

MXV11-A

MXV11A DIAG
CVMXAA0

AH E657A MC

COPYRIGHT 1979

FICHE 1 OF 1

NOV 1979

digital

MADE IN USA

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-E656A-MC
PRODUCT NAME: CVMXAA0 Mxv11A DIAG
PRODUCT DATE: JUNE 1979
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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

COPYRIGHT (C) 1979 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

TABLE OF CONTENTS

1.0	GENERAL PROGRAM INFORMATION.
1.1	PROGRAM PURPOSE (ABSTRACT).
1.2	SYSTEM REQUIREMENTS.
1.3	RELATED DOCUMENTS AND STANDARDS.
1.4	DIAGNOSTIC HIERARCHY PREREQUISITES.
1.5	ASSUMPTIONS.
1.6	VT-100 CONSOLE SETUP.
2.0	OPERATING INSTRUCTIONS.
2.1	LOADING AND STARTING PROCEDURES.
2.2	SPECIAL ENVIRONMENTS.
2.3	OPERATIONAL SWITCH SETTINGS
2.4	PROGRAM OPTIONS & DEFAULTS.
2.5	EXECUTION TIMES.
2.6	POWER FAIL.
3.0	ERROR INFORMATION.
3.1	ERROR REPORTING PROCEDURE.
3.2	ERROR HALTS.
4.0	PERFORMANCE AND PROGRESS REPORTS.
4.1	PERFORMANCE REPORTS.
5.0	DEVICE INFORMATION TABLES.
6.0	SUMMARY OF TESTS.

1.0 GENERAL PROGRAM INFORMATION.

1.1 PROGRAM PURPOSE (ABSTRACT).

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

SERIAL LINE UNIT TESTING IS DONE IN TWO DISTINCT PHASES:

1. EACH OF THE 2 CHANNELS OF THE MXV11-A IS TESTED INDIVIDUALLY.
2. THE MXV11-A 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 PROGRAM WILL TEST TO WHATEVER OPTIONS THE DEVICE MAP (\$DEVN)
IS SET TO. SEE PROGRAM OPTIONS & DEFAULTS SEC 2.4 .
THE PROGRAM WILL PRINT THE CONTENTS OF \$DEVN 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.

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

THE ADDRESSES & VECTOR RANGES ARE AS FOLLOWS:

CHANNEL 0: 176500 THRU 176536
CHANNEL 1: 176500 THRU 176576
 177500 THRU 177576 (CAN BE CONSOLE)
VECTORS : 0 THRU 374

THE DEFAULT ADDRESSES & VECTORS ARE AS FOLLOWS:

CHANNEL 0: 176500 THRU 176506 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 & DEFAULTS SEC. 2.4 .

THIS PROGRAM IS DESIGNED TO RUN ON ANY Q-BUS PDP-11 WITH 4K OF
MEMORY AND AN MXV11-A (Q-BUS) MODULE. IT CAN RUN UNDER XXDP &
APT MONITORS, AND ON PROCESSORS WITH NO HARDWARE
SWITCH REGISTER.

1.2 SYSTEM REQUIREMENTS.

HARDWARE REQUIREMENTS:

ANY Q-BUS PDP-11 FAMILY PROCESSOR
4K MEMORY - MINIMUM
A SPECIAL DATA WRAP AROUND CONNECTOR (PN # H3270-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.

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 (\$DEVN) 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)

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:

1310 START: LOADS ADDRESSES 0 TO 400 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 OCCURRING.
NORMAL TESTING WILL THEN BEGIN.

1332 START: LOADS ADDRESSES 0 TO 400 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 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:
 - A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED). LEADING ZEROS NEED NOT BE TYPED, AND IF MORE THAN 6 DIGITS ARE TYPED THE LAST 6 WILL BE USED. IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.
 - B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 3.
 - C) IF THE INPUT CHARACTER IS NOT ONE OF THE CHARACTERS MENTIONED ABOVE THEN A QUESTION MARK (?) WILL BE TYPED FOLLOWED BY A CARRAGE RETURN AND A LINE FEED SEQUENCE THEN PROCEED FROM STEP 3 (ERASING ALL PREVIOUS INPUT).
- 4) THE DIAGNOSTIC WILL CONTINUE ON TYPING <CR>.

NOTE: BECAUSE OF 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

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	176500	RCSR
	VECT0	1256	300	VECTOR
CHANNEL 1	BASE1	1260	177560	RCSR
	VECT1	1262	60	VECTOR

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

\$DEVN DEFAULT = 000000

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

CHANNEL 0:

BIT 15:	0 = TEST CHANNEL 0	DEFAULT
	1 = BYPASS CHANNEL 0 TEST	= 100000
BIT 14:	0 = 8 BITS/WORD (NO PARITY)	= DEFAULT
	1 = 7 BITS/WORD (WITH PARITY)	= 40000
BIT 12:	0 = ODD PARITY	DEFAULT
	1 = EVEN PARITY	= 10000
BIT 11:	0 = BREAK DETECTION DISABLED	= DEFAULT
	1 = BREAK DETECTION ENABLED	= 4000
BIT 10:	0 = DO DATA WRAP AROUND TESTS	= DEFAULT
	1 = BYPASS DATA WRAP TESTS	= 2000

CHANNEL 1:

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

BIT 7: 0 = 8 BITS/WORD (NO PARITY) = DEFAULT
 1 = 7 BITS/WORD (WITH PARITY) = 200

BIT 5: 0 = ODD PARITY = DEFAULT
 1 = EVEN PARITY = 40

BIT 4: 0 = BREAK DETECTION DISABLED = DEFAULT
 1 = BREAK DETECTION ENABLED = 20

BIT 3: 0 = DO DATA WRAP AROUND TESTS = DEFAULT
 1 = BYPASS DATA WRAP TESTS = 10

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

BIT 1: 0 = ROM PRESENT = DEFAULT
 1 = BYPASS ROM TESTING = 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-A,
 THE CONSOLE MUST NOT BE TESTED SINCE APT SENDS 'BREAKS'
 TO THE CONSOLE WHICH INTERFERES WITH THE TESTS.

SUMMARY OF USER LOCATIONS & DEFAULTS.

	LOC	DEFAULT	
\$DEV	1252	0	DEVICE MAP
SWREG	176	0	SOFTWARE SWITCH REGISTER
BASE0	1254	176500	CHANNEL 0
VECT0	1256	300	CHANNEL 0
BASE1	1260	177560	CHANNEL 1
VECT1	1262	60	CHANNEL 1
LOROM	1264	173000	LOW ROM ADDRESS
HIROM	1266	173776	HIGH ROM ADDRESS (256 WORDS)

2.5 EXECUTION TIMES.

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

CH. 0 AT 38.4K BAUD.
 CH. 1 (CONSOLE) AT 9600 BAUD.

	LSI-11	F-11	
	-----	----	
ARE: FIRST PASS-	25 SEC	15 SEC	FOR SLU TESTS ONLY
ADDITIONAL PASSES	55 SEC	30 SEC	FOR SLU TESTS ONLY

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
	-----		----
4K	15 SEC		10 SEC
8K	40 SEC		15 SEC
12K	1:00	MIN	30 SEC
16K	1:20		40 SEC
20K	1:40		50 SEC
24K	2:00		1:00 MIN
28K	2:20		1:20 MIN

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 4K OF MEMORY THE
ERROR TYPEOUT IS VERY BRIEF. THE FORMAT OF THE ERROR TYPEOUT
IS AS FOLLOWS:

TEST#____,ERROR#____,PC-____,ADDRESS-____,VECTOR=____

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

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

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

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

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

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

3.2 ERROR HALTS. -----

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

4.0 PERFORMANCE AND PROGRESS REPORTS.

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

AS EACH CHANNEL 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
16K RAM
CLOCK ENABLED
NO ERRORS

CVMXAAO MXV11A DIAGNOSTIC

SWR- 000000 NEW- 10000<CR>

(ENABLE PERFORMANCE REPORTS)

DEVM = 000000 NEW - 201<CR>

(CH.1 = 7 BITS/WORD (WITH PARITY), CLOCK ENABLED)

16K MEMORY
CLOCK ENABLED

** PHASE 1 SUMMARY **

CSR: 176500, VECTOR: 000760, ERRORS: 0
CSR: 177500, VECTOR: 000770, ERRORS: 0

(CHANNEL 0)
(CHANNEL 1)

** PHASE 2 SUMMARY **

CSR: 176500, 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' & '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																
RBUF	ERRO	OR	FR	P												
	R	ERR	ERR	ERR												
TCSR																
TBUF																

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
 PERR = PARITY ERROR

6.0 SUMMARY OF TESTS AND SPECIAL SUBROUTINES.

PHASE 1 TESTS

TEST 1 RAM ADDRESS TEST

WRITE ENTIRE MEMORY WITH ADDRESS AS DATA. READ MEMORY TO VERIFY.

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.

TEST 3 ROM TESTS

TEST 4 CLOCK TESTS

TEST 5 ADDRESSABILITY

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

THE FOLLOWING 3 TESTS TEST ALL 'READ WRITE' BITS

TEST 6 BREAK - TCSR 0 SET, CLEAR, RESET

TEST 7 XMITIE - TCSR 6 SET, CLEAR, RESET

TEST 10 RCVRIE - RCSR 6 SET, CLEAR, RESET

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

TEST 13 RCVRDONE IS CLEARED BY READING RBUF
---- --

TEST 14 OVERRUN & ERROR BIT - RBUF 14
---- --

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

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

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

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

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 22 NOT A TEST - SEND BACK TO LOOP

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

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

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

```

634
635      .TITLE  CVMXAAO MXV11A DIAG
636      ;*COPYRIGHT (C) 1978
637      ;*DIGITAL EQUIPMENT CORP.
638      ;*MAYNARD, MASS. 01754
639      ;*
640      ;*PROGRAM BY GARY PAPAZIAN
641      ;*
642      ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
643      ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
644      ;*
645
646      .SBTTL  OPERATIONAL SWITCH SETTINGS
647      ;*
648      ;*      SWITCH                      USE
649      ;*      -----
650      ;*      15                      HALT ON ERROR
651      ;*      14                      LOOP ON TEST
652      ;*      13                      INHIBIT ERROR TYPEOUTS
653      ;*      11                      INHIBIT ITERATIONS
654      ;*      10                      BELL ON ERROR
655      ;*      9                       LOOP ON ERROR
656      ;*      8                      LOOP ON TEST IN SWR<7:0>
657      .SBTTL  BASIC DEFINITIONS
658
659      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
660      001100      STACK= 1100
661      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
662      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
663
664      ;*MISCELLANEOUS DEFINITIONS
665      000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
666      000012      LF= 12      ;;CODE FOR LINE FEED
667      000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
668      000200      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED
669      177776      PS= 177776   ;;PROCESSOR STATUS WORD
670      .EQUIV  PS,PSW
671      177774      STKLMT= 177774 ;;STACK LIMIT REGISTER
672      177772      PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
673      177570      DSWR= 177570 ;;HARDWARE SWITCH REGISTER
674      177570      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
675
676      ;*GENERAL PURPOSE REGISTER DEFINITIONS
677      000000      R0= %0      ;;GENERAL REGISTER
678      000001      R1= %1      ;;GENERAL REGISTER
679      000002      R2= %2      ;;GENERAL REGISTER
680      000003      R3= %3      ;;GENERAL REGISTER
681      000004      R4= %4      ;;GENERAL REGISTER
682      000005      R5= %5      ;;GENERAL REGISTER
683      000006      R6= %6      ;;GENERAL REGISTER
684      000007      R7= %7      ;;GENERAL REGISTER
685      000006      SP= %6      ;;STACK POINTER
686      000007      PC= %7      ;;PROGRAM COUNTER
687
688      ;*PRIORITY LEVEL DEFINITIONS
689      000000      PRO= 0      ;;PRIORITY LEVEL 0

```

690	000040	PR1=	40	::PRIORITY LEVEL 1
691	000100	PR2=	100	::PRIORITY LEVEL 2
692	000140	PR3=	140	::PRIORITY LEVEL 3
693	000200	PR4=	200	::PRIORITY LEVEL 4
694	000240	PR5=	240	::PRIORITY LEVEL 5
695	000300	PR6=	300	::PRIORITY LEVEL 6
696	000340	PR7=	340	::PRIORITY LEVEL 7

;'SWITCH REGISTER' SWITCH DEFINITIONS

698		SW15=	100000
699	100000	SW14=	40000
700	040000	SW13=	20000
701	020000	SW12=	10000
702	010000	SW11=	4000
703	004000	SW10=	2000
704	002000	SW09=	1000
705	001000	SW08=	400
706	000400	SW07=	200
707	000200	SW06=	100
708	000100	SW05=	40
709	000040	SW04=	20
710	000020	SW03=	10
711	000010	SW02=	4
712	000004	SW01=	2
713	000002	SW00=	1
714	000001		
715		.EQUIV	SW09,SW9
716		.EQUIV	SW08,SW8
717		.EQUIV	SW07,SW7
718		.EQUIV	SW06,SW6
719		.EQUIV	SW05,SW5
720		.EQUIV	SW04,SW4
721		.EQUIV	SW03,SW3
722		.EQUIV	SW02,SW2
723		.EQUIV	SW01,SW1
724		.EQUIV	SW00,SW0

;'DATA BIT DEFINITIONS (BIT00 TO BIT15)

726		BIT15=	100000
727	100000	BIT14=	40000
728	040000	BIT13=	20000
729	020000	BIT12=	10000
730	010000	BIT11=	4000
731	004000	BIT10=	2000
732	002000	BIT09=	1000
733	001000	BIT08=	400
734	000400	BIT07=	200
735	000200	BIT06=	100
736	000100	BIT05=	40
737	000040	BIT04=	20
738	000020	BIT03=	10
739	000010	BIT02=	4
740	000004	BIT01=	2
741	000002	BIT00=	1
742	000001		
743		.EQUIV	BIT09,BIT9
744		.EQUIV	BIT08,BIT8
745		.EQUIV	BIT07,BIT7

```

746      .EQUIV BIT06,BIT6
747      .EQUIV BIT05,BIT5
748      .EQUIV BIT04,BIT4
749      .EQUIV BIT03,BIT3
750      .EQUIV BIT02,BIT2
751      .EQUIV BIT01,BIT1
752      .EQUIV BIT00,BIT0
753
754      ; *BASIC "CPU" TRAP VECTOR ADDRESSES
755      ERRVEC= 4      ; TIME OUT AND OTHER ERRORS
756      RESVEC= 10     ; RESERVED AND ILLEGAL INSTRUCTIONS
757      TBITVEC= 14    ; "T" BIT
758      TRIVEC= 14     ; TRACE TRAP
759      BPTVEC= 14     ; BREAKPOINT TRAP (BPT)
760      IOTVEC= 20     ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
761      PWRVEC= 24     ; POWER FAIL
762      EMTVEC= 30     ; EMULATOR TRAP (EMT) **ERROR**
763      TRAPVEC= 34    ; "TRAP" TRAP
764      TKVEC= 60      ; TTY KEYBOARD VECTOR
765      TPVEC= 64      ; TTY PRINTER VECTOR
766      PIRQVEC= 240   ; PROGRAM INTERRUPT REQUEST VECTOR
767
768      ; THE FOLLOWING DEFINITIONS APPLY TO THE GLOBAL SUBS
769      SET= -1
770      CLR= 0
771      TRUE= 1
772
773      ; RCSR REGISTER BIT NAMES
774
775      DONE= BIT07     ; RECEIVER DONE
776      IE= BIT06      ; RECEIVER INTERRUPT ENABLE
777
778      ; RBUF REGISTER BIT NAMES
779
780      ERROR= BIT15    ; ERROR INDICATOR
781      ORERR= BIT14    ; OVERRUN ERROR
782      FRERR= BIT13    ; FRAMING ERROR
783      PERR= BIT12     ; PARITY ERROR
784
785      ; TCSR REGISTER BIT NAMES
786
787      RDY= BIT07      ; TRANSMITTER READY
788      IE= BIT06       ; TRANSMITTER INTERRUPT ENABLE
789      BREAK= BIT00    ; SEND BREAK (CONTINUOUS SPACE)
790

```

```

791
792
793      ;*****
794      .SBTTL  TRAP CATCHER
795
796          .=0
797      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
798      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
799      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
800          .=174
801      000174 000000  DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
802      000176 000000  SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
803      .SBTTL  STARTING ADDRESS(ES)
804      000200 000137 00157J  JMP      @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
805
806          .=100
807      000100 000102      102          ;RETURN FROM ANY CLOCK INTERRUPTS
808      000102 000002      RTI
809
810      .SBTTL  ACT11 HOOKS
811
812      ;*****
813      ;HOOKS REQUIRED BY ACT11
814          $SVPC-.          ;SAVE PC
815          .=46
816      000046 013422      $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEUP
817          .=52
818      000052 000000      .WORD 0          ;;2)SET LOC.52 TO ZERO
819          .=$SVPC          ;; RESTORE PC
820      .-1000
821      .SBTTL  APT PARAMETER BLOCK
822
823      ;*****
824      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
825      ;*****
826          .$X=.          ;;SAVE CURRENT LOCATION
827          .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
828      000024 000200      200          ;;FOR APT START UP
829          .=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
830      000044 001000      $APTHDR ;;POINT TO APT HEADER BLOCK
831          .-.$X          ;;RESET LOCATION COUNTER
832
833      ;*****
834      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
835      ;INTERFACE SPEC.
836
837      $APTHD:
838      $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
839      $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
840      $TSTM:  .WORD 130.       ;;RUN TIME OF LONGEST TEST
841      $PASTM: .WORD 140.       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
842      $UNITM: .WORD 125.       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
843          .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE (WORDS)

```

843
 844
 845
 846
 847
 848
 849 001100
 850 001100 000000
 851 001100 000
 852 001102 000
 853 001103 000
 854 001104 000000
 855 001106 000000
 856 001110 000000
 857 001112 000000
 858 001111, 000
 859 0011 5 001
 860 001 16 000000
 861 00 120 000000
 862 00 122 000000
 863 001124 000000
 864 001126 000000
 865 001130 000000
 866 001132 000000
 867 001134 000
 868 001135 000
 869 001136 000000
 870 001140 177570
 871 001142 177570
 872 001144 177560
 873 001146 177562
 874 001150 177564
 875 001152 177566
 876 001154 000
 877 001155 002
 878 001156 012
 879 001157 000
 880 001160 000000
 881 001162 000000
 882 001164 177607 000377
 883 001170 077
 884 001171 015
 885 001172 000012
 886
 887
 888
 889
 890
 891 001174
 892 001174 000000
 893 001176 000000
 894 001200 000000
 895 001202 000000
 896 001204 000000
 897 001206 000000
 898 001210 000000

.SBTTL COMMON TAGS

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

. =1100

\$CMTAG: ; START OF COMMON TAGS
 .WORD 0
 \$TSTNM: .BYTE 0 ; CONTAINS THE TEST NUMBER
 \$ERFLG: .BYTE 0 ; CONTAINS ERROR FLAG
 \$ICNT: .WORD 0 ; CONTAINS SUBTEST ITERATION COUNT
 \$LPADR: .WORD 0 ; CONTAINS SCOPE LOOP ADDRESS
 \$LPERR: .WORD 0 ; CONTAINS SCOPE RETURN FOR ERRORS
 \$ERTTL: .WORD 0 ; CONTAINS TOTAL ERRORS DETECTED
 \$ITEMB: .BYTE 0 ; CONTAINS ITEM CONTROL BYTE
 \$ERMAX: .BYTE 1 ; CONTAINS MAX. ERRORS PER TEST
 \$ERRPC: .WORD 0 ; CONTAINS PC OF LAST ERROR INSTRUCTION
 \$GDADR: .WORD 0 ; CONTAINS ADDRESS OF 'GOOD' DATA
 \$BDADR: .WORD 0 ; CONTAINS ADDRESS OF 'BAD' DATA
 \$GDDAT: .WORD 0 ; CONTAINS 'GOOD' DATA
 \$BDDAT: .WORD 0 ; CONTAINS 'BAD' DATA
 .WORD 0 ; RESERVED--NOT TO BE USED
 .WORD 0
 \$AUTOB: .BYTE 0 ; AUTOMATIC MODE INDICATOR
 \$INTAG: .BYTE 0 ; INTERRUPT MODE INDICATOR
 .WORD 0
 \$SWR: .WORD DSWR ; ADDRESS OF SWITCH REGISTER
 \$DISPLAY: .WORD DDISP ; ADDRESS OF DISPLAY REGISTER
 \$TKS: 177560 ; TTY KBD STATUS
 \$TKB: 177562 ; TTY KBD BUFFER
 \$TPS: 177564 ; TTY PRINTER STATUS REG. ADDRESS
 \$TPB: 177566 ; TTY PRINTER BUFFER REG. ADDRESS
 \$NULL: .BYTE 0 ; CONTAINS NULL CHARACTER FOR FILLS
 \$FILLS: .BYTE 2 ; CONTAINS # OF FILLER CHARACTERS REQUIRED
 \$FILLC: .BYTE 12 ; INSERT FILL CHARS. AFTER A 'LINE FEED'
 \$TPFLG: .BYTE 0 ; 'TERMINAL AVAILABLE' FLAG (BIT<07>=0 YES)
 \$TIMES: 0 ; MAX. NUMBER OF ITERATIONS
 \$ESCAPE: 0 ; ESCAPE ON ERROR ADDRESS
 \$BELL: .ASCII <207><377><377> ; CODE FOR BELL
 \$QUES: .ASCII /?/ ; QUESTION MARK
 \$CRLF: .ASCII <15> ; CARRIAGE RETURN
 \$LF: .ASCII <12> ; LINE FEED

.SBTTL APT MAILBOX-ETABLE

 .EVEN
 \$MAIL: ; APT MAILBOX
 \$MSGTY: .WORD AMSGTY ; MESSAGE TYPE CODE
 \$FATAL: .WORD AFATAL ; FATAL ERROR NUMBER
 \$TESTN: .WORD ATESTN ; TEST NUMBER
 \$PASS: .WORD APASS ; PASS COUNT
 \$DEVCT: .WORD ADEVCT ; DEVICE COUNT
 \$UNIT: .WORD AUNIT ; I/O UNIT NUMBER
 \$MSGAD: .WORD AMSGAD ; MESSAGE ADDRESS

899	001212	000000	\$MSGLG: .WORD	AMSGLG	::MESSAGE LENGTH
900	001214		\$ETABLE:		::APT ENVIRONMENT TABLE
901	001214	000	\$ENV: .BYTE	AENV	::ENVIRONMENT BYTE
902	001215	000	\$ENVM: .BYTE	AENVM	::ENVIRONMENT MODE BITS
903	001216	000000	\$SWREG: .WORD	ASWREG	::APT SWITCH REGISTER
904	001220	000000	\$USWR: .WORD	AUSWR	::USER SWITCHES
905	001222	000000	\$CPUOP: .WORD	ACPUOP	::CPU TYPE,OPTIONS
906			::		BITS 15-11=CPU TYPE
907			::		11/04=01,11/05=02,11/20=03,11/40-04,11/45 05
908			::		11/70=06,PDQ-07,Q-10
909			::		BIT 10=REAL TIME CLOCK
910			::		BIT 9=FLOATING POINT PROCESSOR
911			::		BIT 8=MEMORY MANAGEMENT
912	001224	000	\$MAMS1: .BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
913	001225	000	\$MTYP1: .BYTE	AMTYP1	::MEM. TYPE,BLK#1
914			::		MEM.TYPE BYTE -- (HIGH BYTE)
915			::		900 NSEC CORE=001
916			::		300 NSEC BIPOLAR-002
917			::		500 NSEC MOS=003
918	001226	000000	\$MADR1: .WORD	AMADR1	::HIGH ADDRESS,BLK#1
919			::		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
920	001230	000	\$MAMS2: .BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
921	001231	000	\$MTYP2: .BYTE	AMTYP2	::MEM.TYPE,BLK#2
922	001232	000000	\$MADR2: .WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
923	001234	000	\$MAMS3: .BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
924	001235	000	\$MTYP3: .BYTE	AMTYP3	::MEM.TYPE,BLK#3
925	001236	000000	\$MADR3: .WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
926	001240	000	\$MAMS4: .BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
927	001241	000	\$MTYP4: .BYTE	AMTYP4	::MEM.TYPE,BLK#4
928	001242	000000	\$MADR4: .WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
929	001244	000000	\$VECT1: .WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
930	001246	000000	\$VECT2: .WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
931	001250	000000	\$BASE: .WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
932	001252	000000	\$DEVN: .WORD	ADEVN	::DEVICE MAP
933	001254		\$ETEND:		
934			.MEXIT		

```

935      .SBTTL  ERROR POINTER TABLE
936
937      ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
938      ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
939      ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TAP E IS PERTINENT.
940      ;*NOTE1:      IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
941      ;*NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
942
943      ;*      EM      ;;POINTS TO THE ERROR MESSAGE
944      ;*      DM      ;;POINTS TO THE DATA HEADER
945      ;*      DT      ;;POINTS TO THE DATA
946      ;*      DF      ;;POINTS TO THE DATA FORMAT
947
948
949      $ERRTB:
950
951      BASE0: 176500      ;DEFAULT CH 0 BASE ADDR
952      VECT0: 300         ;DEFAULT CH 0 VECTOR
953      BASE1: 177560      ;DEFAULT CH 1 BASE ADDR (CONSOLE)
954      VECT1: 60          ;DEFAULT CH 1 VECTOR
955      LOROM: 173000      ;DEFAULT LO ROM ADDR
956      HIROM: 173776      ;DEFAULT HI ROM ADDR (256 WORDS)
957                          ;OPTION 1: 4K 000000 - 017776
958                          ;OPTION 2: 4K 020000 - 037776
959      DLADD: 0           ;CH ADDR UNDER TEST...LOADED BY 'CYCLE'
960      RCSR: 0            ;CH REGS UNDER TEST...LOADED BY 'LOOP'
961      RBUF: 0
962      TCSR: 0
963      TBUF: 0
964      DLVEC: 0           ;CH RECVR VECTOR UNDER TEST...LOADED BY 'CYCLE'
965      CLKVEC= 100        ;LINE CLOCK INTR VECTOR
966      TMP1: 0            ;TEMPORARY REGS
967      TMP2: 0
968      CARR: .ASCIZ <15>
969
970      ;*****
971      ;TRAP CATCHER FOR OFF LINE USE ONLY. BEGIN PROGRAM WITH FRESH LOAD
972      ;*****
973
974      TRPCAT: CLR R0
975      1$: MOV #INTSRV,(R0)+
976          MOV #PR7,(R0)+
977          CMP R0,#1000
978          BNE 1$
979          BR 3$
980
981          CLR R0
982          MOV #2,R1
983      2$: MOV R1,(R0)+
984          CLR (R0)+
985          ADD #4,R1
986          CMP R0,#1000
987          BNE 2$
988      3$: CLR 176
989          JMP START
990
991      001254
992      001254 176500
993      001256 000300
994      001260 177560
995      001262 000060
996      001264 173000
997      001266 173776
998
999      001270 000000
1000     001272 000000
1001     001274 000000
1002     001276 000000
1003     001300 000000
1004     001302 000000
1005     001304 000100
1006     001306 000000
1007     001310 000015
1008
1009     001312 005000
1010     001314 012720 011636
1011     001320 012720 000340
1012     001324 020027 001000
1013     001330 001371
1014     001332 000412
1015
1016     001334 005000
1017     001336 012701 000002
1018     001342 010120
1019     001344 005020
1020     001346 062701 000004
1021     001352 020027 001000
1022     001356 001371
1023     001360 005037 000176
1024     001364 000137 001370
  
```

```

991 001370
992
993
994 001370 012706 001100
995 001374 005026
996 001376 022706 001140
997 001402 001374
998 001404 012706 001100
999
1000 001410 012737 015432 000020
1001 001416 012737 000340 000022
1002 001424 012737 015232 000030
1003 001432 012737 000340 000032
1004 001440 012737 016364 000034
1005 001446 012737 000340 000036
1006 001454 012737 013456 000024
1007 001462 012737 000340 000026
1008 001470 013737 013370 013362
1009 001476 005037 001160
1010 001502 005037 001162
1011 001506 112737 000001 001115
1012 001514 012737 001514 001106
1013 001522 012737 001522 001110
1014
1015
1016 001530 013746 000004
1017 001534 012737 001570 000004
1018 001542 012737 177570 001140
1019 001550 012737 177570 001142
1020 001556 022777 177777 177354
1021 001564 001012
1022
1023 001566 000403
1024 001570 012716 001576
1025 001574 000002
1026 001576 012737 000176 001140
1027 001604 012737 000174 001142
1028 001612 012637 000004
1029
1030 001616 005037 001202
1031 001622 132737 000200 001215
1032 001630 001403
1033 001632 012737 001216 001140
1034 001640
1035
1036
1037 001640 005227 177777
1038 001644 001045
1039 001646 022737 013422 000042
1040 001654 001441
1041 001656 104401 001724
1042
1043 001662 005737 000042
1044 001666 001012
1045 001670 123727 001214 000001
1046 001676 001406

START:
.SBTTL INITIALIZE THE COMMON TAGS
;;CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV $CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;;CLEAR MEMORY LOCATION
CMP $SWR,R6 ;;DONE?
BNE -6 ;;LOOP BACK IF NO
MOV $STACK,SP ;;SETUP THE STACK POINTER
;;INITIALIZE A FEW VECTORS
MOV $SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
MOV #340,@#IOTVEC+2 ;;LEVEL 7
MOV $ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
MOV #340,@#EMTVEC+2 ;;LEVEL 7
MOV $TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
MOV #340,@#TRAPVEC+2 ;;LEVEL 7
MOV $SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
MOV #340,@#PWRVEC+2 ;;LEVEL 7
MOV $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
MOVB #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
MOV #,$SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,$SLPERR ;;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.
MOV @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
MOV #64,$@#ERRVEC ;;SET UP ERROR VECTOR
MOV #DSWR,$SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
MOV #DDISP,$DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
CMP #-1,$@SWR ;;TRY TO REFERENCE HARDWARE SWR
BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
;;AND THE HARDWARE SWR IS NOT -1
BR 65$ ;;BRANCH IF NO TIMECUT
MOV #65,$(SP) ;;SET UP FOR TRAP RETURN
RTI
65$: MOV $SWREG,$SWR ;;POINT TO SOFTWARE SWR
MOV $DISPREG,$DISPLAY
66$: MOV (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR

CLR $PASS ;;CLEAR PASS COUNT
BITB $APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
BEQ 67$ ;;YES,USE NON-APT SWITCH
MOV $SWREG,$SWR ;;NO,USE APT SWITCH REGISTER
67$:
.SBTTL TYPE PROGRAM NAME
;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
INC #-1 ;;FIRST TIME?
BNE 68$ ;;BRANCH IF NO
CMP $SENDAD,@#42 ;;ACT-11?
BEQ 68$ ;;BRANCH IF YES
TYPE ,69$ ;;TYPE ASCIZ STRING
.SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
TST @#42 ;;ARE WE RUNNING UNDER XXDP/A 1?
BNE 70$ ;;BRANCH IF YES
CMQB $ENV,#1 ;;ARE WE RUNNING UNDER APT?
BEQ 70$ ;;BRANCH IF YES

