

LSI-11

SUITCASE TEST
MD-11-DWQAA-A

EP-DWQAA-A-DL-A

OCT 1976

COPYRIGHT ©1976

digital

FICHE 1 OF 1

Made in U.S.A.

This microfiche card contains a grid of 40 frames of test data, arranged in 8 rows and 5 columns. Each frame displays a different set of test results, including various data tables, graphs, and status indicators. The data is presented in a structured, tabular format, typical of digital test equipment output. The frames are separated by small gaps, and the overall layout is consistent across the card.

100
101
102
103
104

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DWGAA-A-D
PRODUCT NAME: LSI-11 SUITCASE 11M03-AA SYSTEM
DATE RELEASED: 21 APRIL 1976
MAINTAINER: DIAGNOSTIC GROUP

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

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

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

COPYRIGHT (C) 1976 DIGITAL EQUIPMENT CORPORATION

PROGRAM HISTORY

PRODUCT CODE: MAINDEC-11-DWQAA-A-D
PRODUCT NAME: LSI-11 SUITCASE 11W03-AA SYSTEM
DATE CREATED: DECEMBER 1975
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: JOHN EGOLF

COPYRIGHT (C) DECEMBER 1975
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS 01754

1. ABSTRACT

MD-11-DW00A-A IS A DIAGNOSTIC WHICH WAS WRITTEN TO TEST THE MS911 MODULE. THIS PROGRAM VERIFIES THE DATA WITHIN THE ROM, VERIFIES FUNCTIONS OF THE 600HZ CLOCK, AND CHECKS THE BUS BUFFER BY MEANS OF THE TEST BOX PROVIDED WITH THE 11MD3-AA SYSTEM. THIS PROGRAM WAS WRITTEN EXPRESSLY FOR THE 11MD3-AA SYSTEM AND TAKES ADVANTAGE OF ALL UNITS WITHIN THE SYSTEM. THIS PROGRAM MUST BE RUN AFTER ALL CPU TESTS, MEMORY TESTS, AND DLV11 TESTS.

2. REQUIREMENTS**2.1 EQUIPMENT**

11MD3-AA SYSTEM, WHICH CONSISTS OF:

- A. KD11-F LSI-11 PROCESSOR (WITH 4K OF MEMORY)
- B. 12K MOS RAM MEMORY
- C. 2 DLV11 SERIAL LINE UNITS
- D. SPECIAL MODULE
 - MS911 - BUS BUFFER
 - BOOTSTRAP ROM
 - 600HZ CLOCK
- E. H780 POWER SUPPLY
- F. ALUMINUM CASE 17 X 21 X 7.5"
- G. SPECIAL OPERATORS PANEL
- H. POWER CORD
- I. CABLE, EXTERNAL BUS -6FT.
- J. H9270 BACKPLANE ASSEMBLY
- K. TEST BOX

2.2 STORAGE

PROGRAM WILL LOAD IN 4K BUT RESERVES THE RIGHT TO USE 16K (ADDRESS 000000 - 077777).

3. LOADING PROCEDURE

THIS PROGRAM MAY BE LOADED LIKE ANY OTHER PROGRAM IS LOADED. THERE ARE NO SPECIAL LOADING PROCEDURES.

4. STARTING PROCEDURE

IF DESIRED, AFTER PROGRAM IS LOADED, SOFTWARE SWITCH REGISTER MAY BE SET AS PER SECTION 4.1 (USING LSI-ODT) BEFORE START OF EXECUTION OF PROGRAM. AS DISCUSSED IN SECTION 4.2, THERE ARE TWO STARTING ADDRESSES:

SA 200 NORMAL START
 SA 210 NORMAL START BUT THE OPPORTUNITY TO "GOTO"
 A SELECTED TEST IS GIVEN TO THE USER.

4.1 CONTROL SWITCH SETTINGS

THE SOFTWARE SWITCH REGISTER MAY BE ALTERED BY USING LSI-ODT AND MODIFYING ADDRESS 000176. AN EASIER WAY TO CHANGE THE SWITCH REGISTER IS THAT AFTER THE DIAGNOSTIC IS STARTED AND WHILE IT IS RUNNING THE USER MAY TYPE "CONTROL G" (<+G>) AND IF THE CPU PRIORITY ALLOWS (AND A RESET INSTRUCTION ISN'T BEING EXECUTED) THE PROGRAM WILL PRINT OUT:

(SMR)=/000000/=/

THE USER MAY THEN HIT <CR> CARRIAGE RETURN OR TYPE (IN OCTAL) NEW SETTINGS. SWITCHES ARE:

SM15	SET: HALT ON ERROR
SM14	SET: LOOP ON TEST
SM13	SET: INHIBIT ERROR PRINT-OUT.
SM12	SET: ESCAPE TO NEXT TEST ON ERROR.
SM11	SET: INHIBIT ITERATIONS
SM10	SET: BELL ON ERROR
SM09	SET: LOOP ON ERROR
SM08	SET: CONFIDENCE; -PRINT TEST # AT 1ST END OF EACH TEST.
SM07	RESERVED
SM06	RESERVED
SM05	RESERVED
SM04	RESERVED
SM03	RESERVED
SM02	RESERVED
SM01	RESERVED
SM00	RESERVED

4.1.2 SWITCH REGISTER RESTRICTIONS

NONE. FOR THE EXCEPTION OF SM12 ALL SWITCH REGISTER OPTIONS ARE "SYSMAC.SML" COMPATABLE.

4.2 STARTING ADDRESS

AS MENTIONED BEFORE THERE ARE TWO STARTING ADDRESSES:

SA 200 NORMAL OPERATION
 SA 210 NORMAL OPERATION WITH OPTION TO SPECIFY
 STARTING TEST NUMBER.

5. OPERATING PROCEDURE

PROGRAM SHOULD BE STARTED AS PER SECTION 4.2. ON INITIAL START PROGRAM WILL PRINT OUT:

MD-11-DWQAA-A
LSI-11 SUIT-CASE ROM, 600 HZ, AND BUS TESTS

AND BEGIN EXECUTION (OR ASK FOR TEST NO. IF SA=210).

5.2 PROGRAM AND/OR OPERATOR ACTION

SWR SHOULD BE SET TO "100000" HALT ON ERROR. AFTER ERROR MODIFY SWR TO "060000" LOOP ON TEST AND INHIBIT PRINTOUT AND START AT ADDRESS 210 SPECIFYING FAILING TEST NUMBER.

6. ERRORS

ALL CONTROLABLE ERRORS WILL PRINT:

1) ERRPC	ERROR PC (IN OCTAL)
2) STSTN	TEST NUMBER (IN OCTAL)
3) SPASS	PASSES MADE (IN DECIMAL)
4) SERTLL	ERRORS MADE (IN DECIMAL)

ALL PROCESSOR REGISTERS ARE SAVED IN MEMORY LOCATIONS ON ERRORS.

SAVR0	LOC: 1420
SAVR1	LOC: 1422
SAVR2	LOC: 1424
SAVR3	LOC: 1426
SAVR4	LOC: 1430
SAVR5	LOC: 1432

ADDITIONAL INFORMATION WILL BE PRINTED AS NEEDED.

7. RESTRICTIONS

7.1 HARDWARE RESTRICTIONS

SYSTEM MUST BE CONFIGURED #EXACTLY# AS OUTLINED FOR
11M03-AA SYSTEM.

8. MISCELLANEOUS

8.1 CONTROL "G" <↑G>

THIS WILL ENABLE USER TO CHANGE SWITCH REGISTER (PROVIDED
PROGRAM IS RUNNING).

8.2 CONTROL "T" <↑T>

THIS WILL ENABLE USER TO "GOTO" SELECTED TEST. (CPU MUST
NOT BE DOING A RESET.)

8.3 PASS COMPLETE

THIS IS A "SYSMAC.SHL" COMPATIBLE END PASS ROUTINE. AT
PASS COMPLETE THE FOLLOWING WILL BE PRINTED:

END PASS # 1
END PASS # 2

PASS COUNT IS IN DECIMAL.

8.4 EXECUTION TIME

WITH NO ERRORS AND SW11=0 PASS TIME IS <8 MINS.

WITH NO ERRORS AND SW11=1 PASS TIME IS <2 MINS.

4715	OPERATIONAL SWITCH SETTINGS
4716	BASIC DEFINITIONS
4717	TRAP CATCHER
(1)	STARTING ADDRESS(ES)
4720	COMMON TAGS
(1)	ERROR POINTER TABLE
4935	T1 ROM "BREPLY" TEST
4961	T2 ROM WRITE TEST
4987	T3 ROM DATA TEST
5007	T4 600 HZ ADDRESS TEST
5038	T5 600 HZ "EVENT ENABLE" R/W TEST
5091	T6 600HZ INTERRUPT TEST
5138	T7 UNEXPECTED "BREPLY" TEST
5178	T10 "DEVICE 'A'" ADDRESS TEST
5217	T11 R/W TEST OF BIT 0 IN DEVICE 'A'
(4)	T12 R/W TEST OF BIT 1 IN DEVICE 'A'
(4)	T13 R/W TEST OF BIT 2 IN DEVICE 'A'
(4)	T14 R/W TEST OF BIT 3 IN DEVICE 'A'
(4)	T15 R/W TEST OF BIT 4 IN DEVICE 'A'
(4)	T16 R/W TEST OF BIT 5 IN DEVICE 'A'
(4)	T17 R/W TEST OF BIT 6 IN DEVICE 'A'
(4)	T20 R/W TEST OF BIT 7 IN DEVICE 'A'
(4)	T21 R/W TEST OF BIT 8 IN DEVICE 'A'
(4)	T22 R/W TEST OF BIT 9 IN DEVICE 'A'
(4)	T23 R/W TEST OF BIT 10 IN DEVICE 'A'
(4)	T24 R/W TEST OF BIT 11 IN DEVICE 'A'
(4)	T25 R/W TEST OF BIT 12 IN DEVICE 'A'
(4)	T26 R/W TEST OF BIT 13 IN DEVICE 'A'
(4)	T27 R/W TEST OF BIT 14 IN DEVICE 'A'
(4)	T30 R/W TEST OF BIT 15 IN DEVICE 'A'
5225	T31 BINARY COUNT TEST
5242	T32 1'S AND 0'S TEST
5264	T33 DEVICE 'A' BYTE OPERATIONS
5303	T34 BUS INITIALIZE TEST FOR DEVICE 'A'
5320	T35 "DEVICE 'B'" ADDRESS TEST (SEL 0)
5361	T36 R/W TEST OF BIT 6 IN DEVICE 'B' (SEL 0)
(4)	T37 R/W TEST OF BIT 7 IN DEVICE 'B' (SEL 0)
5372	T40 DEVICE 'B' INTERRUPT TEST (SEL 0)
5413	T41 "DEVICE 'B'" ADDRESS TEST (SEL 2)
5457	T42 R/W TEST OF BIT 6 IN DEVICE 'B' (SEL 2)
(4)	T43 R/W TEST OF BIT 7 IN DEVICE 'B' (SEL 2)
5468	T44 DEVICE 'B' INTERRUPT TEST (SEL 2)
5510	T45 TEST OF 'EVENT' AT DEVICE 'B'
5545	T46 DLV11 CHARACTER TEST
5624	T47 DLV11 INTERRUPT TEST
5678	T50 MINI MEMORY TEST
5758	END OF PASS ROUTINE
5759	SCOPE HANDLER ROUTINE
5760	ERROR HANDLER ROUTINE
5761	ERROR MESSAGE TIMEOUT ROUTINE
5762	TYPE ROUTINE
5763	BINARY TO OCTAL (ASCII) AND TYPE
5764	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
5765	TTY INPUT ROUTINE
5766	READ AN OCTAL NUMBER FROM THE TTY
5768	TRAP DECODER

MAINDEC-11-11W03A-A MACY11 27(732) 24-AUG-76 16:05
DWGAA.P11 TABLE OF CONTENTS

I01

SEQ 0008

(3) TRAP TABLE

(1)	000000	PR0 =	0	PRIORITY	LEVEL	0
(1)	000040	PR1 =	40	PRIORITY	LEVEL	1
(1)	000100	PR2 =	100	PRIORITY	LEVEL	2
(1)	000140	PR3 =	140	PRIORITY	LEVEL	3
(1)	000200	PR4 =	200	PRIORITY	LEVEL	4
(1)	000240	PR5 =	240	PRIORITY	LEVEL	5
(1)	000300	PR6 =	300	PRIORITY	LEVEL	6
(1)	000340	PR7 =	340	PRIORITY	LEVEL	7

:"SWITCH REGISTER" SWITCH DEFINITIONS

(1)	100000	SW15 =	100000
(1)	040000	SW14 =	40000
(1)	020000	SW13 =	20000
(1)	010000	SW12 =	10000
(1)	004000	SW11 =	4000
(1)	002000	SW10 =	2000
(1)	001000	SW09 =	1000
(1)	000400	SW08 =	400
(1)	000200	SW07 =	200
(1)	000100	SW06 =	100
(1)	000040	SW05 =	40
(1)	000020	SW04 =	20
(1)	000010	SW03 =	10
(1)	000004	SW02 =	4
(1)	000002	SW01 =	2
(1)	000001	SW00 =	1
(1)		.EQUIV	SW09, SW9
(1)		.EQUIV	SW08, SW8
(1)		.EQUIV	SW07, SW7
(1)		.EQUIV	SW06, SW6
(1)		.EQUIV	SW05, SW5
(1)		.EQUIV	SW04, SW4
(1)		.EQUIV	SW03, SW3
(1)		.EQUIV	SW02, SW2
(1)		.EQUIV	SW01, SW1
(1)		.EQUIV	SW00, SW0

:"DATA BIT DEFINITIONS (BIT00 TO BIT15)

(1)	100000	BIT15 =	100000
(1)	040000	BIT14 =	40000
(1)	020000	BIT13 =	20000
(1)	010000	BIT12 =	10000
(1)	004000	BIT11 =	4000
(1)	002000	BIT10 =	2000
(1)	001000	BIT09 =	1000
(1)	000400	BIT08 =	400
(1)	000200	BIT07 =	200
(1)	000100	BIT06 =	100
(1)	000040	BIT05 =	40
(1)	000020	BIT04 =	20
(1)	000010	BIT03 =	10
(1)	000004	BIT02 =	4
(1)	000002	BIT01 =	2
(1)	000001	BIT00 =	1
(1)		.EQUIV	BIT09, BIT9
(1)		.EQUIV	BIT08, BIT8

