#27-1907	#27-1927	#28-1966	#29-2024	#29-2074	#29-2093	#29-2112
	#30-2194	#30-2246	#31-2298	#31-2358	#35-2712	#35-2782				
MSG	#16-966	16-976	#17-1021	17-1037	#19-1330	19-1355	#21-1453	22-1464	#23-1524	23-1532
	#25-1678	25-1682	#26-1765	26-1769	#27-1880	27-1885	#30-2166	30-2173	#31-2267	31-2272
	#35-2683	35-2689	#37-2931	37-2936	#39-3171	39-3186	#40-3434	40-3439	#41-3647	41-3662
MULT	#13-827									
NEWTST	#13-827	#16-976	#17-1037	#18-1159	#19-1355	#22-1464	#23-1532	#24-1602	#25-1682	#26-1769
	#27-1885	#28-1943	#29-2004	#30-2173	#31-2272	#32-2391	#33-2477	#35-2689	#36-2850	#37-2936
	#39-3186	#40-3439	#41-3662							
POP	#13-827	#46-4530	#46-4530	#49-4536	#49-4536	#52-4542				
PUSH	#13-827	#46-4530	#46-4530	#49-4536	#49-4536	#49-4536	#52-4542			
REPORT	#13-785	#13-827								
SETPRI	#13-827	#24-1621	#30-2214	#31-2308	#31-2378	#33-2517	#37-2966	#37-3064	#37-3076	#39-3241
	#39-3292	#39-3309	#40-3470	#40-3549	#40-3561	#41-3725	#41-3768	#41-3785		
SETTRA	#54-4546	54-4546	54-4546	54-4546	54-4546	54-4546	54-4546	54-4546	54-4546	54-4546
	54-4547									
SETUP	#13-827	#16-947								
SKIP	#13-827	23-1548	25-1744	26-1785	26-1811	26-1844	27-1900	27-1902	27-1921	28-1958
	28-1960	28-1982	29-2009	29-2011	29-2042	29-2066	29-2086	29-2105	29-2130	29-2147
	30-2203	30-2228	31-2276	31-2278	31-2326	31-2342	32-2395	32-2397	32-2424	32-2444
	32-2465	33-2595	35-2731	35-2794	35-2820	35-2836	37-3127	40-3602		
SLASH	#13-827									
STARS	#13-827	#14-881	#14-885	#14-887	#14-887	#14-887	#15-888	#15-888	#15-888	#16-921
	#16-923	#16-976	#16-976	#17-1037	#17-1037	#18-1159	#18-1159	#18-1182	#18-1184	#18-1207
	#18-1209	#18-1233	#18-1235	#18-1258	#18-1260	#19-1355	#19-1355	#22-1464	#22-1464	#23-1519
	#23-1521	#23-1532	#23-1532	#24-1602	#24-1602	#25-1682	#25-1682	#26-1769	#26-1769	#27-1885
	#27-1885	#28-1943	#28-1943	#29-2004	#29-2004	#30-2173	#30-2173	#31-2272	#31-2272	#32-2391
	#32-2391	#33-2477	#33-2477	#33-2611	#33-2613	#34-2641	#34-2643	#35-2689	#35-2689	#36-2850
	#36-2850	#37-2936	#37-2936	#39-3186	#39-3186	#39-3373	#39-3375	#40-3439	#40-3439	#41-3662
	#41-3662	#41-3842	#41-3846	#42-4194	#42-4198	#42-4339	#42-4341	#43-4392	#43-4396	#45-4528
	#46-4530	#46-4530	#47-4532	#48-4534	#48-4534	#48-4534	#48-4534	#49-4536	#50-4538	#51-4540
	#52-4542	#53-4544	#54-4546							
SUBSU	#13-827	#16-947	16-947							
TRMTRP	#54-4546									
TYPBIN	#13-827									
TYPDEC	#13-827	#36-2877	#36-2887	#36-2894	#36-2908	#45-4528				

MACRO NAME	REFERENCES									
TYPNAM	#13-827	#16-948								
TYPNUM	#13-827									
TYPOCS	#13-827	#44-4449								
TYPOCT	#13-827	36-2873	36-2875	36-2883	36-2885	36-2890	36-2892	36-2904	36-2906	42-4344
	44-4446	44-4451	44-4457	44-4459	48-4534					
TYPTXT	#13-827	#36-2871	#36-2872	#36-2874	#36-2876	#36-2882	#36-2884	#36-2886	#36-2889	#36-2891
	#36-2893	#36-2900	#36-2903	#36-2905	#36-2907	#44-4445	#44-4447	#44-4450	#44-4456	#44-4458
WAITMS	#13-810									
$$CMRE	#14-888									
$$CMTM	#14-888									
$$ESCA	#13-827									
$$NEWT	#13-827	16-976	17-1037	18-1159	19-1355	22-1464	23-1532	24-1602	25-1682	26-1769
	27-1885	28-1943	29-2004	30-2173	31-2272	32-2391	33-2477	35-2689	36-2850	37-2936
	39-3186	40-3439	41-3662							
$$SET	#54-4546	#54-4546	#54-4546	#54-4546	#54-4546	#54-4546	#54-4546	#54-4546	#54-4546	#54-4546
	#54-4547									
$$SETM	#16-947	16-947								
$$SKIP	#13-827	23-1548	25-1744	26-1785	26-1811	26-1844	27-1900	27-1902	27-1921	28-1958
	28-1960	28-1982	29-2009	29-2011	29-2042	29-2066	29-2086	29-2105	29-2130	29-2147
	30-2203	30-2238	31-2276	31-2278	31-2324	31-2342	32-2395	32-2397	32-2424	32-2444
	32-2465	33-2595	35-2731	35-2794	35-2823	35-2836	37-3127	40-3602		
.EQUAT	#13-784	13-827								
.HEADE	#13-783	13-824								
.SETUP	#13-783	16-942								
.SWRHI	#13-784	13-826								
.SWRLO	#13-826									
.$ACT1	#13-785	14-865								
.$APTB	#13-785	#15-888	#15-888							
.$APTH	#13-785	#14-887								
.$APTY	#13-785	#49-4536								
.$CATC	#13-783	#14-882								
.$CMTA	#13-783	#14-888								
.$EOP	#13-783	45-4528								
.$ERRO	#13-784	50-4538								
.$POWE	#13-784	46-4530								
.$READ	#13-783	#48-4534								
.$SCOP	#13-784	51-4540								
.$TRAP	#13-784	#54-4546								
.$TYPD	#13-784	#52-4542								
.$TYPE	#13-783	47-4532								
.$TYPO	#13-785	#53-4544								