```

```

1047 001700 023727 001140 000176      CMP      SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1048 001706 001005                      BNE      71$           ;;BRANCH IF NO
1049 001710 104406                      GTSWR                      ;;GET SOFT-SWR SETTINGS
1050 001712 000403                      BR      71$
1051 001714 112737 000001 001134 70$:  MOV      #1,$AUTOB      ;;SET AUTO-MODE INDICATOR
1052 001722                      71$:
1053 001722 000416                      BR      68$           ;;GET OVFR THE ASCIZ
1054                      ;;69$: .ASCIZ <CRLF>*CVMXAAO MXV11A DIAGNOSTIC*<CRLF>
1055                      68$:
1056 001760 005227 177777                      INC      #-1           ;1ST TIME?
1057 001764 001005                      BNE      1$           ;BR IF NO
1058 001766 123727 001214 000001          CMPB     $ENV,#1       ;APT?
1059 001774 001401                      BEQ      1$           ;BR IF YES
1060 001776 104412                      GTDEVM          ;ELSE SHOW CURRENT DEVM & GET NEW
1061
1062 002000 004737 011534          1$:  JSR      PC,SIZE      ;SIZE THE MXV11-A
1063
1064 002004 012706 001100          RAMPOM: MOV     #STACK,SP      ;RESET STACK PTR
1065
1066
1067
1068
1069                      ;*****
1070                      ;*TEST 1      RAM ADDRESS TEST
1071                      ;*
1072                      ;*      WRITE ENTIRE MEMORY WITH ADDRESS AS DATA.
1073                      ;*      READ ENTIRE MEMORY TO VERIFY.
1074                      ;*****
1075 002010 000004          TST1:  SCOPE
1076 002012 012737 000001 001160          MOV      #1,$TIMES      ;;DO 1 ITERATION
1077 002020 012737 000001 001200          MOV      #1,$TESTN      ;;SET TEST NUMBER IN APT MAIL BLX
1078
1079 002026 032737 000004 001252          BIT      #BIT2,$DEVM      ;BYPASS RAM TESTS?
1080 002034 001022                      BNE      TST2           ;;EXIT IF YES
1081
1082 002036 012700 017776          MOV      #17776,R0      ;ADDRESS REG
1083
1084 002042 010010          1$:  MOV      R0,(R0)      ;LOAD ADDRESS AS DATA
1085 002044 020037 011634          CMP      R0,MAXMEM      ;LAST ADDR?
1086 002050 001402          BEQ      2$           ;BR IF YES
1087 002052 005720          TST      (R0)+      ;ELSE BUMP ADDR REG
1088 002054 000772          BR      1$
1089
1090 002056 012700 017776          2$:  MOV      #17776,R0      ;VERIFY
1091 002062 020010          3$:  CMP      R0,(R0)      ;READ & CHECK
1092 002064 001401          BEQ      4$           ;DATA NOT - ADDRESS
1093 002066 104010          ERROR  10
1094
1095 002070 020037 011634          4$:  CMP      R0,MAXMEM      ;LAST ADDR?
1096 002074 001402          BEQ      TST2           ;;EXIT IF YES
1097 002076 005720          TST      (R0)+      ;ELSE BUMP ADDR
1098 002100 000770          BR      3$

```

```

1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113 002102 000004
1114 002104 012737 000001 001160
1115 002112 012737 000002 001200
1116
1117 002120 032737 000004 001252
1118 002126 001145
1119
1120 002130 012737 177777 002554
1121 002136 012700 017776
1122
1123 002142 013710 002554
1124 002146 020037 011634
1125 002152 001402
1126 002154 005720
1127 002156 000771
1128
1129 002160 012700 017776
1130 002164 021037 002554
1131 002170 001401
1132 002172 104011
1133
1134 002174 005737 002554
1135 002200 001002
1136 002202 000261
1137 002204 000401
1138 002206 000241
1139
1140 002210 013737 002554 002560
1141 002216 006037 002560
1142 002222 013710 002560
1143 002226 021037 002560
1144 002232 001401
1145 002234 104022

*****
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.
*****

1ST2:  SCOPE
      MOV      #1,$TIMES      ;;DO 1 ITERATION
      MOV      #2,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

      BIT      #BIT2,$DEVM    ;BYPASS RAM TESTS?
      BNE      25$           ;BR IF YES

      MOV      #-1,BKGND     ;BACKGROUND = 1'S
      MOV      #17776,R0     ;ADDRESS
21$:   MOV      BKGND,(R0)    ;WRITE BACKGROUND
      CMP      R0,MAXMEM     ;LAST ADDR?
      BEQ      2$
      TST      (R0)+         ;ELSE BUMP ADDR
      BR       1$

      MOV      #17776,R0     ;RESET ADDR
      CMP      (R0),BKGND    ;READ & CHECK
      BEQ      4$
      ERROR    1$           ;LOC DOES NOT BACKGROUND

      TST      BKGND         ;IF BKGND = 1'S, FLOAT 0'S
      BNE      5$
      SEC
      BR       6$           ;ELSE SET CARRY TO FLOAT 1'S
5$:    CLC                  ;FLOAT 0'S

      MOV      BKGND,SAVBK   ;SAVE
      ROR      SAVBK         ;BEGIN ROTATE
      MOV      SAVBK,(R0)    ;WRITE IT
      CMP      (R0),SAVBK    ;READ & CHECK
      BEQ      8$
      ERROR    22           ;NO COMPARE
  
```

```

1146
1147 002236 005137 002560      8$: COM SAVBK ;DO COMPLEMENT
1148 002242 013710 002560      MOV SAVBK,(R0) ;WRITE IT
1149 002246 011037 002556      MOV (R0),RDWRD ;READ IT
1150 002252 023737 002556 002560 CMP RDWRD,SAVBK ;SAME?
1151 002260 001401      BEQ 9$
1152 002262 104023      ERROR 23 ;NO COMPARE OF COMPLEMENT
1153
1154 002264 005137 002560      9$: COM SAVBK ;RESTORE BACKGROUND
1155 002270 023737 002560 002554 CMP SAVBK,BKGND ;BACK TO ORIG. BACKGROUND?
1156 002276 001407      BEQ 23$ ;BR IF YES
1157
1158 002300 005737 002554      TST BKGND ;IF BKGND = 1, SET CARRY
1159 002304 001002      BNE 22$
1160 002306 000241      CLC ;ELSE CLEAR IT
1161 002310 000742      BR 7$ ;DO NEXT BIT
1162 002312 000261      22$: SEC
1163 002314 000740      BR 7$
1164
1165 002316 013710 002554      23$: MOV BKGND,(R0) ;RESTORE ORIG BACKGROUND
1166 002322 020037 011634      CMP R0,MAXMEM ;LAST ADDR?
1167 002326 001402      BEQ 10$ ;BR IF YES
1168 002330 005720      TST (R0)+ ;ELSE BUMP ADDR
1169 002332 000714      BR 3$ ;& REPEAT
1170
1171 002334 104407      10$: CKSWR
1172 002336 012700 017776      MOV #17776,R0 ;SEE IF ENTIRE MEM STILL SAME BACKGROUND
1173 002342 011037 002556      11$: MOV (R0),RDWRD ;READ
1174 002346 023737 002556 002554 CMP RDWRD,BKGND ;SAME?
1175 002354 001401      BEQ 12$
1176 002356 104025      ERROR 25 ;CHANGE IN BACKGROUND
1177
1178 002360 020037 011634      12$: CMP R0,MAXMEM ;LAST ADDR?
1179 002364 001402      BEQ 13$
1180 002366 005720      TST (R0)+ ;ELSE BUMP ADDR
1181 002370 000764      BR 11$ ;& DO ANOTHER
1182
1183 002372 012737 002444 000004 13$: MOV #16$,ERRVEC ;WAIT > 5 SEC TO TEST FOR VOLITILITY
1184 002400 012737 000200 000006 MOV #200,ERRVEC+2 ;SET UP TIMEOUT TRAP ADDR
1185
1186 002406 013700 011634      MOV MAXMEM,R0
1187 002412 062700 000002      ADD #2,R0 ;THIS ADDR SHOULD GIVE TIMEOUT
1188
1189 002416 012701 000002      MOV #2,R1 ;ITERATION CTR
1190 002422 012702 177777      14$: MOV #-1,R2
1191 002426 011010      15$: MOV (R0),(R0) ;SHOULD TIMEOUT (DO MOV INSTEAD OF TST)
1192 002430 005720      TST (R0)+ ;COME HERE IF NO TIMEOUT & BUMP ADDR
1193 002432 020027 177776      CMP R0,#177776 ;REACHED 32K?
1194 002436 001373      BNE 15$ ;BR IF NO
1195 002440 104042      ERROR 42 ;CANNOT FIND ADDR TO TIMEOUT
1196
1197 002442      25$:
1198 002442 000447      BR TST3 ;EXIT TEST
  
```

```

1199
1200 002444 022626          16$:  CMP      (SP)+,(SP)+      ;CLEAR STACK FROM TRAP
1201 002446 005302          DEC      R2
1202 002450 001366          BNE      15$
1203 002452 005301          DEC      R1
1204 002454 001362          BNE      14$
1205
1206 002456 104407          CKSWR
1207 002460 012737 000006 000004  MOV      #ERRVEC+2,ERRVEC      ;5 SEC UP. RESET TIMEOUT TRAP ADDR.
1208 002466 005037 000006      CLR      ERRVEC+2
1209
1210 002472 012700 017776      MOV      #17776,R0      ;TEST MEMORY FOR VOLITILITY
1211
1212 002476 021037 002554      17$:  CMP      (R0),BKGND      ;READ & CHECK
1213 002502 001401          BEQ      18$
1214 002504 104026          ERROR     26      ;BACKGROUND CHANGE AFTER 5 SEC
1215
1216 002506 020037 011634      18$:  CMP      R0,MAXMEM      ;LAST ADDR?
1217 002512 001402          BEQ      19$
1218 002514 005720          TST      (R0)+      ;E SE BUMP ADDR
1219 002516 000767          BR       17$      ;& DO ANOTHER
1220
1221 002520 005737 002554      19$:  TST      BKGND      ;BACKGROUND ALL 0'S?
1222 002524 001404          BEQ      20$      ;BR IF YES
1223 002526 005037 002554      CLR      BKGND      ;ELSE DO OPPOSITE BACKGROUND
1224 002532 000137 002136      JMP      21$
1225
1226 002536 005737 011614      20$:  TST      NOCHAN      ;ANY CHANNELS TO BE TESTED?
1227 002542 001407          BEQ      TST3      ;:BR IF YES
1228 002544 104401 001310      TYPE     ,CARR
1229 002550 000137 013334      JMP      $EOP      ;ELSE ALL DONE
1230
1231 002554 000000          BKGND:  0      ;BACKGROUND
1232 002556 000000          RDWRD:  0      ;STORE READ WORD HERE
1233 002560 000000          SAVBK:  0      ;ROTATE BACKGROUND HERE
1234

```

```

1235
1236
1237
1238
1239 002562 000004
1240 002564 012737 000010 001160
1241 002572 012737 000003 001200
1242
1243 002600 037737 000002 001252
1244 002606 001057
1245 002610 012737 011636 000004
1246 002616 012737 000200 000006
1247
1248
1249
1250
1251
1252 002624 013703 001264
1253 002630 005037 011644
1254 002634 005713
1255 002636 005737 011644
1256 002642 001403
1257 002644 010337 002744
1258 002650 104007
1259
1260 002652 020337 001266
1261 002656 001403
1262 002660 062703 000002
1263 002664 000761
1264
1265
1266
1267
1268
1269 002666 013703 001264
1270 002672 005037 011644
1271 002676 010013
1272 002700 005737 011644
1273 002704 001003
1274 002706 010337 002744
1275 002712 104035
1276
1277 002714 020337 001266
1278 002720 001403
1279 002722 062703 000002
1280 002726 000761
1281
1282 002730 012737 000006 000004
1283 002736 005037 000006
1284 002742 000401
1285
1286 002744 000000

;*****
;TEST 3 ROM TESTS
;*****
ST3: SCOPE
      MOV #10,$TIMES ;DO 10 ITERATIONS
      MOV #3,$TESTN ;SET TEST NUMBER IN APT MAIL BOX

      BIT #BIT1,$DEVH ;BYPASS ROM TEST?
      BNE TST4 ;EXIT IF YES
      MOV #INTSRV,ERRVEC ;SETUP TIMEOUT TRAP ADDR
      MOV #200,ERRVEC+2

;*****
;TEST 'LOROM' TO 'HIROM' PRESENT
;*****
3$: MOV LOROM,R3
   CLR INTFLG
   TST (R3) ;ACCESS
   TST INTFLG ;TIMEOUT?
   BEQ 4$ ;BR IF NO
   MOV R3,ADDR ;FOR ERROR ANALYSIS
   ERROR 7

4$: CMP R3,HIROM ;ALL DONE?
   BEQ 5$ ;BR IF YES
   ADD #2,R3 ;ELSE BUMP ADDR
   BR 3$

;*****
;TEST THAT ROM CANNOT BE WRITTEN INTO
;*****
5$: MOV LOROM,R3
6$: CLR INTFLG
   MOV R0,(R3) ;TRY TO WRITE INTO ROM
   TST INTFLG ;TIMEOUT?
   BNE 7$ ;BR IF YES
   MOV R3,ADDR ;FOR ERROR ANALYSIS
   ERROR 35

7$: CMP R3,HIROM ;ALL DONE?
   BEQ 8$ ;BR IF YES
   ADD #2,R3 ;ELSE BUMP ADDR
   BR 6$

8$: MOV #ERRVEC+2,ERRVEC ;RESTORE TIMEOUT VECTOR
   CLR ERRVEC+2
   BR TST4 ;EXIT TEST

ADDR: .WORD 0 ;ROM ADDRESS WHERE ERROR OCCURED

```

```

1287
1288
1289
1290
1291 002746 000004
1292 002750 012737 000010 001160
1293 002756 012737 000004 001200
1294
1295 002764 106427
1296 002766 000200
1297
1298 002770 012737 011636 000100
1299 002776 012737 000200 000102
1300 003004 005037 011644
1301
1302 003010 106427
1303 003012 000000
1304
1305 003014 032737 000001 001252
1306 003022 001411
1307
1308 003024 012700 177777
1309 003030 005300
1310 003032 001376
1311 003034 005737 011644
1312 003040 001012
1313 003042 104006
1314 003044 000410
1315
1316 003046 012700 177777
1317 003052 005300
1318 003054 001376
1319 003056 005737 011644
1320 003062 001401
1321 003064 104034
1322
1323 003066 106427
1324 003070 000200
1325
1326 003072 012737 000102 000100
1327 003100 012737 000002 000102
1328
1329 003106 106427
1330 003110 000000
  
```

```

*****
:TEST 4          CLOCK TESTS
*****
TST4:  SCOPE
      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
      MOV      #4,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
      .WORD    106427          ;MIPS
      .WORD    200             ;DISABLE INTERRUPTS
      MOV      #INTSRV,CLKVEC  ;SETUP VECTOR AREA
      MOV      #200,CLKVEC+2   ;LOCKOUT OTHER INTERRUPT
      CLR      INTFLG
      .WORD    106427          ;MIPS
      .WORD    0               ;ALLOW CLOCK TO INTERRUPT THRU INTSRV
      BIT      #BIT0,$DEVM     ;CLOCK OPTION DISABLED?
      BEQ      2$              ;BR IF YES
      MOV      #-1,R0          ;TIMER
      DEC      R0
      BNE      1$
      TST      INTFLG          ;GOT INTERRUPT?
      BNE      4$              ;BR IF YES
      ERRUR    6               ;NO CLOCK INTERRUPT
      BR       4$
      MOV      #-1,R0          ;TIMER
      DEC      R0
      BNE      3$
      TST      INTFLG          ;GOT INTERRUPT?
      BEQ      4$              ;BR IF NO
      ERROR    34              ;UNEXP. CLOCK INTERRUPT
      .WORD    106427          ;MIPS
      .WORD    200             ;DISABLE INTERRUPT
      MOV      #102,CLKVEC     ;RESTORE CLOCK VECTOR AREA
      MOV      #RT1,CLKVEC+2
      .WORD    106427          ;MIPS
      .WORD    0               ;ENABLE INTERRUPTS
  
```

```

1331
1332 003112 005037 012312      SLU:  CLR  PHASE2      ;INIT FOR TESTS
1333 003116 005037 001206      CLR  $UNIT      ;CONTAINS CHAN # UNDER TEST ( 0 OR 1 )
1334 003122 005037 012314      CLR  PICNT
1335
1336 003126 012706 001100      LOOP:  MOV  #STACK,SP      ;RESET STACK POINTER
1337 003132 005737 011614      TST  NOCHAN      ;ANY CHANNELS TO TEST?
1338 003136 001404      BEQ  1$      ;BR IF YES
1339 003140 104401 001310      TYPE  ,CARR
1340 003144 000137 013334      JMP  $EOP      ;ELSE ALL DONE
1341
1342 003150 004737 011646      1$:   JSR  PC,CYCLE      ;ELSE SETUP MODULE AND CHANNEL
1343
1344 003154 013737 001270 001272      MOV  DLADD,RCSR
1345
1346 003162 013737 001270 001274      MOV  DLADD,RBUF
1347 003170 062737 000002 001274      ADD  #2,RBUF
1348
1349 003176 013737 001270 001276      MOV  DLADD,TCSR
1350 003204 062737 000004 001276      ADD  #4,TCSR
1351
1352 003212 013737 001270 001300      MOV  DLADD,TBUF
1353 003220 062737 000006 001300      ADD  #6,TBUF
1354
1355 003226 012700 177777      MOV  #-1,R0      ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
1356 003232 077001      SOB  R0,,      ;FINISH BEFORE RESET
1357 003234 000005      RESET
1358
1359 003236 005237 001204      INC  $DEVCT
1360 003242 123727 012312 000001      CMPB PHASE2,#TRUE      ;DOING PHASE 2?
1361 003250 001002      BNE  2$      ;BR IF NO
1362 003252 000137 007264      JMP  MODTST
1363
1364
2$:

```

```

1365
1366
1367
1368
1369
1370
1371
1372 003256 000004
1373 003260 012737 000002 001160
1374 003266 012737 000005 001200
1375
1376 003274 012737 011636 000004
1377 003302 012737 000340 000006
1378
1379 003310 005000
1380 003312 013701 001272
1381
1382 003316 005037 011644
1383 003322 005721
1384 003324 005737 011644
1385 003330 001401
1386 003332 104001
1387
1388 003334 005200
1389 003336 020027 000004
1390 003342 001365
1391
1392 003344 012737 000006 000004
1393 003352 005037 000006
1394
1395 003356 032777 177477 175706
1396 003364 001401
1397 003366 104027
1398
1399 003370 032777 007400 175676
1400 003376 001401
1401 003400 104030
1402
1403 003402 032777 177476 175666
1404 003410 001401
1405 003412 104031
1406
1407 003414 032777 177400 175656
1408 003422 001401
1409 003424 104032
1410
  
```

```

*****
TEST 5 ADDRESSABILITY
*****
THIS TEST VERIFIES THAT ALL 4 REGISTERS OF THE CHANNEL UNDER TEST
RESPOND TO THEIR ADDRESSES.
*****
TEST: SCOPE
MOV #2,$TIMES ;;DO 2 ITERATIONS
MOV #5,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #INTSRV,ERRVEC ;SETUP TIMEOUT VECTOR
MOV #PR7,ERRVEC+2
CLR R0
MOV RCSR,R1
1$: CLR INTFLG
TST (R1)+ ;DARE TO TIMEOUT
TST INTFLG ;DID IT?
BEQ 2$ ;BR IF NO
ERROR 1 ;TIMEOUT ON CHANNEL ADDRESS
2$: INC R0
CMP R0,#4 ;ALL DONE?
BNE 1$ ;BR IF NO
MOV #6,ERRVEC ;RESTORE TIMEOUT VECTOR
CLR ERRVEC+2
BIT #177477,@RCSR ;CHECK THAT ALL UNUSED BITS ARE 0
BEQ 3$
ERROR 27 ;RCSR HAS UNUSED BITS SET
3$: BIT #7400,@RBUF
BEQ 4$
ERROR 30 ;RBUF HAS UNUSED BITS SET
4$: BIT #177476,@TCSR
BEQ 5$
ERROR 31 ;TCSR HAS UNUSED BITS SET
5$: BIT #177400,@TBUF
BEQ 6$
ERROR 32 ;EXIT IF OK
;TBUF HAS ;UNUSED BITS SET
  
```

```

1411      ;*****
1412      ;* THE FOLLOWING 3 TESTS TEST ALL 'READ WRITE' BITS
1413      ;*****
1414
1415
1416      ;*****
1417      ;* TEST 6      BREAK - TCSR 0 SET, CLEAR, RESET
1418      ;*            THIS BIT IS THE ONLY ONE IN THIS POSITION
1419      ;*            THAT IS READ AND WRITE.
1420      ;*****
1421      003426 000004      TST6:  SCOPE
1422      003430 012737 000010 001160      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
1423      003436 012737 000006 001200      MOV      #6,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1424
1425      003444 005737 012304      TST      BRK      ;BREAK DETECTION ENABLED?
1426      003450 001403      BEQ      1$      ;BR IF NO
1427      003452 005737 012310      TST      CONSOLE      ;ELSE ARE WE ON CONSOLE?
1428      003456 001055      BNE      TST7      ;;EXIT IF YES
1429      003460      1$:
1430      ; SEE IF IT IS CLEAR
1431      003460      LPADR
1432      003460 012737 003466 001110      MOV      #64$,$LPERR
1433      003466      64$:
1434
1435      003466      IF      #BREAK SETIN @TCSR THEN
1436      003466 032777 000001 175602      BIT      #BREAK,@TCSR
1437      003474 001401      BEQ      $1
1438      ; BREAK DID NOT RESET IN TCSR
1439      003476 104002      ERROR 2
1440      003500      ENDIF
1441      003500      $1:
1442
1443      ; TRY TO SET BREAK BIT
1444      003500      LPADR
1445      003500 012737 003506 001110      MOV      #64$,$LPERR
1446      003506      64$:
1447      003506      LET      @TCSR := @TCSR SET.BY #BREAK
1448      003506 052777 000001 175562      BIS      #BREAK,@TCSR
1449      ; STUCK TO 0
1450      003514      IF      #BREAK NOTSETIN @TCSR THEN
1451      003514 032777 000001 175554      BIT      #BREAK,@TCSR
1452      003522 001001      BNE      $2
1453      ; BREAK DID NOT SET IN TCSR
1454      003524 104003      ERROR 3
1455      003526      ENDIF
1456      003526      $2:
1457
1458      ; TRY TO CLEAR A SET BIT
1459      003526      LPADR
1460      003526 012737 003534 001110      MOV      #64$,$LPERR
1461      003534      64$:
1462
1463      003534      LET      @TCSR := @TCSR CLW.BY #BREAK
1464      003534 042777 000001 175534      BIC      #BREAK,@TCSR
1465      ; SHOULD HAVE CLEARED
1466      003542      IF      #BREAK SETIN @TCSR THEN
  
```

6 3
(VMXAA) MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 34
(VMXAA.P11 30-MAY-79 15:57 T6 BREAK - TCSR 0 SET, CLEAR, RESET

SEQ 0032

```
1467 003542 032777 000001 175526 BIT #BREAK,@TCSR
1468 003550 001401 BLQ $3
1469 ; BREAK DID NOT CLEAR IN TCSR
1470 003552 104004 ERROR 4
1471 003554
1472 003554 $3:
1473
1474 ; NOW SEE IF RESET CLEARS IT
1475 003554 LPADR
1476 003554 012737 003562 001110 MOV #64$,$LPERR
1477 003562 64$:
1478
1479 003562 LET @TCSR := @TCSR SET.BY #BREAK
1480 003562 052777 000001 175506 BIS #BREAK,@TCSR
1481 ; ISSUE BUS RESET
1482 003570 012700 177777 MOV #-1,R0 ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
1483 003574 077001 SOB R0,, ;FINISH BEFORE RESET
1484 003576 000005 RESET
1485 003600 IF #BREAK SET IN @TCSR THEN
1486 003600 032777 000001 175470 BIT #BREAK,@TCSR
1487 003606 001401 BEQ $4
1488 ; BREAK DID NOT RESET IN TCSR
1489 003610 104005 ERROR 5
1490 003612
1491 003612 $4:
ENDIF
```

```

1492
1493
1494
1495
1496 003612 000304
1497 003614 012737 000010 001160
1498 003622 012737 000007 001200
1499
1500 003630
1501
1502
1503 003642
1504 003642 012737 003650 001110
1505 003650
1506
1507 003650
1508 003650 032777 000100 175420
1509 003656 001401
1510
1511 003660 104012
1512 003662
1513 003662
1514
1515
1516 003662
1517 003662 012737 003670 001110
1518 003670
1519 003670
1520 003670 052777 000100 175400
1521
1522 003676
1523 003676 032777 000100 175372
1524 003704 001001
1525
1526 003706 104013
1527 003710
1528 003710
1529
1530
1531 003710
1532 003710 012737 003716 001110
1533 003716
1534
1535 003716
1536 003716 042777 000100 175352
1537
1538 003724
1539 003724 032777 000100 175344
1540 003732 001401
1541
1542 003734 104014
1543 003736
1544 003736
1545
1546
1547 003736

;*****
;TEST 7 IF - TCSR 6 SET, CLEAR, RESET
;*****
TST7: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #7,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
; USE PRIORITY OF 7
SETPRI #PR7

; SEE IF IT IS CLEAR
LPADR
65$: MOV #65,$LPERR

IF #IE SETIN @TCSR THEN
BIT #IE,@TCSR
BEQ $5
; IE DID NOT RESET IN TCSR
ERROR 12
ENDIF
$5:

; TRY TO SET IE BIT
LPADR
64$: MOV #64,$LPERR

LET @TCSR : @TCSR SET.BY #IE
; STUCK TO 0
IF #IE NOTSETIN @TCSR THEN
BIT #IE,@TCSR
BNE $6
; XMIT DID NOT SET IN TCSR
ERROR 15
ENDIF
$6:

; TRY TO CLEAR A SET BIT
LPADR
64$: MOV #64,$LPERR

LET @TCSR := @TCSR CLR.BY #IE
; SHOULD HAVE CLEARED
IF #IE SETIN @TCSR THEN
BIT #IE,@TCSR
BEQ $7
; XMIT DID NOT CLEAR IN TCSR
ERROR 14
ENDIF
$7:

; NOW SEE IF RESET CLEARS IT
LPADR

```

CVMXAAO MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 36
CVMXAA.P11 30-MAY-79 15:57 17 IE - TCSR 6 SET, CLEAR, RESET

SEQ 0034

1548 003736 012737 003744 001110
1549 003744 648: MOV #648,\$LPERR

1550

1551 003744

1552 003744 052777 000100 175324

1553

1554 003752 000005

1555 003754

1556 003754 032777 000100 175314

1557 003762 001401

1558

1559 003764 104015

1560 003766

1561 003766

\$10:

ERROR 15

LET @TCSR := @TCSR SET.BY #IE

; ISSUE BUS RESET

IF #IE SET IN @TCSR THEN

; XMIT DID NOT RESET IN TCSR

ENDIF

```

1562
1563
1564
1565
1566
1567
1568 003766 000004
1569 003770 012737 000010 001160
1570 003776 012737 000010 001200
1571
1572 004004
1573 004004 012737 004012 001110
1574 004012
1575
1576 004012
1577 004012 032777 000100 175252
1578 004020 001401
1579
1580 004022 104035
1581 004024
1582 004024
1583
1584
1585 004024
1586 004024 012737 004032 001110
1587 004032
1588 004032
1589 004032 052777 000100 175232
1590
1591 004040
1592 004040 032777 000100 175224
1593 004046 001001
1594
1595 004050 104036
1596 004052
1597 004052
1598
1599
1600 004052
1601 004052 012737 004060 001110
1602 004060
1603
1604 004060
1605 004060 042777 000100 175204
1606
1607 004066
1608 004066 032777 000100 175176
1609 004074 001401
1610
1611 004076 104037
1612 004100
1613 004100
1614
1615
1616 004100
1617 004100 012737 004106 001110

*****
*TEST 10      IE - RCSR 6      SET, CLEAR, RESET
*              THIS BIT IS THE ONLY ONE IN THIS POSITION
*              THAT IS READ AND WRITE.
*****
TST10:  SCOPE
        MOV     #10,$TIMES      ;;DO 10 ITERATIONS
        MOV     #10,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
        ; SEE IF IT IS CLEAR
        LPADR
64$:   MOV     #64$,$LPERR

        IF      #IE SETIN @RCSR THEN
        BIT     #IE,@RCSR
        BEQ     $11
        ; IE DID NOT RESET IN RCSR
        ENDIF

$11:   ; TRY TO SET IE BIT
        LPADR
        MOV     #64$,$LPERR
64$:   MOV     #64$,$LPERR
        BIS     #IE,@RCSR
        LET     @RCSR := @RCSR SET.BY #IE
        ; STUCK TO 0
        IF      #IE NOTSETIN @RCSR THEN
        ; IE DID NOT SET IN RCSR
        ENDIF

$12:   ; TRY TO CLEAR A SET BIT
        LPADR
        MOV     #64$,$LPERR
64$:   MOV     #64$,$LPERR
        BIC     #IE,@RCSR
        LET     @RCSR := @RCSR CLR.BY #IE
        ; SHOULD HAVE CLEARED
        IF      #IE SETIN @RCSR THEN
        ; IE DID NOT CLEAR IN RCSR
        ENDIF

$13:   ; NOW SEE IF RESET CLEARS IT
        LPADR
        MOV     #64$,$LPERR
  