4760	001262	003716	DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL		
4761										
4762										
4763	001264	001741								
4764	001266	003210	EM10	;DEVICE "A" R/W FAILURE.						
4765	001270	003664	DH10	;ERRPC	TSTNO	PASCNT	ERRCNT	DEV.A	WANTED	FOUND
4766	001272	003716	DT2	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD	SAVRS	SAVR4
4767			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL	OCTAL	OCTAL
4768										
4769	001274	001771								
4770	001276	003074	EM11	;NO "BREPLY" FROM DEVICE 'A'.						
4771	001300	003650	DH6	;ERRPC	TSTNO	PASCNT	ERRCNT	DEV.A		
4772	001302	003716	DT1	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD		
4773			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL		
4774										
4775	001304	002026								
4776	001306	003142	EM12	;NO "BREPLY" FROM DEVICE 'B'.						
4777	001310	003650	DH7	;ERRPC	TSTNO	PASCNT	ERRCNT	DEV.B		
4778	001312	003716	DT1	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD		
4779			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL		
4780										
4781	001314	002063								
4782	001316	003276	EM13	;DEVICE "B" R/W FAILURE.						
4783	001320	003664	DH11	;ERRPC	TSTNO	PASCNT	ERRCNT	DEV.B	WANTED	FOUND
4784	001322	003716	DT2	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD	SAVRS	SAVR4
4785			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL	OCTAL	OCTAL
4786										
4787	001324	002113								
4788	001326	003364	EM14	;UNEXPECTED "BREPLY" FROM UNKNOWN DEVICE.						
4789	001330	003650	DH12	;ERRPC	TSTNO	PASCNT	ERRCNT	DEV.		
4790	001332	003716	DT1	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD		
4791			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL		
4792										
4793	001334	002164								
4794	001336	003142	EM15	;NO INTERRUPT FROM DEVICE 'B'.						
4795	001340	003650	DH7	;ERRPC	TSTNO	PASCNT	ERRCNT	DEV.B		
4796	001342	003716	DT1	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD		
4797			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL		
4798										
4799	001344	002220								
4800	001346	003433	EM16	;GENERAL ERROR!						
4801	001350	003704	DH13	;ERRPC	TSTNO	PASCNT	ERRCNT			
4802	001352	003716	DT3	;SERRPC	MSKTST	SPASS	SERTTL			
4803			DF1	;OCTAL	OCTAL	DEC	DEC			
4804										
4805	001354	002237								
4806	001356	003474	EM17	;DLV11 DATA ERROR.						
4807	001360	003664	DH14	;ERRPC	TSTNO	PASCNT	ERRCNT	DLV11	WANTED	FOUND
4808	001362	003716	DT2	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD	SAVRS	SAVR4
4809			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL	OCTAL	OCTAL
4810										
4811	001364	002261								
4812	001366	003562	EM20	;MEMORY ERROR.						
4813	001370	003664	DH15	;ERRPC	TSTNO	PASCNT	ERRCNT	MEMORY	WANTED	FOUND
4814	001372	003716	DT2	;SERRPC	MSKTST	SPASS	SERTTL	SAVRD	SAVRS	SAVR4
4815			DF1	;OCTAL	OCTAL	DEC	DEC	OCTAL	OCTAL	OCTAL

4817 001374 100000
4818
4819
4820
4821 001376 177546
4822 001400 000100
4823 001402 000102
4824 001404 172524
4825 001406 165250
4826 001410 000340
4827 001412 000342
4828 001414 000344
4829 001416 000346
4830 001420 000000
4831 001422 000000
4832 001424 000000
4833 001426 000000
4834 001428 000000
4835 001430 000000
4836 001432 000000
4837 001434 000000
4838 001436 000000
4839 001440 000000

MEMSIZ: 100000
CLKCSR: 177546
CLKVEC: 100
CLKPTY: 102
DEV.A: 172524
DEV.B: 165250
B.AVEC: 000340
B.APTY: 000342
B.BVEC: 000344
B.BPTY: 000346
SAVR0: 000000
SAVR1: 000000
SAVR2: 000000
SAVR3: 000000
SAVR4: 000000
SAVR5: 000000
MSKTST: 000000
TSTFLG: 000000
XISCP: 000000

16K 100000
12K 060000
8K 040000
4K 020000
;CLOCK CONTROL REGISTER

4840 001442 047516 021040 051102
4841 001500 047522 020115 040504
4842 001533 116 020117 041042
4843 001572 046103 041517 020113
4844 001624 047125 054105 042520
4845 001651 116 020117 047111
4846 001701 103 047514 045503
4847 001741 104 053105 041511
4848 001771 116 020117 041042
4849 002026 047516 021040 051102
4850 002063 104 053105 041511
4851 002113 125 042516 050130
4852 002164 047516 044440 052116
4853 002220 042507 042516 040522
4854 002237 104 053114 030461
4855 002261 115 046505 051117
4856 002277 015 044412 041516
4857 002336 015 041412 040510
4858 002403 015 051412 055111
4859 002440 005015 046442 046505
4860 002520 051105 050122 020103
4861 002574 051105 050122 020103
4862 002662 051105 050122 020103
4863 002731 105 051122 041520
4864 003017 105 051122 041520
4865 003074 051105 050122 020103
4866 003142 051105 050122 020103
4867 003210 051105 050122 020103
4868 003276 051105 050122 020103
4869 003364 051105 050122 020103
4870 003433 105 051122 041520
4871 003474 051105 050122 020103

EM1: .ASCIZ /NO "BREPLY" FROM ROM ADDRESS./
EM2: .ASCIZ /ROM DATA COMPARISON ERROR./
EM3: .ASCIZ /NO "BREPLY" FROM 600 HZ CLOCK./
EM4: .ASCIZ 'CLOCK R/W OF BIT6 FAILED.'
EM5: .ASCIZ /UNEXPECTED INTERRUPT./
EM6: .ASCIZ /NO INTERRUPT FROM CLOCK./
EM7: .ASCIZ /CLOCK INTERRUPTS NOT CONSISTANT./
EM10: .ASCIZ 'DEVICE "A" R/W FAILURE.'
EM11: .ASCIZ /NO "BREPLY" FROM DEVICE 'A':./
EM12: .ASCIZ /NO "BREPLY" FROM DEVICE 'B':./
EM13: .ASCIZ 'DEVICE "B" R/W FAILURE.'
EM14: .ASCIZ /UNEXPECTED "BREPLY" FROM UNKNOWN DEVICE./
EM15: .ASCIZ /NO INTERRUPT FROM DEVICE 'B'./
EM16: .ASCIZ /GENERAL ERROR!/
EM17: .ASCIZ /DLV11 DATA ERROR./
EM20: .ASCIZ /MEMORY ERROR./
EM21: .ASCIZ (<15><12>)/INCORRECT MEMORY SIZE FOUND./
.ASCIZ (<15><12>)/CHANGE LOCATION "MEMSIZ" TO SPECIFIC/
.ASCIZ (<15><12>)/SIZE IN YOUR CONFIGURATION./
.ASCIZ (<15><12>)/"MEMSIZ" SHOULD BE 100000 FOR A 16K SYSTEM./(<15><12><0>)
DH1: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT ROM ADDRESS/
DH2: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT ROMADD WANTED FOUND/
DH3: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT CLKCSR/
DH4: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT CLKCSR WANTED FOUND/
DH5: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT (SP)-TRAP PC/
DH6: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT DEV.A/
DH7: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT DEV.B/
DH10: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT DEV.A WANTED FOUND/
DH11: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT DEV.B WANTED FOUND/
DH12: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT DEVICE/
DH13: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT /
DH14: .ASCIZ /ERRPC TSTNO PASCNT ERRCNT DLV11 WANTED FOUND/

Address	Offset	Pointer	Value	Label	Instruction	Comment
4894	003726	005237	001436		SETTST: INC	
4895	003732				START:	
(1)	003732	012706	001100		NOV #SCHTAG,R6	;; FIRST LOCATION TO BE CLEARED
(1)	003736	005026			CLR (R6)+	;; CLEAR MEMORY LOCATION
(1)	003740	022706	001126		CHP #SDDAT,R6	;; DONE?
(1)	003744	001374			BNE .-6	;; LOOP BACK IF NO
(1)	003746	012706	001100		NOV #STACK,SP	;; SETUP THE STACK POINTER
(1)	003752	012737	011270	000020	NOV #SCOPE,#IOTVEC	;; IOT VECTOR FOR SCOPE ROUTINE
(1)	003760	012737	000340	000022	NOV #340,#IOTVEC+2	;; LEVEL 7
(1)	003766	012737	011602	000030	NOV #SENSOR,#ENTVEC	;; ENT VECTOR FOR ERROR ROUTINE
(1)	003774	012737	000340	000032	NOV #340,#ENTVEC+2	;; LEVEL 7
(1)	004002	012737	013454	000034	NOV #TRAP,#TRAPVEC	;; TRAP VECTOR FOR TRAP CALLS
(1)	004010	012737	000340	000036	NOV #340,#TRAPVEC+2	;; LEVEL 7
(1)	004016	013737	011202	011174	NOV SENDCT,SEOPCT	;; SETUP END-OF-PROGRAM COUNTER
(1)	004024	005037	001160		CLR STIMES	;; INITIALIZE NUMBER OF ITERATIONS
(1)	004030	005037	001162		CLR SESCAPE	;; CLEAR THE ESCAPE ON ERROR ADDRESS
(1)	004034	112737	000001	001115	NOVB #1,SERMAX	;; ALLOW ONE ERROR PER TEST
(1)	004042	012737	004042	001106	NOV #.,SLPADR	;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1)	004050	012737	004050	001110	NOV #.,SLPERR	;; SETUP THE ERROR LOOP ADDRESS
(2)	004056	013746	000004		NOV #84,-(SP)	;; SAVE ERROR VECTOR
(2)	004062	013746	000006		NOV #86,-(SP)	
(2)	004066	012737	004102	000004	NOV #64#,4	;; SET UP TIME OUT VECTOR
(2)	004074	005777	175036		TST #SMR	;; TRY TO REFERENCE HARDWARE SMR
(2)	004100	000407			BR #55	;; BRANCH IF NO TIMEOUT TRAP OCCURS
(2)	004102	012737	000176	001136	NOV #SMREG,SMR	;; POINT TO SOFTWARE SMR
(2)	004110	012737	000174	001140	NOV #DISPREG,DISPLAY	;; POINT TO SOFTWARE DISPLAY REG
(2)	004116	022626			CHP (SP)+,(SP)+	;; RESTORE STACK
(2)	004120	012637	000006		NOV (SP)+,#86	;; RESTORE ERROR VECTOR
(2)	004124	012637	000004		NOV (SP)+,#84	
4896	004130	005227	177777		INC #-1	;; FIRST TIME?
(1)	004134	001055			BNE #66	;; BRANCH IF NO
(1)	004136	022737	011234	000042	CHP #SENDAD,#842	;; ACT-11 AUTO-ACCEPT?
(1)	004144	001451			BEQ #66	;; BRANCH IF NO
(1)	004146	104400	004154		TYPE #67S	;; TYPE ASCIZ STRING
(1)	004152	000446			BR #66	;; GET OVER THE ASCIZ
(1)	004270				;;67S: .ASCIZ	<200>/MD-11-11M03A-A/<15><12>/LSI-11 SUIT-CASE ROM, 600 HZ CLK, AND BUS
4897	004270	012737	000006	000004	NOV #6,4	
4898	004276	005037	000006		CLR #6	
4899	004302	013777	001402	175070	NOV CLKPTY,#CLKVEC	
4900	004310	012777	000002	175064	NOV #2,CLKPTY	
4901	004316	013777	001412	175064	NOV #A,PTY,#B,AVEC	
4902	004324	005077	175062		CLR #B,PTY	
4903	004330	013777	001416	175056	NOV #B,PTY,#B,BVEC	
4904	004336	005077	175054		CLR #B,PTY	
4905	004342	005037	000022		CLR #IOTVEC+2	;; ZERO SCOPE PRIO.
4906	004346	005037	001440		CLR #XSCOP	
4907	004352	012737	014516	000060	NOV #KBSR,#860	;; SET VECTOR
4908	004360	012737	000200	000062	NOV #200,#862	;; SET PRIO
4909	004366	012777	000100	174546	NOV #100,#8TKS	;; SET INTR ENABLE
4910	004374	106427	000000		HTPS #0	;; SET PSM TO 0
4911	004400	005737	001436		TST TSTFLG	;; FLAG SET?
4912	004404	001431			TST1	
4913	004406	005037	001436		CLR TSTFLG	;; ZERO FLAG
4914	004412	104400	014631		TYPE #XTSTN	;; TYPE "TEST NO: "
4915	004416	104407			RDOCT	;; GET NUMBER

```

4916 004420 012600
4917 004422 001773
4918 004424 020037 015140
4919 004430 101370
4920 004432 110037 001102
4921 004436 000241
4922 004440 006100
4923 004442 016037 015012 001106
4924 004450 062737 000002 001106
4925 004456 013737 001106 001110
4926 004464 000177 174416

```

```

MOV (SP)+,RO
BEQ X1$ ;BR IF NO NUMBER INPUT
CMP RO, LASTN ;VALID TEST?
BHI X1$ ;BR IF NO
MOVB RO, STSTN ;LOAD INITIAL TEST NO.
CLC
ROL RO ;MAKE POWER OF 2.
MOV TEST.TABLE(RO), SLPADR
ADD #2, SLPADR ;GET TEST PC.
MOV SLPADR, SLPERR ;GET PAST SCOPE
JMP #SLPADR ;GOTO TEST.

```

```

*****
;TEST 1 ROM "BREPLY" TEST
;TEST TO REFERENCE ALL 256. ADDRESS
;OF THE ROM ONLY MAKING SURE THAT
;THERE IS A "BREPLY" RESPONSE - NO DATA IS
;CHECKED IN THIS TEST.
*****

```

```

(3)
(4)
(4)
(4)
(4)
(3)
(2)
4936 004470 000004
4937 004472 012700 173000
4938 004476 012701 000400
4939 004502 013746 000004
4940 004506 013746 000006
4941 004512 012737 004544 000004
4942 004520 012737 000200 000006
4943 004526 005710
4944 004530 005710
4945 004532 062700 000002
4946 004536 005301
4947 004540 001372
4948 004542 000404
4949 004544 104001
4950 004546 012716 004532
4951 004552 000002
4952 004554 012637 000006
4953 004560 012637 000004

```

```

TST1: SCOPE
MOV #173000, R0 ;SET INITIAL START ADDRESS OF ROM.
MOV #256, R1 ;SET NUMBER OF WORDS TO BE TESTED.
MOV 4, -(SP) ;SAVE LOC 4.
MOV 6, -(SP) ;SAVE LOC 6.
MOV #38, 4 ;SET TIME-OUT TRAP VECTOR.
MOV #200, 6 ;LOAD PRIORITY = 4
1$: TST (R0) ;REFERENCE ROM ADDRESS.
TST (R0) ;DO IT AGAIN.
2$: ADD #2, R0 ;UPDATE ADDRESS.
DEC R1 ;ALL DONE?
BNE 1$ ;BR IF NO
BR 4$ ;CONTINUE TEST
3$: ERROR 1 ;ROM DID NOT ISSUE "BREPLY".
MOV #2$, (SP) ;SET RETURN ADDRESS
RTI ;RETURN
4$: MOV (SP)+, 6 ;RESTORE ADD 6.
MOV (SP)+, 4 ;RESTORE ADD 4.