```

1618	004106				64\$:		
1619							
1620	004106					LET	@RCSR := @RCSR SET BY #IE
1621	004106	052777	000100	175156	BIS	#IE,@RCSR	
1622							; ISSUE BUS RESET
1623	004114	000005			RESET		
1624	004116					IF	#IE SET IN @RCSR THEN
1625	004116	032777	000100	175146	BIT	#IE,@RCSR	
1626	004124	001401			BFO	\$14	
1627							; IE DID NOT RESET IN RCSR
1628	004126	104040			ERROR	40	
1629	004130						ENDIF
1630	004130				\$14:		
1631							
1632	004130	012737	004136	001110	MOV	#1\$,SLPERR	
1633							;XMIT RDY SHOULD BE SET BY RESET ABOVE
1634	004136	032777	000200	175132	1\$: BIT	#RDY,@TCSR	
1635	004144	001002			BNE	TST11	::EXIT IF SET
1636	004146	104042			ERROR	42	
1637							
1638	004150	000005			RESET		;ALLOW LOOPING ON ERROR
1639							

```

1640
1641
1642 *****
1643 :TEST 11      TEST THAT XMIT RDY - TCSR 7 - CLEARS
1644 :             WHEN TBUF IS LOADED WITH A CHARACTER
1645 :             AND THAT IT SETS WITHIN A REASONABLE AMOUNT OF TIME.
1646 :*****
1646 004152 000004 TST11: SCOPE
1647 004154 012737 000010 001160 MOV #10,$TIMES ;;DO 10 ITERATIONS
1648 004162 012737 000011 001200 MOV #11,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
1649
1650 004170 005737 012310 TST CONSOLE ;ARE WE ON CONSOLE?
1651 004174 001077 BNE TST12 ;;EXIT IF YES
1652 004176 005037 004366 CLR PASS LET PASS :- #0 ;INIT COUNT OF TIMES THRU
1653 004202 $15: LOOP ; START OF LOOP
1654 004202 ; MAX OF 2 TIMES THRU
1655 004202 LET ERRFLG :- #0
1656 004202 CLR ERRFLG
1657 004202 005037 004370 CLR EXFLG LET EXFLG :- #0
1658 004206 005037 004372 CLR EXFLG
1659 004206 005037 004372 ; LOAD TBUF WITH ONE CHARACTER
1660 ; WAIT FOR READY TO SET
1661 ; (SHOULD BE VERY SHORT WAIT
1662 ; SINCE UART DOUBLE BUFFERS ITS INPUT)
1663
1664 ;SEND A CHARACTER
1665 LET @TBUF :B- #0
1666
1667 004212 CLR @TBUF
1668 004212 105077 175062 JSR R5,TIMER ;WAIT FOR XMIT READY
1669
1670 004216 004537 012512 TCSR
1671 004222 001276 RDY
1672 004224 000200 ERROR 66 ;XMIT RDY DID NOT SET IN TCSR
1673 004226 104066 BR TST12 ;;EXIT TEST
1674 004230 000461
1675
1676 004232 105077 175042 CLR @TBUF ;SHIP 1ST CHAR
1677 004236 105777 175034 TST @TCSR ;WAIT FOR RDY
1678 004242 100375 BPL 1$
1679
1680 ; LOAD TBUF WITH A SECOND CHARACTER (TO DOUBLE BUFFER)
1681 ; CHECK IMMEDIATELY THAT RDY IS CLEAR
1682 ; AND THEN WAIT FOR IT TO SET
1683
1684 ;SEND SECOND CHARACTER
1685 LET @TBUF :B- #0
1686 004244 105077 175030 CLR @TBUF
1687 004250 000240 NOP ; GIVE IT TIME TO CLEAR
1688 ; RDY SHOULD HAVE CLEARED UPON
1689 ; RECEIPT OF A CHARACTER
1690 004252 032777 000200 175016 BIT #RDY,@TCSR
1691 004260 001404 BEQ $17 IF #RDY SET IN @TCSR THEN
1692
1693 ; RDY DID NOT CLEAR IN TCSR
1694 ; WILL RESULT IN ERR 67 IF FAILS 2X
1695 004262 012737 177777 004370 MOV #SET,ERRFLG LET ERRFLG :- #SET
  
```

```

1696                                     ; DEFER ERROR TYPEOUT
1697
1698 004270                                     ELSE
1699 004270 000406                               BR      $20
1700 004272                                     $17:
1701 004272 004537 012512                     JSR      R5,TIMER
1702 004276 001276                               TCSR
1703 004300 000200                               RDY
1704 004302 104070                     ERROR 70
1705 004304 000433                     BR      TST12
1706                                     ;:EXIT TEST
1707                                     ; XMIT RDY DID NOT SET IN TCSR
1708 004306                                     ENDIF ; OF DEFERED ERROR CALL
1709 004306                                     IF ERRFLG EQ #SET THEN
1710 004306 023727 004370 177777                     CMP      ERRFLG,#SET
1711 004314 001013                     BNE      $21
1712 004316                                     LET PASS := PASS + #1
1713 004316 005237 004366                     INC      PASS
1714 004322                                     IF PASS GT #1 THEN
1715 004322 023727 004366 000001                     CMP      PASS,#1
1716 004330 003404                     BLE      $22
1717                                     ; CALL ERROR IF 2ND TRY
1718                                     ; ON XMIT RDY NOT CLEARING
1719 004332 104067                     ERROR 67
1720 004334                                     LET EXFLG := #SET
1721 004334 012737 177777 004372                     MOV      #SET,EXFLG
1722 004342                                     ENDIF
1723 004342                                     ELSE
1724 004342                                     ; NO ERROR
1725 004342 000403                               BR      $23
1726 004344                                     LET EXFLG := #SET
1727 004344                                     ENDIF
1728 004344 012737 177777 004372                     MOV      #SET,EXFLG
1729 004352                                     EXIF      EXFLG EQ #SET
1730 004352                                     ENDLOOP
1731 004352                                     $23:
1732 004352 023727 004372 177777                     CMP      EXFLG,#SET
1733 004360 001401                     BEQ      $16
1734 004362                                     $16:
1735 004362 000707                               BR      $15
1736 004364                                     BR      .+10
1737 004364 000403                               ;:EXIT TEST
1738
1739 004366 000000                     PASS:      0
1740 004370 000000                     ERRFLG:   0
1741 004372 000000                     EXFLG:    0

```

```

1742
1743
1744
1745
1746
1747
1748 004374 000004
1749 004376 012737 000010 001160
1750 004404 012737 000012 001200
1751
1752 004412 005737 012306
1753 004416 001027
1754 004420 005737 012310
1755 004424 001024
1756 004426
1757 004426 012737 004434 001110
1758 004434
1759
1760 004434
1761 004434 105077 174640
1762
1763 004440 004537 012512
1764 004444 001272
1765 004446 000200
1766 004450 104071
1767 004452 000411
1768
1769 004454
1770 004454 012737 004462 001110
1771 004462
1772
1773 004462 000005
1774 004464
1775 004464 032777 000200 174600
1776 004472 001401
1777
1778 004474 104072
1779 004476
1780 004476

*****
TEST 12      TEST THAT OUTPUTTING A CHAR FROM TBUF (WITH WRAP CONNECTED)
              RESULTS IN DONE SETTING WITHIN A REASONABLE AMOUNT OF TIME
              AND THAT RESET CLEARS THE BIT.
*****
TST12:  SCOPE
        MOV    #10,$TIMES      ;;DO 10 ITERATIONS
        MOV    #12,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

        TST    WRAP           ;DOING DATA WRAP TESTS?
        BNE    TST13          ;;EXIT IF NO
        TST    CONSOLE        ;ARE WE ON CONSOLE?
        BNE    TST13          ;;EXIT IF YES
                                LPADR
        MOV    #64,$LPERR

64$:
        ; SEND A CHARACTER AND LET IT WRAP AROUND
        ; LET @TBUF :B- #0
        CLRB   @TBUF

        JSR    R5,TIMER        ;WAIT FOR RECV DONE
                                RCSR
                                DONE
        ERROR   71
        BR     TST13          ;RECV DONE DID NOT SET IN RCSR
                                ;;EXIT TEST
                                ..
                                LPADR
        MOV    #65,$LPERR

65$:
        ; NOW THAT IT IS SET SEE IF IT CAN BE RESET
        RESET
                                IF #DONE SETIN @RCSR THEN
        BIT     #DONE,@RCSR
        BEQ    $24

                                ; DONE DID NOT RESET IN RCSR.
        ERROR 72
                                ENDIF

$24:

```

```

1781
1782
1783 .....
1784 :TEST 13      TEST THAT DONE IS CLEARED BY READING RBUF
1785 :.....
1786 TST13: SCOPE
1787 MOV      #10,$TIMES      ;;DO 10 ITERATIONS
1788 MOV      #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1789 TST      WRAP            ;DOING DATA WRAP TESTS?
1790 BNE      TST14           ;;EXIT IF NO
1791 TST      CONSOLE         ;ARE WE ON CONSOLE?
1792 BNE      TST14           ;;EXIT IF YES
1793                      LPADR
1794 MOV      #64$,$LPERR
1795 64$:
1796          ; OUTPUT A CHARACTER AND WAIT FOR DONE TO SET.
1797          ; OUTPUT A CHARACTER
1798          LET @TBUF :B #0
1799 CLRB      @TBUF
1800
1801 JSR      R5,TIMER          ;WAIT FOR RECV DONE
1802          RCSR
1803          DONE
1804 ERROR    73               ;RECV DONE DID NOT SET IN RCSR
1805 BR       TST14           ;;EXIT TEST
1806
1807          ; NOW THAT IT IS SET LETS SEE IF READING THE
1808          ; BUFFER CLEARS DONE.
1809
1810          ;READ BUFFER
1811          LET R0 :B- @RBUF
1812
1813          IF #DONE SETIN @RCSR THEN
1814 BIT      #DONE,@RCSR
1815 BEQ      $25
1816          ;DONE DID NOT CLEAR IN RCSR
1817 ERROR    74
1818          ; SET IT BACK TO CONTINUE
1819          ENDIF
1820 $25:
  
```

```

1821
1822
1823      ::*****
1824      ::TEST 14      TEST THE OVERRUN & ERROR BITS - RBUF 14
1825      ::*****
1825      TST14: SCOPE
1826      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
1827      MOV      #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1828
1829      TST      WRAP      ;DOING DATA WRAP TESTS?
1830      BNE      TST15      ;;EXIT IF NO
1831      TST      CONSOLE      ;ARE WE ON CONSOLE?
1832      BNE      TST15      ;;EXIT IF YES
1833
1834      004626
1835      004626      012737      004634      001110
1836      004634      64$:      MOV      #64$,$LPERR      LPADR
1837
1838      ;OUTPUT 2 CHARACTERS
1839      ;THIS SHOULD AN CAUSE OVERRUN ERROR.
1840
1841      ;OUTPUT 1 CHARACTER
1842      LET @TBUF :B #0
1843
1844      CLR      @TBUF
1845      JSR      R5,TIMER      ;WAIT FOR RECV DONE
1846      RCSR
1847      DONE
1848      ERROR      16      ;RECV DONE DID NOT SET IN RCSR
1849      BR      TST15      ;;EXIT TEST
1850
1851      ;OU*PUT 2ND CHARACTER
1852      LET @TBUF :B- #0
1853
1854      CLR      @TBUF
1855      JSR      PC,WAIT      ;LET OVERRUN HAPPEN
1856
1857      ;READ BUFFER AND ERROR BITS
1858      LET R4 := @RBUF
1859
1860      ;IT DIDN'T SET
1861      IF #ORERR NOTSET IN R4 THEN
1862      004670      032704      040000
1863      004674      001102
1864      004676      104101
1865
1866      ERROR 101
1867
1868      ;NO USE COMPOUNDING ERRORS
1869      BR      TST15      ;;EXIT TEST
1870      ENDIF
1871
1872      ;NOW SEE IF ERROR BIT SET WITH OVERRUN ERROR.
1873      LPADR
1874      004702      012737      004710      001110
1875      004710      64$:      MOV      #64$,$LPERR
1876      00471C      032704      100000
1877      BIT      #ERROR,R4
  
```

```

1877 004714 C01302 BNE $27
1878
1879 ;ERROR DID NOT SET IN RBUF
1880 004716 104102 ERROR 102
1881
1882 ;-WHEN ORERR SET.
1883 ;GET OUT NOW.
1884 004720 000437 BR TST15 ;;EXIT TEST
1885 004722 ENDF
1886 004722 $27:
1887
1888 LPADR
1889 004722 012737 004730 001110 MOV #64$, $LPERR
1890 004730 64$:
1891 ;CHECK REAL RBUF TO SEE IF ORERR IS STILL SET.
1892
1893 IF #ORERR NOTSET IN @RBUF THEN
1894 004730 032777 040000 174336 BIT #ORERR, @RBUF
1895 004736 001002 BNE $30
1896
1897 ;READING RBUF CLEARED ORERR.
1898 004740 104103 ERROR 103
1899 ;SKIP REST OF TEST
1900 004742 C00426 BR TST15 ;;EXIT TEST
1901 004744 ENDF
1902 004744 $30:
1903
1904 LPADR
1905 004744 012737 004752 001110 MOV #64$, $LPERR
1906 004752 64$:
1907 ;READING RBUF ABOVE SHOULD ENABLE ERROR TO BE CLEARED
1908 ;BY NEXT TRANSFER.
1909 ;NOW SEE IF THEY CLEAR WHEN ANOTHER CHAR. IS RECEIVED
1910
1911 ;SEND A CHARACTER AROUND.
1912 004752 LET @TBUF :B- #0
1913 004752 105077 174322 CLRB @TBUF
1914
1915 004756 004537 012512 JSR R5, TIMER ;WAIT FOR RECV DONE
1916 004762 001272 RCSR
1917 004764 000200 DONE
1918 004766 104020 ERROR 20 ;RECV DONE DID NOT SET IN RCSR
1919 004770 000413 BR TST15 ;;EXIT TEST
1920
1921 IF #ORERR SET IN @RBUF THEN
1922 004772 032777 040000 174274 BIT #ORERR, @RBUF
1923 005000 001402 BEQ $51
1924
1925 ;ORERR DID NOT CLEAR IN RBUF
1926 005002 104104 ERROR 104
1927
1928 ;-AFTER RECEIVING ANOTHER CHAR
1929 005004 000405 BR TST15 ;;EXIT TEST
1930 005006 ENDF
1931 005006 $51:
1932

```

CVMXAA0 MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 45
CVMXAA.P11 30-MAY-79 15:57 T14 TEST THE OVERRUN & ERROR BITS - PRUF 14

[illegible]

CVMXAA0 MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 46
 CVMXAA.P11 30-MAY-79 15:57 T14 TEST THE OVERRUN & ERROR BITS - RBUF 14

SEQ 0044

```

1941
1942
1943 *****
1944 *TEST 15 TRANSMITTER INTERRUPT LOGIC TEST
1945 * LOGICALLY THIS IS 4 SEPARATE TESTS
1946 * A) DOES TRANSMITTER INTERRUPT LOGIC WORK
1947 * B) AT PRIORITY OF 0
1948 * C) AND ONLY ONCE
1949 * D) BUT NOT WITH INTERRUPT ENABLE CLEAR
1950 *****
1950 005020 000004 TST15: SCOPE
1951 005022 012737 000010 001160 MOV #10,$TIMES ;;DO 10 ITERATIONS
1952 005030 012737 000015 001200 MOV #15,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
1953
1954 005036 005037 011644 CLR INTFLG
1955
1956 005042 013703 001302 MOV DLVEC,R3
1957 005046 062703 000004 ADD #4,R3 ;GET XMIT VECT ADDR
1958 005052 012723 011636 MOV #INTSRV,(R3)+
1959 005056 012713 000340 MOV #PR7,(R3)
1960 005062 LPADR
1961 005062 012737 005070 001110 MOV #648,$LPERR
1962 005070 648:
1963 005070 004537 012512 JSR R5,TIMER ;WAIT FOR XMIT READY
1964 005074 001276 TCSR
1965 005076 000200 RDY
1966 005100 104066 ERROR 66 ;XMIT RDY DID NOT SET IN TCSR
1967 005102 000453 BR TST16 ;;EXIT TEST
1968
1969 005104 ;CLEAR INTERRUPT ENABLE
1970 005104 042777 000100 174164 BIC #IE,@TCSR LET @TCSR := @TCSR CLR.BY #IE
1971 005112 SETPRI #PRO ;SET IT TO 0
1972
1973 ;NOW SET I.E. BIT
1974 005124 052777 000100 174144 BIS #IE,@TCSR LET @TCSR := @TCSR SET.BY #IE
1975 005132 004537 012566 JSR R5,TIMER1 ;SEE IF FLAG SETS
1976 005136 011644 INTFLG
1977 005140 104106 ERROR 106 ;INTERR FLAG DID NOT SET
1978 005142 000433 BR TST16 ;;EXIT TEST
1979
1980 ;BYPASS THIS PART IF CONSOLE
1981 005144 023737 012310 000000 CMP CONSOLE,0 IF CONSOLE EQ 0 THEN
1982 005152 001007 BNE $33
1983
1984 005154 004737 012630 JSR PC,WAIT ;LET POSSIBLE 2'ND INTERR OCCUR
1985
1986 ;DID EXACTLY 1 INTERRUPT OCCUR
1987 005160 IF INTFLG GT #1 THEN
1988 005160 023727 011644 000001 CMP INTFLG,#1
1989 005166 003401 BLE $34
1990
1991 ;TRANSMITTER INTERRUPTED TWICE
1992 005170 104107 ERROR 107
1993
1994 ENDIF
1995
1996 $34:
1997 $33:
1998
1999
2000

```

```

;CLEAR INTERRUPT ENABLE
LET @TCSR := @TCSR.CLR.BY #IE

;CLEAR 'INTERRUPT OCCURED' FLAG
LET INTFLG := #0

```

```
;NO INTERRUPTS SHOULD OCCUR, PSW STILL AT 0.
;DARE IT TO HAPPEN
```

```

LEI AIBUF := #()

```

```
;SEE IF INTERR FLAG EVER SETS
```

AR 1E NO

```

;INTERR FLAG OCCURED WITH IF DISABLED

```

```

2018
2019
2020 .....
2021 *TEST 16 RECEIVER INTERRUPT LOGIC TEST
2022 * THIS TEST COVERS ALL OF THE RECEIVER
2023 * SIDE OF THE INTERRUPT LOGIC IN
2024 * CHARACTER MODE.
2025 .....
2025 005232 000004 TST16: SCOPE
2026 005234 012737 000010 001160 MOV #10,STIMES ;;DO 10 ITERATIONS
2027 005242 012737 000016 001200 MOV #16,STESTN ;;SET TEST NUMBER IN APT MAIL BOX
2028
2029 005250 005737 012306 TST WRAP ;;DOING DATA WRAP TESTS?
2030 005254 001103 BNE TST17 ;;EXIT IF NO
2031 005256 005737 012310 TST CONSOLE ;;ARE WE ON CONSOLE?
2032 005262 001100 BNE TST17 ;;EXIT IF YES
2033
2034 ;;CLEAR INTERRUPT OCCURED FLAG
2035 005264 013701 001302 MOV DLVEC,R1 ;;SET UP RECEIVER INTER.VECTOR
2036 005270 012721 011636 MOV #INTSRV,(R1)+
2037 005274 012711 000340 MOV #PR7,(R1)
2038
2039 ;;PRIORITY 0 AND MULTIPLE INTERRUPT TEST.-IE
2040 005300 LPADR
2041 005300 012737 005306 001110 MOV #648,SLPERR
2042 005306
2043 005306 648:
2044 005306 005037 011644 CLR INTFLG LET INTFLG := #0
2045
2046 005312 ;;CLEAR INTERRUPTS
2047 005312 042777 000100 173752 BIC #IE,@RCSR LET @RCSR := @RCSR CLR.BY #IE
2048
2049 ;;CHANGE PRIORITY
2050 005320 SETPRI #PRO ;;...TO 0
2051
2052 ;;SEND A CHARACTER
2053 005332 LET @TBUF :=B- #0
2054 005332 105077 173742 CLRB @TBUF
2055
2056 005336 004537 012512 JSR R5,TIMER ;;WAIT FOR XMIT READY
2057 005342 001276 TCSR
2058 005344 000200 RDY
2059 005346 104066 ERROR 66 ;;XMIT RDY DID NOT SET IN TCSR
2060 005350 000445 BR TST17 ;;EXIT TEST
2061
2062 ;;SET INTERRUPT ENABLE
2063 005352 LET @RCSR := @RCSR SET.BY #IE
2064 005352 052777 000100 173712 BIS #IE,@RCSR
2065
2066 005360 004537 012566 JSR R5,TIMER1 ;;SEE IF FLAG SETS
2067 005364 011644 INTFLG
2068 005366 104111 ERROR 111 ;;INTERR FLAG DID NOT SET
2069 005370 000435 BR TST17 ;;EXIT TEST
2070
2071 005372 004737 012630 JSR PC,WAIT ;;LET POSSIBLE 2ND INTERR OCCUR
2072
2073 005376 ;;EXACTLY 1 INTERRUPT?
                IF INTFLG GT #1 THEN
  
```

```

2074 005376 023727 011644 000001      CMP      INTFLG,#1
2075 005404 003401                      BIE      $35
2076                                     ;RECEIVER INTERRUPTED TWICE
2077 005406 104112                      ERROR    112
2078 005410                                     ENDIF
2079 005410      $35:
2080
2081                                     ;INTERRUPT WITHOUT IE SET.
2082 005410                                     LPADR
2083 005410 012737 005416 001110      MOV      #64$,$LPERR
2084 005416      64$:
2085                                     ;CLEAR INTERRUPT
2086 005416      LET @RCSR := @RCSR CLR.BY #IE
2087 005416 042777 000100 173646      BIC      #IE,@RCSR
2088                                     ;CLEAR INTERRUPT FLAG
2089 005424      LET INTFLG := #0
2090 005424 005037 011644      CLR      INTFLG
2091                                     ; SEND A CHARACTER
2092 005430      LET @TBUF :B- #0
2093 005430 105077 173644      CLRB     @TBUF
2094                                     ; DARE IT
2095 005434 004737 012630      JSR      PC,WAIT ;SEE IF INTERR FLAG EVER SETS
2096 005440 005737 011644      TST      INTFLG
2097 005444 001401                      BEQ      .+4 ;BR IF NO
2098 005446 104113                      ERROR    113 ;INTERR FLAG OCCURED WITH IE DISABLED
2099
2100 005450      SETPRI #PR7 ;RAISE CPU PRIORITY
2101 005462 000005      RESET ;CLEAR THE WORLD
2102

```

```

2103
2104
2105
2106
2107 005464 000004
2108 005466 012737 000001 001160
2109 005474 012737 000017 001200
2110
2111 005502 005737 012306
2112 005506 001062
2113 005510 005737 012310
2114 005514 001057
2115
2116 005516
2117 005516 005002
2118 005520 000401
2119 005522
2120 005522 005202
2121 005524
2122 005524 020227 000377
2123 005530 003051
2124
2125 005532 004537 012512
2126 005536 001276
2127 005540 000200
2128 005542 104123
2129 005544 000443
2130
2131
2132 005546
2133 005546 110277 173526
2134
2135 005552 004537 012512
2136 005556 001272
2137 005560 000200
2138 005562 104124
2139 005564 000433
2140
2141
2142 005566
2143 005566 017703 173502
2144
2145 005572
2146 005572 032703 100000
2147 005576 001401
2148
2149 005600 104200
2150 005602
2151 005602
2152 005602 020302
2153 005604 001412
2154 005606 005737 012300
2155 005612 001002
2156 005614 104017
2157 005616 000416
2158

*****
*TEST 17      TEST DATA WRAP AROUND BINARY COUNT: FLAG MODE
*****
TST17: SCOPE
MOV      #1,$TIMES      ;;DO 1 ITERATION
MOV      #17,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

TST      WRAP           ;DOING DATA WRAP TESTS?
BNE      TST20          ;;EXIT IF NO
TST      CONSOLE        ;ARE WE ON CONSOLE?
BNE      TST20          ;;EXIT IF YES

                        ;BINARY COUNT PATTERN
                        INCR R2 FROM #0 TO #377 BY #1

$37:     CLR      R2
          BR       $36
          INC      R2
$36:     CMP      R2,#377
          BGT      $40

          JSR      R5,TIMER      ;WAIT FOR XMIT READY
          TCSR
          RDY
          ERROR    123
          BR       TST20        ;XMIT RDY DID NOT SET IN TCSR
                                   ;;EXIT TEST

                                   ;START IT ON ITS WAY
                                   LET @TBUF :=@R2

          MOV      R2,@TBUF

          JSR      R5,TIMER      ;WAIT FOR RECV DONE
          RCSR
          DONE
          ERROR    124
          BR       TST20        ;RCV DONE DID NOT SET IN RCSR
                                   ;;EXIT TEST

                                   ;RETRIEVE
                                   LET R3 := @RBUF

                                   ;CHECK FOR ERROR DURING TRANSFER
                                   IF #ERROR SET IN R3 THEN

                                   ;ERROR BIT SET IN HIGH BYTE OF RBUF
                                   ENDIF

$41:     CMP      R3,R2          ;COMPARE DATA
          BEQ      $5          ;BR IF OK
          TST      WORDL        ;8 BITS/WORD?
          BNE      $5          ;BR IF NO
          ERROR    17          ;DATA COMPARE ERR IN 8 BIT WORD
          BR       TST20        ;;EXIT TEST
  
```

CVMXAAO MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 51
 CVMXAA.P11 30-MAY-79 15:57 117 TEST DATA WRAP AROUND BINARY COUNT: FLAG MODE

SEQ 0049

2159	005620	020227	000200	2\$:	CMP	R2,#200		;200 OR MORE?
2160	005624	002012			BGE	4\$		;BR IF YES
2161	005626	104117			ERROR	117		;DATA COMPARE ERR IN 7 BIT WORD
2162	005630	000411			BR	TS120	::EXIT	TEST
2163								
2164	005632	005737	012300	3\$:	TST	WORDL		;8 BITS/WORD?
2165	005636	001405			BEQ	4\$		;BR IF YES
2166	005640	020227	000177		CMP	R2,#177		;MORE THAN 177?
2167	005644	003402			BLE	4\$		;BR IF NO
2168	005646	104022			ERROR	22		;GETTING 8 BITS ON 7 BIT XMIT
2169								;IS SDEVN SETUP OK?
2170	005650	000401			BR	.+4		;EXIT
2171								
2172	005652			4\$:			ENDINC	;R2
2173	005652	000723			BR	\$37		
2174	005654			\$40:				

```

2175
2176
2177
2178
2179 005654 000004
2180 005656 012737 000001 001160
2181 005664 012737 000020 001200
2182
2183 005672 005737 012306
2184 005676 001003
2185 005700 005737 012310
2186 005704 001402
2187 005706 000137 006176
2188 005712
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205 005712
2206
2207 005724
2208 005724 013701 001302
2209
2210 005730
2211 005730 012721 006242
2212 005734
2213 005734 012721 000340
2214
2215
2216 005740
2217 005740 012721 006204
2218 005744
2219 005744 012711 000340
2220
2221
2222 005750
2223 005750 005037 006200
2224 005754
2225 005754 005037 006240
2226
2227 005760 005737 012300
2228 005764 001004
2229 005766 012737 000400 006202
2230 005774 000403
  
```

```

*****
:TEST 20      TEST DATA WRAP AROUND  BINARY COUNT: INTERRUPT MODE
*****
TST20:  SCOPE
        MOV     #1,$TIMES      ;;DO 1 ITERATION
        MOV     #20,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

        TST     WRAP           ;DOING DATA WRAP TESTS?
        BNE     1$             ;BR IF NO
        TST     CONSOLE        ;ARE WE ON CONSOLE?
        BEQ     2$             ;BR IF NO
        JMP     EX1            ;EXIT TEST

1$:
2$:
        ; THIS TEST WILL RUN BOTH TRANSMITTER AND
        ; RECIEVER AT FULL SPEED TESTING
        ; THE ABILITY OF THE MODULE
        ; TO HANDLE INTERRUPTS FROM BOTH SIDES AT ONCE.

        ;DOUBLE BUFFERING IS NOT FULLY TESTED BECAUSE OF
        ;APT CONSIDERATIONS. I.E. 'BREAK' FROM APT
        ;CAUSES OVERRUN ERRORS. THEREFORE TRANSMIT INTR IS
        ;ENABLED ONLY AFTER THE RECVR HAS OBTAINED THE LAST WORD

        ; THIS TEST WILL TRANSFER A MAXIMUM OF 400(8)
        ; CHARACTERS THROUGH THE MODULE, BUT IF AN ERROR
        ; IS DETECTED BY THE TEST A PREMATURE SHUTDOWN OCCURS.

        ;CHANGE PRIORITY
        ;...TO 0
        SETPRI #PRO

        ;GET VECTOR ADDRESS
        LET R1 := DLVEC

        ;RCVR VECTOR
        LET (R1)+ := #REC

        LET (R1)+ := #PR7

        ;POINT TO TRANSMITTER VECTOR
        ;AND SET IT UP ALSO
        LET (R1)+ := #TRAN

        LET (R1) := #PR7

        ; CLEAR ERROR COUNTER
        LET ERRCNT := #0

        LET DATA := #0 ;XMIT DATA

        TST     WORDL          ;8 BITS/WORD?
        BNE     3$             ;BR IF NO
        MOV     #400,NUMBER
        BR      4$
  
```

Address	Op Code	Op 1	Op 2	Op 3	Op 4	Op 5	Op 6	Op 7	Op 8	Op 9	Op 10	Op 11	Op 12	Op 13	Op 14	Op 15	Op 16	Op 17	Op 18	Op 19	Op 20	Op 21	Op 22	Op 23	Op 24	Op 25	Op 26	Op 27	Op 28	Op 29	Op 30	Op 31	Op 32	Op 33	Op 34	Op 35	Op 36	Op 37	Op 38	Op 39	Op 40	Op 41	Op 42	Op 43	Op 44	Op 45	Op 46	Op 47	Op 48	Op 49	Op 50	Op 51	Op 52	Op 53	Op 54	Op 55	Op 56	Op 57	Op 58	Op 59	Op 60	Op 61	Op 62	Op 63	Op 64	Op 65	Op 66	Op 67	Op 68	Op 69	Op 70	Op 71	Op 72	Op 73	Op 74	Op 75	Op 76	Op 77	Op 78	Op 79	Op 80	Op 81	Op 82	Op 83	Op 84	Op 85	Op 86	Op 87	Op 88	Op 89	Op 90	Op 91	Op 92	Op 93	Op 94	Op 95	Op 96	Op 97	Op 98	Op 99	Op 100	Op 101	Op 102	Op 103	Op 104	Op 105	Op 106	Op 107	Op 108	Op 109	Op 110	Op 111	Op 112	Op 113	Op 114	Op 115	Op 116	Op 117	Op 118	Op 119	Op 120	Op 121	Op 122	Op 123	Op 124	Op 125	Op 126	Op 127	Op 128	Op 129	Op 130	Op 131	Op 132	Op 133	Op 134	Op 135	Op 136	Op 137	Op 138	Op 139	Op 140	Op 141	Op 142	Op 143	Op 144	Op 145	Op 146	Op 147	Op 148	Op 149	Op 150	Op 151	Op 152	Op 153	Op 154	Op 155	Op 156	Op 157	Op 158	Op 159	Op 160	Op 161	Op 162	Op 163	Op 164	Op 165	Op 166	Op 167	Op 168	Op 169	Op 170	Op 171	Op 172	Op 173	Op 174	Op 175	Op 176	Op 177	Op 178	Op 179	Op 180	Op 181	Op 182	Op 183	Op 184	Op 185	Op 186	Op 187	Op 188	Op 189	Op 190	Op 191	Op 192	Op 193	Op 194	Op 195	Op 196	Op 197	Op 198	Op 199	Op 200	Op 201	Op 202	Op 203	Op 204	Op 205	Op 206	Op 207	Op 208	Op 209	Op 210	Op 211	Op 212	Op 213	Op 214	Op 215	Op 216	Op 217	Op 218	Op 219	Op 220	Op 221	Op 222	Op 223	Op 224	Op 225	Op 226	Op 227	Op 228	Op 229	Op 230	Op 231	Op 232	Op 233	Op 234	Op 235	Op 236	Op 237	Op 238	Op 239	Op 240	Op 241	Op 242	Op 243	Op 244	Op 245	Op 246	Op 247	Op 248	Op 249	Op 250	Op 251	Op 252	Op 253	Op 254	Op 255	Op 256	Op 257	Op 258	Op 259	Op 260	Op 261	Op 262	Op 263	Op 264	Op 265	Op 266	Op 267	Op 268	Op 269	Op 270	Op 271	Op 272	Op 273	Op 274	Op 275	Op 276	Op 277	Op 278	Op 279	Op 280	Op 281	Op 282	Op 283	Op 284	Op 285	Op 286	Op 287	Op 288	Op 289	Op 290	Op 291	Op 292	Op 293	Op 294	Op 295	Op 296	Op 297	Op 298	Op 299	Op 300	Op 301	Op 302	Op 303	Op 304	Op 305	Op 306	Op 307	Op 308	Op 309	Op 310	Op 311	Op 312	Op 313	Op 314	Op 315	Op 316	Op 317	Op 318	Op 319	Op 320	Op 321	Op 322	Op 323	Op 324	Op 325	Op 326	Op 327	Op 328	Op 329	Op 330	Op 331	Op 332	Op 333	Op 334	Op 335	Op 336	Op 337	Op 338	Op 339	Op 340	Op 341	Op 342	Op 343	Op 344	Op 345	Op 346	Op 347	Op 348	Op 349	Op 350	Op 351	Op 352	Op 353	Op 354	Op 355	Op 356	Op 357	Op 358	Op 359	Op 360	Op 361	Op 362	Op 363	Op 364	Op 365	Op 366	Op 367	Op 368	Op 369	Op 370	Op 371	Op 372	Op 373	Op 374	Op 375	Op 376	Op 377	Op 378	Op 379	Op 380</
---------	---------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	----------

```

2287 006166 000401          BR      $51
2288 006170          $50:
2289                                ;UNKNOWN ERROR
2290 006170 104024          ERROR 24
2291 006172                                ENDIF
2292 006172          $51:
2293 006172          $47:
2294 006172          $45:
2295 006172                                ELSE
2296 006172 000401          BR      $52
2297 006174          $43:
2298                                ;DATA COMPARE ERROR
2299 006174 104120          ERROR 120
2300 006176                                ENDIF
2301 006176          $52:
2302 006176                                ENDIF
2303 006176          $42:
2304 006176          EX1:
2305 006176 000464          BR      TST21          ;;EXIT TEST
2306
2307
2308 006200 000000          ERRCNT: 0
2309 006202 000000          NUMBER: 0
2310
2311
2312
2313          ;;*****
2314                                ;TRANSMIT INTERRUPT HANDLER
2315          ;;*****
2316
2317 006204          TRAN:          IF DATA NE NUMBER AND ERRCNT EQ #0 THEN
2318 006204 023737 006240 006202          CMP      DATA,NUMBER
2319 006212 001406          BEQ      $53
2320 006214 005737 006200          TST      ERRCNT
2321 006220 001003          BNE      $53
2322                                ;SHIP OUT WORD
2323 006222                                LET @TBUF := DATA
2324 006222 013777 006240 173050          MOV      DATA,@TBUF
2325 006230                                ENDIF
2326 006230          $53:
2327                                ;STOP INTERRUPT, NOT EXER DOUBL BLFFER
2328 006230                                LET @TCSR := @TCSR CLR.BY #IE
2329 006230 042777 000100 173040          BIC      #IE,@TCSR
2330 006236 000002          RTI
2331
2332 006240 000000          DATA: 0
  
```

```

2333
2334 ;:.....
2335 ;RECEIVER INTERRUPT HANDLER
2336 ;:.....
2337
2338 006242 REC: ;GET CHAR
2339 006242 LET RMLD := @RBUF
2340 006242 017737 173026 006346 MOV @RBUF,RMLD
2341 ;CHECK ERROR
2342 006250 IF #ERROR SETIN RMLD OR RMLD NE DATA THEN
2343 006250 032737 100000 006346 BIT #ERROR,RMLD
2344 006256 001004 BNE $54
2345 006260 023737 006346 006240 CMP RMLD,DATA
2346 006266 001411 BEQ $55
2347 006270 $54:
2348 ;STOP ALL INTERR PROC & GET OUT
2349 006270 LET DATA := NUMBER
2350 006270 013737 006202 006240 MOV NUMBER,DATA
2351 006276 LET @RCSR := @RCSR CLR.BY #IE
2352 006276 042777 000100 172766 BIC #IE,@RCSR
2353 006304 LET ERRCNT := ERRCNT + #1
2354 006304 005237 006200 INC ERRCNT
2355 006310 ELSE
2356 006310 000415 BR $56
2357 006312 $55:
2358 006312 LET DATA := DATA + #1
2359 006312 005237 006240 INC DATA
2360 006316 IF DATA EQ NUMBER THEN
2361 006316 023737 006240 006202 CMP DATA,NUMBER
2362 006324 001004 BNE $57
2363 006326 LET @RCSR := @RCSR CLR.BY #IE
2364 006326 042777 000100 172736 BIC #IE,@RCSR
2365 006334 ELSE
2366 006334 000403 BR $60
2367 006336 $57:
2368 ;ALLOW NEXT XMIT INTERR
2369 006336 LET @TCSR := @TCSR SET.BY #IE
2370 006336 052777 000100 172732 BIS #IE,@TCSR
2371 006344 ENDIF
2372 006344 $60:
2373 006344 $56:
2374 006344 RTI
2375 006344 000002
2376 006346 000000 RMLD: 0
2377
2378

```

```

2379
2380
2381 .....
2382 *TEST 21      TEST BREAK LOGIC
2383 *             TRANSMIT KNOWN CHAR WITH BREAK SET
2384 *             AND COMPARE RECEIVED WITH 0.
2385 *             FRAMING ERROR WILL ALSO BE CHECKED
2386 *             IF ERROR BITS ARE ENABLED.
2387 .....
2387 006350 000004 TST21: SCOPE
2388 006352 012737 000010 001160 MOV #10,$TIMES ;;DO 10 ITERATIONS
2389 006360 012737 000021 001200 MOV #21,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
2390
2391 006366 005737 012306 TST WRAP ;WRAP TESTS?
2392 006372 001006 BNE 1$ ;BR IF NO
2393 006374 005737 012310 TST CONSOLE ;ELSE ARE WE ON CONSOLE?
2394 006400 001003 BNE 1$ ;BR IF YES...DONT TEST
2395 006402 005737 012304 TST BRK ;ELSE IS BREAK DET ENABLED?
2396 006406 001402 BEQ 2$ ;BR IF NO
2397 006410 000137 006744 1$: JMP EX2
2398 006414 2$:
2399 006414 LPADR
2400 006414 012737 006422 001110 MOV #64,$LPERR
2401 006422 64$:
2402
2403 006422 LET ERRCHK := #0 ; CLEAR ERROR WORD
2404 006422 005037 006746 CLR ERRCHK
2405 ;SET BREAK BIT
2406 006426 LET @TCSR := @TCSR SET.BY #BREAK
2407 006426 052777 000001 172642 BIS #BREAK,@TCSR
2408 ;NON-ZERO CHAR. ''
2409 006434 LET @TBUF := #125
2410 006434 012777 000125 172636 MOV #125,@TBUF
2411
2412 006442 004537 012512 JSR R5,TIMER ;WAIT FOR RECV DONE
2413 006446 001272 RCSR
2414 006450 000200 DONE
2415 006452 104115 ERROR 115 ;RECV DONE DID NOT SET IN RCSR
2416 006454 000535 BR TST22 ;;EXIT TEST
2417
2418 006456 LET R0 := @RBUF
2419 006456 017700 172612 MOV @RBUF,R0
2420 006462 IFB R0 NE #0 THEN
2421 006462 105700 TSTB R0
2422 006464 001403 BEQ $61
2423
2424 ; BREAK DID NOT EQUAL 0
2425 006466 052737 000001 006746 BIS #BIT0,ERRCHK
2426 006474 LET ERRCHK := ERRCHK SET.BY #BIT0
2427 006474 $61: ENDF
2428 006474 IF #FRERR NOTSETIN R0 THEN
2429 006474 032700 020000 BIT #FRERR,R0
2430 006500 001003 BNE $62
2431 006502 LET ERRCHK := ERRCHK SET.BY #BIT1
2432 006502 052737 000002 006746 BIS #BIT1,ERRCHK
2433 006510 ENDF
2434 006510 $62:

```

```

2435 006510 005737 012300          TST      WORDL          ;PARITY ENABLED?
2436 006514 001420          BEQ      4$              ;BR IF NO
2437 006516 005737 012302          TST      ODD              ;ELSE ODD?
2438 006522 001407          BEQ      3$              ;BR IF YES
2439 006524 032700 010000          BIT      #PERR,R0          ;DONT EXPECT PARITY ERR
2440 006530 001412          BEQ      4$              ;BR IF NONE
2441 006532 052737 000010 006200    BIS      #BIT3,ERRCNT      ;PARITY ERR WHEN NOT EXPECTED
2442 006540 000406          BR        4$
2443
2444 006542 032700 010000          3$:    BIT      #PERR,R0          ;EXPECT PARITY ERR
2445 006546 001003          BNE      4$              ;BR IF THERE
2446 006550 052737 000004 006746    BIS      #BIT2,ERRCHK      ;NO PARITY ERR WHEN EXPECTED
2447
2448 006556 000005          4$:    RESET
2449 006560 032777 170000 172506    BIT      #170000,RBUF
2450 006566 001401          BEQ      1$
2451 006570 104033          ERROR    33          ;RESET DID NOT CLEAR ERROR,FR ERR,OR PERR IN RBUF
2452 006572          1$:
2453
2454 006572          -
2455 006572 032737 000001 006746    BIT      #BIT0,ERRCHK          IF #BIT0 SET IN ERRCHK THEN
2456 006600 001401          BEQ      $63              ;BREAK ERROR
2457 006602 104121          ERROR    121              ENDIF
2458 006604
2459 006604          $63:
2460 006604
2461 006604 032737 000002 006746    BIT      #BIT1,ERRCHK          IF #BIT1 SET IN ERRCHK THEN
2462 006612 001401          BEQ      $64              ;FRAMING ERROR
2463 006614 104122          ERROR    122              ENDIF
2464 006616
2465 006616          $64:
2466 006616
2467 006616 032737 000004 006746    BIT      #BIT2,ERRCHK          IF #BIT2 SET IN ERRCHK THEN
2468 006624 001401          BEQ      $65              ;NO PARITY ERROR WHEN
2469 006626 104235          ERROR    235              ;THERE SHOULD BE
2470
2471 006630          $65:
2472 006630
2473 006630
2474 006630 032737 000010 006746    BIT      #BIT3,ERRCHK          IF #BIT3 SET IN ERRCHK THEN
2475 006636 001401          BEQ      $66              ;PARITY ERROR SHOULD NOT HAVE
2476 006640 104236          ERROR    236              ;OCCURED WITH EVEN PARITY
2477
2478
2479 006642          $66:
2480 006642
2481 006642
2482 006642 012737 006650 001110    MOV      #64$,$LPERR          LPADR
2483 006650          64$:
2484
2485 006650          ;SET BREAK BIT
2486 006650 052777 000001 172420    BIS      #BREAK,$TCSR        LET $TCSR := $TCSR SET BY #BREAK
2487
2488 006656 004537 012512          JSR      R5,TIMER          ;WAIT FOR XMIT READY
2489 006662 001276          TCSR
2490 006664 000001          BREAK

```

```

2491 006666 104021          ERROR 21          ;BREAK DID NOT SET IN TCSR
2492 006670 000427          BR      TST22      ;;EXIT TEST
2493
2494                                ;CLEAR BREAK BIT
2495 006672                                LET @TCSR := @TCSR CLR.BY #BREAK
2496 006672 042777 000001 172376      BIC      #BREAK,@TCSR
2497
2498 006700 004737 012630          JSR      PC,WAIT
2499                                ;READ RBUF TO CLEAR ERRORS & REC DONE
2500 006704                                LET RO := @RBUF
2501 006704 017700 172364          MOV      @RBUF,RO
2502                                ;SEND CHAR
2503 006710                                LET @TBUF := #125
2504 006710 012777 000125 172362      MOV      #125,@TBUF
2505
2506 006716 004537 012512          JSR      R5,TIMER          ;WAIT FOR RECV DONE
2507 006722 001272                                RCSR
2508 006724 000200                                DONE
2509 006726 104230          ERROR 230          ;RECV DONE DID NOT SET IN RCSR
2510 006730 000407          BR      TST22      ;;EXIT TEST
2511
2512                                ;WAS CHAR AFTER BREAK RECEIVED
2513 006732                                IFB @RBUF NE #125 THEN
2514 006732 127727 172336 000125      CMPB     @RBUF,#125
2515 006740 001401          BEQ      $67
2516                                ;CHAR AFTER BREAK NOT RECEIVED CORRECTLY
2517 006742 104231          ERROR 231
2518 006744          ENDIF
2519 006744          $67:
2520 006744          EX2:
2521 006744 000401          BR      TST22      ;;EXIT TEST
2522
2523 006746 000000          ERRCHK: .WORD 0
  
```

```

2524
2525 006750
2526
2527
2528
2529 006750 000004
2530 006752 012737 000001 001160
2531
2532 006760 032777 010000 172152
2533 006766 001513
2534 006770 023727 012312 000001
2535 006776 001021
2536 007000
2537 007040 000425
2538
2539 007042 005737 012314
2540 007046 001022
2541 007050
2542 007110 005237 012314
2543
2544 007114
2545 007130
2546 007136
2547 007156
2548 007164
2549 007204
2550 007212 104401 001171
2551
2552 007216 005037 001112
2553 007222 023727 012312 000001
2554 007230 001010
2555 007232 005037 012312
2556 007236 005037 001206
2557 007242 104401 001310
2558 007246 000137 013334
2559
2560 007252 112737 000005 001102
2561 007260 000137 003126
2562

NOTST:
*****
;TEST 22 NOT A TEST - SEND BACK TO LOOP
*****
TST22: SCOPE
MOV #1,$TIMES ;;DO 1 ITERATION
BIT #BIT12,$SWR ;WANT SUMMARY?
BEQ 3$ ;BR IF NO
CMP PHASE2,$TRUE ;ELSE IN PHASE 2?
BNE 1$ ;BR IF NO
TYPTXT <<CRLF>/ ** PHASE 2 SUMMARY **/<CRLF>>
BR 2$

1$: TST P1CNT
BNE 2$
TYPTXT <<CRLF>/ ** PHASE 1 SUMMARY **/<CRLF>>
INC P1CNT

2$: TYPTXT <*,CSR: *>
TYPOCT DLADD
TYPTXT <*,VECTOR: *>
TYPOCT DLVEC
TYPTXT <*,ERRORS: *>
TYPDEC $ERTTL
TYPE ,SCRLF

3$: CLR $ERTTL ; RESET FOR NEXT DEVICE/PASS
CMP PHASE2,$TRUE ;IN PHASE 2?
BNE 4$ ;BR IF NO
CLR PHASE2
CLR $UNIT
TYPE ,CARR
JMP $EOP

4$: MOVB #5,$ISTNM ;FAKE OUT FOR 2ND CHANNEL
JMP LOOP ;BACK UP TO THE BEGINNING
  
```

CVMXAAO MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 60
 CVMXAA.P11 30-MAY-79 15:57 T22 NOT A TEST - SEND BACK TO LOOP

SEQ 0058

```

2563
2564 007264          MODTST:
2565
2566                ;*****
2567                ;*TEST 23      TEST THAT CHANNELS INTR AT ASSIGNED PRIORITY.
2568                ;*              INTERRUPTS WILL BE ENABLED ON ALL ACTIVE CHANNELS.
2569                ;*              RECEIVER AND TRANSMITTER. THEN WE'LL CHECK TO
2570                ;*              SEE IF THEY INTERRUPTED IN THE ASSIGNED SEQUENCE.
2571                ;*****
2572 007264 000004      TST23: SCOPE
2573 007266 012737 000001 001160      MOV    #1,$TIMES      ;;DO 1 ITERATION
2574 007274 012737 000023 001200      MOV    #23,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
2575
2576 007302 005037 010040      CLR    INTABL          ;CLEAR OUT INTERRUPT TABLE
2577 007306 005037 010042      CLR    INTABL+2
2578 007312 005037 010044      CLR    INTABL+4
2579 007316 005037 010046      CLR    INTABL+6
2580
2581 007322          SETPRI #PR7
2582
2583 007334 032737 102000 001252      BIT    #<BIT15:BIT10>,$DEV0    ;CH 0 DROPPED OR NO WRAP?
2584 007342 001040          BNE    1$              ;BR IF EITHER SET
2585
2586 007344 013700 001256      MOV    VECT0,R0          ;SETUP CH 0 VECTOR AREA
2587 007350 012720 007742      MOV    #CH0R,(R0)+        ;PCV
2588 007354 012720 000340      MOV    #PR7,(R0)+
2589 007360 012720 007756      MOV    #CH0X,(R0)+        ;XMIT
2590 007364 012710 000340      MOV    #PR7,(R0)
2591
2592 007370 013700 001254      MOV    BASE0,R0          ;ENABLE INTERRUPTS
2593 007374 052710 000100      BIS    #IE,(R0)          ;RCV
2594 007400 062700 000004      ADD    #4,R0
2595 007404 052720 000100      BIS    #IE,(R0)+        ;XMIT
2596
2597 007410 012710 000125      MOV    #125,(R0)          ;XMIT CHAR TO PRIME RECEIVERS
2598 007414 162700 000006      SUB    #6,R0              ;GO BACK TO RCSR
2599 007420 012737 100000 001304      MOV    #100000,TMP1
2600 007426 032710 000200          BIT    #DONE,(R0)      ;DONE?
2601 007432 001004          BNE    1$              ;BR IF YES
2602 007434 005337 001304      DEC    TMP1              ;TIMED OUT?
2603 007440 001372          BNE    7$              ;BR IF NO
2604 007442 104070          ERROR  70              ;NO RCVR DONE
2605
2606 007444 032737 000410 001252 1$: BIT    #<BIT18:BIT13>,$DEV0    ;CH 1 DROPPED OR NO WRAP?
2607 007452 001044          BNE    2$              ;BR IF EITHER SET
2608 007454 023737 001260 001144      CMP    BASE1,$TKS    ;CH 1 CONSOLE?
2609 007462 001440          BEQ    2$              ;BR IF YES
2610
2611 007464 013700 001262      MOV    VECT1,R0          ;SETUP CH 1 VECTOR AREA
2612 007470 012720 010000      MOV    #CH1R,(R0)+        ;RCV
2613 007474 012720 000340      MOV    #PR7,(R0)+
2614 007500 012720 010016      MOV    #CH1X,(R0)+        ;XMIT
2615 007504 012710 000340      MOV    #PR7,(R0)
2616
2617 007510 013700 001260      MOV    BASE1,R0          ;ENABLE INTERRUPTS
2618 007514 052710 000100      BIS    #IE,(R0)          ;RCV

```

CVMXAA0 MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 61
 CVMXAA.P11 30-MAY-79 15:57 123 TEST THAT CHANNELS INTR AT ASSIGNED PRIORITY.

SEQ 0059

```

2619 007520 062700 000004      ADD      #4,R0
2620 007524 052720 000100      BIS      #1E,(R0)+      ;XMIT
2621
2622 007530 012710 000125      MOV      #125,(R0)      ;XMIT CHAR TO PRIME RECEIVERS
2623 007534 162700 000006      SJB      #6,R0      ;GO BACK TO RCSR
2624 007540 012737 100000 001304      MOV      #100000,TMP1
2625 007546 032710 000200      3$:      BIT      #DONE,(R0)      ;DONE?
2626 007552 001004      BNE      2$      ;BR IF YES
2627 007554 005337 001304      DEC      TMP1      ;TIMED OUT?
2628 007560 001372      BNE      8$      ;BR IF NO
2629 007562 104071      ERROR      71      ;NO RCVR DONE
2630
2631      ;ALL XMIT & REC INTERRUPTS SHOULD BE
2632      ;IN CONTENTION.
2633      ;CHAN 0 HAS PRIORITY OVER CHAN 1.
2634      ;RECEIVE HAS PRIORITY OVER XMIT INTERRUPTS.
2635      ;THEREFORE, ONCE CPU PRIORITY IS LOWERED,
2636      ;INTERRUPTS S/B IN THE ORDER SHOWN IN THE
2637      ;CHANNEL IDENTIFIER TABLE (RCH0:)
2638
2639 007564 032737 102000 001252 2$:      BIT      #<BIT15:BIT10>,$DEV0      ;CH0 DROPPED OR NO WRAP?
2640 007572 001003      BNE      5$      ;BR IF EITHER SET
2641
2642 007574 012704 010040      MOV      #INTABL,R4      ;SETUP FOR SERVICE ROUTINES
2643 007600 000402      BR      6$
2644
2645 007602 012704 010044      5$:      MOV      #INTABL+4,R4
2646
2647 007606      6$:      SETPRI      #PRO      ;LET'EM GO!
2648 007620 012700 177777      MOV      #-1,R0      ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
2649 007624 077001      SOB      R0,,      ;FINISH BEFORE RESET
2650 007626 000005      RESET      ;DISABLE ALL INTERRUPTS
2651 007630
2652
2653      ;NOW LETS SEE IF INTABL HAS ENTRIES IN THE CORRECT ORDER
2654
2655 007642 032737 102000 001252      BIT      #<BIT15:BIT10>,$DEV0      ;CH 0 DROPPED OR NO WRAP?
2656 007650 001011      BNE      3$      ;BR IF EITHER SET
2657
2658 007652 005737 010040      IST      INTABL      ;1ST ENTRY 0?
2659 007656 001401      BEQ      .+4      ;BR IF YES
2660 007660 104250      ERROR      250      ;DID NOT INTERR AT ASSIGNED PRIOR
2661
2662 007662 023727 010042 000001      CMP      INTABL+2,#1      ;2ND ENTRY 1?
2663 007670 001401      BEQ      .+4      ;BR IF YES
2664 007672 104250      ERROR      250
2665
2666 007674 032737 000410 001252 3$:      BIT      #<BIT8:BIT3>,$DEV0      ;CH 1 DROPPED OR NO WRAP?
2667 007702 001016      BNE      4$      ;BR IF EITHER SET
2668 007704 023737 001260 001144      CMF      BASE1,$TKS      ;CH 1 CONSOLE?
2669 007712 001412      BEQ      4$      ;BR IF YES
2670
2671 007714 023727 010044 000002      CMP      INTABL+4,#2      ;3RD ENTRY 2?
2672 007722 001401      BEQ      .+4      ;BR IF YES
2673 007724 104250      ERROR      250
2674

```

CVMXAA0 MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 62
CVMXAA.P11 30-MAY-79 15:57 123 TEST THAT CHANNELS INTR AT ASSIGNED PRIORITY.

SEQ 0060

2675	007726	023727	010046	000003	CMP	INTABL+6,#3	;4TH ENTRY 3?
2676	007734	001401			BEG	.+4	;BR IF ;YES
2677	007736	104250			ERROR	250	
2678							
2679	007740		48:				
2680	007740	000443			BR	TST24	::EXIT TEST
2681							

```

2682
2683 007742 STARS
2684 ;* START OF SERVICE ROUTINES
2685 007742 STARS
2686
2687 007742 005024 CHOR: CLR (R4)+ ;PUT IDENTIFIER IN TABLE
2688 007744 013700 001254 MOV BASE0,R0
2689 007750 042710 000100 BIC #IE,(R0) ; 1 INTERR IS ALL WE WANT
2690 007754 000002 RTI
2691
2692 007756 012724 000001 CHOX: MOV #1,(R4)+ ;PUT IDENTIFIED IN TABLE
2693 007762 013700 001254 MOV BASE0,R0
2694 007766 062700 000004 ADD #4,R0
2695 007772 042710 000100 BIC #IE,(R0) ; 1 INTERR IS ALL WE WANT
2696 007776 000002 RTI
2697
2698 010000 012724 000002 CHIR: MOV #2,(R4)+ ;PUT IDENTIFIER IN TABLE
2699 010004 013700 001260 MOV BASE1,R0
2700 010010 042710 000100 BIC #IE,(R0) ; 1 INTERR IS ALL WE WANT
2701 010014 000002 RTI
2702
2703 010016 012724 000003 CHIX: MOV #3,(R4)+ ;PUT IDENTIFIED IN TABLE
2704 010022 013700 001260 MOV BASE1,R0
2705 010026 062700 000004 ADD #4,R0
2706 010032 042710 000100 BIC #IE,(R0) ; 1 INTERR IS ALL WE WANT
2707 010036 000002 RTI
2708
2709
2710 ;THIS TABLE WILL CONTAIN ENTRIES
2711 ;REPRESENTING EACH INTR IN THE ORDER
2712 ;THAT IT OCCURED
2713 ; S/B IN THE FOLLOWING ORDER: 0,1,2,3
2714
2715 010040 000004 INTAB: .BLKW 4
2716
2717
  
```

2718
 2719
 2720
 2721
 2722
 2723
 2724
 2725
 2726
 2727
 2728
 2729
 2730
 2731
 2732
 2733
 2734
 2735
 2736
 2737
 2738
 2739
 2740
 2741
 2742
 2743
 2744
 2745
 2746
 2747
 2748
 2749
 2750
 2751
 2752
 2753
 2754
 2755
 2756
 2757
 2758
 2759
 2760
 2761
 2762
 2763
 2764
 2765
 2766
 2767
 2768

010050 000004
 010052 012737 000001 001160
 010060 012737 000024 001200
 010066 005037 011324
 010072 012737 000'00 011330
 010100 012700 0'0724
 010104 005020
 010106 020027 011324
 010112 001374
 010114
 010126 032737 102000 001252
 010134 001035
 010136 013700 001256
 010142 012720 010550
 010146 012720 000340
 010152 012720 010610
 010156 012710 000340
 010162 013700 001254
 010166 052710 000100
 010172 062700 000004
 010176 052710 000100
 010202 032737 040000 001252
 010210 001404
 010212 012737 000037 011326
 010220 000403
 010222 012737 000077 011326
 010230 032737 000410 001252
 010236 001041
 010240 023737 001260 001144
 010246 001435

 TEST 24 TEST DATA TRANSFERS WITH 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.