```

```

*****
;TEST 2 ROM WRITE TEST
;TEST THAT WRITING THE ROM
;PRODUCES A TIME-OUT TRAP.
;
*****

```

```

(3)
(4)
(4)
(4)
(3)
(2)
4962 004564 000004
4963 004566 012700 173000
4964 004572 012701 000400
4965 004576 013746 000004
4966 004602 013746 000006
4967 004606 012737 004632 000004
4968 004614 012737 000200 000006
4969 004622 005020
4970 004624 000240

```

```

TST2: SCOPE
MOV #173000, R0 ;SET ROM ADDRESS
MOV #256, R1 ;DO ALL ADDRESS
MOV 4, -(SP) ;STORE 4
MOV 6, -(SP) ;STORE 6
MOV #2$, 4 ;SET TRAP VECTOR
MOV #200, 6 ;PRIO
1$: CLR (R0)+ ;WRITE ROM WITH ZERO.
NOP

```

4970	004626	104016	
4971	004630	024646	
4972	004632	105301	
4973	004634	001403	
4974	004636	012716	004622
4975	004642	000002	
4976	004644	105427	000000
4977	004650	022626	
4978	004652	012637	000006
4979	004656	012637	000004

```

ERROR 16 ;WRITE ROM AND NO TRAP.
CMP -(SP),-(SP) ;FAKE AN INTERRUPT.
25: DECB R1 ;256 DONE?
BEQ 35 ;BR IF YES
MOV @15,(SP) ;SET RETURN
RTI ;RETURN
35: MTPS @0 ;SET PTY TO 0
CMP (SP)+,(SP)+ ;FAKE RTI
MOV (SP)+,6 ;RESTORE 6
NOV (SP)+,4 ;RESTORE 4

```

4980
4987
(3)
(4)
(4)
(4)
(4)
(3)
(2)

```

*****
;TEST 3 ROM DATA TEST
;TEST TO READ AND COMPARE ALL
;256 ROM ADDRESS TO THE ROM DATA
;MAP IN MEMORY. ALL ADDRESSES ARE
;VERIFIED FOR GOOD DATA.
*****

```

4988	004662	000004	
4989	004664	012700	173000
4990	004670	012701	000400
4991	004674	012702	013516
4992	004700	011004	
4993	004702	011205	
4994	004704	020504	
4995	004706	001401	
4996	004710	104002	
4997	004712	022022	
4998	004714	005301	
4999	004716	001370	

```

TST3: SCOPE
MOV @173000,R0 ;SET START OF ROM ADDRESS.
MOV @256,R1 ;SET NUMBER OF WORDS TO CHECK.
MOV @ROMMAP,R2 ;GET SOFTWARE ADDRESS
15: MOV (R0),R4 ;READ ROM.
MOV (R2),R5 ;READ SOFTWARE IMAGE
CMP R5,R4 ;ARE THEY GOOD?
BEQ +4 ;ROM DATA ERROR?
ERROR 2 ;BAD DATA IN ROMS!!
CMP (R0)+,(R2)+ ;POP POINTERS
DEC R1 ;ALL DONE?
BNE 15 ;BR IF NOT DONE.

```

5006
5007
(3)
(4)
(4)
(4)
(4)
(3)
(2)

```

*****
;TEST 4 600 HZ ADDRESS TEST
;TEST TO VERIFY A "BREPPLY" RESPONSE
;FROM THE 600 HZ CLOCK.
;IT IS ASSUMED THAT THE CLOCK
;IS AT ADDRESS "177546".
*****

```

5008	004720	000004	
5009	004722	013746	000004
5010	004726	013746	000006
5011	004732	012701	000017
5012	004736	013700	001376
5013	004742	012737	004766 000004
5014	004750	012737	000200 000006
5015	004756	005710	
5016	004760	005301	
5017	004762	001375	
5018	004764	000404	
5019	004766	104003	
5020	004770	012716	004760
5021	004774	000002	
5022	004776	012637	000006
5023	005002	012637	000004

```

TST4: SCOPE
MOV 4,-(SP) ;SAVE ADDRESS 4.
MOV 6,-(SP) ;SAVE ADDRESS 6.
MOV @15,R1 ;SET INTERNAL ICOUNT TO 15.
MOV CLKCSR,R0 ;GET CLOCK CSR.
MOV @35,4 ;SET TIME-OUT VECTOR.
MOV @200,6 ;SET PRIO.
15: TST (R0) ;REFERENCE CLOCK CSR.
25: DEC R1 ;ICOUNT = 0?
BNE 15 ;BR IF NO.
BR 45 ;CONT TEST
35: ERROR 3 ;TIME-OUT ERROR.
MOV @25,(SP) ;SET RETURN ADD.
RTI ;RETURN.
45: MOV (SP)+,6 ;RESTORE 6
MOV (SP)+,4 ;RESTORE 4

```



```

5078 005204 022626          55:  CMP      (SP)+,(SP)+
5079 005206 012677 174170    65:  MOV      (SP)+,2CLKPTY  ;RESTORE CLKPTY
5080 005212 012677 174162    MOV      (SP)+,2CLKVEC  ;RESTORE CLKVEC
5081 005216 106427 000000    MTPS    80              ;SET PRIO TO LVL 0

```

```

5082
5090
5091
(3)
(4)
(4)
(4)
(4)
(3)

```

```

;*****
;#TEST 6      600HZ INTERRUPT TEST
;#TEST THAT THE 600HZ CLOCK
;#CAN INTERRUPT AND THAT THE
;#INTERRUPTS ARE WITHIN THE SAME
;#TIME SPAN. 200. INTERRUPTS WILL BE
;#EXECUTED
;*****

```

```

(2)
5092 005222 000004
5093 005224 017746 174150
5094 005226 017746 174146
5095 005228 012777 005310 174136
5096 005230 012777 000200 174132
5097 005232 012704 000310
5098 005234 012705 000002
5099 005236 005001
5100 005238 106427 000000
5101 005240 013700 001376
5102 005242 052777 000100 174076
5103 005300 005201
5104 005302 001376
5105 005304 104006
5106 005306 000423
5107 005310 005305
5108 005312 001414
5109 005314 100014
5110 005316 010102
5111 005320 062702 000010
5112 005324 160302
5113 005326 100001
5114 005330 005402
5115 005332 020227 000270
5116 005336 101401
5117 005340 104007
5118 005342 000401
5119 005344 010103
5120 005346 005001
5121 005350 005304
5122 005352 001401
5123 005354 000002
5124 005356 042777 000100 174012
5125 005364 022626
5126 005366 012677 174010
5127 005372 012677 174002
5128 005376 106427 000000

```

```

TST6:  SCOPE
        MOV      2CLKVEC,-(SP)  ;SAVE CLKVEC
        MOV      2CLKPTY,-(SP) ;SAVE CLKPTY
        MOV      825,2CLKVEC   ;SET INTERRUPT VECTOR
        MOV      8200,2CLKPTY  ;SET PRIO.
        MOV      8200,R4       ;SET ICOUNT
        MOV      82,R5         ;SET COUNT POSITION
        CLR      R1            ;ZERO COUNT UP.
        MTPS    80            ;CLEAR PSM
        MOV      CLKCSR,R0     ;SAVE FOR PRINTOUT
        BIS      8BIT6,2CLKCSR ;SET EVENT ENABLE
15:     INC      R1            ;COUNT TIME
        BNE     ERROR        ;=0?
        BR      55           ;NO INTERRUPT
25:     DEC      R5           ;EXIT TEST
        BEQ    38           ;HOW MANY INTERRUPTS?
        BPL    48           ;BR IF 2ND!
        BPL    48           ;BR IF 1ST!
        MOV     R1,R2        ;MUST BE 3RD OR MORE!
        ADD    810,R2       ;GIVE TOLARENCE OF +10
        SUB    R3,R2        ;GET DIFFERENCE
        BPL    +4           ;SSKIP IF POSITIVE
        NEG    R2           ;MAKE POSITIVE
        CMP    R2,8270     ;GIVE +270 TOLARENCE
        BLOS   ERROR        ;INTERRUPTS OUT OF TOLERANCE
        BR     48           ;CONT TEST
35:     MOV     R1,R3       ;SAVE COUNT
45:     CLR    R1          ;ZERO COUNTER
        DEC    R4          ;ICOUNT =0?
        BEQ   55          ;BR IF YES
        RTI   55          ;EXIT
55:     BIC    8BIT6,2CLKCSR ;CLEAR EVENT ENABLE
        CMP   (SP)+,(SP)+  ;POP INTR OFF STACK
        MOV   (SP)+,2CLKPTY ;RESTORE CLKPTY
        MOV   (SP)+,2CLKVEC ;RESTORE CLKVEC.
        MTPS 80           ;SET PRIO TO LEVEL 0

```

```

5129
5130
(3)
(4)

```

```

;*****
;#TEST 7      UNEXPECTED "BREPLY" TEST.
;#TEST TO SCAN THROUGH ALL ADDRESS

```

```

(4)
(4)
(4)
(4)
(4)
(3)
(2) 005402 000004
(1) 005404 012737 000003 001160
5139 005412 013746 000004
5140 005416 013746 000006
5141 005422 012737 005444 000004
5142 005430 005037 000006
5143 005434 005000
5144 005436 005710 1S:
5145 005440 005720
5146 005442 000775
5147 005444 022626 2S:
5148 005446 023700 001374
5149 005452 001403
5150 005454 104016
5151 005456 104400 002277
5152
5153 005462 012737 005520 000004
5154 005470 005710 3S:
5155 005472 012701 014650
5156 005476 022100 4S:
5157 005500 001002
5158 005502 062100
5159 005504 000771
5160 005506 005721 5S:
5161 005510 005711
5162 005512 001371
5163 005514 104014
5164 005516 000401
5165 005520 022626 6S:
5166 005522 062700 000002
5167 005526 022700 177700 7S:
5168 005532 001356
5169 005534 012637 000006
5170 005540 012637 000004
5177
5178
(3)
(4)
(4)
(4)
(4)
(3)
(2) 005544 000004
5179 005546 013746 000004
5180 005552 013746 000006
5181 005556 012701 000017
5182 005562 013700 001404
5183 005566 012737 005612 000004
5184 005574 012737 000200 000006
5185 005602 005710 1S:

```

```

: #VERIFYING THAT ONLY EXPECTED DEVICES
: #RETURN "BREPLY". THIS TEST CHECKS
: #ONLY THAT "UNEXPECTED BREPLYS" AREN'T RECEIVED
: #NOT THAT ALL DEVICE ISSUE "BREPLY".
: *
: *****
TST7: SCOPE
      MOV      #3,STIMES          ; DO 3 ITERATIONS
      MOV      4,-(SP)           ; SAVE 4
      MOV      6,-(SP)           ; SAVE 6
      MOV      #2S,4            ; SET TIME-OUT TRAP
      CLR      6
      CLR      RO                ; SET FIRST ADDRESS TO 0
1S:   TST      (RO)              ; READ ADDRESS
      TST      (RO)+            ; POP ADDRESS
      BR       1S               ; CONTINUE TEST
2S:   CMP      (SP)+,(SP)+      ; FAKE AN RTI
      CMP      MEMSIZ,RO        ; 16K ?
      BEQ      .+10             ; BR IF MEMORY SIZE IS CORRECT
      ERROR   16                ; INCORRECT MEMORY SIZE FOUND
      TYPE    ,EM21            ; REPORT CHANGE MSG.
      MOV      #6S,4            ; LOC SAVRO HAS MEMORY FOUND.
      TST      (RO)              ; RESET TIME-OUT TRAP
      MOV      #BREPLY,TABLE,R1 ; SET EXPECTED DEVICES
4S:   CMP      (R1)+,RO         ; EXPECTED DEVICE?
      BNE     5S                ; BR IF NO
      ADD     (R1)+,RO          ; POP MODULO OFFSET
      BR      3S                ; CONT TEST
5S:   TST      (R1)+            ; POP PAST OFFSET
      TST      (R1)             ; END OF TABLE?
      BNE     4S                ; BR IF NO
      ERROR  14                ; UNEXPECTED "BREPLY" FROM DEVICE.
      BR      7S                ; SKIP FAKE RTI
6S:   CMP      (SP)+,(SP)+      ; FAKE RTI
7S:   ADD      #2,RO            ; POP TO NEXT ADDRESS
      CMP      #177700,RO       ; ALL DONE?
      BNE     3S
      MOV     (SP)+,6           ; RESTOE 6
      MOV     (SP)+,4           ; RESTORE 4
: *****
: #TEST 10 "DEVICE 'A'" ADDRESS TEST
: #TEST TO VERIFY A "BREPLY" RESPONSE
: #FROM "DEVICE 'A'".
: #IT IS ASSUMED THAT THE DEVICE
: #IS AT ADDRESS "172524".
: *****
TST10: SCOPE
      MOV      4,-(SP)          ; SAVE ADDRESS 4.
      MOV      6,-(SP)          ; SAVE ADDRESS 6.
      MOV      #15,R1           ; SET INTERNAL ICOUNT TO 15.
      MOV      DEV,A,RO         ; GET DEVICE CSR
      MOV      #3S,4            ; SET TIME-OUT VECTOR.
      MOV      #200,6           ; SET PRIO.
1S:   TST      (RO)              ; REFERENCE DEVICE CSR.