 TST24: SCOPE
 MOV #1,\$TIMES ;;DO 1 ITERATION
 MOV #24,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
 CLR COXMIT ;1'ST WORD OF CH 0 TO XMIT
 MOV #100,C1XMIT ;1'ST WORD OF CH 1 TO XMIT
 MOV #CHOTAB,R0 ;CLEAR OUT RECEIVER TABLES
 CLR (R0)+
 CMP R0,\$STATEND ;ALL DONE?
 BNE .-6 ;BR IF NO
 SETPRI #PR7
 BIT #<BIT15:BIT10>,\$DEV0 ;CH 0 DROPPED OR NO WRAP?
 BNE 1\$;BR IF EITHER SET
 MOV VECT0,R0 ;SETUP CH 0 VECTOR AREA
 MOV #ROSrv,(R0)+ ;RCVR
 MOV #PR7,(R0)+
 MOV #XOSrv,(R0)+ ;XMIT
 MOV #PR7,(R0)
 MOV BASE0,R0 ;ENABLE INTERRUPTS
 BIS #1E,(R0) ;RCVR
 ADD #4,R0
 BIS #1E,(R0) ;XMIT
 BIT #BIT14,\$DEV0 ;8 BITS/WORD?
 BEQ .+12 ;BR IF YES
 MOV #37,COEND
 BR .+10
 MOV #77,COEND
 1\$: BIT #<BIT8:BIT3>,\$DEV0 ;CH 1 DROPPED OR NO WRAP?
 BNE 2\$;BR IF EITHER SET
 CMP BASF1,\$TKS ;CH 1 CONSOLE?
 BEQ 2\$;BR IF YES

```

2769
2770 010250 013700 001262      MOV      VECT1,R0      ;SETUP CH 1 VECTOR AREA
2771 010254 012720 010636      MOV      #R1SRV,(R0)+  ;RCVR
2772 010260 012720 000340      MOV      #PR7,(R0)+
2773 010264 012720 010676      MOV      #X1SRV,(R0)+  ;XMIT
2774 010270 012710 000340      MOV      #PR7,(R0)
2775
2776 010274 013700 001260      MOV      BASE1,R0      ;ENABLE INTERRUPTS
2777 010300 052710 000100      BIS      #IE,(R0)      ;RCVR
2778 010304 062700 000004      ADD      #4,R0
2779 010310 052710 000100      BIS      #IE,(R0)      ;XMIT
2780
2781 010314 032737 000200 001252  BIT      #BIT7,$DEV0      ;8 BITS/WORD?
2782 010322 001404      BEQ      .+12      ;BR IF YES
2783 010324 012737 000137 011332  MOV      #137,C1END
2784 010332 000403      BR      .+10
2785 010334 012737 000177 011332  MOV      #177,C1END
2786
2787 010342 012700 010724      2$: MOV      #CH0TAB,R0      ;INIT BUFFER PTRS FOR SERV ROUTINES
2788 010346 012701 011124      MOV      #CH1TAB,R1
2789
2790 010352      SETPRI      #PRO      ;LET'EM LOOSE!
2791
2792 010364 012702 177777      MOV      #-1,R2      ;DELAY TO ALLOW ANY PREVIOUS XMIT TO
2793 010370 077201      SOB      R2,.      ;FINISH BEFORE RESET
2794 010372 000005      RESET      ;DISABLE ALL INTERR
2795
2796 010374      SETPRI      #PR7
2797
2798      ;NOW LETS CHECK TO SEE IF TABLE(S) ARE IN CORRECT ORDER
2799
2800 010406 032737 102000 001252  BIT      #<BIT15:BIT10>,$DEV0  ;CH 0 DROPPED OR NO WRAP
2801 010414 001021      BNE      5$      ;BR IF EITHER SET
2802
2803 010416 005000      CLR      R0      ;EXPECTED WORD
2804 010420 012701 010724      MOV      #CH0TAB,R1      ;TABLE PTR
2805
2806 010424 020011      3$: CMP      R0,(R1)      ;EXPECT = ACTUAL?
2807 010426 001405      BEQ      4$      ;BR IF YES
2808
2809 010430 005711      TST      (R1)      ;ERROR SET?
2810 010432 100002      BPL      .+6      ;BR IF NO
2811 010434 104270      ERROR      270      ;ERROR FLAG AFTER XFER
2812 010436 000401      BR      .+4
2813 010440 104271      ERROR      271      ;DATA COMPARE ERROR

```

```

2814
2815 010442 020037 011326      4$:  CMP      RO,COEND      ;ALL DONE?
2816 010446 001404              BEQ      5$              ;BR IF YES
2817 010450 005200              INC      RO
2818 010452 062701 000002      ADD      #2,R1
2819 010456 000762              BR       3$              ;ELSE DO AGAIN
2820
2821 010460 032737 000410 001252 5$:  BIT      #<BIT8:BIT3>,$DEV      ;CH 1 DROPPED OR NO WRAP
2822 010466 001026              BNE      8$              ;BR IF EITHER SET
2823 010470 023737 001260 001144      CMP      BASE1,$TKS      ;CH 1 CONSOLE?
2824 010476 001422              BEQ      8$              ;BR IF YES
2825
2826 010500 012700 000100      MOV      #100,RO      ;EXPECTED WORD
2827 010504 012701 011124      MOV      #CM1TAB,R1    ;TABLE PTR
2828
2829 010510 020011              6$:  CMP      RO,(R1)      ;EXPECT = ACTUAL?
2830 010512 001405              BEQ      7$              ;BR IF YES
2831
2832 010514 005711              TST      (R1)          ;ERROR SET?
2833 010516 10C002              BPL      ,+6           ;BR IF NO
2834 010520 104270              ERROR    270        ;ERROR FLAG UP AFTER XFER
2835 010522 000401              BR       ,+4
2836 010524 104271              ERROR    271        ;DATA COMPARE ERROR
2837
2838 010526 020037 011332      7$:  CMP      RO,C1END      ;ALL DONE?
2839 010532 001404              BEQ      8$              ;BR IF YES
2840 010534 005200              INC      RO
2841 010536 062701 000002      ADD      #2,R1
2842 010542 000762              BR       6$              ;ELSE DO AGAIN
2843
2844 010544 000137 006750      8$:  JMP      NOTST
2845

```

```

2846
2847
2848
2849
2850 010550 013702 001254 ROSRV: MOV BASE0,R2
2851 010554 062702 000002 ADD #2,R2 ;POINT TO RBUF
2852 010560 011220 MOV (R2),(R0)+ ;PUT IN TABL
2853 010562 023737 011324 011326 CMP COXMIT,COEND ;LAST CHAR?
2854 010570 061406 BEQ 1$ ;BR IF YES
2855 010572 005237 011324 INC COXMIT ;ELSE BUMP CHAR TO XMIT NEXT
2856 010576 062702 000002 ADD #2,R2 ;POINT TO XCSR
2857 010602 052712 000100 BIS #1E,(R2) ;RE-ENABLE
2858 010606 000002 1$: RTI
2859
2860 010610 013702 001254 XOSRV: MOV BASE0,R2
2861 010614 062702 000006 ADD #6,R2 ;POINT TO XBUF
2862 010620 013712 011324 MOV COXMIT,(R2) ;SHIP WORD
2863 010624 162702 000002 SUB #2,R2 ;POINT TO XCSR
2864 010630 042712 000100 BIC #1E,(R2) ;DISABLE XMIT INTERR
2865 ;RE-ENABLE IN REC HANDLER
2866 010634 000002 RTI
2867
2868
2869 010636 013702 001260 R1SRV: MOV BASE1,R2
2870 010642 062702 000002 ADD #2,R2 ;POINT TO RBUF
2871 010646 011221 MOV (R2),(R1)+ ;PUT IN TABLE
2872 010650 023737 011330 011332 CMP C1XMIT,C1END ;LAST CHAR?
2873 010656 001406 BEQ 1$ ;BR IF YES
2874 010660 005237 011330 INC C1XMIT ;ELSE BUMP CHAR TO XMIT NEXT
2875 010664 062702 000002 ADD #2,R2 ;POINT TO XCSR
2876 010670 052712 000100 BIS #1E,(R2) ;RE-ENABLE
2877 010674 000002 1$: RTI
2878
2879 010676 013702 001260 X1SRV: MOV BASE1,R2
2880 010702 062702 000006 ADD #6,R2 ;POINT TO XBUF
2881 010706 013712 011330 MOV C1XMIT,(R2) ;SHIP WORD
2882 010712 162702 000002 SUB #2,R2 ;POINT TO XCSR
2883 010716 042712 000100 BIC #1E,(R2) ;DISABLE XMIT INTERR
2884 ;RE-ENABLE IN REC HANDLER
2885 010722 000002 RTI
2886
2887
2888
2889
2890
2891 010724 000100
2892 011124 000100
2893 011324
2894
2895 011324 000000
2896 011326 000000
2897 011330 000000
2898 011332 000000
2899

; ** RECEIVER STATUS TABLES **
CHOTAB: .BLKW 100 ; 7 BIT WDS
LH1TAB: .BLKW 100 ; 8 BIT WDS
STATEND: ; 0- 37 ; 0- 77
; 40- 77 ; 100-177

COXMIT: 0 ; START WORD FOR CH 0: 0
COEND: 0 ; END WORD FOR CH 0: 37 OR 77
C1XMIT: 0 ; START WORD FOR CH 1: 100
C1END: 0 ; END WORD FOR CH 1: 137 OR 177
  
```

```

2900
2901          .SBTTL ROUTINE TO SIZE THE DEVICE MAP ($DEVN) & MEMORY
2902
2903 011334 032737 100000 001252 SIZE: BIT #BIT15,$DEVN ;TEST CH 0?
2904 011342 001413 BEQ 1$ ;BR IF YES
2905 011344 104401 013042 TYPE ,CODROP
2906 011350 032737 000400 001252 BIT #BIT8,$DEVN ;TEST CH 1?
2907 011356 001414 BEQ 2$ ;BR IF YES
2908 011360 104401 013072 TYPE ,C1DROP
2909 011364 005237 011614 INC NOCHAN ;SET FLAG
2910 011370 000417 BR 4$
2911
2912 011372 032737 000400 001252 1$: BIT #BIT8,$DEVN ;TEST CH 1?
2913 011400 001403 BEQ 2$ ;BR IF YES
2914 011402 104401 013072 TYPE ,C1DROP
2915 011406 000406 BR 3$
2916
2917 011410 023737 001260 001144 2$: CMP BASE1,$TKS ;IS IT CONSOLE?
2918 011416 001002 BNE 3$ ;BR IF NO
2919 011420 104401 013122 TYPE ,C1CON
2920 011424 005037 011614 3$: CLR NOCHAN
2921
2922 011430 032737 000004 001252 4$: BIT #BIT2,$DEVN ;DO RAM TEST?
2923 011436 001402 BEQ 10$ ;BR IF YES
2924 011440 104401 013170 TYPE ,NORAM
2925
2926 011444 032737 000002 001252 10$: BIT #BIT1,$DEVN ;DO ROM TEST?
2927 011452 001402 BEQ 5$ ;BR IF YES
2928 011454 104401 013145 TYPE ,NOROM
2929
2930 011460 032737 000001 001252 5$: BIT #BIT0,$DEVN ;CLOCK DISABLED?
2931 011466 001402 BEQ 6$ ;BR IF YES
2932 011470 104401 013214 TYPE ,CLKEN
2933
2934 011474 012737 011636 000004 6$: MOV #INTSRV,ERRVEC ;SETUP TIMEOUT TRAP ADDR
2935 011502 012737 000200 000006 MOV #200,ERRVEC+2 ;LOCKOUT INTERR
2936 011510 005037 011644 CLR INTFLG
2937
2938 011514 012700 020000 MOV #20000,R0 ;SIZE MEM. START WITH BOT OF 2ND 4K
2939 011520 005001 CLR R1 ;MEM MAP. 0-4K, 1-8K, ETC
2940
2941 011522 011010 7$: MOV (R0),(R0) ;DO MOV INSTEAD OF 'TST'
2942 011524 005737 011644 TST INTFLG ;GOT TIMEOUT INTERR?
2943 011530 001006 BNE 8$ ;BR IF YES
2944 011532 005201 INC R1
2945 011534 062700 020000 ADD #20000,R0 ;GO TO BOT OF NEXT 4K
2946 011540 020027 160000 CMP R0,#160000 ;BOT OF 32K?
2947 011544 001366 BNE 7$ ;BR IF NO
  
```

(VMXAA0 MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 69
 (VMXAA.P11 30-MAY-79 15:57 ROUTINE TO SIZE THE DEVICE MAP (\$JEVM) & MEMORY

SEQ 0067

```

2948
2949 011546 010037 011634 8$: MOV R0,MAXMEM ;SAVE
2950 011552 162737 000002 011634 SUB #2,MAXMEM ;GO BACK TO TOP OF LAST VALID BLOCK
2951 011560 006301 ASL R1 ;MULT BY 2
2952 011562 016137 011636 011572 MOV MEMTBL(R1),9$
2953 011570 104401 TYPE
2954 011572 000000 9$: .WORD 0 ;MEM SIZE
2955 011574 104401 013314 TYPE ,MEM
2956
2957 011600 012737 000006 000004 .: MOV #ERRVEC+2,ERRVEC ;RESET TMO VECTOR
2958 011606 005037 000006 CLR ERRVEC+2
2959
2960 011612 000207 RTS PC
2961
2962 011614 000000 NOCHAN: 0 ;0 - CHANNEL(S) TO TEST
2963 ;1 = NO CHANNELS TO TEST
2964
2965 ;MSG ADDRESSES
2966 011616 013244 MEMTBL: M4K : 0 - 17776
2967 011620 013251 M8K : 20000 - 37776
2968 011622 013256 M12K : 40000 - 57776
2969 011624 013264 M16K : 60000 - 77776
2970 011626 013272 M20K : 100000 - 117776
2971 011630 013300 M24K : 120000 - 137776
2972 011632 013306 M28K : 140000 - 157776
2973
2974 011634 000000 MAXMEM: 0 ;MAX MEMORY
2975
2976
2977
2978 .SBTTL INTSRV INTERRUPT SERVICE ROUTINE
2979
2980 ;* THIS GLOBAL ROUTINE DOES NOTHING BUT INCREMENT
2981 ;* 'INTFLG' EACH TIME IT IS CALLED. IT ASSUMES
2982 ;* THAT THE MAIN CALLING ROUTINE WILL KNOW WHAT
2983 ;* TO LOOK FOR.
2984
2985 011636 005237 011644 INTSRV: INC INTFLG ;ADD 1 TO 'INTERRUPT OCCURED' FLAG
2986 011642 000002 RTI ;THAT'S ALL
2987
2988 011644 000000 INTFLG: 0
2989

```

```

2990
2991      .SBTTL ROUTINE TO SET UP FOR NEXT CHANNEL ADDRESS
2992
2993      ;*****
2994      ;      THIS ROUTINE CAUSES ADRS TO POINT TO THE
2995      ;      ADDRESS OF CHANNEL UNDER TEST, ADRS +2 TO
2996      ;      POINT TO THE VECTOR OF THE CHANNEL UNDER TEST.
2997      ;*****
2998
2999 011646 005737 011614      CYCLE: TST      NOCHAN      ;ANY CHANNELS TO BE TESTED?
3000 011652 001401      BEG      1$      ;BR IF YES
3001 011654 000207      RTS      PC      ;ELSE RETURN
3002
3003 011656 023727 001206 000002 1$: CMP      $UNIT,#2      ;DONE ALL AVAIL CHANNELS?
3004 011664 001024      BNE      3$      ;BR IF NO
3005 011666 005037 001206      CLR      $UNIT
3006 011672 032737 002000 001252      BIT      #BIT10,$DEV0      ;CHAN 0 DATA WRAP?
3007 011700 001410      BEQ      2$      ;BR IF YES
3008 011702 032737 000010 001252      BIT      #BIT3,$DEV0      ;CHAN 1 DATA WRAP?
3009 011710 001404      BEQ      2$      ;BR IF YES
3010 011712 104401 001310      TYPE      ,CARR
3011 011716 000137 013334      JMP      $EOP      ;ELSE ALL DONE
3012
3013 011722 012737 000001 012312 2$: MOV      #TRUE,PHASE2
3014 011730 005037 012314      CLR      PICNT
3015 011734 000207      RTS      PC
3016
3017 011736 005737 001206      3$: TST      $UNIT      ;CHAN 0 TO BE TESTED?
3018 011742 001066      BNE      12$      ;BR IF NO
3019
3020 011744 032737 100000 001252      BIT      #BIT15,$DEV0      ;CHAN 0 TO BE TESTED?
3021 011752 001403      BEQ      4$      ;BR IF YES
3022 011754 005237 001206      INC      $UNIT
3023 011760 000732      BR      CYCLE      ;TRY OVER AGAIN FOR CHAN 1
3024
3025 011762 013737 001254 001270 4$: MOV      BASE0,DLADD      ;SET CHAN 0 RCSR ADDR
3026 011770 013737 001256 001302      MOV      VECTO,DLVEC      ; & VECTOR
3027
3028 011776 032737 040000 001252      BIT      #BIT14,$DEV0      ;8 BITS/WORD?
3029 012004 001003      BNE      5$      ;BR IF NO
3030 012006 005037 012300      CLR      WORDL      ;ELSE CLEAR TO DEFAULT & NO PARITY
3031 012012 000402      BR      .+6
3032 012014 005237 012300      5$: INC      WORDL      ;7 BITS/WORD WITH PARITY
3033
3034 012020 032737 010000 001252      BIT      #BIT12,$DEV0      ;ODD PARITY?
3035 012026 001003      BNE      7$      ;BR IF NO
3036 012030 005037 012302      CLR      ODD      ;ELSE CLEAR TO DEFAULT
3037 012034 000402      BR      .+6
3038 012036 005237 012302      7$: INC      ODD
3039
3040 012042 032737 004000 001252      BIT      #BIT11,$DEV0      ;BREAK DETECTION ENABLED?
3041 012050 001003      BNE      8$      ;BR IF YES
3042 012052 005037 012304      CLR      BRK      ;ELSE CLEAR TO DEFAULT
3043 012056 000402      BR      .+6
3044 012060 005237 012304      8$: INC      BRK

```

```

3045
3046 012064 032737 002000 001252      BIT      #BIT10,$DEV0      ;DATA WRAP AROUND?
3047 012072 001003                      BNE      9$              ;BR IF NO
3048 012074 005037 012306                      CLR      WRAP              ;ELSE CLEAR TO DEFAULT
3049 012100 000402                      BR       .+6
3050 012102 005237 012306      9$:      INC      WRAP
3051
3052 012106 005037 012310      10$:     CLR      CONSOLE      ;CHAN 0 IS NO CONSOLE
3053 012112 005237 001206      11$:     INC      $UNIT        ;SETUP FOR NEXT TIME
3054 012116 000207                      RTS      PC
3055
3056 012120 032737 000400 001252 12$:     BIT      #BIT18,$DEV0      ;CHAN 1 TO BE TESTED?
3057 012126 001403                      BEQ      13$              ;BR IF YES
3058 012130 005237 001206                      INC      $UNIT
3059 012134 000644                      BR       CYCLE            ;TRY OVER AGAIN FOR CHAN 1
3060
3061 012136 013737 001260 001270 13$:     MOV      BASE1,DLADD      ;SET CHAN 1 RCSR ADDR
3062 012144 013737 001262 001302                      MOV      VECT1,DLVEC      ; & VECTOR
3063
3064 012152 032737 000200 001252      BIT      #BIT17,$DEV0      ;8 BITS/WORD?
3065 012160 001003                      BNE      14$              ;BR IF NO
3066 012162 005037 012300                      CLR      WORDL            ;ELSE CLEAR TO DEFAULT & NO PARITY
3067 012166 000402                      BR       .+6
3068 012170 005237 012300      14$:     INC      WORDL          ;7 BITS/WORD WITH PARITY
3069
3070 012174 032737 000040 001252      BIT      #BIT15,$DEV0      ;ODD PARITY?
3071 012202 001003                      BNE      16$              ;BR IF NO
3072 012204 005037 012302                      CLR      ODD              ;ELSE CLEAR TO DEFAULT
3073 012210 000402                      BR       .+6
3074 012212 005237 012302      16$:     INC      ODD
3075
3076 012216 032737 000020 001252      BIT      #BIT14,$DEV0      ;BREAK DETECTION ENABLED?
3077 012224 001003                      BNE      17$              ;BR IF YES
3078 012226 005037 012304                      CLR      BRK              ;ELSE CLEAR TO DEFAULT
3079 012232 000402                      BR       .+6
3080 012234 005237 012304      17$:     INC      BRK
3081
3082 012240 032737 000010 001252      BIT      #BIT13,$DEV0      ;DATA WRAP AROUND?
3083 012246 001003                      BNE      18$              ;BR IF NO
3084 012250 005037 012306                      CLR      WRAP              ;ELSE CLEAR TO DEFAULT
3085 012254 000402                      BR       .+6
3086 012256 005237 012306      18$:     INC      WRAP
3087
3088 012262 023737 001260 001144      CMP      BASE1,$TKS      ;CHAN 1 CONSOLE?
3089 012270 001306                      BNE      10$              ;BR IF NO
3090 012272 005237 012310                      INC      CONSOLE
3091 012276 000705                      BR       11$
3092
3093 012300 000000      WORDL: 0      ;WORD LENGTH      0 = 8 BITS (NO PARITY), 1 = 7 BITS (WITH PARITY)
3094 012302 000000      ODD: 0      ;ODD PARITY      0 = ODD, 1 = EVEN
3095 012304 000000      BRK: 0      ;BREAK DET      0 = DISABL, 1 = ENABL
3096 012306 000000      WRAP: 0      ;DATA WRAP      0 = YES, 1 = NO
3097 012310 000000      CONSOLE: 0      ;CONSOLE      0 = CH UNDER TEST NOT CONSOLE
3098 . . . . .      ; . . . . .      1 = CH UNDER TEST IS CONSOLE
3099 012312 000000      PHASE2: 0
3100 012314 000000      P1CNT: 0      ;CLR TO PRINT 'PHASE 1' ONLY ONCE

```

```

3101
3102
3103
3104
3105
3106
3107 012316 104401 013233
3108 012322 013746 001252
3109 012326 104402
3110 012330 104401 014752
3111
3112 012334 005046
3113 012336 005046
3114
3115 012340 105777 166600
3116 012344 100375
3117 012346 117746 166574
3118 012352 042716 177600
3119
3120 012356 021627 000015
3121 012362 001017
3122 012364 005766 000004
3123 012370 001403
3124 012372 016637 000002 001252
3125 012400 104401 001171
3126 012404 004737 011334
3127 012410 062706 000006
3128 012414 012716 002004
3129 012420 000002
3130
3131 012422 004737 014052
3132 012426 021627 000060
3133 012432 002420
3134 012434 021627 000067
3135 012440 003015
3136 012442 042726 000060
3137 012446 005766 000002
3138 012452 001403
3139 012454 006316
3140 012456 006316
3141 012460 006316
3142
3143 012462 005266 000002
3144 012466 056616 177776
3145 012472 000722
3146
3147 012474 104401 001170
3148 012500 104401 001171
3149 012504 005726
3150 012506 005016
3151 012510 000713

*****
ROUTINE TO DISPLAY CURRENT DEVM & ALLOW OPERATOR TO INPUT NEW VALUE.
*****

GT$DEV: TYPE ,DEVM ;SHOW CURRENT
MOV $DEV,-(SP) ;SAVE $DEV FOR TYPEOUT
TYPE ,SMNEW ;GO TYPE--OCTAL ASCII(ALL DIGITS)
;GET NEW

1$: CLR -(SP) ;CLR CTR
CLR -(SP) ;CLR NEW DEVM

2$: TSTB @STKS ;CHAR THERE?
BPL 2$ ;BR IF NO
MOVB @STKB,-(SP) ;ELSE GET CHAR
BIC #^C177,(SP) ;MAKE IT 7 BIT ASCII

3$: CMP (SP),#15 ;<CR>?
BNE 7$ ;BR IF NO
TST 4(SP) ;ELSE IS IT 1'ST CHAR?
BEQ 4$ ;BR IF YES
MOV 2(SP),$DEV ;ELSE SAVE NEW DEVM

4$: TYPE ,$CRLF
6$: JSR PC,SIZE ;RE-SIZE
ADD #6,SP ;RESET STACK
MOV #RAMROM,(SP)
RTI

7$: JSR PC,$TYPEC ;ECHO CHAR
CMP (SP),#60 ;CHAR < 0?
BLT 9$ ;BR IF YES
CMP (SP),#67 ;CHAR > 7?
ZGT 9$ ;BR IF YES
BIC #60,(SP)+ ;STRIP OFF ASCII
TST 2(SP) ;1'ST CHAR?
BEQ 8$ ;BR IF YES
ASL (SP) ;ELSE SHIFT PRESENT CHAR
ASL (SP) ;TO MAKE ROOM
ASL (SP) ;FOR NEW ONE

8$: INC 2(SP) ;KEEP CHAR CT
BIS -2(SP),(SP) ;SET IN NEW CHAR
BR 2$ ;GET NEW ONE

9$: TYPE ,SQUES
TYPE ,$CRLF
TST (SP)+ ;RESET PTR
CLR (SP) ;CLR DEVM
BR 2$ ;TRY AGAIN

```

CVMXAA0 MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 73
 CVMXAA.P11 30-MAY-79 15:57 ROUTINE TO SET UP FOR NEXT CHANNEL ADDRESS

SEQ 0071

```

3152
3153
3154
3155
3156
3157
3158
3159 012512 013537 012562
3160 012516 012537 012564
3161
3162 012522 012701 000010
3163 012526 012700 177777
3164 012532 033777 012564 000022
3165 012540 001005
3166 012542 005300
3167 012544 001372
3168 012546 005301
3169 012550 001366
3170 012552 000402
3171
3172 012554 062705 000004
3173 012560 000205
3174
3175 012562 000000
3176 012564 000000
3177
3178
3179 012566 012537 012562
3180
3181 012572 012701 000010
3182 012576 012700 177777
3183 012602 005777 177754
3184 012606 001005
3185 012610 005300
3186 012612 001373
3187 012614 005301
3188 012616 001367
3189 012620 000402
3190
3191 012622 062705 000004
3192 012626 000205
3193
3194
3195 012630 012700 177777
3196 012634 005300
3197 012636 001376
3198 012640 000207
3199

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

TIMER:  MOV      @ (R5)+,REGHLD
        MOV      (R5)+,BITHLD

        MOV      #10,R1          ;DOUBLL LOOP
1$:      MOV      #-1,R0
2$:      BIT      BITHLD,@REGHLD  ;BIT THERE?
        BNE      3$              ;BR IF YES
        DEC      R0
        BNE      2$
        DEC      R1
        BNE      1$
        BR       4$              ;ALL TIMED OUT

3$:      ADD      #4,R5          ;JUMP OVER ERR ON RETURN
4$:      RTS      R5

REGHLD: 0
BITHLD: 0
;REGISTER TO BE TESTED
;BIT TO BE TESTED IN REGISTER

TIMER1:  MOV      (R5)+,REGHLD

        MOV      #10,R1
1$:      MOV      #-1,R0
2$:      TST      @REGHLD        ;ZERO?
        BNE      3$              ;BR IF NO
        DEC      R0
        BNE      2$
        DEC      R1
        BNE      1$
        BR       4$              ;ALL TIMED OUT

3$:      ADD      #4,R5          ;JUMP OVER ERR ON RETURN
4$:      RTS      R5

WAIT:    MOV      #-1,R0
        DEC      R0
        BNE      .-2
        RTS      PC

```

```

3200
3201 012642 MYTYPE: TYPTXT <<CRLF>*TEST #*>
3202 012662 TYPOCT $TESTN
3203 012670 TYPTXT <*,ERROR #*>
3204 012710 1:3737 001114 001176 MOVB $ITEMB,$FATAL ; APT FATAL ERROR NUMBER
3205 012716 TYPOCS $FATAL
3206 012726 TYPTXT <*,PC =*>
3207 012744 TYPOCT $ERRPC
3208
3209 012752 023727 001200 000005 CMP $TESTN,#5 ;DOING SLU TESTS?
3210 012760 100425 BMI 1$ ;BR IF NO
3211
3212 012762 TYPTXT <*,CSR:*> ;ELSE PROCEED
3213 013000 TYPOCT DLADD
3214 013006 TYPTXT <*,VECTOR:*>
3215 013026 TYPOCT DLVEC
3216 013034 104401 001171 1$: TYPE ,SCRLF
3217 013040 000207 RTS PC
3218
3219
3220 013042 041600 040510 020116 CODROP: .ASCIZ <CRLF>/CHAN 0 TESTING DROPPED/
3221 013050 020060 042524 052123
3222 013056 047111 020107 051104
3223 013064 050117 042520 000104
3224 013072 041600 040510 020116 C1DROP: .ASCIZ <CRLF>/CHAN 1 TESTING DROPPED/
3225 013100 020061 042524 052123
3226 013106 047111 020107 051104
3227 013114 050117 042520 000104
3228 013122 041600 040510 020116 C1CON: .ASCIZ <CRLF>/CHAN 1 IS CONSOLE/
3229 013130 020061 051511 041440
3230 013136 047117 047523 042514
3231 013144 000
3232 013145 200 047522 020115 NOROM: .ASCIZ <CRLF>/ROM TEST BYPASSED/
3233 013152 042524 052123 041040
3234 013160 050131 051501 042523
3235 013166 000104
3236 013170 005015 040522 020115 NORAM: .ASCIZ <CR><LF>/RAM TEST BYPASSED/
3237 013176 042524 052123 041040
3238 013204 050131 051501 042523
3239 013212 000104
3240 013214 041600 047514 045503 CLKEN: .ASCIZ <CRLF>/CLOCK ENABLED/
3241 013222 042440 040516 046102
3242 013230 042105 000
3243 013233 200 042504 046526 DEVM: .ASCIZ <CRLF>/DEVM - /
3244 013240 036440 000040
3245 013244 005015 045464 000 M4K: .ASCIZ <CR><LF>/4K/
3246 013251 015 034012 000113 M8K: .ASCIZ <CR><LF>/8K/
3247 013256 005015 031061 000113 M12K: .ASCIZ <CR><LF>/12K/
3248 013264 005015 033061 000113 M16K: .ASCIZ <CR><LF>/16K/
3249 013272 005015 030062 000113 M20K: .ASCIZ <CR><LF>/20K/
3250 013300 005015 032062 000113 M24K: .ASCIZ <CR><LF>/24K/
3251 013306 005015 034062 000113 M28K: .ASCIZ <CR><LF>/28K/
3252 013314 046440 046505 050040 MEM: .ASCIZ / MEM PRESENT/<CR><LF>
3253 013322 042522 042523 052116
3254 013330 005015 000
3255 013334 .EVEN
  
```

```

3256      .SB*TL  END OF PASS ROUTINE
3257
3258      ;*****
3259      ;*INCREMENT THE PASS NUMBER ($PASS)
3260      ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
3261      ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
3262      ;*IF THERES A MONITOR GO TO IT
3263      ;*IF THERE ISN'T JUMP TO RAMROM
3264
3265      $EOP:
3266      013334 000004      SCOPE
3267      013336 005037 001102  CLR      $TSTNM      ;;ZERO THE TEST NUMBER
3268      013342 005037 001160  CLR      $TIMES      ;;ZERO THE NUMBER OF ITERATIONS
3269      013346 005237 001202  INC      $PASS      ;;INCREMENT THE PASS NUMBER
3270      013352 042737 100000  BIC      #100000,$PASS  ;;DON'T ALLOW A NEG. NUMBER
3271      013360 005327      DEC      (PC)+      ;;LOOP?
3272      013362 000001      $EOPCT: .WORD 1
3273      013364 003022      BGT      $DOAGN      ;;YES
3274      013366 012737      MOV      (PC)+,@(PC)+  ;;RESTORE COUNTER
3275      013370 000001      $ENDCT: .WORD 1
3276      013372 013362      $EOPCT
3277      013374 104401 013441  TYPE      ,SENDMG      ;;TYPE 'END PASS #'
3278      013400 013746 001202  MOV      $PASS,-(SP)  ;;SAVE $PASS FOR TYPEOUT
3279      013404 104405      TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
3280      013406 104401 013436  TYPE      ,SENUL      ;;TYPE A NULL CHARACTER
3281      013412 013700 000042  $GET42: MOV      @#42,R0      ;;GET MONITOR ADDRESS
3282      013416 001405      BEQ      $DOAGN      ;;BRANCH IF NO MONITOR
3283      013420 000005      RESET      ;;CLEAR THE WORLD
3284      013422 004710      $ENDAD: JSR      PC,(R0)      ;;GO TO MONITOR
3285      013424 000240      NOP      ;;SAVE ROOM
3286      013426 000240      NOP      ;;FOR
3287      013430 000240      NOP      ;;ACT11
3288      013432      $DOAGN:
3289      013432 000137      JMP      @(PC)+      ;;RETURN
3290      013434 002004      $RTNAD: .WORD      RAMROM
3291      013436 377 377 000  $ENULL: .BYTE      -1,-1,0      ;;NULL CHARACTER STRING
3292      013441 015 042412 042116  $ENDMG: .ASCIIZ  <15><12>/END PASS #/
3293      013446 050040 051501 020123
3294      013454 000043
  
```

```

3295 .SBTTL POWER DOWN AND UP ROUTINES
3296
3297 ::*****
3298 :POWER DOWN ROUTINE
3299 013456 012737 013622 000024 $PWRDN: MOV $SILLUP,@#PWRVEC ;;SET FOR FAST UP
3300 013464 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
3301 013472 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
3302 013474 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
3303 013476 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
3304 013500 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
3305 013502 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
3306 013504 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
3307 013506 017746 165426 MOV @SWR,-(SP) ;;PUSH @SWR ON STACK
3308 013512 010637 013626 MOV SP,$SAVR6 ;;SAVE SP
3309 013516 012737 013530 000024 MOV $PWRUP,@#PWRVEC ;;SET UP VECTOR
3310 013524 000000 HALT
3311 013526 000776 BR -2 ;;HANG UP
3312
3313 ::*****
3314 :POWER UP ROUTINE
3315 013530 012737 013622 000024 $PWRUP: MOV $SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
3316 013536 013706 013626 MOV $SAVR6,SP ;;GET SP
3317 013542 005037 013626 CLR $SAVR6 ;;WAIT LOOP FOR THE ITY
3318 013546 005237 013626 1$: INC $SAVR6 ;;WAIT FOR THE INC
3319 013552 001375 BNE 1$ ;;OF WORD
3320 013554 012677 165360 MOV (SP)+,@SWR ;;POP STACK INTO @SWR
3321 013560 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
3322 013562 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
3323 013564 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
3324 013566 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
3325 013570 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
3326 013572 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
3327 013574 012737 013456 000024 MOV $PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
3328 013602 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;PRIO:7
3329 013610 104401 TYPE ;;REPORT THE POWER FAILURE
3330 013612 013630 $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
3331 013614 012716 MOV (PC)+,(SP) ;;RESTART AT START
3332 013616 001370 $PWRAD: .WORD START ;;RESTART ADDRESS
3333 013620 000002 RTI
3334 013622 000000 $SILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
3335 013624 000776 BR -2 ;; BEFORE THE POWER DOWN WAS COMPLETE
3336 013626 000000 $SAVR6: 0 ;;PUT THE SP HERE
3337 013630 005015 047520 042527 $POWER: .ASCII <15><12>'POWER'
3338 013636 000122
3339 .EVEN

```

```

3340 .SBTTL TYPE ROUTINE
3341
3342 ;*****
3343 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3344 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3345 ;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3346 ;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3347 ;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
3348 ;*
3349 ;*CALL:
3350 ;*1) USING A TRAP INSTRUCTION
3351 ;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3352 ;*OR
3353 ;* TYPE
3354 ;* MESADR
3355 ;*
3356
3357 013640 105737 001157 $TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
3358 013644 100002 BPL 1$ ;;BR IF YES
3359 013646 000000 HALT ;;HALT HERE IF NO TERMINAL
3360 013650 000430 BR 3$ ;;LEAVE
3361 013652 010046 1$: MOV R0,-(SP) ;;SAVE R0
3362 013654 017600 000002 MOV @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
3363 013660 122737 000001 001214 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
3364 013666 001011 BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
3365 013670 132737 000100 001215 BITB #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT
3366 013676 001405 BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
3367 013700 010037 013710 MOV R0,61$ ;;SETUP MESSAGE ADDRESS FOR APT
3368 013704 004737 014772 JSR PC,$ATY3 ;;SPOOL MESSAGE TO APT
3369 013710 000000 61$: .WORD 0 ;;MESSAGE ADDRESS
3370 013712 132737 000040 001215 62$: BITB #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
3371 013720 001003 BNE 60$ ;;YES,SKIP TYPE OUT
3372 013722 112046 2$: MOVB (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
3373 013724 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
3374 013726 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
3375 013730 012600 60$: MOV (SP)+,R0 ;;RESTORE R0
3376 013732 062716 000002 3$: ADD #2,(SP) ;;ADJUST RETURN PC
3377 013736 000002 RTI ;;RETURN
3378 013740 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
3379 013744 001430 BEQ 8$
3380 013746 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
3381 013752 001006 BNE 5$
3382 013754 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
3383 013756 104401 TYPE ;;TYPE A CR AND LF
3384 013760 001171 $CRLF
3385 013762 105037 014116 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
3386 013766 000755 BR 2$ ;;GET NEXT CHARACTER
3387 013770 004737 014052 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
3388 013774 123726 001156 6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
3389 014000 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
3390 014002 013746 001154 MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
3391 ;;AND THE NULL CHAR.
3392 014006 105366 000001 7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
3393 014012 002770 BLT 6$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
3394 014014 004737 014052 JSR PC,$TYPEC ;;GO TYPE A NULL
3395 014020 105337 014116 DECB $CHARCNT ;;DO NOT COUNT AS A COUNT

```

```

3396 014024 00 770          BR      7$          ;;LOOP
3397
3398          ;HORIZONTAL TAB PROCESSOR
3399
3400 014026 112716 000040      8$:   MOVB   #'',(SP)      ;;REPLACE TAB WITH SPACE
3401 014032 004737 014052      9$:   JSR    PC,$TYPEC      ;;TYPE A SPACE
3402 014036 132737 000007 014116  BITB   #7,$CHARCNT      ;;BRANCH IF NOT AT
3403 014044 001372          BNE     9$          ;;TAB STOP
3404 014046 005726          TST     (SP)+          ;;POP SPACE OFF STACK
3405 014050 000724          BR      2$          ;;GET NEXT CHARACTER
3406 014052 105777 165072      $TYPEC: TSTB   @ $TPS      ;;WAIT UNTIL PRINTER IS READY
3407 014056 100375          BPL     $TYPEC
3408 014060 116677 000002 165064  MOVB   2(SP),@ $TPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3409 014066 122766 000015 000002  CMPB   #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
3410 014074 001003          BNE     1$          ;;BRANCH IF NO
3411 014076 105037 014116      CLRB   $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
3412 014102 000406          BR      $TYPEX      ;;EXIT
3413 014104 122766 000012 000002 1$:   CMPB   #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
3414 014112 001402          BEQ     $TYPEX      ;;BRANCH IF YES
3415 014114 105227          INCB   (PC)+          ;;COUNT THE CHARACTER
3416 014116 000000      $CHARCNT: .WORD 0          ;;CHARACTER COUNT STORAGE
3417 014120 000207      $TYPEX: RTS    PC
3418

```

```

3419 .SBTTL TTY INPUT ROUTINE
3420
3421 ::*****
3422 .ENABL LSB
3423
3424 ::*****
3425 *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3426 *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
3427 *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
3428 *WHEN OPERATING IN TTY FLAG MODE.
3429 014122 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;;IS THE SOFT-SWR SELECTED?
3430 014130 001114 BNE 15$ ;;BRANCH IF NO
3431 014132 105777 165006 TSTB @STKS ;;CHAR THERE?
3432 014136 100111 BPL 15$ ;;IF NO, DON'T WAIT AROUND
3433 014140 117746 165002 MOVB @STKB,-(SP) ;;SAVE THE CHAR
3434 014144 042716 177600 BIC #^C177,(SP) ;;STRIP-OFF THE ASCII
3435 014150 022726 000007 CMP #7,(SP)+ ;;IS IT A CONTROL G?
3436 014154 001102 BNE 15$ ;;NO, RETURN TO USER
3437 014156 123727 001134 000001 CMPB $AUTOB,#1 ;;ARE WE RUNNING IN AUTO-MODE?
3438 014164 001476 BEQ 15$ ;;BRANCH IF YES
3439
3440 014166 104401 014734 $GTSWR: TYPE , $CNTLG ;;ECHO THE CONTROL-G (^G)
3441 014172 104401 014741 TYPE , $MSWR ;;TYPE CURRENT CONTENTS
3442 014176 013745 000176 MOV SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
3443 014202 104402 TYPEOF ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3444 014204 104401 014752 TYPE , $MNEW ;;PROMPT FOR NEW SWR
3445 014210 005046 19$: CLR -(SP) ;;CLEAR COUNTER
3446 014212 005046 CLR -(SP) ;;THE NEW SWR
3447 014214 105777 164724 7$: TSTB @STKS ;;CHAR THERE?
3448 014220 100375 BPL 7$ ;;IF NOT TRY AGAIN
3449
3450 014222 117746 164720 MOVB @STKB,-(SP) ;;PICK UP CHAR
3451 014226 042716 177600 BIC #^C177,(SP) ;;MAKE IT 7-BIT ASCII
3452
3453 014232 021627 000003 CMP (SP),#3 ;;IS IT A CONTROL-C?
3454 014236 001015 BNE 9$ ;;BRANCH IF NOT
3455 014240 104401 014722 TYPE , $CNTLC ;;YES, ECHO CONTROL-C (^C)
3456 014244 062706 000006 ADD #6,SP ;;CLEAN UP STACK
3457 014250 123727 001135 000001 CMPB $INTAG,#1 ;;REENABLE TTY KEYBOARD INTERRUPTS?
3458 014256 001003 BNE 8$ ;;BRANCH IF NO
3459 014260 012777 000100 164656 MOV #100,@STKS ;;ALLOW TTY KEYBOARD INTERRUPTS
3460 014266 000137 012316 8$: JMP GT$DEV ;;CONTROL-C RESTART
3461
3462
3463 014272 021627 000025 9$: CMP (SP),#25 ;;IS IT A CONTROL-U?
3464 014276 001005 BNE 10$ ;;BRANCH IF NOT
3465 014300 104401 014727 TYPE , $CNTLU ;;YES, ECHO CONTROL-U (^U)
3466 014304 062706 000006 20$: ADD #6,SP ;;IGNORE PREVIOUS INPUT
3467 014310 000737 BR 19$ ;;LET'S TRY IT AGAIN
3468
3469
3470 014312 021627 000015 10$: CMP (SP),#15 ;;IS IT A <CR>?
3471 014316 001022 BNE 16$ ;;BRANCH IF NO
3472 014320 005766 000004 TST 4(SP) ;;YES, IS IT THE FIRST CHAR?
3473 014324 001403 BEQ 11$ ;;BRANCH IF YES
3474 014326 016677 000002 164604 MOV 2(SP),@SWR ;;SAVE NEW SWR

```

3475	014334	062706	000006		11\$:	ADD	#6,SP	::CLEAR UP STACK
3476	014340	104401	001171		14\$:	TYPE	,SCLF	::ECHO <CR> AND <LF>
3477	014344	123727	001135	000001		CMPB	\$INTAG,#1	::RE-ENABLE TTY KBD INTERRUPTS?
3478	014352	001003				BNE	15\$	::BRANCH IF NOT
3479	014354	012777	000100	164562		MOV	#100,@STKS	::RE-ENABLE TTY KBD INTERRUPTS
3480	014362	000002			15\$:	RTI		::RETURN
3481	014364	004737	014052		16\$:	JSR	PC,\$TYPEC	::ECHO CHAR
3482	014370	021627	000060			CMP	(SP),#60	::CHAR < 0?
3483	014374	002420				BLT	18\$	::BRANCH IF YES
3484	014376	021627	000067			CMP	(SP),#67	::CHAR > 7?
3485	014402	003015				BGT	18\$	::BRANCH IF YES
3486	014404	042726	000060			BIC	#60,(SP)+	::STRIP-OFF ASCII
3487	014410	005766	000002			TST	2(SP)	::IS THIS THE FIRST CHAR
3488	014414	001403				BEQ	17\$	::BRANCH IF YES
3489	014416	006316				ASL	(SP)	::NO, SHIFT PRESENT
3490	014420	006316				ASL	(SP)	::CHAR OVER TO MAKE
3491	014422	006316				ASL	(SP)	::ROOM FOR NEW ONE.
3492	014424	005266	000002		17\$:	INC	2(SP)	::KEEP COUNT OF CHAR
3493	014430	056616	177776			BIS	-2(SP),(SP)	::SET IN NEW CHAR
3494	014434	000667				BR	7\$	::GET THE NEXT ONE
3495	014436	104401	001170		18\$:	TYPE	,SQUES	::TYPE ?<CR><LF>
3496	014442	000720				BR	20\$	::SIMULATE CONTROL-U
3497					.DSABL	LSB		
3498								
3499								
3500								
3501								
3502								
3503								
3504								
3505								
3506								
3507								
3508	014444	011646			\$RDCHR:	MOV	(SP),-(SP)	::PUSH DOWN THE PC
3509	014446	016666	000004	000002		MOV	4(SP),2(SP)	::SAVE THE PS
3510	014454	105777	164464		1\$:	TSTB	@STKS	::WAIT FOR
3511	014460	100375				BPL	1\$	::A CHARACTER
3512	014462	117766	164460	000004		MOVB	@STKB,4(SP)	::READ THE TTY
3513	014470	042766	177600	000004		BIC	#^C<177>,4(SP)	::GET RID OF JUNK IF ANY
3514	014476	026627	000004	000023		CMP	4(SP),#23	::IS IT A CONTROL-S?
3515	014504	001013				BNE	3\$	::BRANCH IF NO
3516	014506	105777	164432		2\$:	TSTB	@STKS	::WAIT FOR A CHARACTER
3517	014512	100375				BPL	2\$	::LOOP UNTIL ITS THERE
3518	014514	117746	164426			MOVB	@STKB,-(SP)	::GET CHARACTER
3519	014520	042716	177600			BIC	#^C<177>,(SP)	::MAKE IT 7-BIT ASCII
3520	014524	022627	000021			CMP	(SP)+,#21	::IS IT A CONTROL-Q?
3521	014530	001366				BNE	2\$	::IF NOT DISCARD IT
3522	014532	000750				BR	1\$	::YES, RESUME
3523	014534	026627	000004	000140	3\$:	CMP	4(SP),#140	::IS IT UPPER CASE?
3524	014542	002407				BLT	4\$	::BRANCH IF YES
3525	014544	026627	000004	000175		CMP	4(SP),#175	::IS IT A SPECIAL CHAR?
3526	014552	003003				BGT	4\$	::BRANCH IF YES
3527	014554	042766	000040	000004		BIC	#40,4(SP)	::MAKE IT UPPER CASE
3528	014562	000002			4\$:	RTI		::GO BACK TO USER
3529								
3530								

 *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
 *

 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY

```

3531      ::CALL:
3532      ::      RDLIN
3533      ::      RETURN HERE
3534      ::      INPUT A STRING FROM THE TTY
3535      ::      ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
3536      ::      TERMINATOR WILL BE A BYTE OF ALL 0'S
3537
3538      SRDLIN: MOV      R3, -(SP)      ::SAVE R3
3539      1$:      MOV      #STTYIN, R3  ::GET ADDRESS
3540      2$:      CMP      #STTYIN+8., R3 ::BUFFER FULL?
3541      BLOS     4$                  ::BR IF YES
3542      RDCHR    (SP)+, (R3)          ::GO READ ONE CHARACTER FROM THE TTY
3543      MOVB     #3, (R3)             ::GET CHARACTER
3544      CMPB     #3, (R3)             ::IS IT A CONTROL-C?
3545      BNE      10$                  ::BRANCH IF NO
3546      TYPE     , $CNTLC             ::TYPE A CONTROL-C (^C)
3547      MOV      (SP)+, R3            ::RESTORE R3
3548      JMP      GTSDEV               ::GOTO CONTROL-C RESTART
3549      10$:     CMPB     #177, (R3)   ::IS IT A RUBOUT
3550      BNE      3$                  ::SKIP IF NOT
3551      TYPE     , $QUES              ::TYPE A '?'
3552      BR       1$                  ::CLEAR THE BUFFER AND LOOP
3553      3$:      MOVB     (R3), 9$     ::ECHO THE CHARACTER
3554      TYPE     , 9$
3555      CMPB     #15, (R3)+          ::CHECK FOR RETURN
3556      BNE      2$                  ::LOOP IF NOT RETURN
3557      CLRB     -1(R3)              ::CLEAR RETURN (THE 15)
3558      TYPE     , $LF               ::TYPE A LINE FEED
3559      MOV      (SP)+, R3            ::RESTORE R3
3560      MOV      (SP), -(SP)          ::ADJUST THE STACK AND PUT ADDRESS OF THE
3561      MOV      4(SP), 2(SP)         ::      FIRST ASCII CHARACTER ON IT
3562      MOV      #STTYIN, 4(SP)
3563      RTI
3564      9$:      .BYTE     0          ::RETURN
3565      .BYTE     0                  ::STORAGE FOR ASCII CHAR. TO TYPE
3566      .BLKB     8.                 ::TERMINATOR
3567      STTYIN:  .BLKB     8.         ::RESERVE 8 BYTES FOR TTY INPUT
3568      $CNTLC:  .ASCIIZ   /^C/<15><12> ::CONTROL '^C'
3569      $CNTLU:  .ASCIIZ   /^U/<15><12> ::CONTROL '^U'
3570      $CNTLG:  .ASCIIZ   /^G/<15><12> ::CONTROL '^G'
3571      $MSWR:   .ASCIIZ   <15><12>/SWR - /
3572      $MNEW:   .ASCIIZ   / NEW = /
3573      .EVEN

```

```

3573      .SBTTL  APT COMMUNICATIONS ROUTINE
3574
3575      ;*****
3576 014764 112737 000001 015230 $ATY1:  MOVB  #1,$FFLG      ;;TO REPORT FATAL ERROR
3577 014772 112737 000001 015226 $ATY3:  MOVB  #1,$MFLG      ;;TO TYPE A MESSAGE
3578 015000 000403          BR      $ATYC
3579 015002 112737 000001 015230 $ATY4:  MOVB  #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
3580 015010          $ATYC:
3581 015010 010046          MOV    R0,-(SP)      ;;PUSH R0 ON STACK
3582 015012 010146          MOV    R1,-(SP)      ;;PUSH R1 ON STACK
3583 015014 105737 015226          TSTB  $MFLG      ;;SHOULD TYPE A MESSAGE?
3584 015020 001450          BEQ     5$           ;;IF NOT:  BR
3585 015022 122737 000001 001214          CMPB  #APTENV,$ENV      ;;OPERATING UNDER APT?
3586 015030 001031          BNE     3$           ;;IF NOT:  BR
3587 015032 132737 000100 001215          BITB  #APTPOOL,$ENVM    ;;SHOULD SPOOL MESSAGES?
3588 015040 001425          BEQ     3$           ;;IF NOT:  BR
3589 015042 017600 000004          MOV    @4(SP),R0      ;;GET MESSAGE ADDR.
3590 015046 062766 000002 000004          ADD    #2,4(SP)      ;;BUMP RETURN ADDR.
3591 015054 005737 001174          1$:  TST    $MSGTYPE      ;;SEE IF DONE W/ LAST XMISSION?
3592 015060 001375          BNE     1$           ;;IF NOT:  WAIT
3593 015062 010037 001210          MOV    R0,$MSGAD      ;;PUT ADDR IN MAILBOX
3594 015066 105720          2$:  TSTB  (R0)+      ;;FIND END OF MESSAGE
3595 015070 001376          BNE     2$
3596 015072 163700 001210          SUB     $MSGAD,R0      ;;SUB START OF MESSAGE
3597 015076 006200          ASR     R0           ;;GET MESSAGE LGTH IN WORDS
3598 015100 010037 001212          MOV    R0,$MSGGLT      ;;PUT LENGTH IN MAILBOX
3599 015104 012737 000004 001174          MOV    #4,$MSGTYPE      ;;TELL APT TO TAKE MSG.
3600 015112 000413          BR      5$
3601 015114 017637 000004 015140 3$:  MOV    @4(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
3602 015122 062766 000002 000004          ADD    #2,4(SP)      ;;BUMP RETURN ADDRESS
3603 015130 013746 177776          MOV    177776,-(SP)      ;;PUSH 177776 ON STACK
3604 015134 004737 013640          JSR     PC,$TYPE      ;;CALL TYPE MACRO
3605 015140 000000          4$:  .WORD  0
3606 015142          5$:
3607 015142 105737 015230          10$: TSTB  $FFLG      ;;SHOULD REPORT FATAL ERROR?
3608 015146 001416          BEQ     12$          ;;IF NOT:  BR
3609 015150 005737 001214          TST    $ENV      ;;RUNNING UNDER APT?
3610 015154 001413          BEQ     12$          ;;IF NOT:  BR
3611 015156 005737 001174          11$: TST    $MSGTYPE      ;;FINISHED LAST MESSAGE?
3612 015162 001375          BNE     11$          ;;IF NOT:  WAIT
3613 015164 017637 000004 001176          MOV    @4(SP),$FATAL      ;;GET ERROR #
3614 015172 062766 000002 000004          ADD    #2,4(SP)      ;;BUMP RETURN ADDR.
3615 015200 005237 001174          INC     $MSGTYPE      ;;TELL APT TO TAKE ERROR
3616 015204 105037 015230          12$: CLRB  $FFLG      ;;CLEAR FATAL FLAG
3617 015210 105037 015227          CLRB  $LFLG      ;;CLEAR LOG FLAG
3618 015214 105037 015226          CLRB  $MFLG      ;;CLEAR MESSAGE FLAG
3619 015220 012601          MOV    (SP)+,R1      ;;POP STACK INTO R1
3620 015222 012600          MOV    (SP)+,R0      ;;POP STACK INTO R0
3621 015224 000207          RTS     PC           ;;RETURN
3622 015226          000          $MFLG: .BYTE  0      ;;MESSG. FLAG
3623 015227          000          $LFLG: .BYTE  0      ;;LOG FLAG
3624 015230          000          $FFLG: .BYTE  0      ;;FATAL FLAG
3625          015232          .EVEN
3626          000200          APTSIZE=200
3627          000001          APTENV=001
3628          000100          APTPOOL=100

```

CVHAAO MKV11A DIAG MACV11 30G(1063) 30-MAY-79 15:58 PAGE 83^{D 7}
CVHAA.P11 30-MAY-79 15:57 APT COMMUNICATIONS ROUTINE

SEQ 0081

3629 000040 APT(SUP-040)

```

3630 .SBTTL ERROR HANDLER ROUTINE
3631
3632 ;*****
3633 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3634 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3635 ;*AND GO TO MYTYPE ON ERROR
3636 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3637 ;*SW15=1      HALT ON ERROR
3638 ;*SW13=1      INHIBIT ERROR TYPEOUTS
3639 ;*SW10=1      BELL ON ERROR
3640 ;*SW09=1      LOOP ON ERROR
3641 ;*CALL
3642 ;*      ERROR      N      ;;ERROR-EMT AND N-ERROR ITEM NUMBER
3643
3644 $ERROR:
3645 015232 104407 001103 7$: CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
3646 015234 105237 001103 7$: INCB      $ERFLG      ;;SET THE ERROR FLAG
3647 015240 001775 001102 163672 BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
3648 015242 013777 002000 163662 MOV      $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
3649 015250 032777 001402 163662 BIT      #BIT10,@SWR  ;;BELL ON ERROR?
3650 015256 001402 001164 1$      BEQ      1$      ;;NO - SKIP
3651 015260 104401 001112 1$: TYPE      $BELL      ;;RING BELL
3652 015264 005237 001116 1$: INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
3653 015270 011637 000002 001116 MOV      (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
3654 015274 162737 000002 001116 SUB      #2,$ERRPC
3655 015302 117737 163610 001114 MOV      @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
3656 015310 032777 020000 163622 BIT      #BIT13,@SWR  ;;SKIP TYPEOUT IF SET
3657 015316 001004 012642 BNE      20$      ;;SKIP TYPEOUTS
3658 015320 004737 001171 JSR      PC,MYTYPE  ;;GO TO USER ERROR ROUTINE
3659 015324 104401 001171 TYPE      $CRLF
3660 015330 122737 000001 001214 20$: CMPB      #APTENV,$ENV  ;;RUNNING IN APT MODE
3661 015336 001007 001114 015352 BNE      2$      ;;NO,SKIP APT ERROR REPORT
3662 015340 113737 015002 21$: MOV      $ITEMB,21$  ;;SET ITEM NUMBER AS ERROR NUMBER
3663 015346 004737 015002 21$: JSR      PC,$ATY4  ;;REPORT FATAL ERROR TO APT
3664 015352 000 21$: .BYTE      0
3665 015353 000 22$: .BYTE      0
3666 015354 000777 163556 22$: BR      22$      ;;APT ERROR LOOP
3667 015356 005777 163556 2$: TST      @SWR      ;;HALT ON ERROR
3668 015362 100002 3$: BPL      3$      ;;SKIP IF CONTINUE
3669 015364 000000 3$: HALT      ;;HALT ON ERROR!
3670 015366 104407 001000 163542 3$: CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
3671 015370 032777 001402 163542 3$: BIT      #BIT09,@SWR  ;;LOOP ON ERROR SWITCH SET?
3672 015376 001402 001110 4$: BEQ      4$      ;;BR IF NO
3673 015400 013716 001162 4$: MOV      $LPERR,(SP)  ;;FUDGE RETURN FOR LOOPING
3674 015404 005737 001162 4$: TST      $ESCAPE  ;;CHECK FOR AN ESCAPE ADDRESS
3675 015410 001402 001162 5$: BEQ      5$      ;;BR IF NONE
3676 015412 013716 000042 5$: MOV      $ESCAPE,(SP)  ;;FUDGE RETURN ADDRESS FOR ESCAPE
3677 015416 022737 013422 000042 5$: CMP      #SENDAD,@#42  ;;ACT-11 AUTO-ACCEPT?
3678 015424 001001 6$: BNE      6$      ;;BRANCH IF NO
3679 015426 000000 6$: HALT      ;;YES
3680 015430 000002 RTI      ;;RETURN
3681
3682
3683

```

```

3684 .SBTTL SCOPE HANDLER ROUTINE
3685
3686 *****
3687 *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3688 *AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3689 *AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
3690 *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3691 *SW14=1 LOOP ON TEST
3692 *SW11=1 INHIBIT ITERATIONS
3693 *SW09=1 LOOP ON ERROR
3694 *SW08=1 LOOP ON TEST IN SWR<7:0>
3695 *CALL
3696 * SCOPE ;:SCOPE=101
3697
3698 $SCOPE:
3699 015432 104407 040000 163476 1$: CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
3700 015434 032777 040000 163476 1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
3701 015442 001114 040000 163476 1$: BNE $OVER ;:YES IF SW14=1
3702 ;:*****START OF CODE FOR THE XOR TESTER*****
3703 015444 000416 040000 163476 $XTSTR: BR 6$ ;:IF RUNNING ON THE "XOR" TESTER CHANGE
3704 ;:THIS INSTRUCTION TO A "NOP" (NOP 240)
3705 015446 013746 000004 000004 MOV @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
3706 015452 012737 015472 000004 MOV #5$,@#ERRVEC ;:SET FOR TIMEOUT
3707 015460 005737 177060 000004 TST @#177060 ;:TIME OUT ON XOR?
3708 015464 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
3709 015470 000463 000004 BR $SVLAD ;:GO TO THE NEXT TEST
3710 015472 022626 000004 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
3711 015474 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
3712 015500 000423 000004 BR 7$ ;:LOOP ON THE PRESENT TEST
3713 015502 032777 000400 163430 6$;:*****END OF CODE FOR THE XOR TESTER*****
3714 015510 001404 163422 001102 BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
3715 015512 127737 163422 001102 BEQ 2$ ;:BR IF NO
3716 015520 001465 001103 163372 CMPB @SWR,$STNM ;:ON THE RIGHT TEST? SWR<7:0>
3717 015522 105737 001103 163372 BEQ $OVER ;:BR IF YES
3718 015526 001421 001103 163372 2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
3719 015530 123737 001115 001103 BEQ 3$ ;:BR IF NO
3720 015536 101015 001103 163372 CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
3721 015540 032777 001000 163372 BHI 3$ ;:BR IF NO
3722 015546 001404 001110 001106 BIT #BIT09,@SWR ;:LOOP ON ERROR?
3723 015550 013737 001110 001106 BEQ 4$ ;:BR IF NO
3724 015556 000446 001103 001160 7$: MOV $LPERR,$LPADR ;:SET LOOP ADDRESS TO LAST SCOPE
3725 015560 105037 001103 001160 BR $OVER ;:ZERO THE ERROR FLAG
3726 015564 005037 001160 001160 4$: CLRB $ERFLG ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
3727 015570 000415 004000 163340 3$: CLR $TIMES ;:ESCAPE TO THE NEXT TEST
3728 015572 032777 004000 163340 3$: BIT #BIT11,@SWR ;:INHIBIT ITERATIONS?
3729 015600 001011 001202 001104 BNE 1$ ;:BR IF YES
3730 015602 005737 001202 001104 TST $PASS ;:IF FIRST PASS OF PROGRAM
3731 015606 001406 001104 001160 BEQ 1$ ;: INHIBIT ITERATIONS
3732 015610 005237 001104 001160 INC $ICNT ;:INCREMENT ITERATION COUNT
3733 015614 023737 001160 001104 CMP $TIMES,$ICNT ;:CHECK THE NUMBER OF ITERATIONS MADE
3734 015622 002024 000001 001104 BGE $OVER ;:BR IF MORE ITERATION REQUIRED
3735 015624 012737 000001 001104 1$: MOV #1,$ICNT ;:REINITIALIZE THE ITERATION COUNTER
3736 015632 013737 015710 001160 MOV $MXCNT,$TIMES ;:SET NUMBER OF ITERATIONS TO DO
3737 015640 105237 001102 001200 $SVLAD: INCB $STNM ;:COUNT TEST NUMBERS
3738 015644 113737 001102 001200 MOV $STNM,$TESTN ;:SET TEST NUMBER IN APT MAILBOX
3739