```

```

5186 005604 005301
5187 005606 001375
5188 005610 000404
5189 005612 104011
5190 005614 012716 005604
5191 005620 000002
5192 005622 012637 000006
5193 005626 012637 000004
5217

```

```

2S: DEC R1 ;ICOUNT = 0?
     BNE 1S ;BR IF NO.
     BR 4S ;CONT TEST
3S: ERROR 11 ;TIME-OUT ERROR.
     MOV #2S, (SP) ;SET RETURN ADD.
     RTI ;RETURN.
4S: MOV (SP)+, 6 ;RESTORE 6
     MOV (SP)+, 4 ;RESTORE 4

```

```

(5)
(4)
(5)
(5)
(5)
(5)
(4)

```

```

*****
#TEST 11 R/W TEST OF BIT 0 IN DEVICE 'A'
#R/W TEST OF BIT 0 IN DEVICE 'A'.
#WRITE BIT 0 VERIFY ONLY BIT 0 IS SET.
#CLEAR BIT 0 VERIFY BIT 0 IS CLEARED.
*
*****

```

```

(3) 005632 000004
(1) 005634 013700 001404
(1) 005640 012705 000001
(1) 005644 010510
(1) 005646 011004
(1) 005650 020504
(2) 005652 001401
(1) 005654 104010
(1) 005656 040510
(1) 005660 011004
(1) 005662 042705 000001
(1) 005666 005704
(2) 005670 001401
(1) 005672 104010
(1)
(5)
(4)
(5)
(5)
(5)
(5)
(4)

```

```

TST11: SCOPE
        MOV DEV.A,R0 ;LOAD DEVICE 'A' CSR.
        MOV #BIT0,R5 ;SET EXPECTED.
        MOV R5,(R0) ;WRITE BIT0
        MOV (R0),R4 ;READ CSR
        CMP R5,R4 ;BIT OK?
        BEQ .+4 ;BR IF OK
        ERROR 10 ;REGISTER HAS WRONG DATA.
        BIC R5,(R0) ;CLEAR BIT0
        MOV (R0),R4 ;READ REGISTER.
        BIC #BIT0,R5 ;ZERO EXPECTED
        TST R4 ;REGISTER OK?
        BEQ .+4 ;BR IF YES
        ERROR 10 ;REGISTER NOT =0.

```

```

(3) 005674 000004
(1) 005676 013700 001404
(1) 005702 012705 000002
(1) 005706 010510
(1) 005710 011004
(1) 005712 020504
(2) 005714 001401
(1) 005716 104010
(1) 005720 040510
(1) 005722 011004
(1) 005724 042705 000002
(1) 005730 005704
(2) 005732 001401
(1) 005734 104010
(1)
(5)
(4)

```

```

*****
#TEST 12 R/W TEST OF BIT 1 IN DEVICE 'A'
#R/W TEST OF BIT 1 IN DEVICE 'A'.
#WRITE BIT 1 VERIFY ONLY BIT 1 IS SET.
#CLEAR BIT 1 VERIFY BIT 1 IS CLEARED.
*
*****

```

```

TST12: SCOPE
        MOV DEV.A,R0 ;LOAD DEVICE 'A' CSR.
        MOV #BIT1,R5 ;SET EXPECTED.
        MOV R5,(R0) ;WRITE BIT1
        MOV (R0),R4 ;READ CSR
        CMP R5,R4 ;BIT OK?
        BEQ .+4 ;BR IF OK
        ERROR 10 ;REGISTER HAS WRONG DATA.
        BIC R5,(R0) ;CLEAR BIT1
        MOV (R0),R4 ;READ REGISTER.
        BIC #BIT1,R5 ;ZERO EXPECTED
        TST R4 ;REGISTER OK?
        BEQ .+4 ;BR IF YES
        ERROR 10 ;REGISTER NOT =0.

```

```

(1)
(5)
(4)

```

```

*****
#TEST 13 R/W TEST OF BIT 2 IN DEVICE 'A'

```

```

(5)
(5)
(5)
(5)
(4)
(3) 005736 000004
(1) 005740 013700 001404
(1) 005744 012705 000004
(1) 005750 010510
(1) 005752 011004
(1) 005754 020504
(2) 005756 001401
(1) 005760 104010
(1) 005762 040510
(1) 005764 011004
(1) 005766 042705 000004
(1) 005772 005704
(2) 005774 001401
(1) 005776 104010

```

```

:R/W TEST OF BIT 2 IN DEVICE 'A'.
:WRITE BIT 2 VERIFY ONLY BIT 2 IS SET.
:CLEAR BIT 2 VERIFY BIT 2 IS CLEARED.
:
:*****
TST13: SCOPE
      MOV     DEV.A,R0      ;LOAD DEVICE 'A' CSR.
      MOV     #BIT2,R5     ;SET EXPECTED.
      MOV     R5,(R0)      ;WRITE BIT2
      MOV     (R0),R4      ;READ CSR
      CMP     R5,R4        ;BIT OK?
      BEQ     .+4          ;BR IF OK
      ERROR   10           ;REGISTER HAS WRONG DATA.
      BIC     R5,(R0)      ;CLEAR BIT2
      MOV     (R0),R4      ;READ REGISTER.
      BIC     #BIT2,R5     ;ZERO EXPECTED
      TST     R4           ;REGISTER OK?
      BEQ     .+4          ;BR IF YES
      ERROR   10           ;REGISTER NOT =0.

```

```

(5)
(4)
(5)
(5)
(5)
(5)
(4)
(3) 006000 000004
(1) 006002 013700 001404
(1) 006006 012705 000010
(1) 006012 010510
(1) 006014 011004
(1) 006016 020504
(2) 006020 001401
(1) 006022 104010
(1) 006024 040510
(1) 006026 011004
(1) 006030 042705 000010
(1) 006034 005704
(2) 006036 001401
(1) 006040 104010

```

```

:*****
:TEST 14      R/W TEST OF BIT 3 IN DEVICE 'A'
:R/W TEST OF BIT 3 IN DEVICE 'A'
:WRITE BIT 3 VERIFY ONLY BIT 3 IS SET.
:CLEAR BIT 3 VERIFY BIT 3 IS CLEARED.
:
:*****
TST14: SCOPE
      MOV     DEV.A,R0      ;LOAD DEVICE 'A' CSR.
      MOV     #BIT3,R5     ;SET EXPECTED.
      MOV     R5,(R0)      ;WRITE BIT3
      MOV     (R0),R4      ;READ CSR
      CMP     R5,R4        ;BIT OK?
      BEQ     .+4          ;BR IF OK
      ERROR   10           ;REGISTER HAS WRONG DATA.
      BIC     R5,(R0)      ;CLEAR BIT3
      MOV     (R0),R4      ;READ REGISTER.
      BIC     #BIT3,R5     ;ZERO EXPECTED
      TST     R4           ;REGISTER OK?
      BEQ     .+4          ;BR IF YES
      ERROR   10           ;REGISTER NOT =0.

```

```

(5)
(4)
(5)
(5)
(5)
(5)
(4)
(3) 006042 000004
(1) 006044 013700 001404
(1) 006050 012705 000020
(1) 006054 010510
(1) 006056 011004
(1) 006060 020504
(2) 006062 001401

```

```

:*****
:TEST 15      R/W TEST OF BIT 4 IN DEVICE 'A'
:R/W TEST OF BIT 4 IN DEVICE 'A'
:WRITE BIT 4 VERIFY ONLY BIT 4 IS SET.
:CLEAR BIT 4 VERIFY BIT 4 IS CLEARED.
:
:*****
TST15: SCOPE
      MOV     DEV.A,R0      ;LOAD DEVICE 'A' CSR.
      MOV     #BIT4,R5     ;SET EXPECTED.
      MOV     R5,(R0)      ;WRITE BIT4
      MOV     (R0),R4      ;READ CSR
      CMP     R5,R4        ;BIT OK?
      BEQ     .+4          ;BR IF OK

```

```

(1) 006064 104010
(1) 006066 040510
(1) 006070 011004
(1) 006072 042705 000020
(1) 006076 005704
(2) 006100 001401
(1) 006102 104010

```

```

ERROR 10 ;REGISTER HAS WRONG DATA.
BIC R5,(R0) ;CLEAR BIT4
MOV (R0),R4 ;READ REGISTER.
BIC @BIT4,R5 ;ZERO EXPECTED
TST R4 ;REGISTER OK?
BEQ .+4 ;BR IF YES
ERROR 10 ;REGISTER NOT =0.

```

```

*****
#TEST 16 R/W TEST OF BIT 5 IN DEVICE 'A'
#R/W TEST OF BIT 5 IN DEVICE 'A'.
#WRITE BIT 5 VERIFY ONLY BIT 5 IS SET.
#CLEAR BIT 5 VERIFY BIT 5 IS CLEARED.
#
*****

```

```

(3) 006104 000004
(1) 006106 013700 001404
(1) 006112 012705 000040
(1) 006116 010510
(1) 006120 011004
(1) 006122 020504
(2) 006124 001401
(1) 006126 104010
(1) 006130 040510
(1) 006132 011004
(1) 006134 042705 000040
(1) 006140 005704
(2) 006142 001401
(1) 006144 104010

```

```

TST16: SCOPE
MOV DEV,A,R0 ;LOAD DEVICE 'A' CSR.
MOV @BIT5,R5 ;SET EXPECTED.
MOV R5,(R0) ;WRITE BIT5
MOV (R0),R4 ;READ CSR
CMP R5,R4 ;BIT OK?
BEQ .+4 ;BR IF OK
ERROR 10 ;REGISTER HAS WRONG DATA.
BIC R5,(R0) ;CLEAR BIT5
MOV (R0),R4 ;READ REGISTER.
BIC @BIT5,R5 ;ZERO EXPECTED
TST R4 ;REGISTER OK?
BEQ .+4 ;BR IF YES
ERROR 10 ;REGISTER NOT =0.

```

```

*****
#TEST 17 R/W TEST OF BIT 6 IN DEVICE 'A'
#R/W TEST OF BIT 6 IN DEVICE 'A'.
#WRITE BIT 6 VERIFY ONLY BIT 6 IS SET.
#CLEAR BIT 6 VERIFY BIT 6 IS CLEARED.
#
*****

```

```

(3) 006146 000004
(1) 006150 013700 001404
(1) 006154 012705 000100
(1) 006160 010510
(1) 006162 011004
(1) 006164 020504
(2) 006166 001401
(1) 006170 104010
(1) 006172 040510
(1) 006174 011004
(1) 006176 042705 000100
(1) 006202 005704
(2) 006204 001401
(1) 006206 104010

```

```

TST17: SCOPE
MOV DEV,A,R0 ;LOAD DEVICE 'A' CSR.
MOV @BIT6,R5 ;SET EXPECTED.
MOV R5,(R0) ;WRITE BIT6
MOV (R0),R4 ;READ CSR
CMP R5,R4 ;BIT OK?
BEQ .+4 ;BR IF OK
ERROR 10 ;REGISTER HAS WRONG DATA.
BIC R5,(R0) ;CLEAR BIT6
MOV (R0),R4 ;READ REGISTER.
BIC @BIT6,R5 ;ZERO EXPECTED
TST R4 ;REGISTER OK?
BEQ .+4 ;BR IF YES
ERROR 10 ;REGISTER NOT =0.

```

```

*****
#TEST 20 R/W TEST OF BIT 7 IN DEVICE 'A'
#R/W TEST OF BIT 7 IN DEVICE 'A'.
#WRITE BIT 7 VERIFY ONLY BIT 7 IS SET.

```

```

(1)
(5)
(4)
(5)

```

(5)
(5)
(4)
(3) 006210 000004
(1) 006212 013700 001404
(1) 006216 012705 000200
(1) 006222 010510
(1) 006224 011004
(1) 006226 020504
(2) 006230 001401
(1) 006232 104010
(1) 006234 040510
(1) 006236 011004
(1) 006240 042705 000200
(1) 006244 005704
(2) 006246 001401
(1) 006250 104010

```
*****  
: #CLEAR BIT 7 VERIFY BIT 7 IS CLEARED.  
: #  
*****  
TST20: SCOPE  
      MOV     DEV,A,R0      ;LOAD DEVICE 'A' CSR.  
      MOV     @BIT7,R5     ;SET EXPECTED.  
      MOV     R5,(R0)      ;WRITE BIT7  
      MOV     (R0),R4      ;READ CSR  
      CMP     R5,R4        ;BIT OK?  
      BEQ     .+4          ;BR IF OK  
      ERROR  10           ;REGISTER HAS WRONG DATA.  
      BIC     R5,(R0)      ;CLEAR BIT7  
      MOV     (R0),R4      ;READ REGISTER.  
      BIC     @BIT7,R5     ;ZERO EXPECTED  
      TST     R4           ;REGISTER OK?  
      BEQ     .+4          ;BR IF YES  
      ERROR  10           ;REGISTER NOT =0.
```

(5)
(4)
(5)
(5)
(5)
(5)
(4)
(3) 006252 000004
(1) 006254 013700 001404
(1) 006260 012705 000400
(1) 006264 010510
(1) 006266 011004
(1) 006270 020504
(2) 006272 001401
(1) 006274 104010
(1) 006276 040510
(1) 006300 011004
(1) 006302 042705 000400
(1) 006306 005704
(2) 006310 001401
(1) 006312 104010

```
*****  
: #TEST 21 R/W TEST OF BIT 8 IN DEVICE 'A'  
: #R/W TEST OF BIT 8 IN DEVICE 'A'.  
: #WRITE BIT 8 VERIFY ONLY BIT 8 IS SET.  
: #CLEAR BIT 8 VERIFY BIT 8 IS CLEARED.  
: #  
*****  
TST21: SCOPE  
      MOV     DEV,A,R0      ;LOAD DEVICE 'A' CSR.  
      MOV     @BIT8,R5     ;SET EXPECTED.  
      MOV     R5,(R0)      ;WRITE BIT8  
      MOV     (R0),R4      ;READ CSR  
      CMP     R5,R4        ;BIT OK?  
      BEQ     .+4          ;BR IF OK  
      ERROR  10           ;REGISTER HAS WRONG DATA.  
      BIC     R5,(R0)      ;CLEAR BIT8  
      MOV     (R0),R4      ;READ REGISTER.  
      BIC     @BIT8,R5     ;ZERO EXPECTED  
      TST     R4           ;REGISTER OK?  
      BEQ     .+4          ;BR IF YES  
      ERROR  10           ;REGISTER NOT =0.
```

(5)
(4)
(5)
(5)
(5)
(5)
(4)
(3) 006314 000004
(1) 006316 013700 001404
(1) 006322 012705 001000
(1) 006326 010510
(1) 006330 011004
(1) 006332 020504
(2) 006334 001401
(1) 006336 104010
(1) 006340 040510

```
*****  
: #TEST 22 R/W TEST OF BIT 9 IN DEVICE 'A'  
: #R/W TEST OF BIT 9 IN DEVICE 'A'.  
: #WRITE BIT 9 VERIFY ONLY BIT 9 IS SET.  
: #CLEAR BIT 9 VERIFY BIT 9 IS CLEARED.  
: #  
*****  
TST22: SCOPE  
      MOV     DEV,A,R0      ;LOAD DEVICE 'A' CSR.  
      MOV     @BIT9,R5     ;SET EXPECTED.  
      MOV     R5,(R0)      ;WRITE BIT9  
      MOV     (R0),R4      ;READ CSR  
      CMP     R5,R4        ;BIT OK?  
      BEQ     .+4          ;BR IF OK  
      ERROR  10           ;REGISTER HAS WRONG DATA.  
      BIC     R5,(R0)      ;CLEAR BIT9
```

(1) 006342 011004
(1) 006344 042705 001000
(1) 006350 005704
(2) 006352 001401
(1) 006354 104010

```
MOV (R0),R4 ;READ REGISTER.  
BIC @BIT9,R5 ;ZERO EXPECTED  
TST R4 ;REGISTER OK?  
BEQ +4 ;BR IF YES  
ERROR 10 ;REGISTER NOT =0.
```

(1)
(5)
(4)
(5)
(5)
(5)
(5)
(4)

#TEST 23 R/W TEST OF BIT 10 IN DEVICE 'A'
#R/W TEST OF BIT 10 IN DEVICE 'A'.
#WRITE BIT 10 VERIFY ONLY BIT 10 IS SET.
#CLEAR BIT 10 VERIFY BIT 10 IS CLEARED.

(3) 006356 000004
(1) 006360 013700 001404
(1) 006364 012705 002000
(1) 006370 010510
(1) 006372 011004
(1) 006374 020504
(2) 006376 001401
(1) 006400 104010
(1) 006402 040510
(1) 006404 011004
(1) 006406 042705 002000
(1) 006412 005704
(2) 006414 001401
(1) 006416 104010

```
TST23: SCOPE  
MOV DEV,A,R0 ;LOAD DEVICE 'A' CSR.  
MOV @BIT10,R5 ;SET EXPECTED.  
MOV R5,(R0) ;WRITE BIT10  
MOV (R0),R4 ;READ CSR  
CMP R5,R4 ;BIT OK?  
BEQ +4 ;BR IF OK  
ERROR 10 ;REGISTER HAS WRONG DATA.  
BIC R5,(R0) ;CLEAR BIT10  
MOV (R0),R4 ;READ REGISTER.  
BIC @BIT10,R5 ;ZERO EXPECTED  
TST R4 ;REGISTER OK?  
BEQ +4 ;BR IF YES  
ERROR 10 ;REGISTER NOT =0.
```

(1)
(5)
(4)
(5)
(5)
(5)
(5)
(4)

#TEST 24 R/W TEST OF BIT 11 IN DEVICE 'A'
#R/W TEST OF BIT 11 IN DEVICE 'A'.
#WRITE BIT 11 VERIFY ONLY BIT 11 IS SET.
#CLEAR BIT 11 VERIFY BIT 11 IS CLEARED.

(3) 006420 000004
(1) 006424 013700 001404
(1) 006428 012705 004000
(1) 006432 010510
(1) 006434 011004
(1) 006436 020504
(2) 006440 001401
(1) 006442 104010
(1) 006444 040510
(1) 006446 011004
(1) 006450 042705 004000
(1) 006454 005704
(2) 006456 001401
(1) 006460 104010

```
TST24: SCOPE  
MOV DEV,A,R0 ;LOAD DEVICE 'A' CSR.  
MOV @BIT11,R5 ;SET EXPECTED.  
MOV R5,(R0) ;WRITE BIT11  
MOV (R0),R4 ;READ CSR  
CMP R5,R4 ;BIT OK?  
BEQ +4 ;BR IF OK  
ERROR 10 ;REGISTER HAS WRONG DATA.  
BIC R5,(R0) ;CLEAR BIT11  
MOV (R0),R4 ;READ REGISTER.  
BIC @BIT11,R5 ;ZERO EXPECTED  
TST R4 ;REGISTER OK?  
BEQ +4 ;BR IF YES  
ERROR 10 ;REGISTER NOT =0.
```

(1)
(5)
(4)
(5)
(5)
(5)
(5)

#TEST 25 R/W TEST OF BIT 12 IN DEVICE 'A'
#R/W TEST OF BIT 12 IN DEVICE 'A'.
#WRITE BIT 12 VERIFY ONLY BIT 12 IS SET.
#CLEAR BIT 12 VERIFY BIT 12 IS CLEARED.

```

(4)
(3) 006462 000004
(1) 006464 013700 001404
(1) 006470 012705 010000
(1) 006474 010510
(1) 006476 011004
(1) 006500 020504
(2) 006502 001401
(1) 006504 104010
(1) 006506 040510
(1) 006510 011004
(1) 006512 042705 010000
(1) 006516 005704
(2) 006520 001401
(1) 006522 104010

```

```

*****
TST25: SCOPE
MOV DEV A,R0 ;LOAD DEVICE 'A' CSR.
MOV @BIT12,R5 ;SET EXPECTED.
MOV R5,(R0) ;WRITE BIT12
MOV (R0),R4 ;READ CSR
CMP R5,R4 ;BIT OK?
BEQ .+4 ;BR IF OK
ERROR 10 ;REGISTER HAS WRONG DATA.
BIC R5,(R0) ;CLEAR BIT12
MOV (R0),R4 ;READ REGISTER.
BIC @BIT12,R5 ;ZERO EXPECTED
TST R4 ;REGISTER OK?
BEQ .+4 ;BR IF YES
ERROR 10 ;REGISTER NOT =0.

```

```

(5)
(4)
(5)
(5)
(5)
(5)
(4)

```

```

*****
#TEST 26 R/W TEST OF BIT 13 IN DEVICE 'A'
#R/W TEST OF BIT 13 IN DEVICE 'A'
#WRITE BIT 13 VERIFY ONLY BIT 13 IS SET.
#CLEAR BIT 13 VERIFY BIT 13 IS CLEARED.
#
*****

```

```

(3) 006524 000004
(1) 006526 013700 001404
(1) 006532 012705 020000
(1) 006536 010510
(1) 006540 011004
(1) 006542 020504
(2) 006544 001401
(1) 006546 104010
(1) 006550 040510
(1) 006552 011004
(1) 006554 042705 020000
(1) 006560 005704
(2) 006562 001401
(1) 006564 104010

```

```

TST26: SCOPE
MOV DEV A,R0 ;LOAD DEVICE 'A' CSR.
MOV @BIT13,R5 ;SET EXPECTED.
MOV R5,(R0) ;WRITE BIT13
MOV (R0),R4 ;READ CSR
CMP R5,R4 ;BIT OK?
BEQ .+4 ;BR IF OK
ERROR 10 ;REGISTER HAS WRONG DATA.
BIC R5,(R0) ;CLEAR BIT13
MOV (R0),R4 ;READ REGISTER.
BIC @BIT13,R5 ;ZERO EXPECTED
TST R4 ;REGISTER OK?
BEQ .+4 ;BR IF YES
ERROR 10 ;REGISTER NOT =0.

```

```

(5)
(4)
(5)
(5)
(5)
(5)
(4)

```

```

*****
#TEST 27 R/W TEST OF BIT 14 IN DEVICE 'A'
#R/W TEST OF BIT 14 IN DEVICE 'A'
#WRITE BIT 14 VERIFY ONLY BIT 14 IS SET.
#CLEAR BIT 14 VERIFY BIT 14 IS CLEARED.
#
*****

```

```

(3) 006566 000004
(1) 006570 013700 001404
(1) 006574 012705 040000
(1) 006600 010510
(1) 006602 011004
(1) 006604 020504
(2) 006606 001401
(1) 006610 104010
(1) 006612 040510
(1) 006614 011004
(1) 006616 042705 040000

```

```

TST27: SCOPE
MOV DEV A,R0 ;LOAD DEVICE 'A' CSR.
MOV @BIT14,R5 ;SET EXPECTED.
MOV R5,(R0) ;WRITE BIT14
MOV (R0),R4 ;READ CSR
CMP R5,R4 ;BIT OK?
BEQ .+4 ;BR IF OK
ERROR 10 ;REGISTER HAS WRONG DATA.
BIC R5,(R0) ;CLEAR BIT14
MOV (R0),R4 ;READ REGISTER.
BIC @BIT14,R5 ;ZERO EXPECTED

```

(1) 006622 005704
(2) 006624 001401
(1) 006626 104010

TST R4 ; REGISTER OK?
BEQ +4 ; BR IF YES
ERROR 10 ; REGISTER NOT =0.

:TEST 30 R/W TEST OF BIT 15 IN DEVICE 'A'
:R/W TEST OF BIT 15 IN DEVICE 'A'
:WRITE BIT 15 VERIFY ONLY BIT 15 IS SET.
:CLEAR BIT 15 VERIFY BIT 15 IS CLEARED.
:*

(3) 006630 000004
(1) 006632 013700 001404
(1) 006634 012705 100000
(1) 006636 010510
(1) 006638 011004
(1) 006640 020504
(2) 006642 001401
(1) 006644 104010
(1) 006646 040510
(1) 006648 011004
(1) 006650 042705 100000
(1) 006652 005704
(2) 006654 001401
(1) 006656 104010

TST30: SCOPE
MOV DEV.A,R0 ; LOAD DEVICE 'A' CSR.
MOV @BIT15,R5 ; SET EXPECTED.
MOV R5,(R0) ; WRITE BIT15
MOV (R0),R4 ; READ CSR
CMP R5,R4 ; BIT OK?
BEQ +4 ; BR IF OK
ERROR 10 ; REGISTER HAS WRONG DATA.
BIC R5,(R0) ; CLEAR BIT15
MOV (R0),R4 ; READ REGISTER.
BIC @BIT15,R5 ; ZERO EXPECTED
TST R4 ; REGISTER OK?
BEQ +4 ; BR IF YES
ERROR 10 ; REGISTER NOT =0.

:TEST 31 BINARY COUNT TEST
:TEST TO WRITE A BINARY COUNT PATTERN
: (000000-177777) THRU DEVICE 'A' REGISTER
:*

(2) 006672 000004
(1) 006674 012737 000003 001160
(1) 006676 013700 001404
(1) 006678 005005
(1) 006680 010510
(1) 006682 011004
(1) 006684 020504
(1) 006686 001401
(1) 006688 104010
(1) 006690 005205
(1) 006692 001371

TST31: SCOPE
MOV #3,STIMES ; DO 3 ITERATIONS
MOV DEV.A,R0 ; GET DEVICE 'A' CSR.
CLR R5 ; ZERO EXPECTED
18: MOV R5,(R0) ; LOAD DATA
MOV (R0),R4 ; READ DATA
CMP R5,R4 ; WAS DATA GOOD?
BEQ +4 ; BR IF YES
ERROR 10 ; REGISTER DATA ERROR.
INC R5 ; UPDATE DATA
BNE 18 ; BR IF NOT DONE

:TEST 32 1'S AND 0'S TEST
:TEST TO WRITE ALTERNATE 1'S AND 0'S
: (125252 AND 052525) INTO DEVICE 'A'.
:*

(2) 006726 000004
(1) 006730 012737 000012 001160
(1) 006732 013700 001404
(1) 006734 005003

TST32: SCOPE
MOV #10,STIMES ; DO 10. ITERATIONS
MOV DEV.A,R0 ; GET DEVICE 'A' CSR
CLR R3 ; SET TO DO 128. TIMES


```

(4)
(4)
(4)
(3)
(2) 007104 000004
(1) 007106 012737 000005 001160
53304 007114 013700 001404
53305 007120 012710 177777
53306 007124 005005
53307 007126 000005
(1) 007130 052777 000100 172004
53308 007136 011004
53309 007140 001401
53310 007142 104010
53311

```

```

;#TEST THAT ALL BITS IN DEVICE 'A'
;#REGISTER CAN BE CLEARED BY "INIT" (RESET INSTR).
;#
;#*****
TST34: SCOPE
MOV #5,STIMES ;:DO 5 ITERATIONS
MOV DEV.A,R0 ;:GET CSR
MOV @-1,(R0) ;:GET ALL BITS
CLR RS ;:SET EXPECTED TO ALL ZEROS
RESET ;:ISSUE INIT
BIS #100,2STKS ;:SET INTR ENABLE
MOV (R0),R4 ;:READ REGISTER
BEQ +4 ;:BR IF ALL ZEROS
ERROR i0 ;:REGISTER NOT ALL ZEROS

```



```

(5)
(4)
(3) 007300 000004
(1) 007302 106427 000200
(1) 007306 013700 001406
(1) 007312 012705 000200
(1) 007316 010510
(1) 007320 011004
(1) 007322 020504
(2) 007324 001401
(1) 007326 104013
(1) 007330 040510
(1) 007332 011004
(1) 007334 042705 000200
(1) 007340 005704
(2) 007342 001401
(1) 007344 104013

```

```

*****
*
*ST37: SCOPE
MTPS 8200
MOV DEV,B,RO ;LOAD CSR INTO RO
MOV 8BIT7,RS ;SET BIT 7 INTO RS
MOV RS,(RO) ;WRITE BIT
MOV (RO),R4 ;READ REGISTER
CMP RS,R4 ;OK?
BEQ .+4 ;BR IF OK.
ERROR 13 ;REGISTER R/W ERROR
BIC RS,(RO) ;CLEAR BIT 7
MOV (RO),R4 ;READ DEVICE
BIC 8BIT7,RS ;CLEAR EXPECTED
TST R4 ;REGISTER =0?
BEQ .+4 ;BR IF =0!
ERROR 13 ;REGISTER ERROR

```

```

S362
S371
S372
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2)
S373
S374
S375
S376
S377
S378
S379
S380
S381
S382
S383
S384
S385
S386
S387
S388
S389
S390
S391
S392
S393
S394
S395
S396
S397
S398
S399

```

```

007346 000004
007350 106427 000000
007354 017746 172030
007360 017746 172026
007364 012777 007500 172016
007372 012777 000200 172012
007400 005001
007402 013700 001406
007406 052710 000100
007412 000240
007414 042710 000100
007420 052710 000200
007424 000240
007426 013777 007446 171754
007434 052710 000100
007440 000240
007442 104015
007444 000407
007446 005010
007450 105203
007452 100403
007454 012716 007406
007460 000002
007462 022626
007464 005010
007466 012677 171720
007472 012677 171712
007476 000404