```

CVMXAAO MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 86
CVMXAA.P11 30-MAY-79 15:57 SCOPE HANDLER ROUTINE

SEQ 0084

3740	015652	011637	001106	MOV	(SP), \$LPADR	::: SAVE SCOPE LOOP ADDRESS
3741	015656	011637	001110	MOV	(SP), \$LPERR	::: SAVE ERROR LOOP ADDRESS
3742	015662	005037	001162	CLR	\$ESCAPE	::: CLEAR THE ESCAPE FROM ERROR ADDRESS
3743	015666	112737	000001	MOVB	#1, \$ERMAX	::: ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3744	015674	013777	001102	SOVER: MOV	\$ISTNM, @DISPLAY	::: DISPLAY TEST NUMBER
3745	015702	013716	001106	MOV	\$LPADR, (SP)	::: FUDGE RETURN ADDRESS
3746	015706	000002		RTI		::: FIXES PS
3747	015710	003720		\$MXCNT: 2000.		::: MAX. NUMBER OF ITERATIONS

```

3748 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3749
3750 *****
3751 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3752 *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
3753 *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3754 *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3755 *REPLACED WITH SPACES.
3756 *CALL:
3757 *      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
3758 *      TYPDS    ;;GO TO THE ROUTINE
3759
3760 015712 $TYPDS:
3761 015712 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
3762 015714 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
3763 015716 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
3764 015720 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
3765 015722 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
3766 015724 012746 020200      MOV      #20200,-(SP) ;;SET BLANK SWITCH AND SIGN
3767 015730 016605 000020      MOV      20(SP),R5 ;;GET THE INPUT NUMBER
3768 015734 100004      BPL      1$          ;;BR IF INPUT IS POS.
3769 015736 005405      NEG      R5          ;;MAKE THE BINARY NUMBER POS.
3770 015740 112766 000055 000001      MOVB   #'-,1(SP) ;;MAKE THE ASCII NUMBER NEG.
3771 015746 005000      CLR      R0          ;;ZERO THE CONSTANTS INDEX
3772 015750 012703 016126      MOV      #SDBLK,R3 ;;SETUP THE OUTPUT POINTER
3773 015754 112723 000040      MOVB   #'',(R3)+ ;;SET THE FIRST CHARACTER TO A BLANK
3774 015760 005002      CLR      R2          ;;CLEAR THE BCD NUMBER
3775 015762 016001 016116      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
3776 015766 160105      3$:      SUB      R1,R5 ;;FORM THIS BCD DIGIT
3777 015770 002402      BLT      4$          ;;BR IF DONE
3778 015772 005202      INC      R2          ;;INCREASE THE BCD DIGIT BY 1
3779 015774 000774      BR       3$
3780 015776 060105      4$:      ADD      R1,R5 ;;ADD BACK THE CONSTANT
3781 016000 005702      TST      R2          ;;CHECK IF BCD DIGIT-0
3782 016002 001002      BNE      5$          ;;FALL THROUGH IF 0
3783 016004 105716      TSTB   (SP)          ;;STILL DOING LEADING 0'S?
3784 016006 100407      BMI      7$          ;;BR IF YES
3785 016010 106316      5$:      ASLB   (SP)          ;;MSD?
3786 016012 103003      BCC      6$          ;;BR IF NO
3787 016014 116663 000001 177777      MOVB   1(SP),-1(R3) ;;YES--SET THE SIGN
3788 016022 052702 000060      BIS      #'0,R2 ;;MAKE THE BCD DIGIT ASCII
3789 016026 052702 000040      7$:      BIS      #' ,R2 ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
3790 016032 110223      MOVB   R2,(R3)+ ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
3791 016034 005720      TST      (R0)+      ;;JUST INCREMENTING
3792 016036 020027 000010      CMP      R0,#10 ;;CHECK THE TABLE INDEX
3793 016042 002746      BLT      2$          ;;GO DO THE NEXT DIGIT
3794 016044 003002      BGT      8$          ;;GO TO EXIT
3795 016046 010502      MOV      R5,R2          ;;GET THE LSD
3796 016050 000764      BR       6$          ;;GO CHANGE TO ASCII
3797 016052 105726      8$:      TSTB   (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
3798 016054 100003      BPL      9$          ;;BR IF NO
3799 016056 116663 177777 177776      MOVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
3800 016064 105013      9$:      CLRB   (R3)          ;;SET THE TERMINATOR
3801 016066 012605      MOV      (SP)+,R5 ;;POP STACK INTO R5
3802 016070 012603      MOV      (SP)+,R3 ;;POP STACK INTO R3
3803 016072 012602      MOV      (SP)+,R2 ;;POP STACK INTO R2

```

(VMXAA0 MXV11A DIAG MACV11 30G(1063) 30-MAY-79 15:58 PAGE 88
(VMXAA.P11 30-MAY-79 15:57 CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

SEQ 0086

3804	016074	012601		MOV	(SP)+,R1	::POP STACK INTO R1
3805	016076	012600		MOV	(SP)+,R0	::POP STACK INTO R0
3806	016100	104401	016126	TYPE	,SDBLK	::NOW TYPE THE NUMBER
3807	016104	016666	000002 000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
3808	016112	012616		MOV	(SP)+,(SP)	
3809	016114	000002		RTI		::RETURN TO USER
3810	016116	023420		\$DTBL:	10000.	
3811	016120	001750			1000.	
3812	016122	000144			100.	
3813	016124	000012			10.	
3814	016126	000004		\$SDBLK:	.BLKW 4	

```

3815 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
3816
3817 *****
3818 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3819 *OCTAL (ASCII) NUMBER AND TYPE IT.
3820 *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3821 *CALL:
3822 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3823 *      TYPOS      ;;CALL FOR TYPEOUT
3824 *      .BYTE     N      ;;N-1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3825 *      .BYTE     M      ;;M 1 OR 0
3826 *                               ;;1=TYPE LEADING ZEROS
3827 *                               ;;0=SUPPRESS LEADING ZEROS
3828
3829 *$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3830 *$TYPOS OR $TYPOC
3831 *CALL:
3832 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3833 *      TYPON      ;;CALL FOR TYPEOUT
3834
3835 *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3836 *CALL:
3837 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3838 *      TYPOC      ;;CALL FOR TYPEOUT
3839
3840 016136 017646 000000 016361 $TYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
3841 016142 116637 000001      MOV      1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
3842 016150 112637 016363      MOV      (SP)+,SOMODE+1      ;;NUMBER OF DIGITS TO TYPE
3843 016154 062716 000002      ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
3844 016160 000406      BR      $TYPON
3845 016162 112737 000001 016361 $TYPOC: MOV      #1,SOFILL      ;;SET THE ZERO FILL SWITCH
3846 016170 112737 000006 016363      MOV      #6,SOMODE+1      ;;SET FOR SIX(6) DIGITS
3847 016176 112737 000005 016360 $TYPON: MOV      #5,SOCNT      ;;SET THE ITERATION COUNT
3848 016204 010346      MOV      R3,-(SP)      ;;SAVE R3
3849 016206 010446      MOV      R4,-(SP)      ;;SAVE R4
3850 016210 010546      MOV      R5,-(SP)      ;;SAVE R5
3851 016212 113704 016363      MOV      SOMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
3852 016216 005404      NEG      R4
3853 016220 062704 000006      ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
3854 016224 110437 016362      MOV      R4,SOMODE      ;;SAVE IT FOR USE
3855 016230 113704 016361      MOV      SOFILL,R4      ;;GET THE ZERO FILL SWITCH
3856 016234 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
3857 016240 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
3858 016242 006105      1$: ROL      R5      ;;ROTATE MSB INTO 'C'
3859 016244 000404      BR      3$      ;;GO DO MSB
3860 016246 006105      2$: ROL      R5      ;;FORM THIS DIGIT
3861 016250 006105      ROL      R5
3862 016252 006105      ROL      R5
3863 016254 010503      MOV      R5,R3
3864 016256 006103      3$: ROL      R3      ;;GET LSB OF THIS DIGIT
3865 016260 105337 016362      DECB     SOMODE      ;;TYPE THIS DIGIT?
3866 016264 100016      BPL      7$      ;;BR IF NO
3867 016266 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK
3868 016272 001002      BNE      4$      ;;TEST FOR 0
3869 016274 005704      TST      R4      ;;SUPPRESS THIS 0?
3870 016276 001403      BEQ      5$      ;;BR IF YES
  
```

3871	016300	005204		4\$:	INC	R4	::DON'T SUPPRESS ,NYMORE 0'S
3872	016302	052703	000060		BIS	#'0,R3	::MAKE THIS DIGIT ASCII
3873	016306	052703	000040	5\$:	BIS	#',R3	::MAKE ASCII IF NOT ALREADY
3874	016312	110337	016356		MOVB	R3,8\$	::SAVE FOR TYPING
3875	016316	104401	016356		TYPE	,8\$	::GO TYPE THIS DIGIT
3876	016322	105337	016360	7\$:	DECB	\$OCNT	::COUNT BY 1
3877	016326	003347			BGT	2\$	::BR IF MORE TO DO
3878	016330	002402			BLT	6\$	::BR IF DONE
3879	016332	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
3880	016334	000744			BR	2\$	::GO DO THE LAST DIGIT
3881	016336	012605		6\$:	MOV	(SP)+,R5	::RESTORE R5
3882	016340	012604			MOV	(SP)+,R4	::RESTORE R4
3883	016342	012603			MOV	(SP)+,R3	::RESTORE R3
3884	016344	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
3885	016352	012616			MOV	(SP)+,(SP)	
3886	016354	000002			RTI		::RETURN
3887	016356	000		8\$:	.BYTE	0	::STORAGE FOR ASCII DIGIT
3888	016357	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
3889	016360	000		\$OCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
3890	016361	000		\$OFILL:	.BYTE	0	::ZERO FILL SWITCH
3891	016362	000000		\$OMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE

```

3892      .SBTTL  TRAP DECODER
3893
3894      ;*****
3895      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3896      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3897      ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3898      ;*GO TO THAT ROUTINE.
3899
3900      016364 010046      $TRAP:  MOV    R0,-(SP)      ;;SAVE R0
3901      016366 016600 000002      MOV    2(SP),R0      ;;GET TRAP ADDRESS
3902      016372 005740      TST     -(R0)      ;;BACKUP BY 2
3903      016374 111000      MOVB   (R0),R0      ;;GET RIGHT BYTE OF TRAP.
3904      016376 006300      ASL     R0      ;;POSITION FOR INDEXING
3905      016400 016000 016420      MOV    $TRPAD(R0),R0      ;;INDEX TO TABLE
3906      016404 000200      RTS     R0      ;;GO TO ROUTINE
3907
3908
3909      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
3910
3911      016406 011646      $TRAP2: MOV    (SP),-(SP)      ;;MOVE THE PC DOWN
3912      016410 016666 000004 000002      MOV    4(SP),2(SP)      ;;MOVE THE PSW DOWN
3913      016416 000002      RTI      ;;RESTORE THE PSW
3914
3915      .SBTTL  TRAP TABLE
3916
3917      ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3918      ;*BY THE "TRAP" INSTRUCTION.
3919
3920      ;
3921      ; ROUTINE
3922      ;-----
3922      016420 016406      $TRPAD: .WORD  $TRAP2
3923      016422 013640      $TYPE  ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
3924      016424 016162      $TYPOC ;;CALL=TYPOC      TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3925      016426 016136      $TYPOS ;;CALL=TYPOS      TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
3926      016430 016176      $TYPON ;;CALL=TYPON      TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
3927      016432 015712      $TYPDS ;;CALL=TYPDS      TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
3928
3929      016434 014172      $GTSWR  ;;CALL=GTSWR      TRAP+6(104406)  GET SOFT-SWR SETTING
3930
3931      016436 014122      $CKSWR  ;;CALL=CKSWR      TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
3932      016440 014444      $RDCHR  ;;CALL=RDCHR      TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
3933      016442 014564      $RDLIN  ;;CALL=RDLIN      TRAP+11(104411) TTY TYPEIN STRING ROUTINE
3934      016444 012316      $DEV    ;;CALL=GTDEVM     TRAP+12(104412) ^C WILL OPEN UP DEV MAP
3935
3936
3937      ;MAX ADDR FOR 4K ALLOWED FOR APT = 17400
3938      ;MAX ADDR FOR 4K FOR ABS LOADER ONLY = 17500
3939      000001      .END
  
```


[illegible]

DISPLA	001142	871#	1019*	1027*	3648*	3744*											
DISPRE	000174	801#	1027														
DLADD	001270	959#	1344	1346	1349	1352	2546	3025*	3061*	3214							
DLVEC	J01302	964#	1956	2035	2208	2548	3026*	3062*	3216								
DONE	= J00200	775#	1765	1775	1803	1814	1846	1917	2137	2414	2508	2600	2625				
DSWR	= 177570	673#	870	1018													
EMTVEC=	000030	762#	1002*	1003*													
ERRCHK	006746	2404*	2425*	2432*	2446*	2455	2461	2467	2474	2523#							
ERRCNT	006200	2223*	2253	2262	2308#	2320	2354*	2441*									
ERRFLG	004370	1658*	1695*	1710	1740#												
ERROR	= 100000	780#	1876	1934	2146	2265	2343										
ERRVEC=	000004	755#	1016	1017*	1028*	1183*	1184*	1207*	1208*	1245*	1246*	1282*	1283*	1376*			
		1377*	1392*	1393*	2934*	2935*	2957*	2958*	3705	3706*	3708*	3711*					
EXFLG	004372	1660*	1721*	1728*	1732	1741#											
EX1	006176	2187	2304#														
EX2	006744	2397	2520#														
FRERR	= 020000	782#	2275	2429													
GNS	= ***** U	800	1054	2537	2542	2545	2547	2549	3202	3204	3207	3213	3215	3923			
		3924	3925	3926	3927	3929	3931	3932	3933	3934							
GTDEVM=	104412	1060	3934#														
GTSWR	= 104406	1049	3929#														
GT\$DEV	012316	3107#	3460	3546	3934												
HIROM	001266	956#	1260	1277													
HT	= 000011	665#	3378	3419													
IE	= 000100	776#	788#	1508	1520	1523	1536	1539	1552	1556	1577	1589	1592	1605			
		1608	1621	1625	1970	1974	2003	2047	2064	2087	2235	2238	2258	2260			
		2329	2352	2364	2370	2593	2595	2618	2620	2689	2695	2700	2706	2755			
		2757	2777	2779	2857	2864	2876	2883									
INTABL	010040	2576*	2577*	2578*	2579*	2642	2645	2658	2662	2671	2675	2715#					
INTFLG	011644	1253*	1255	1270*	1272	1300*	1311	1319	1382*	1384	1954*	1976	1988	2006*			
		2013	2044*	2067	2074	2090*	2096	2936*	2942	2985*	2988#						
INTSRV	011636	975	1245	1298	1376	1958	2036	2934	2985#								
IOTVEC=	000020	760#	1000*	1001*													
LF	= 000012	666#	3236	3245	3246	3247	3248	3249	3250	3251	3252	3413	3419				
LOOP	003126	1336#	2561														
LOROM	001264	955#	1252	1269													
MAXMEM	011634	1085	1095	1124	116	1178	1186	1216	2949*	2950*	2974#						
MEM	013314	2955	3252#														
MEMTBL	011616	2952	2966#														
MODTST	007264	1362	2564#														
MYTYPE	012642	3201#	3658														
M12K	013256	2968	3247#														
M16K	013264	2969	3248#														
M20K	013272	2970	3249#														
M24K	013300	2971	3250#														
M28K	C13306	2972	3251#														
M4K	013244	2966	3245#														
M8K	013251	2967	3246#														
NOCHAN	011614	1226	1337	2909*	2920*	2962#	2999										
NORAM	013170	2924	3236#														
NOROM	013145	2928	3232#														
NOTST	006750	2525#	2844														
NUMBER	006202	2229*	2231*	2244	2309#	2318	2350	2361									
ODD	012302	2437	3036*	3038*	3072*	3074*	3094#										
ORERR	= 040000	781#	186	1894	1922	2268											
PASS	004366	1653*	1713*	1715	1739#												

CVMXAAO MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 97
CVMXAA.P11 30-MAY-79 15:57 CR JS REFERENCE TABLE -- USER SYMBOLS

SE 0 0094

[illegible]

		2220	2223#	2224	2225#	2226	2235#	2236	2238#	2239	2258#	2259	2260#	2261
		2324#	2325	2329#	2330	2340#	2341	2350#	2351	2352#	2353	2354#	2355	2359#
		2360	2364#	2365	2370#	2371	2404#	2405	2407#	2408	2410#	2411	2419#	2420
		2425#	2426	2432#	2433	2486#	2487	2496#	2497	2501#	2502	2504#	2505	
SERROR	015232	1002	3644#											
SERRPC	001116	860#	3208	3653*	3654*	3655	3684							
SERRTB	001254	949#												
SERTTL	001112	857#	2550	2552*	3652*	3684								
SESCAP	001162	881#	1010*	3675	3677	3684	3742*							
SETABL	001214	900#												
SETEND	001254	842	933#											
SFATAL	001176	893#	3204*	3206	3613*									
SFFLG	015230	3576*	3579*	3607	3616*	3624#								
SFILLC	001156	878#	3388	3419										
SFILLS	001155	877#	3419											
SFSAND=	000310	634#	1438	1453	1469	1488	1510	1525	1541	1558	1579	1594	1610	1627
		1692	1712	1717	1777	1816	1863	1878	1896	1924	1936	1983	1990	2076
		2148	2264	2267	2270	2277	2284	2320	2345	2363	2423	2431	2457	2463
		2469	2476	2516										
SFSBAD-	000401	634#	1438	1453	1469	1488	1510	1525	1541	1558	1579	1594	1610	1627
		1692	1712	1717	1777	1816	1863	1878	1896	1924	1936	1983	1990	2076
		2148	2264	2267	2270	2277	2284	2322	2348	2363	2423	2431	2457	2463
		2469	2476	2516										
SFSBLA-	000170	634#												
SFSCAS=	000150	634#												
SFSDEC=	000220	634#												
SFSGOO-	000400	634#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625
		1690	1710	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074
		2146	2262	2265	2268	2275	2282	2318	2343	2361	2421	2429	2455	2461
		2467	2474	2514										
SFSIF	000110	634#	1436	1441	1451	1456	1467	1472	1486	1491	1508	1513	1523	1528
		1539	1544	1556	1561	1577	1582	1592	1597	1608	1613	1625	1630	1690
		1699	1708	1710	1715	1723	1725	1730	1775	1780	1814	1820	1861	1869
		1876	1886	1894	1902	1922	1931	1934	1940	1981	1988	1993	1995	2074
		2079	2146	2151	2262	2265	2268	2273	2275	2280	2282	2287	2292	2296
		2301	2303	2318	2320	2326	2343	2345	2356	2361	2366	2372	2374	2421
		2427	2429	2434	2455	2459	2461	2465	2467	2472	2474	2480	2514	2519
SFSINC	000210	634#	2117	2173										
SFSLOO=	000200	634#	1655	1732	1735									
SFSNAM-	000160	634#												
SFSNO -	000403	634#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625
		1690	1710	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074
		2146	2262	2265	2268	2275	2282	2318	2320	2343	2345	2361	2429	2455
		2461	2467	2474										
SFSOR =	000320	634#	1438	1453	1469	1488	1510	1525	1541	1558	1579	1594	1610	1627
		1692	1712	1717	1777	1816	1863	1878	1896	1924	1936	1983	1990	2076
		2148	2264	2267	2270	2277	2284	2320	2322	2345	2363	2423	2431	2457
		2463	2469	2476	2516									
SFSRTN-	000300	634#												
SFSSEL=	000140	634#												
SFSUNT-	000130	634#												
SFSWHI-	000120	634#	2320	2345										
SFSYES	000402	634#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625
		1690	1710	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074
		2146	2262	2265	2268	2275	2282	2318	2320	2343	2345	2361	2421	2429
		2455	2461	2467	2474	2514								

CVMXAAO MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 100
 CVMXAA.P11 30-MAY-79 15:57 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0097

\$GDADR 001120	861#													
\$GDDAT 001124	863#													
\$GET42 013412	3281#													
\$GTSWR 014172	3441#	3929												
\$HD = 000000	645													
\$HIBTS 001000	837#													
\$ICNT 001104	854#	3733*	3734	3736*	3747									
\$IFLEV= 177777	634#	1436#	1441#	1451#	1456#	1467#	1472#	1486#	1491#	1508#	1513#	1523#	1528#	
	1539#	1544#	1556#	1561#	1577#	1582#	1592#	1597#	1608#	1613#	1625#	1630#	1690#	
	1708#	1710#	1715#	1723#	1730#	1775#	1780#	1814#	1820#	1861#	1869#	1876#	1886#	
	1894#	1902#	1922#	1931#	1934#	1940#	1981#	1988#	1993#	1995#	2074#	2079#	2146#	
	2151#	2262#	2265#	2268#	2277	2284	2292#	2301#	2303#	2318#	2326#	2343#	2361#	
	2372#	2374#	2421#	2427#	2429#	2434#	2455#	2459#	2461#	2465#	2467#	2472#	2474#	
	2480#	2514#	2519#											
\$ILLUP 013622	3299	3315	3334#											
\$INTAG 001135	868#	3457	3477	3572										
\$ISKO = 000001	1436#	1441	1451#	1456	1457#	1472	1486#	1491	1508#	1513	1523#	1528	1539#	
	1544	1556#	1561	1577#	1582	1592#	1597	1608#	1613	1625#	1630	1690#	1708	
	1710#	1730	1775#	1780	1814#	1820	1861#	1869	1876#	1886	1894#	1902	1922#	
	1931	1934#	1940	1981#	1995	2074#	2079	2146#	2151	2262#	2303	2318#	2326	
	2343#	2374	2421#	2427	2429#	2434	2455#	2459	2461#	2465	2467#	2472	2474#	
	2480	2514#	2519											
\$ISK1 = 000001	1715#	1723	1988#	1993	2265#	2301	2361#	2372						
\$ISK2 = 000003	2268#	2277#	2284#	2292										
\$ITEMB 001114	858#	3204	3655*	3663	3684									
\$LF 001172	885#	3419	3556	3565	3684									
\$LFLG 015227	3617*	3623#												
\$LOCTA= 177777	634#	1437	1438	1441	1442	1452	1453	1456	1457	1468	1469	1472	1473	
	1487	1488	1491	1492	1509	1510	1513	1514	1524	1525	1528	1529	1540	
	1541	1544	1545	1557	1558	1561	1562	1578	1579	1582	1583	1593	1594	
	1597	1598	1609	1610	1613	1614	1626	1627	1630	1631	1655	1656	1691	
	1692	1699	1700	1701	1708	1709	1711	1712	1716	1717	1723	1724	1725	
	1726	1727	1730	1731	1733	1734	1735	1736	1737	1776	1777	1780	1781	
	1815	1816	1820	1821	1862	1863	1869	1870	1877	1878	1886	1887	1895	
	1896	1902	1903	1923	1924	1931	1932	1935	1936	1940	1941	1982	1983	
	1989	1990	1993	1994	1995	1996	2075	2076	2079	2080	2118	2119	2120	
	2121	2122	2123	2124	2147	2148	2151	2152	2173	2174	2175	2263	2264	
	2266	2267	2269	2270	2273	2274	2275	2276	2277	2280	2281	2282	2283	
	2284	2287	2288	2289	2292	2293	2294	2295	2296	2297	2298	2301	2302	
	2303	2304	2319	2320	2321	2322	2326	2327	2344	2345	2346	2347	2348	
	2356	2357	2358	2362	2363	2366	2367	2368	2372	2373	2374	2375	2422	
	2423	2427	2428	2430	2431	2434	2435	2456	2457	2459	2460	2462	2463	
	2465	2466	2468	2469	2472	2473	2475	2476	2480	2481	2515	2516	2519	
	2520													
\$LPADR 001106	855#	1012*	3724*	3740*	3745	3747								
\$LPERR 001110	856#	1013*	1432*	1445*	1460*	1476*	1504*	1517*	1532*	1548*	1573*	1586*	1601*	
	1617*	1632*	1757*	1770*	1794*	1835*	1873*	1889*	1905*	1961*	1999*	2041*	2083*	
	2400*	2482*	3674	3724	3741*	3747								
\$LSTCN 177777	634#	1436	1438	1441	1451	1453	1456	1467	1469	1472	1486	1488	1491	
	1508	1510	1513	1523	1525	1528	1539	1541	1544	1556	1558	1561	1577	
	1579	1582	1592	1594	1597	1608	1610	1613	1625	1627	1630	1655	1656	
	1690	1692	1700	1701	1708	1710	1712	1715	1717	1723	1726	1727	1730	
	1735	1736	1775	1777	1780	1814	1816	1820	1861	1863	1869	1876	1878	
	1886	1894	1896	1902	1922	1924	1931	1934	1936	1940	1981	1983	1988	
	1990	1993	1995	2074	2076	2079	2117	2119	2120	2121	2124	2146	2148	
	2151	2173	2174	2262	2264	2265	2267	2268	2270	2274	2275	2277	2281	

	2282	2284	2288	2289	2292	2293	2294	2297	2298	2301	2303	2318	2322
	2326	2343	2348	2357	2358	2361	2363	2367	2368	2372	2374	2421	2423
	2427	2429	2431	2434	2455	2457	2459	2461	2463	2465	2467	2469	2472
	2474	2476	2480	2514	2516	2519							
SLSTIN- 000000	634#	1436	1437	1438	1448	1449	1451	1452	1453	1464	1465	1467	1468
	1469	1480	1481	1486	1487	1488	1508	1509	1510	1520	1521	1523	1524
	1525	1536	1537	1539	1540	1541	1552	1553	1556	1557	1558	1577	1578
	1579	1589	1590	1592	1593	1594	1605	1606	1608	1609	1610	1621	1622
	1625	1626	1627	1653	1654	1658	1659	1660	1661	1668	1669	1685	1686
	1690	1691	1692	1695	1696	1699	1700	1710	1711	1712	1713	1714	1715
	1716	1717	1721	1722	1725	1726	1728	1729	1732	1733	1734	1735	1736
	1761	1762	1775	1776	1777	1799	1800	1811	1812	1814	1815	1816	1842
	1843	1852	1853	1857	1858	1861	1862	1863	1876	1877	1878	1894	1895
	1896	1913	1914	1922	1923	1924	1934	1935	1936	1970	1971	1974	1975
	1981	1982	1983	1988	1989	1990	2003	2004	2006	2007	2010	2011	2044
	2045	2047	2048	2054	2055	2064	2065	2074	2075	2076	2087	2088	2090
	2091	2093	2094	2117	2118	2119	2120	2121	2122	2123	2124	2133	2134
	2143	2144	2146	2147	2148	2173	2174	2208	2209	2211	2212	2213	2214
	2217	2218	2219	2220	2223	2224	2225	2226	2235	2236	2238	2239	2258
	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2273
	2274	2275	2276	2277	2280	2281	2282	2283	2284	2287	2288	2296	2297
	2318	2319	2320	2321	2322	2324	2325	2329	2330	2340	2341	2343	2344
	2345	2346	2347	2350	2351	2352	2353	2354	2355	2356	2357	2359	2360
	2361	2362	2363	2364	2365	2366	2367	2370	2371	2404	2405	2407	2408
	2410	2411	2419	2420	2421	2422	2423	2425	2426	2429	2430	2431	2432
	2433	2455	2456	2457	2461	2462	2463	2467	2468	2469	2474	2475	2476
SLSTST= 177777	2486	2487	2496	2497	2501	2502	2504	2505	2514	2515	2516		
	634#	1436	1438	1441	1451	1453	1456	1467	1469	1472	1486	1488	1491
	1508	1510	1513	1523	1525	1528	1539	1541	1544	1556	1558	1561	1577
	1579	1582	1592	1594	1597	1608	1610	1613	1625	1627	1630	1655	1656
	1690	1692	1699	1700	1701	1708	1710	1712	1715	1717	1723	1725	1726
	1727	1730	1732	1735	1736	1775	1777	1780	1814	1816	1820	1861	1863
	1869	1876	1878	1886	1894	1896	1902	1922					

SHAMS1	001224	912#												
SHAMS2	001230	920#												
SHAMS3	001234	923#												
SHAMS4	001240	926#												
SMBADR	001002	838#												
SMCALL=	000000	1#	634											
SMFLG	015226	3577#	3583	3618#	3622#									
SMNEW	014752	3110	3444	3570#										
SMSGAD	001210	898#	3593#	3596										
SMSGLG	001212	899#	3598#											
SMSGTY	001174	892#	3591	3599#	3611	36'5#								
SMSWR	014741	3441	3568#											
SMTYP1	001225	913#												
SMTYP2	001231	921#												
SMTYP3	001235	924#												
SMTYP4	001241	927#												
SMXCNT	015710	3737	3747#											
SNESTL-	177777	634#	1436#	1441#	1451#	1456#	1467#	1472#	1486#	1491#	1508#	1513#	1523#	1528#
		1539#	1544#	1556#	1561#	1577#	1582#	1592#	1597#	1608#	1613#	1625#	1630#	1655#
		1690#	1699	1708#	1710#	1715#	1723#	1725	1730#	1732	1735#	1775#	1780#	1814#
		1820#	1861#	1869#	1876#	1886#	1894#	1902#	1922#	1931#	1934#	1940#	1981#	1988#
		1993#	1995#	2074#	2079#	2117#	2146#	2151#	2173#	2262#	2265#	2268#	2273	2280
		2287	2292#	2296	2301#	2303#	2318#	2326#	2343#	2356	2361#	2366	2372#	2374#
		2421#	2427#	2429#	2434#	2455#	2459#	2461#	2465#	2467#	2472#	2474#	2480#	2514#
		2519#												
SNSK0 -	000110	1436#	1441	1451#	1456	1467#	1472	1486#	1491	1508#	1513	1523#	1528	1539#
		1544	1556#	1561	1577#	1582	1592#	1597	1608#	1613	1625#	1630	1655#	1732
		1735	1775#	1780	1814#	1820	1861#	1869	1876#	1886	1894#	1902	1922#	1931
		1934#	1940	1981#	1995	2074#	2079	2117#	2173	2262#	2303	2318#	2326	2343#
		2356	2374	2421#	2427	2429#	2434	2455#	2459	2461#	2465	2467#	2472	2474#
		2480	2514#	2519										
SNSK1 -	000110	1690#	1699	1708	1710#	1725	1730	1988#	1993	2146#	2151	2265#	2296	2301
		2361#	2366	2372										
SNSK2	000110	1715#	1723	2268#	2273	2280	2287	2292						
SNULL	001154	876#	3390	3419										
SNWTST	000001	1068#	1070	1100#	1102	1236#	1288#	1365#	1367	1416#	1418	1493#	1563#	1565
		1641#	1643	1743#	1745	1782#	1822#	1942#	1944	2019#	2021	2104#	2176#	2380#
		2382	2526#	2566#	2568	2719#	2721							
SOCNT	016360	3847#	3876#	3889#										
SOMODE	016362	3842#	3846#	3851	3854#	3865#	3891#							
SOVER	015674	3701	3717	3725	3735	3744#								
SPASS	001202	895#	1030#	3269#	3270#	3278	3291	3731	3748					
SPASTM	001006	840#												
SPOWER	013630	3330	3337#											
SPWRAD	013616	3332#												
SPWRDN	013456	1006	3299#	3327										
SPWRMG	013612	3330#												
SPWRUP	013530	3309	3315#											
SQUES	001170	883#	3147	3419	3495	3549	3565	3684						
SRDCHR	014444	3508#	3932											
SRDECI=	***** U	3934												
SRDLIN	014564	3536#	3933											
SRDOCI=	***** U	3934												
SRDSZ =	000010	3529#												
SRINAD	013434	3290#												
SRZA -	***** U	3934												