```

```

*****
*TEST 40 DEVICE 'B' INTERRUPT TEST (SEL 0)
*TEST OF DEVICE 'B' INTERRUPTS (SEL 0)
*SET BIT6 VERIFY NO INTERRUPT
*CLEAR BIT6; SET BIT7 -VERIFY NO INTERRUPT
*NOW SET BIT6 AGAIN -EXPECT INTERUPT AT
*VECTOR "340".
*
*
*****
*ST40: SCOPE
MTPS 80
MOV 28.AVEC,-(SP) ;SAVE DVB VECTOR
MOV 28.APTY,-(SP) ;SAVE DVB PRIO
MOV 25,28.AVEC ;LOAD VECTOR
MOV 200,28.APTY ;LOAD PRIO
CLR R1
MOV DEV,B,RO
15: BIS 8BIT6,(RO) ;SET EVENT ENABLE
NOP ;WASTE TIME
BIC 8BIT6,(RO) ;CLEAR IT
BIS 8BIT7,(RO) ;SET OTHER INTR
NOP ;WASTE TIME
MOV 25,28.AVEC ;SET GOOD VECTOR
BIS 8BIT6,(RO) ;SET IE
NOP ;WASTE TIME
ERROR 15 ;NO INTERRUPT
BR 45 ;CONT TEST
25: CLR (RO) ;ZERO REGISTER
INCB R3 ;UPDATE ICOUNT
BMI 35 ;DONE?
MOV 815,(SP) ;SET RETURN
RTI ;EXIT
35: CMP (SP)+,(SP)+ ;POP PC+PSW
45: CLR (RO) ;DSABLE DEVICE
MOV (SP)+,28.APTY ;RESTORE PTY
MOV (SP)+,28.AVEC ;RESTORE VEC
BR 65 ;CONT TEST

```

5400	007500	010002	
5401	007502	011600	
5402	007504	104005	
5403	007506	010200	
5404	007510	106427	000000

```

5S:  MOV R0,R2      ;SAVE R0
      MOV (SP),R0   ;SAVE PC
      ERROR S       ;UNEXPECTED INTERUPT
      MOV R2,R0     ;RESTORE R0
6S:  MTPS 80        ;ZERO PTY

```

5412
5413
(3)
(4)
(4)
(4)
(4)
(3)
(2)

```

*****
;TEST 41 "DEVICE 'B' ADDRESS TEST (SEL 2)
;TEST TO VERIFY A "BRIEFLY" RESPONSE
;FROM "DEVICE 'B'" (SEL 2).
;IT IS ASSUMED THAT THE DEVICE
;IS AT ADDRESS "165252" (SEL 2).
*****

```

5414	007514	000004	
5415	007516	013746	000004
5416	007522	013746	000006
5417	007526	012701	000017
5418	007532	013700	001406
5419	007536	062700	000002
5420	007542	012737	007556 000004
5421	007550	012737	000200 000006
5422	007556	005710	
5423	007560	005301	
5424	007562	001375	
5425	007564	000404	
5426	007566	104012	
5427	007570	012716	007560
5428	007574	000002	
5429	007576	012637	000006
5430	007602	012637	000004

```

TST41: SCOPE
      MOV 4,-(SP)    ;SAVE ADDRESS 4.
      MOV 6,-(SP)    ;SAVE ADDRESS 6.
      MOV #15,R1     ;SET INTERNAL ICOUNT TO 15.
      MOV DEV,B,R0   ;GET DEVICE CSR.
      ADD #2,R0      ;MAKE IT SEL 2
      MOV #3,R4      ;SET TIME-OUT VECTOR.
      MOV #200,R6    ;SET PRIO.
1S:   TST (R0)       ;REFERENCE DEVICE CSR.
2S:   DEC R1         ;ICOUNT = 0?
      BNE 1S         ;BR IF NO.
      BR 4S          ;CONT TEST
3S:   ERROR 12      ;TIME-OUT ERROR.
      MOV #2S,(SP)  ;SET RETURN ADD.
      RTI           ;RETURN.
4S:   MOV (SP)+,6   ;RESTORE 6
      MOV (SP)+,4   ;RESTORE 4

```

5431
5432
(5)
(4)
(5)
(5)
(5)
(5)
(5)
(4)

```

*****
;TEST 42 R/W TEST OF BIT 6 IN DEVICE 'B' (SEL 2)
;TEST THAT BIT 6 IN DEVICE 'B' (SEL 2)
;IS R/W.
;SET BIT 6 VERIFY IT IS SET.
;CLEAR BIT 6 VERIFY IT IS CLEAR.
;
*****

```

5433	007606	000004	
5434	007610	106427	000200
5435	007614	013700	001406
5436	007620	062700	000002
5437	007624	012705	000100
5438	007630	010510	
5439	007632	011004	
5440	007634	020504	
5441	007636	001401	
5442	007640	104013	
5443	007642	040510	
5444	007644	011004	
5445	007646	042705	000100
5446	007652	005704	
5447	007654	001401	

```

TST42: SCOPE
      MTPS #200
      MOV DEV,B,R0   ;LOAD CSR INTO R0
      ADD #2,R0      ;MAKE IT SEL 2
      MOV #BIT6,RS   ;SET BIT 6 INTO RS
      MOV RS,(R0)    ;WRITE BIT
      MOV (R0),R4    ;READ REGISTER
      CMP RS,R4      ;OK?
      BEQ .+4        ;BR IF OK.
      ERROR 13      ;REGISTER R/W ERROR
      BIC RS,(R0)    ;CLEAR BIT 6
      MOV (R0),R4    ;READ DEVICE
      BIC #BIT6,RS   ;CLEAR EXPECTED
      TST R4         ;REGISTER =0?
      BEQ .+4        ;BR IF =0!

```

```

(1) 007656 104013
(1) 007660 106427 000000
(1)
(5)
(4)
(5)
(5)
(5)
(5)
(5)
(4)
(3) 007664 000004
(1) 007666 106427 000200
(1) 007672 013700 001406
(1) 007676 062700 000002
(1) 007702 012705 000200
(1) 007706 010510
(1) 007710 011004
(1) 007712 020504
(2) 007714 001401
(1) 007716 104013
(1) 007720 040510
(1) 007722 011004
(1) 007724 042705 000200
(1) 007730 005704
(2) 007732 001401
(1) 007734 104013
(1) 007736 106427 000000

```

```

ERROR 13 :REGISTER ERROR
MTPS 80 :SET PTY TO 0
*****
:TEST 43 R/W TEST OF BIT 7 IN DEVICE 'B' (SEL 2)
:TEST THAT BIT 7 IN DEVICE 'B' (SEL 2)
:IS R/W.
:SET BIT 7 VERIFY IT IS SET.
:CLEAR BIT 7 VERIFY IT IS CLEAR.
:
*****
TST43: SCOPE
MTPS 800
MOV DEV.B,RO :LOAD CSR INTO RO
ADD #2,RO :MAKE IT SEL 2
MOV #BIT7,RS :SET BIT 7 INTO RS
MOV RS,(RO) :WRITE BIT
MOV (RO),R4 :READ REGISTER
CMP RS,R4 :OK?
BEQ +4 :BR IF OK.
ERROR 13 :REGISTER R/W ERROR
BIC RS,(RO) :CLEAR BIT 7
MOV (RO),R4 :READ DEVICE
BIC #BIT7,RS :CLEAR EXPECTED
TST R4 :REGISTER =0?
BEQ +4 :BR IF =0!
ERROR 13 :REGISTER ERROR
MTPS 80 :SET PTY TO 0

```

```

5458
5467
5468
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 007742 000004
5469 007744 106427 000000
5470 007750 017746 171440
5471 007754 017746 171436
5472 007760 012777 010100 171426
5473 007766 012777 000200 171422
5474 007774 005001
5475 007776 013700 001406
5476 010002 062700 000002
5477 010006 052710 000100
5478 010012 000240
5479 010014 042710 000100
5480 010020 052710 000200
5481 010024 000240
5482 010026 012777 010046 171360
5483 010034 052710 000100
5484 010040 000240

```

```

*****
:TEST 44 DEVICE 'B' INTERRUPT TEST (SEL 2)
:TEST OF DEVICE 'B' INTERRUPTS (SEL 2)
:SET BIT6 VERIFY NO INTERRUPT
:CLEAR BIT6; SET BIT7 -VERIFY NO INTERRUPT
:NOW SET BIT6 AGAIN -EXPECT INTERUPT AT
:VECTOR "344".
:
*****
TST44: SCOPE
MTPS 80
MOV #28,BVEC,-(SP) :SAVE DVB VECTOR
MOV #28,BPTY,-(SP) :SAVE DVB PRIO
MOV #55,28,BVEC :LOAD VECTOR
MOV #200,28,BPTY :LOAD PRIO
CLR R1
MOV DEV.B,RO
ADD #2,RO :MAKE IT SEL 2
IS: BIS #BIT6,(RO) :SET EVENT ENABLE
NOP :WASTE TIME
BIC #BIT6,(RO) :CLEAR IT
BIS #BIT7,(RO) :SET OTHER INTR
NOP :WASTE TIME
MOV #25,28,BVEC :SET GOOD VECTOR
BIS #BIT6,(RO) :SET IE
NOP :WASTE TIME

```

5485	010042	104015		ERROR	15		:NO INTERRUPT
5486	010044	000407		BR	45		:CONT TEST
5487	010046	005010		25: CLR	(R0)		:ZERO REGISTER
5488	010050	105203		INCB	R3		:UPDATE ICOUNT
5489	010052	100403		BMI	35		:DONE?
5490	010054	012716	010006	MOV	815,(SP)		:SET RETURN
5491	010060	000002		RTI			:EXIT
5492	010062	022626		35: CMP	(SP)+,(SP)+		:POP PC+PSW
5493	010064	005010		45: CLR	(R0)		:DSABLE DEVICE
5494	010066	012677	171324	MOV	(SP)+,28.BPTY		:RESTORE PTY
5495	010072	012677	171316	MOV	(SP)+,28.BVEC		:RESTORE VEC
5496	010076	000404		BR	65		:CONT TEST
5497	010100	010002		55: MOV	R0,R2		:SAVE R0
5498	010102	011600		MOV	(SP),R0		:SAVE PC
5499	010104	104005		ERROR	5		:UNEXPECTED INTERUPT
5500	010106	010200		MOV	R2,R0		:RESTORE R0
5501	010110	106427	000000	65: MTPS	80		:ZERO PTY

```

*****
: *TEST 45      TEST OF 'EVENT' AT DEVICE 'B'
: *TEST THAT SETTING BIT1 OF DEVICE 'B' (SEL 2)
: *CAUSES AN 'EVENT' VECTORING TO ADDRESS 100.
: *
: *
*****

```

5510	(3)						
5511	(4)						
5512	(4)						
5513	(4)						
5514	(3)						
5515	(2)	010114	000004	TST45: SCOPE			
5516		010116	106427	MTPS	80		:ZERO PSW
5517		010122	017746	MOV	2CLKVEC,-(SP)		:SAVE VEC
5518		010126	017746	MOV	2CLKPTY,-(SP)		:SAVE PTY
5519		010132	013700	MOV	DEV.B,R0		:GET CSR
5520		010136	062700	ADD	82,R0		:MAKE IT SEL 2
5521		010142	005003	CLR	R3		:ICOUNTER=0
5522		010144	012777	MOV	825,2CLKVEC		:SET VECTOR
5523		010150	012777	MOV	8200,2CLKPTY		:SET PRIO
5524		010160	005010	15: CLR	(R0)		:ZERO REGISTER
5525		010162	052710	BIS	8BIT1,(R0)		:SET EVENT
5526		010166	005027	CLR	(PC)+		:STALL FOR TIME.
5527		010170	000000	645: 0			
5528		010172	062737	ADD	81,645		:INC TIMER
5529		010200	001374	BNE	-6		:TIMER DONE?
5530		010202	104015	ERROR	15		:NO INTERRUPT
5531		010204	000406	BR	45		:CONT
5532		010206	105203	25: INCB	R3		:ICOUNT=ICOUT+1
5533		010210	100403	BMI	35		:BR IF DONE
5534		010212	012716	MOV	815,(SP)		:SET RTN
5535		010216	000002	RTI			:EXIT
5536		010220	022626	35: CMP	(SP)+,(SP)+		:POP PC+PSW
5537		010222	005010	45: CLR	(R0)		:DSABLE DEVICE
5538		010224	012677	MOV	(SP)+,2CLKPTY		:RESTORE PTY
5539		010230	012677	MOV	(SP)+,2CLKVEC		:RESTORE VEC
5540		010234	106427	MTPS	80		:PTY 0

```

*****
: *TEST 46      DLV11 CHARACTER TEST
*****

```

```

(4)
(4)
(4)
(4)
(4)
(3)
(2) 010240 000004
(1) 010242 012737 000002 001160
5546 010250 012700 175610
5547 010254 012701 000002
5548 010260 005003
5549 010262 005005
5550 010264 005760 000002
5551 010270 005760 000002
5552 010274 105760 000004
5553 010300 100407
5554 010302 023737 000000 000000
5555 010310 062703 000001
5556 010314 001367
5557 010316 104016
5558 010320 110560 000006
5559 010324 005003
5560 010326 105710
5561 010330 100407
5562 010332 023737 000000 000000
5563 010340 062703 000001
5564 010344 001370
5565 010346 104016
5566 010350 016004 000002
5567 010354 020504
5568 010356 001401
5569 010360 104017
5570 010362 105205
5571 010364 001343
5572 010366 005301
5573 010370 001341
5574 010372 105760 000004
5575 010376 100375
5576 010400 112760 000015 000006
5577 010406 105760 000004
5578 010412 100375
5579 010414 112760 000012 000006
5580 010422 012705 000005
5581 010426 012703 000040
5582 010430 005203
5583 010434 022703 000176
5584 010440 001772
5585 010442 012701 000110
5586 010446 010304
5587 010450 105760 000004
5588 010454 100375
5589 010456 112760 000015 000006
5590 010464 105760 000004
5591 010470 100375
5592 010472 112760 000012 000006
5593 010500 022704 000176

```

```

: *TEST TO OUTPUT CHARACTERS ON THE DLV11
: *NOT BEING USED AS THE CONSOLE INTERFACE
: *PROGRAM WILL DO 2 FULL BINARY COUNT PATTERNS
: *AND THEN A SLIDE ASCII PATTERN FOR 5 ROWS.
: *
: *****
TST46: SCOPE
MOV 82,STIMES
MOV 8175610,R0
MOV 82,R1
CLR R3
CLR R5
TST 2(R0)
TST 2(R0)
15: TSTB 4(R0)
BMI 64$
CMP 0,0
ADD 81,R3
BNE 15$
ERROR 16
64$: MOVB 85,6(R0)
CLR R3
65$: TSTB (R0)
BMI 66$
CMP 0,0
ADD 81,R3
BNE 65$
ERROR 16
66$: MOV 2(R0),R4
CMP R5,R4
BEQ 67$
ERROR 17
67$: INCB R5
BNE 15$
DEC R1
BNE 15$
TSTB 4(R0)
BPL -4
MOV 815,6(R0)
TSTB 4(R0)
BPL -4
MOV 812,6(R0)
MOV 85,R5
25: MOV 840,R3
35: INC R3
CMP 8176,R3
BEQ 25$
MOV 872,R1
MOV R3,R4
TSTB 4(R0)
BPL -4
MOV 815,6(R0)
TSTB 4(R0)
BPL -4
45: MOV 812,6(R0)
CMP 8176,R4

```

```

: :DO 2 ITERATIONS
: :SET DEVICE ADDRESS
: :SET BINARY COUNTER
:
: :CLEAR COUNTER
: :CLR RX DONE
:
: :PRINTER READY?
: :BR IF YES
: :WASTE TIME
: :DELAY COUNTER+1
: :DELAY DONE? NO!
: :TPS BIT7 (<)1 (NOT READY)
: :LOAD DATA CHAR.
: :CLEAR POINTER
: :RECEIVER READY(DONE)?
: :BR IF YES
: :WASTE TIME.
:
: :DELAY DONE? NO!
: :RECEIVER NOT DONE.
: :READ DATA
: :DATA GOOD?
: :BR IF YES
: :DATA COMPARE ERROR.
: :NEXT CHAR

```

```

5594 010504 001002
5595 010506 012704 000040
5596 010512 105760 000004
5597 010516 100375
5598 010520 110460 000006
5599 010524 005204
5600 010528 005301
5601 010532 001363
5602 010536 005306
5603 010540 001336
5604 010544 105760 000004
5605 010548 100375
5606 010552 005060 000006
5607 010556 105760 000004
5608 010560 100375
5609 010564 005060 000006
5610 010568 105760 000004
5611 010572 100375
5612 010576 005005
5613 010580 005205
5614 010584 001376
5615 010588 005760 000002

```

```

BNE .+6
MOV #40,R4
TSTB 4(R0)
BPL .+4
MOVB R4,6(R0)
INC R4
DEC R1
BNE 456
DEC R5
BNE 306
TSTB 4(R0)
BPL .+4
CLR 6(R0)
TSTB 4(R0)
BPL .+4
CLR 6(R0)
TSTB 4(R0)
BPL .+4
CLR R5
INC R5
BNE .-2
TST 2(R0)

```

```

TP READY?
BR IF NO
XMIT A ZERO.
READY?
BR IF NO
LOAD A ZERO.
READY?
BR IF NO
CLEAR RX FLAG.

```

```

(3)
(4)
(4)
(4)
(4)
(3)
(2)
(1)
5626 010602 000004
5627 010604 012737 000002 001160
5628 010612 012700 175610
5629 010616 013746 000300
5630 010622 013746 000302
5631 010626 013746 000304
5632 010632 013746 000306
5633 010636 012737 010732 000304
5634 010644 012737 000200 000306
5635 010652 012737 010760 000300
5636 010660 012737 000200 000302
5637 010666 005005
5638 010670 052760 000100 000004
5639 010676 005003
5640 010700 062703 000001
5641 010704 001375
5642 010706 104016
5643 010710 012637 000306
5644 010714 012637 000304
5645 010720 012637 000302
5646 010724 012637 000300
5647 010730 000444
5648 010732
5649 010732 110560 000006

```

```

*****
#TEST 47 DLV11 INTERRUPT TEST
#TEST OF DLV11 INTERRUPTS.
#THIS TEST WILL EXECUTE 32. INTERRUPTS ON
#BOTH THE RECEIVER AND TRANSMITTER AND
#WILL ALSO CHECK THE DATA.
*****
TST47: SCOPE
MOV #2,STIMES ;DO 2 ITERATIONS
MOV #175610,R0 ;GET DLV11 CSR
MOV 300,-(SP) ;SAVE
MOV 302,-(SP) ;VECTORS
MOV 304,-(SP) ;ON THE
MOV 306,-(SP) ;STACK.
MOV #45,304 ;SET TX VECTOR.
MOV #200,306 ;SET PRIO.
MOV #55,300 ;SET RX VECTOR
MOV #200,302 ;SET PRIO.
CLR R5 ;SET DATA POINTER TO ZERO
BIS #100,4(R0) ;SET TX IE.
15: CLR R3 ;SET TIMER TO ZERO.
25: ADD #1,R3 ;TIME WAITING FOR INTERRUPTS.
BNE 25 ;KEEP COUNTING.
ERROR 16 ;DLV11 INTERRUPT PROBLEM.
35: MOV (SP)+,306 ;RESTORE
MOV (SP)+,304 ;ALL
MOV (SP)+,302 ;DLV11
MOV (SP)+,300 ;VECTORS.
BR TST50 ;EXIT TEST

45: ;TRANSMITTER INTERRUPTS TO HERE.
MOVB R5,6(R0) ;LOAD DATA CHAR.

```

```

5648 010736 042760 000100 000004      BIC      #100,4(R0)      ;DSABLE TX INTERUPTS.
5649 010744 052760 000100 000000      BIS      #100,0(R0)      ;ENABLE RX INTERUPTS.
5650 010752 012716 010676              MOV      #1$, (SP)      ;SET RETURN
5651 010756 000002      RTI                      ;EXIT ISR.
5652
5653 010760              5$:      ;RECEIVER INTERUPTS TO HERE.
5654 010760 000240      NOP
5655 010762 116004 000002      MOV      2(R0),R4      ;GET DATA REVEIVERD
5656 010766 020504      CMP      R5,R4      ;DATA GOOD?
5657 010770 001401      BEQ      6$           ;BE IF GOOD DATA
5658 010772 104017      ERROR   17          ;DLV11 DATA ERROR (INTERUPTS)
5659 010774 005205      6$:      INC      R5          ;UPDATE DATA CHAR.
5660 010776 022705 000040      CMP      #32.,R5      ;ALL CHARS DONE?
5661 011002 001006      BNE      7$           ;BR IF NO
5662 011004 042760 000100 000000      BIC      #100,0(R0)      ;DSABLE RX INTERUPTS.
5663 011012 012716 010710      MOV      #3$, (SP)      ;SET RETURN
5664 011016 000002      RTI                      ;RETURN
5665 011020 042760 000100 000000      7$:      BIC      #100,0(R0)      ;DSABLE RX INTERUPTS
5666 011026 052760 000100 000004      BIS      #100,4(R0)      ;ENABLE TX INTERUPTS
5667 011034 012716 010676      MOV      #1$, (SP)      ;SET RETURN
5668 011040 000002      RTI
5669
5677
5678
(3)
(4)
(4)
(4)
(4)
(4)
(4)
(3)
(2) 011042 000004
(1) 011044 012737 000002 001160
5679 011052 013701 001374
5680 011056 162701 000310
5681 011062 012700 015142
5682 011066 012720 177777      1$:
5683 011072 020100      CMP      R1,R0
5684 011074 001374      BNE      1$
5685 011076 012700 015142      MOV      #CORMAX,R0
5686 011102 042710 000001      2$:      BIC      #BIT0, (R0)
5687 011106 012705 177776      MOV      #C<1>,R5
5688 011112 011004      3$:      MOV      (R0),R4
5689 011114 020504      CMP      R5,R4
5690 011116 001401      BEQ      4$
5691 011120 104020      ERROR   20
5692 011122 000261      4$:      SEC
5693 011124 006105      ROL      R5
5694 011126 000261      SEC
5695 011130 006110      ROL      (R0)
5696 011132 103767      BCS      3$
5697 011134 005720      TST      (R0)+
5698 011136 100401      BMI     +4
5699 011140 104016      ERROR   16
5700 011142 020100      CMP      R1,R0
5701 011144 001356      BNE      2$
;*****
;#TEST 50      MINI MEMORY TEST
;#MINI MEMORY TESTS
;#ALL UNUSED MEMORY WILL BE FILLED WITH ALL 1'S
;#AND THEN EACH ADDRESS WILL BE TESTED
;#WITH A FLOATING ZERO.
;#
;#*****
TST50:  SCOPE
MOV      #2,STIMES      ;DO 2 ITERATIONS
MOV      #MEMSIZ,R1      ;GET MAX MEMORY SIZE.
SUB      #310,R1        ;PROTECT ABL
MOV      #CORMAX,R0      ;GET LAST ADDRESS USED BY PROGRAM
1$:      MOV      #177777,(R0)+
CMP      R1,R0          ;ALL MEMORY FILLED?
BNE      1$
MOV      #CORMAX,R0
2$:      BIC      #BIT0, (R0)
MOV      #C<1>,R5
3$:      MOV      (R0),R4
CMP      R5,R4
BEQ      4$
ERROR   20
4$:      SEC
ROL      (R0)
BCS      3$
TST      (R0)+
BMI     +4
ERROR   16
CMP      R1,R0
BNE      2$
;GENERAL ERROR. BIT15<>1!
;ALL MEMORY DONE?