\$SAVLE= 177777	634#	1735#	1736#	2120#	2124#								
\$SAVRE= ***** U	3934												
\$SAVR6 013626	3308#	3316	3317#	3318#	3336#								
\$SCOPE 015432	1000	3698#											
\$SETUP= 000137	991#	999	1000	1002	1004	1006	1008	1009	1010	1012	1039	1042	3267
	3424	3572	3645	3671	3679	3699							
\$SSKO = 000037	1735#	1736	2120#	2124									
\$STUP = 177777	991#												
\$SVLAD 015640	3709	3738#											
\$SVPC = 000104	814#	819											
\$SWR 167400	634#	645	650	651	652	653	654	655	656	880	881	882	1009
	1010	1012	1013	1076	1114	1240	1292	1373	1422	1497	1569	1647	1749
	1786	1826	1951	2026	2108	2180	2388	2530	2573	2732	3262	3268	3283
	3289	3291	3333	3636	3637	3638	3639	3640	3649	3656	3668	3672	3684
	3690	3691	3692	3693	3694	3700	3712	3714	3715	3718	3719	3720	3727
	3728	3729	3741	3744	3747								
\$SWREG 001216	903#	1033											
\$SWRMK- 000000	656	657	3694	3695	3716								
\$TAGLE- 177777	634#	1438#	1441#	1453#	1456#	1469#	1472#	1488#	1491#	1510#	1513#	1525#	1528#
	1541#	1544#	1558#	1561#	1579#	1582#	1594#	1597#	1610#	1613#	1627#	1630#	1656#
	1692#	1700#	1701#	1708#	1712#	1717#	1723#	1726#	1727#	1730#	1732	1735#	1777#
	1780#	1816#	1820#	1863#	1869#	1878#	1886#	1896#	1902#	1924#	1931#	1936#	1940#
	1983#	1990#	1993#	1995#	2076#	2079#	2119#	2121#	2124#	2148#	2151#	2173#	2174#
	2264#	2267#	2270#	2274#	2275#	2277#	2281#	2282#	2284#	2288#	2289#	2292#	2293#
	2294#	2297#	2298#	2301#	2303#	2322#	2326#	2348#	2357#	2358#	2363#	2367#	2368#
	2372#	2374#	2423#	2427#	2431#	2434#	2457#	2459#	2463#	2465#	2469#	2472#	2476#
	2480#	2516#	2519#										
\$TAGNU- 000070	634#	1437	1438#	1452	1453#	1468	1469#	1487	1488#	1509	1510#	1524	1525#
	1540	1541#	1557	1558#	1578	1579#	1593	1594#	1609	1610#	1626	1627#	1655
	1656#	1691	1692#	1699	1701#	1711	1712#	1716	1717#	1725	1727#	1776	1777#
	1815	1816#	1862	1863#	1877	1878#	1895	1896#	1923	1924#	1935	1936#	1982
	1983#	1989	1990#	2075	2076#	2118	2119#	2120#	2123	2124#	2147	2148#	2263
	2264#	2266	2267#	2269	2270#	2273	2275#	2276	2277#	2280	2282#	2283	2284#
	2287	2289#	2296	2298#	2319	2321	2322#	2344	2346	2347	2348#	2356	2358#
	2362	2363#	2366	2368#	2422	2423#	2430	2431#	2456	2457#	2462	2463#	2468
	2469#	2475	2477	2515	2516#								
\$TEMP 000067	1441#	1448#	1449#	1456#	1464#	1465#	1472#	1480#	1481#	1491#	1513#	1520#	1521#
	1528#	1536#	1537#	1544#	1552#	1553#	1561#	1582#	1589#	1590#	1597#	1605#	1606#
	1613#	1621#	1622#	1630#	1653#	1654#	1658#	1659#	1660#	1661#	1668#	1669#	1685#
	1686#	1695#	1696#	1699#	1700#	1708#	1713#	1714#	1721#	1722#	1723#	1725#	1726#
	1728#	1729#	1730#	1732#	1733	1735#	1736#	1761#	1762#	1780#	1799#	1800#	1811#
	1812#	1820#	1842#	1843#	1852#	1853#	1857#	1858#	1869#	1886#	1902#	1913#	1914#
	1931#	1940#	1970#	1971#	1974#	1975#	1993#	1995#	2003#	2004#	2006#	2007#	2010#
	2011#	2044#	2045#	2047#	2048#	2054#	2055#	2064#	2065#	2079#	2087#	2088#	2090#
	2091#	2093#	2094#	2121#	2124#	2133#	2134#	2143#	2144#	2151#	2173#	2174#	2208#
	2209#	2211#	2212#	2213#	2214#	2217#	2218#	2219#	2220#	2223#	2224#	2225#	2226#
	2235#	2236#	2238#	2239#	2258#	2259#	2260#	2261#	2273#	2274#	2277#	2280#	2281#
	2284#	2287#	2288#	2292#	2293#	2294#	2296#	2297#	2301#	2303#	2324#	2325#	2326#
	2329#	2330#	2340#	2341#	2350#	2351#	2352#	2353#	2354#	2355#	2356#	2357#	2359#
	2360#	2364#	2365#	2366#	2367#	2370#	2371#	2372#	2374#	2404#	2405#	2407#	2408#
	2410#	2411#	2419#	2420#	2425#	2426#	2427#	2432#	2433#	2434#	2459#	2465#	2472#
	2480#	2486#	2487#	2496#	2497#	2501#	2502#	2504#	2505#	2519#			
\$TESTN 001200	894#	1077#	1115#	1241#	1293#	1374#	1423#	1498#	1570#	1648#	1750#	1787#	1827#
	1952#	2027#	2109#	2181#	2389#	2574#	2733#	3203	3209	3739#			
\$TIMES 001160	880#	1009#	1076#	1114#	1240#	1292#	1373#	1422#	1497#	1569#	1647#	1749#	1786#
	1826#	1951#	2026#	2108#	2180#	2388#	2530#	2573#	2732#	3268#	3727#	3734	3737#

		3747												
\$TKB	001146	873#	3117	3422	3433	3450	3512	3518						
\$TKS	001144	872#	2608	2668	2767	2823	2917	3088	3115	3422	3431	3447	3459*	3470*
		3510	3516											
\$TN	= 000025	634#	645	1068	1076#	1077	1080	1096	1100	1114#	1115	1198	1227	1236
		1240#	1241	1244	1284	1288	1292#	1293	1365	1373#	1374	1416	1422#	1423
		1428	1493	1497#	1498	1563	1569#	1570	1635	1641	1647#	1648	1651	1674
		1705	1743	1749#	1750	1753	1755	1767	1782	1786#	1787	1790	1792	1805
		1822	1826#	1827	1830	1832	1848	1867	1884	1900	1919	1929	1942	1951#
		1952	1967	1978	2019	2026#	2027	2030	2032	2060	2069	2104	2108#	2109
		2112	2114	2129	2139	2157	2162	2176	2180#	2181	2305	2380	2388#	2389
		2416	2492	2510	2521	2526	2530#	2566	2573#	2574	2680	2719	2732#	2733
\$TPB	001152	875#	3408*	3419										
\$TPFLG	001157	879#	3357	3419										
\$TPS	001150	874#	3406	3419										
\$TRAP	016364	1004	3900#											
\$TRAP2	016406	3911#	3922											
\$TRP	= 000013	3915#	3924#	3925#	3926#	3927#	3928#	3929	3930#	3931	3932#	3933#	3934#	3935#
\$TRPAD	016420	3905	3922#											
\$TSK0	- 000067	1438#	1441	1453#	1456	1469#	1472	1488#	1491	1510#	1513	1525#	1528	1541#
		1544	1558#	1561	1579#	1582	1594#	1597	1610#	1613	1627#	1630	1656#	1735
		1777#	1780	1816#	1820	1863#	1869	1878#	1886	1896#	1902	1924#	1931	1936#
		1940	1983#	1995	2076#	2079	2119#	2121	2124#	2174	2264#	2303	2322#	2326
		2348#	2357	2358#	2374	2423#	2427	2431#	2434	2457#	2459	2463#	2465	2469#
		2472	2476#	2480	2516#	2519								
\$TSK1	- 000060	1656#	1732	1735	1990#	1993	2124#	2173	2267#	2297	2298#	2301	2363#	2367
		2368#	2372											
\$TSK2	- 000045	1692#	1700	1701#	1708	1712#	1726	1727#	1730	2148#	2151	2270#	2274	2275#
		2294												
\$TSK3	- 000047	1717#	1723	2277#	2281	2282#	2293							
\$TSK4	= 000051	2284#	2288	2289#	2292									
\$TSTM	001004	839#												
\$TSTNM	001102	852#	2560*	3267*	3648	3684	3689	3716	3738*	3739	3744	3748		
\$TTYIN	014712	3537	3538	3560	3564#									
\$TYPBN=	***** U	3928												
\$TYPDS	015712	3760#	3927											
\$TYPE	013640	3357#	3604	3915	3923									
\$TYPEC	014052	3131	3387	3394	3401	3406#	3407	3481						
\$TYPEX	014120	3412	3414	3417#										
\$TYPOC	016162	3845#	3924											
\$TYPON	016176	3844	3847#	3926										
\$TYPOS	016136	3840#	3925											
\$UNIT	001206	897#	1333*	2556*	3003	3005*	3017	3022*	3053*	3058*				
\$UNITM	001010	841#												
\$USWR	001220	904#												
\$VECT1	001244	929#												
\$VECT2	001246	930#												
\$XTSTR	015444	3703#												
\$YESNO=	000001	1448#	1449#	1464#	1465#	1480#	1481#	1520#	1521#	1536#	1537#	1552#	1553#	1589#
		1590#	1605#	1606#	1621#	1622#	1713#	1714#	1970#	1971#	1974#	1975#	2003#	2004#
		2047#	2048#	2064#	2065#	2087#	2088#	2120#	2121#	2235#	2236#	2238#	2239#	2258#
		2259#	2260#	2261#	2329#	2330#	2352#	2353#	2354#	2355#	2359#	2360#	2364#	2365#
		2370#	2371#	2407#	2408#	2425#	2426#	2432#	2433#	2486#	2487#	2496#	2497#	
\$SBYTE-	000402	1436#	1451#	1467#	1486#	1508#	1523#	1539#	1556#	1577#	1592#	1608#	1625#	1690#
		1710#	1715#	1775#	1814#	1861#	1876#	1894#	1922#	1934#	1981#	1988#	2074#	2146#
		2262#	2265#	2268#	2275#	2282#	2318#	2320#	2343#	2345#	2361#	2421#	2429#	2455#

SEQ 0104

BEGIN	1#														
CALL	1#														
CASE	1#														
COMMEN	1#	767#													
DECR	1#														
DECRU	1#														
DEFAULT	1#														
ELSE	1#	1608	1724	2272	2279	2286	2295	2355	2365						
END	1#														
ENDCOM	1#	767#													
ENDDEC	1#														
ENDDO	1#														
ENDIF	1#	1440	1455	1471	1490	1512	1527	1543	1560	1581	1596	1612	1629	1707	1722
	1729	1779	1819	1868	1885	1901	1930	1939	1992	1994	2078	2150	2291	2300	2302
	2325	2371	2373	2426	2433	2458	2464	2471	2479	2518					
ENDINC	1#	2172													
ENDLOO	1#	1734													
ENDRTN	1#														
ENDSEL	1#														
ERROR	661#	1093	1132	1145	1152	1176	1195	1214	1258	1275	1313	1321	1386	1397	1401
	1405	1409	1439	1454	1470	1489	1511	1526	1542	1559	1580	1595	1611	1628	1636
	1673	1704	1719	1766	1778	1804	1817	1847	1864	1880	1898	1918	1925	1937	1966
	1977	1991	2015	2059	2068	2077	2098	2128	2138	2149	2156	2161	2168	2250	2271
	2278	2285	2290	2299	2415	2451	2457	2463	2469	2476	2491	2509	2517	2604	2629
	2660	2664	2673	2677	2811	2813	2834	2836							
ESCAPE	1#	767#													
EXIF	1#	1731													
EXIFB	1#														
GETPRI	1#	767#													
GETSWR	1#	767#	1042#												
IF	1#	1435	1450	1466	1485	1507	1522	1538	1555	1576	1591	1607	1624	1689	1709
	1714	1774	1813	1860	1875	1893	1921	1933	1980	1987	2073	2145	2261	2264	2267
	2317	2342	2360	2428	2454	2460	2466	2473							
IFB	1#	2420	2513												
IFCOND	1#														
IF.ERR	1#														
IF.NO.	1#														
INCR	1#	2116													
INCRU	1#														
INLINE	1#														
LEAVE	1#														
LET	1#	1447	1463	1479	1519	1535	1551	1588	1604	1620	1652	1657	1659	1667	1684
	1694	1712	1720	1727	1760	1798	1810	1841	1851	1856	1912	1969	1973	2002	2005
	2009	2043	2046	2053	2063	2086	2089	2092	2132	2142	2207	2210	2212	2216	2219
	2222	2224	2234	2237	2257	2259	2323	2328	2339	2349	2351	2353	2358	2363	2369
	2403	2406	2409	2418	2424	2431	2485	2495	2500	2503					
LOCAL	1#														
LOOP	1#	1654													
LPADR	634#	1431	1444	1459	1475	1503	1516	1531	1547	1572	1585	1600	1616	1756	1769
	1793	1834	1872	1888	1904	1960	1998	2040	2082	2399	2481				
MSG	1068#	1070	1100#	1102	1365#	1367	1416#	1418	1563#	1565	1641#	1643	1743#	1745	1942#
	1944	2019#	2021	2380#	2382	2566#	2568	2719#	2721						
MULT	1#	767#													
NEWTST	1#	767#	1068	1100	1236	1288	1365	1416	1493	1563	1641	1743	1782	1822	1942
	2019	2104	2176	2380	2526	2566	2719								
NOLOCA	1#														

POP	1#	767#	3320	3321	3619	3620	3801								
PUSH	1#	767#	3301	3307	3580	3582	3603	3760							
REPEAT	1#														
REPORT	1#	634#	767#												
RETURN	1#														
ROUTIN	1#														
SAVR14	1#														
SCOPE	662#	1075	1113	1239	1291	1372	1421	1496	1568	1646	1748	1785	1825	1950	2025
	2107	2179	2387	2529	2572	2731	3266								
SELECT	1#														
SETPRI	1#	767#	1500	1971	2050	2100	2205	2581	2647	2651	2743	2790	2796		
SETTRA	3915#	3924	3925	3926	3927	3929	3931	3932	3933	3934					
SETUP	1#	767#	991												
SKIP	1#	767#	1080	1096	1197	1227	1244	1284	1428	1635	1651	1674	1705	1753	1755
	1767	1790	1792	1805	1830	1832	1848	1867	1884	1900	1919	1929	1967	1978	2030
	2032	2060	2069	2112	2114	2129	2139	2157	2162	2304	2416	2492	2510	2520	2679
SLASH	1#	767#													
SPACE	767#														
STARS	1#	767#	793	812	823	825	832	845	886	889	970	972	1068	1074	1100
	1112	1236	1238	1248	1250	1265	1267	1288	1290	1365	1371	1411	1413	1416	1420
	1493	1495	1563	1567	1641	1645	1743	1747	1782	1784	1822	1824	1942	1949	2019
	2024	2104	2106	2176	2178	2313	2315	2334	2336	2380	2386	2526	2528	2566	2571
	2683	2685	2719	2730	2846	2848	2993	2997	3103	3105	3153	3157	3258	3297	3313
	3342	3421	3424	3500	3529	3575	3632	3686	3750	3817	3894				
STRUCT	1#	634#													
SWRSU	1#	767#	1014#												
TRMTRP	3915#														
TYPBIN	1#	767#													
TYPDEC	1#	767#	2549	3278											
TYPNAM	1#	767#	1035												
TYPNUM	1#	767#													
TYPOCS	1#	767#	3205												
TYPOCT	1#	767#	2545	2547	3108	3201	3207	3213	3215	3442					
TYPTXT	1#	767#	2536	2541	2544	2546	2548	3201	3203	3206	3212	3214			
UNTIL	1#														
UNTILB	1#														
WAITMS	634#														
WHILE	1#														
WHILEB	1#														
\$ADDON	1#	1436	1438	1451	1453	1467	1469	1486	1488	1508	1510	1523	1525	1539	1541
	1556	1558	1577	1579	1592	1594	1608	1610	1625	1627	1655	1656	1690	1692	1701
	1710	1712	1715	1717	1727	1735	1775	1777	1814	1816	1861	1863	1876	1878	1894
	1896	1922	1924	1934	1936	1981	1983	1988	1990	2074	2076	2117	2119	2120	2124
	2146	2148	2262	2264	2265	2267	2268	2270	2275	2277	2282	2284	2289	2298	2318
	2322	2343	2348	2358	2361	2363	2368	2421	2423	2429	2431	2455	2457	2461	2463
	2467	2469	2474	2476	2514	2516									
\$AND	1#	2318													
\$BRANC	1#	1437	1452	1468	1487	1509	1524	1540	1557	1578	1593	1609	1626	1691	1699
	1711	1716	1725	1733	1735	1776	1815	1862	1877	1895	1923	1935	1982	1989	2075
	2118	2123	2147	2173	2263	2266	2269	2273	2276	2280	2283	2287	2296	2319	2321
	2344	2346	2356	2362	2366	2422	2430	2456	2462	2468	2475	2515			
\$BRCOND	1#	1732	2122	2343											
\$CALL	1#														
\$CHECK	1#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625	1690	1710
	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074	2146	2262	2265	2268
	2275	2282	2318	2343	2361	2421	2429	2455	2461	2467	2474	2514			

SCHK1	1#	1653	1658	1660	1668	1685	1695	1721	1728	1761	1799	1811	1842	1852	1857
	1913	2006	2010	2044	2054	2090	2093	2117	2120	2133	2143	2208	2211	2213	2217
	2219	2223	2225	2324	2340	2350	2404	2410	2419	2501	2504				
SCKDP2	1#	1448	1464	1480	1520	1536	1552	1589	1605	1621	1713	1970	1974	2003	2047
	2064	2087	2235	2238	2258	2260	2329	2352	2354	2359	2364	2370	2407	2425	2432
	2486	2496													
SCKR6	1#														
SCMND	1#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625	1690	1710
	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074	2146	2262	2265	2268
	2275	2282	2318	2320	2343	2345	2361	2421	2429	2455	2461	2467	2474	2514	
SCOMPA	1#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625	1690	1710
	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074	2117	2146	2262	2265
	2268	2275	2282	2313	2343	2361	2421	2429	2455	2461	2467	2474	2514		
SCOUNT	1#														
SDO	1#														
SELSE	1#	2275	2282												
SERRMS	1#														
SEXIFA	1#														
SEXIFO	1#														
SEXIF2	1#														
SEXIF3	1#														
SGENBR	1#	1437	1452	1468	1487	1509	1524	1540	1557	1578	1593	1609	1626	1691	1699
	1711	1716	1725	1733	1735	1776	1815	1862	1877	1895	1923	1935	1982	1989	2075
	2118	2123	2147	2173	2263	2266	2269	2273	2276	2280	2283	2287	2296	2319	2321
	2344	2346	2356	2362	2366	2422	2430	2456	2462	2468	2475	2515			
SGENTA	1#	1441	1456	1472	1491	1513	1528	1544	1561	1582	1597	1613	1630	1655	1700
	1708	1723	1726	1730	1736	1780	1820	1869	1886	1902	1931	1940	1993	1995	2079
	2119	2121	2151	2174	2274	2281	2288	2292	2293	2294	2297	2301	2303	2326	2347
	2357	2367	2372	2374	2427	2434	2459	2465	2472	2480	2519				
SIF	1#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625	1690	1710
	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074	2146	2262	2265	2268
	2275	2282	2318	2343	2361	2421	2429	2455	2461	2467	2474	2514			
SIFCOD	1#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625	1690	1710
	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074	2146	2262	2265	2268
	2275	2282	2318	2320	2345	2361	2421	2429	2455	2461	2467	2474	2514		
SIFCON	1#														
SIFOPR	1#	1437	1452	1468	1487	1509	1524	1540	1557	1578	1593	1609	1626	1691	1711
	1716	1776	1815	1862	1877	1895	1923	1935	1982	1989	2075	2147	2263	2266	2269
	2276	2283	2319	2321	2346	2362	2422	2430	2456	2462	2468	2475	2515		
SLET	1#	1448	1464	1480	1520	1536	1552	1589	1605	1621	1653	1658	1660	1668	1685
	1695	1713	1721	1728	1761	1799	1811	1842	1852	1857	1913	1970	1974	2003	2006
	2010	2044	2047	2054	2064	2087	2090	2093	2133	2143	2208	2211	2213	2217	2219
	2223	2225	2235	2238	2258	2260	2324	2329	2340	2350	2352	2354	2359	2364	2370
	2404	2407	2410	2419	2425	2432	2486	2496	2501	2504					
SLPCNT	1#	2117													
SOPADD	1#	1713	2120	2354	2359										
SOPAND	1#														
SOPCD1	1#	1448	1464	1480	1520	1536	1552	1589	1605	1621	1713	1970	1974	2003	2047
	2064	2087	2120	2235	2238	2258	2260	2329	2352	2354	2359	2364	2370	2407	2425
	2432	2486	2496												
SOPCD2	1#														
SOPCOD	1#	1448	1464	1480	1520	1536	1552	1589	1605	1621	1713	1970	1974	2003	2047
	2064	2087	2120	2235	2238	2258	2260	2329	2352	2354	2359	2364	2370	2407	2425
	2432	2486	2496												
SOPCOM	1#														
SOPDEF	1#	1436	1437	1448	1451	1452	1464	1467	1468	1480	1486	1487	1508	1509	1520

CVMXAAO MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 111
 CVMXAA.P11 30-MAY-79 15:57 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0107

	1523	1524	1536	1539	1540	1552	1556	1557	1577	1578	1589	1592	1593	1605	1608
	1609	1621	1625	1626	1653	1658	1660	1668	1685	1690	1691	1695	1699	1710	1711
	1713	1715	1716	1721	1725	1728	1732	1733	1735	1761	1775	1776	1799	1811	1814
	1815	1842	1852	1857	1861	1862	1876	1877	1894	1895	1913	1922	1923	1934	1935
	1970	1974	1981	1982	1988	1989	2003	2006	2010	2044	2047	2054	2064	2074	2075
	2087	2090	2093	2117	2118	2120	2122	2123	2133	2143	2146	2147	2173	2208	2211
	2213	2217	2219	2223	2225	2235	2238	2258	2260	2262	2263	2265	2266	2268	2269
	2273	2275	2276	2280	2282	2283	2287	2296	2318	2319	2320	2321	2324	2329	2340
	2343	2344	2345	2346	2350	2352	2354	2356	2359	2361	2362	2364	2366	2370	2404
	2407	2410	2419	2421	2422	2425	2429	2430	2432	2455	2456	2461	2462	2467	2468
	2474	2475	2486	2496	2501	2504	2514	2515							
\$OPEQU	1#														
\$OPNAN	1#														
\$OPNEG	1#														
\$OPNOR	1#														
\$OPNOT	1#	1464	1536	1605	1970	2003	2047	2087	2258	2260	2329	2352	2364	2496	
\$OPOR	1#	1448	1480	1520	1552	1589	1621	1974	2064	2235	2238	2370	2407	2425	2432
	2486														
\$OPROT	1#														
\$OPRO	1#	1653	1658	1660	1668	1685	1695	1721	1728	1761	1799	1811	1842	1852	1857
	1913	2006	2010	2044	2054	2090	2093	2117	2133	2143	2208	2211	2213	2217	2219
	2223	2225	2324	2340	2350	2404	2410	2419	2501	2504					
\$OPR1	1#	2120													
\$OPR2	1#	1448	1464	1480	1520	1536	1552	1589	1605	1621	1713	1970	1974	2003	2047
	2064	2087	2235	2238	2258	2260	2329	2352	2354	2359	2364	2370	2407	2425	2432
	2486	2496													
\$OPSHF	1#														
\$OPSUB	1#														
\$OPSWB	1#														
\$OPXOR	1#														
\$OR	1#	2343													
\$PUT	1#														
\$STRUC	1#														
\$SUBON	1#	1441	1456	1472	1491	1513	1528	1544	1561	1582	1597	1613	1630	1700	1708
	1723	1726	1730	1735	1736	1780	1820	1869	1886	1902	1931	1940	1993	1995	2079
	2121	2124	2151	2173	2174	2274	2281	2288	2292	2293	2294	2297	2301	2303	2326
	2357	2367	2372	2374	2427	2434	2459	2465	2472	2480	2519				
\$THEN	1#	1436	1451	1467	1486	1508	1523	1539	1556	1577	1592	1608	1625	1690	1710
	1715	1775	1814	1861	1876	1894	1922	1934	1981	1988	2074	2146	2262	2265	2268
	2275	2282	2320	2345	2361	2421	2429	2455	2461	2467	2474	2514			
\$TILA	1#														
\$TILO	1#														
\$UNT12	1#														
\$UNT13	1#														
\$WHILE	1#														
\$SCMRE	843#														
\$SCMTM	843#														
\$SDEFA	1#														
\$SENDS	1#														
\$SERRO	1#														
\$SESCA	1#	767#													
\$SGEN	1#	1441	1456	1472	1491	1513	1528	1544	1561	1582	1597	1613	1630	1655	1700
	1708	1723	1726	1730	1736	1780	1820	1869	1886	1902	1931	1940	1993	1995	2079
	2119	2121	2151	2174	2274	2281	2288	2292	2293	2294	2297	2301	2303	2326	2347
	2357	2367	2372	2374	2427	2434	2459	2465	2472	2480	2519				
\$SGETS	1#	1441	1456	1472	1491	1513	1528	1544	1561	1582	1597	1613	1630	1699	1700

	1708	1723	1725	1726	1730	1732	1735	1736	1780	1820	1869	1886	1902	1931	1940
	1993	1995	2079	2121	2124	2151	2173	2174	2273	2274	2277	2280	2281	2284	2287
	2288	2292	2293	2294	2296	2297	2301	2303	2326	2356	2357	2366	2367	2372	2374
	2427	2434	2459	2465	2472	2480	2519								
SSGETT	1#	1699	1725	1732	2273	2277	2280	2284	2287	2296	2356	2366			
SSLPLN	1#	2120													
SSNEWI	1#	767#	1068	1100	1236	1288	1365	1416	1493	1563	1641	1743	1782	1822	1942
	2019	2104	2176	2380	2526	2566	2719								
SSPOP	1#	1441	1456	1472	1491	1513	1528	1544	1561	1582	1597	1613	1630	1700	1708
	1723	1726	1730	1735	1736	1780	1820	1869	1886	1902	1931	1940	1993	1995	2079
	2121	2124	2151	2173	2174	2274	2281	2288	2292	2293	2294	2297	2301	2303	2326
	2357	2367	2372	2374	2427	2434	2459	2465	2472	2480	2519				
SSPUSH	1#	1436	1438	1451	1453	1467	1469	1486	1488	1508	1510	1523	1525	1539	1541
	1556	1558	1577	1579	1592	1594	1608	1610	1625	1627	1655	1656	1690	1692	1701
	1710	1712	1715	1717	1727	1735	1775	1777	1814	1816	1861	1863	1876	1878	1894
	1896	1922	1924	1934	1936	1981	1983	1988	1990	2074	2076	2117	2119	2120	2124
	2146	2148	2262	2264	2265	2267	2268	2270	2275	2277	2282	2284	2289	2298	2318
	2322	2343	2348	2358	2361	2363	2368	2421	2423	2429	2431	2455	2457	2461	2463
	2467	2469	2474	2476	2514	2516									
SSSELE	1#														
SSSET	3915#	3924	3925	3926	3927	3929	3931	3932	3933	3934					
SSSETM	1030#														
SSSETS	1#	1436	1438	1451	1453	1467	1469	1486	1488	1508	1510	1523	1525	1539	1541
	1556	1558	1577	1579	1592	1594	1608	1610	1625	1627	1655	1656	1690	1692	1701
	1710	1712	1715	1717	1727	1735	1775	1777	1814	1816	1861	1863	1876	1878	1894
	1896	1922	1924	1934	1936	1981	1983	1988	1990	2074	2076	2117	2119	2120	2124
	2146	2148	2262	2264	2265	2267	2268	2270	2275	2277	2282	2284	2289	2298	2318
	2322	2343	2348	2358	2361	2363	2368	2421	2423	2429	2431	2455	2457	2461	2463
	2467	2469	2474	2476	2514	2516									
SSSETT	1#	2277	2284												
SSSKIP	1#	767#	1080	1096	1198	1227	1244	1284	1428	1635	1651	1674	1705	1753	1755
	1767	1790	1792	1805	1830	1832	1848	1867	1884	1900					

CVMXAA0 MXV11A DIAG MACY11 30G(1063) 30-MAY-79 15:58 PAGE 113
 CVMXAA.P11 30-MAY-79 15:57 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0109

.SRDOC	1#		
.SREAD	1#	634#	3419
.SR2AZ	1#		
.SSAVE	1#		
.SSB2D	1#		
.SSB2O	1#		
.SSCOP	1#	634#	3684
.SSIZE	1#		
.SSUPR	1#		
.STRAP	1#	634#	3892
.STYPB	1#		
.STYPD	1#	634#	3748
.STYPE	1#	634#	3340
.STYPO	1#	634#	3815
.S4OCA	1#		
.1170	1#		

. ABS. 016446 000 OVR RW REL LCL ;

ERRORS DETECTED: 0

Z:CVMXAA,Z:CVMXAA/CRF/SOL=CVMXAA.SML,SYSMAC.SML,CVMXAA.P11
 RUN-TIME: 44 45 3 SECONDS
 RUN-TIME RATIO: 172/93-1.8
 CORE USED: 42K (83 PAGES)