```

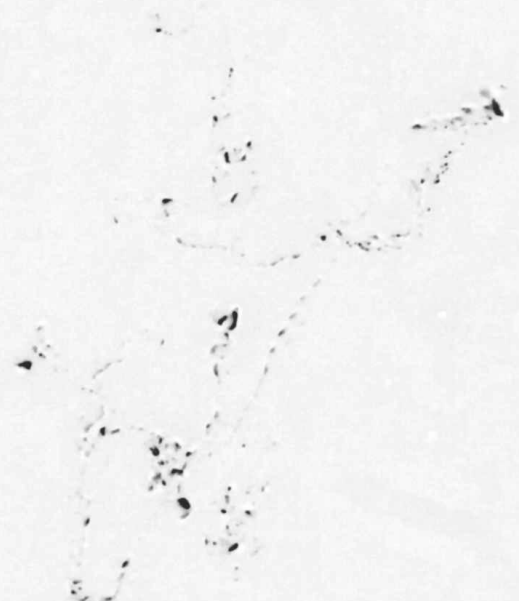
MAINDEC-11-11W03A-A
DW00A.P11 TSO

MACY11 27(732) 24-AUG-76 16:05 PAGE 55-8
MINI MEMORY TEST

804

SEQ 0040

5702



(3)	011276	001425			BEG	995			BR IF NO
(3)	011300	123737	001102	001440	CHPB	\$TSTNM,XXSCOP			HAS THIS TEST ALREADY TYPED OUT?
(3)	011306	001421			BEG	995			BR IF YES
(3)	011310	123737	001102	015140	CHPB	\$TSTNM,LASTN			LEGAL TEST 0
(3)	011316	003015			BGT	995			BR IF NO
(3)	011320	105737	001102		TSTB	\$TSTNM			TEST 0?
(3)	011324	001412			BEG	995			BR IF YES (THERE IS NO TEST 0)
(3)	011326	113737	001102	001440	MOVB	\$TSTNM,XXSCOP			LOAD FOR NEXT TIME
(3)	011334	104400	014644		TYPE	XTST			TYPE "T".
(3)	011340	013746	001440		MOV	XXSCOP,-(SP)			
(3)	011344	104401			TYPOC				
(3)	011346	104400	014610		TYPE	,XSPA			
(3)	011352								
(1)	011352	032777	040000	167556	995:	BIT	\$BIT14,\$SMR		:: LOOP ON PRESENT TEST?
(1)	011360	001101			15:	BNE	\$OVER		:: YES IF SM14=1
(1)									TESTER#####
(1)	011362	000416			#####	START OF CODE FOR THE XOR			
(1)					\$XTSTR:	BR	65		:: IF RUNNING ON THE "XOR" TESTER CHANGE
(1)									THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)	011364	013746	000004		MOV	\$ERRVEC,-(SP)			SAVE THE CONTENTS OF THE ERROR VECTOR
(1)	011370	012737	011410	000004	MOV	\$55,\$ERRVEC			SET FOR TIMEOUT
(1)	011376	005737	177060		TST	\$177060			TIME OUT ON XOR?
(1)	011402	012637	000004		MOV	(SP)+,\$ERRVEC			RESTORE THE ERROR VECTOR
(1)	011406	000453			BR	\$SVLAD			GO TO THE NEXT TEST
(1)	011410	022626			55:	CHP	(SP)+,(SP)+		CLEAR THE STACK AFTER A TIME OUT
(1)	011412	012637	000004		MOV	(SP)+,\$ERRVEC			RESTORE THE ERROR VECTOR
(1)	011416	000413			BR	75			LOOP ON THE PRESENT TEST
(1)	011420				65:;	#####	END OF CODE FOR THE XOR		TESTER#####
(1)	011420	105737	001103		25:	TSTB	\$ERFLG		:: HAS AN ERROR OCCURRED?
(1)	011424	001421			BEG	35			BR IF NO
(1)	011426	123737	001115	001103	CHPB	\$ERMAX,\$ERFLG			MAX. ERRORS FOR THIS TEST OCCURRED?
(1)	011434	101015			BHI	35			BR IF NO
(1)	011436	032777	001000	167472	BIT	\$BIT09,\$SMR			LOOP ON ERROR?
(1)	011444	001404			BEG	45			BR IF NO
(1)	011446	013737	001110	001106	75:	MOV	\$LPERR,\$LPADR		:: SET LOOP ADDRESS TO LAST SCOPE
(1)	011454	000443			BR	\$OVER			
(1)	011456	105037	001103		45:	CLRB	\$ERFLG		:: ZERO THE ERROR FLAG
(1)	011462	005037	001160		CLR	\$TIMES			CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)	011466	000415			BR	15			ESCAPE TO THE NEXT TEST
(1)	011470	032777	004000	167440	35:	BIT	\$BIT11,\$SMR		INHIBIT ITERATIONS?
(1)	011476	001011			BNE	15			BR IF YES
(1)	011500	005737	001100		TST	\$PASS			IF FIRST PASS OF PROGRAM
(1)	011504	001406			BEG	15			INHIBIT ITERATIONS
(1)	011506	005237	001104		INC	\$ICNT			INCREMENT ITERATION COUNT
(1)	011512	023737	001160	001104	CHP	\$TIMES,\$ICNT			CHECK THE NUMBER OF ITERATIONS MADE
(1)	011520	002021			BGE	\$OVER			BR IF MORE ITERATION REQUIRED
(1)	011522	012737	000001	001104	15:	MOV	\$1,\$ICNT		REINITIALIZE THE ITERATION COUNTER
(1)	011530	013737	011600	001160	MOV	\$MXCNT,\$TIMES			SET NUMBER OF ITERATIONS TO DO
(1)	011536	105237	001102		SSVLAD:	INCB	\$TSTNM		COUNT TEST NUMBERS
(1)	011542	011637	001106		MOV	(SP),\$LPADR			SAVE SCOPE LOOP ADDRESS
(1)	011546	011637	001110		MOV	(SP),\$LPERR			SAVE ERROR LOOP ADDRESS
(1)	011552	005037	001162		CLR	\$ESCAPE			CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)	011556	112737	000001	001115	MOVB	\$1,\$ERMAX			ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)	011564	013777	001102	167346	SOVER:	MOV	\$TSTNM,\$DISPLAY		DISPLAY TEST NUMBER
(1)	011572	013716	001106		MOV	\$LPADR,(SP)			FUDGE RETURN ADDRESS
(1)	011576	000002			RTI				FIXES PS
(1)	011600	000100			SMXCNT:	100			MAX. NUMBER OF ITERATIONS


```

(3) 012016 020120          CMP      R1,(R0)+      ;LOOK FOR CURRENT TEST.
(3) 012020 001379          BNE     91$           ;BR IF NOT FOUND
(3) 012022 011037 012076  MOV     (R0),92$     ;SAVE TEST PC
(3) 012026 062737 000002 012076  ADD     82,92$       ;POP PAST SCOPE
(3) 012034 012737 000001 001104  MOV     81,STCNT     ;
(3) 012042 013737 011600 001160  MOV     SMCNT,STIMES ;
(3) 012050 012706 001100          MOV     8STACK,SP    ;SET SP
(3) 012054 012746 000000          MOV     80,-(SP)     ;SET PTYO
(3) 012060 013746 012076          MOV     92$,-(SP)   ;
(3) 012064 000137 011536          JMP     $SVLAD       ;GO INTO SCOPE ROUTINE
(3) 012070 012601          90$:  MOV     (SP)+,R1    ;RESTORE R1
(3) 012072 012600          MOV     (SP)+,R0    ;RESTORE R0
(3) 012074 000002          RTI                    ;EXIT
(3) 012076 000000          92$:  0                    ;SAVE TEST PC

5761 012100 010037 001420  XERR1: MOV     R0,SAVR0
(1) 012104 010137 001422  MOV     R1,SAVR1
(1) 012110 010237 001424  MOV     R2,SAVR2
(1) 012114 010337 001426  MOV     R3,SAVR3
(1) 012120 010437 001430  MOV     R4,SAVR4
(1) 012124 010537 001432  MOV     R5,SAVR5
(1) 012130 113737 001102 001434  MOVB   STSTIM,MSKTST

;*****
(2)
(2)
(2) .SBTTL ERROR MESSAGE TYPEOUT ROUTINE
(2) ;#THIS ROUTINE USES THE "ITEM CONTROL BYTE" (SITEMB) TO DETERMINE WHICH
(2) ;#ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (SERRTB),
(2) ;#AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(2)
(2) SERRTYP:
(2) 012136 104400 001171      TYPE    SCRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(2) 012142 010046          MOV     R0,-(SP)      ;SAVE R0
(2) 012144 005000          CLR     R0            ;PICKUP THE ITEM INDEX
(2) 012146 153700 001114      BISB   2#SITEMB,R0
(2) 012152 001004          BNE     1$           ;IF ITEM NUMBER IS ZERO, JUST
(2) 012154 013746 001116      MOV     SERRPC,-(SP) ;TYPE THE PC OF THE ERROR
(2) 012160 104401          TYP0C          ;SAVE SERRPC FOR TYPEOUT
(2) 012162 000445          BR     10$          ;ERROR ADDRESS
(2) 012164 005300          1$:  DEC     R0        ;GO TYPE--OCTAL ASCII(ALL DIGITS)
(2) 012166 006300          ASL    R0           ;GET OUT
(2) 012170 006300          ASL    R0           ;ADJUST THE INDEX SO THAT IT WILL
(2) 012172 006300          ASL    R0           ;WORK FOR THE ERROR TABLE
(2) 012174 062700 001174      ADD     #SERRTB,R0  ;FORM TABLE POINTER
(2) 012200 012037 012210      MOV     (R0)+,2$    ;PICKUP "ERROR MESSAGE" POINTER
(2) 012204 001404          BEQ    3$           ;SKIP TYPEOUT IF NO POINTER
(2) 012206 104400          TYPE    "ERROR MESSAGE"
(2) 012210 000000          2$:  .WORD 0          ;"ERROR MESSAGE" POINTER GOES HERE
(2) 012212 104400 001171      TYPE    SCRLF        ;;"CARRIAGE RETURN" & "LINE FEED"
(2) 012216 012037 012226      MOV     (R0)+,4$    ;PICKUP "DATA HEADER" POINTER
(2) 012222 001404          BEQ    5$           ;SKIP TYPEOUT IF 0
(2) 012224 104400          TYPE    "DATA HEADER"
(2) 012226 000000          3$:  .WORD 0          ;"DATA HEADER" POINTER GOES HERE
(2) 012230 104400 001171      TYPE    SCRLF        ;;"CARRIAGE RETURN" & "LINE FEED"

```

```

(2) 012234 010146      5S:  MOV  R1, -(SP)      ;; SAVE R1
(2) 012236 012001      MOV  (R0)+, R1      ;; PICKUP "DATA TABLE" POINTER
(2) 012240 001415      BEQ  9S              ;; BR IF NO DATA TO BE TYPED
(2) 012242 012000      MOV  (R0)+, R0      ;; PICKUP "DATA FORMAT" POINTER
(2) 012244 105720      6S:  TSTB (R0)+       ;; "OCTAL" OR "DECIMAL"
(2) 012246 001003      BNE  7S              ;; BR IF DECIMAL
(2) 012250 013146      MOV  @ (R1)+, -(SP) ;; SAVE @ (R1)+ FOR TYPEOUT
(2) 012252 104401      TYPOC                ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
(2) 012254 000402      BR   8S
(2) 012256 013146      7S:  MOV  @ (R1)+, -(SP) ;; SAVE @ (R1)+ FOR TYPEOUT
(2) 012260 104404      TYPOS                ;; GO TYPE--DECIMAL ASCII WITH SIGN
(2) 012262 005711      8S:  TST  (R1)         ;; IS THERE ANOTHER NUMBER?
(2) 012264 001403      BEQ  9S              ;; BR IF NO
(2) 012266 104400      TYPE 11S            ;; TYPE TWO(2) SPACES
(2) 012272 000764      BR   6S              ;; LOOP
(2) 012274 012601      9S:  MOV  (SP)+, R1    ;; RESTORE R1
(2) 012276 012600      10S: MOV  (SP)+, R0   ;; RESTORE R0
(2) 012300 104400      TYPE $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
(2) 012304 000207      RTS  PC              ;; RETURN
(2) 012306 020040      11S: .ASCIZ / /        ;; TWO(2) SPACES
(2) 012312 012312      .EVEN

```

5762 ;*****

.SBTTL TYPE ROUTINE

```

;#ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;#THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;#NOTE1: SNUL contains the character to be used as the filler character.
;#NOTE2: SFILLS contains the number of filler characters required.
;#NOTE3: SFILLC contains the character to fill after.
;#
;#CALL:
;#1) USING A TRAP INSTRUCTION
;# TYPE ,MESADR ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;#OR
;# TYPE MESADR
;#

```

```

(1) 012312 105737 001155 STYPE: TSTB STPFLG      ;; IS THERE A TERMINAL?
(1) 012316 100002      BPL  1S              ;; BR IF YES
(1) 012320 000000      HALT                ;; HALT HERE IF NO TERMINAL
(1) 012322 000407      BR   3S              ;; LEAVE
(1) 012324 010046      1S:  MOV  R0, -(SP)    ;; SAVE R0
(1) 012326 017600 000002      MOV  @2(SP), R0     ;; GET ADDRESS OF ASCIZ STRING
(1) 012332 112046      2S:  MOVB (R0)+, -(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
(1) 012334 001005      BNE  4S              ;; BR IF IT ISN'T THE TERMINATOR
(1) 012336 005726      TST  (SP)+           ;; IF TERMINATOR POP IT OFF THE STACK
(1) 012340 012600      60S: MOV  (SP)+, R0    ;; RESTORE R0
(1) 012342 062716 000002      3S:  ADD  #2, (SP)    ;; ADJUST RETURN PC
(1) 012346 000002      RTI                  ;; RETURN
(1) 012350 122716 000011      4S:  CMPB #HT, (SP)   ;; BRANCH IF <HT>
(1) 012354 001426      BEQ  8S              ;; BRANCH IF NOT <CRLF>
(1) 012356 122716 000200      CMPB #TCRLF, (SP)

```

```

(1) 012362 001004 BNE 55
(1) 012364 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
(1) 012366 104400 TYPE ;;TYPE A CR AND LF
(1) 012370 001171 SCRLF
(1) 012372 000757 BR 25 ;;GET NEXT CHARACTER
(1) 012374 004737 012456 55: JSR PC,STYPEC ;;GO TYPE THIS CHARACTER
(1) 012400 123726 001154 65: CMPB SFILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
(1) 012404 001352 BNE 25 ;;IF NO GO GET NEXT CHAR.
(1) 012406 013746 001152 MOV SNULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
(1) ;;AND THE NULL CHAR.
(1) 012412 105366 000001 75: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
(1) 012416 002770 BLT 65 ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1) 012420 004737 012456 JSR PC,STYPEC ;;GO TYPE A NULL
(1) 012424 105337 012522 DECB SCHARCNT ;;DO NOT COUNT AS A COUNT
(1) 012430 000770 BR 75 ;;LOOP
  
```

;;HORIZONTAL TAB PROCESSOR

```

(1) 012432 112716 000040 85: MOVB #40,(SP) ;;REPLACE TAB WITH SPACE
(1) 012436 004737 012456 95: JSR PC,STYPEC ;;TYPE A SPACE
(1) 012442 132737 000007 012522 BITB #7,SCHARCNT ;;BRANCH IF NOT AT
(1) 012450 001372 BNE 95 ;;TAB STOP
(1) 012452 005726 TST (SP)+ ;;POP SPACE OFF STACK
(1) 012454 000726 BR 25 ;;GET NEXT CHARACTER
(1) 012456 105777 166464 STYPEC: TSTB #STPS ;;WAIT UNTIL PRINTER IS READY
(1) 012462 100375 BPL STYPEC
(1) 012464 116677 000002 166456 MOVB 2(SP),#STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 012472 122766 000015 000002 CMPB #15,2(SP) ;;BRANCH IF
(1) 012500 001003 BNE 15 ;;NOT <CR>
(1) 012502 105037 012522 CLRB SCHARCNT
(1) 012506 000406 BR STYPEX ;;EXIT
(1) 012510 122766 000012 000002 15: CMPB #12,2(SP) ;;BRANCH IF
(1) 012516 002002 BGE STYPEX ;;<LF>
(1) 012520 105227 INCB (PC)+ ;;INC SPACE
(1) 012522 000000 SCHARCNT: WORD 0 ;;COUNT
(1) 012524 000207 STYPEX: RTS PC
  
```

```

;;
;;THT=11
;;TCRLF=200
  
```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

;;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
;;OCTAL (ASCII) NUMBER AND TYPE IT.
;;STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
;;CALL:
  
```

```

;; MOV NUM,-(SP) ;;NUMBER TO BE TYPED
;; TYPOS ;;CALL FOR TYPEOUT
;; .BYTE N ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;; .BYTE M ;;M=1 OR 0
;; ;;1=TYPE LEADING ZEROS
;; ;;0=SUPPRESS LEADING ZEROS
  
```

```

;;*STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
  
```

5763

```

(1)                                     :#STYPOS OR STYPOC
(1)                                     :#CALL:
(1)                                     :#      MOV      NUM,-(SP)          ;;NUMBER TO BE TYPED
(1)                                     :#      TYPON                    ;;CALL FOR TYPEOUT
(1)                                     :#
(1)                                     :#STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                     :#CALL:
(1)                                     :#      MOV      NUM,-(SP)          ;;NUMBER TO BE TYPED
(1)                                     :#      STYPOC                    ;;CALL FOR TYPEOUT
(1)
(1) 012526 017646 000000          STYPOS: MOV      2(SP),-(SP)          ;;PICKUP THE MODE
(1) 012532 116637 000001 012751  MOVB     1(SP),SOFILL          ;;LOAD ZERO FILL SWITCH
(1) 012540 112637 012753          MOVB     (SP)+,SOMODE+1        ;;NUMBER OF DIGITS TO TYPE
(1) 012544 062716 000002          ADD      #2,(SP)              ;;ADJUST RETURN ADDRESS
(1) 012550 000406                    BR      STYPOC
(1) 012552 112737 000001 012751  STYPOC: MOVB     #1,SOFILL          ;;SET THE ZERO FILL SWITCH
(1) 012560 112737 000006 012753  MOVB     #6,SOMODE+1          ;;SET FOR SIX(6) DIGITS
(1) 012566 112737 000005 012750  STYPON: MOVB     #5,SOCNT          ;;SET THE ITERATION COUNT
(1) 012574 010346                    MOV      R3,-(SP)              ;;SAVE R3
(1) 012576 010446                    MOV      R4,-(SP)              ;;SAVE R4
(1) 012600 010546                    MOV      R5,-(SP)              ;;SAVE R5
(1) 012602 113704 012753          MOVB     SOMODE+1,R4          ;;GET THE NUMBER OF DIGITS TO TYPE
(1) 012606 005404                    NEG      R4
(1) 012610 062704 000006          ADD      #6,R4                ;;SUBTRACT IT FOR MAX. ALLOWED
(1) 012614 110437 012752          MOVB     R4,SOMODE            ;;SAVE IT FOR USE
(1) 012620 113704 012751          MOVB     SOFILL,R4            ;;GET THE ZERO FILL SWITCH
(1) 012624 016605 000012          MOV      12(SP),R5            ;;PICKUP THE INPUT NUMBER
(1) 012630 005003                    CLR      R3                    ;;CLEAR THE OUTPUT WORD
(1) 012632 006105                    1S:   ROL      R5              ;;ROTATE MSB INTO "C"
(1) 012634 000404                    BR      3S                    ;;GO DO MSB
(1) 012636 006105                    2S:   ROL      R5              ;;FORM THIS DIGIT
(1) 012640 006105
(1) 012642 006105
(1) 012644 010503                    MOV      R5,R3
(1) 012646 006103                    3S:   ROL      R3              ;;GET LSB OF THIS DIGIT
(1) 012650 105337 012752          DECB    SOMODE                ;;TYPE THIS DIGIT?
(1) 012654 100016                    BPL     7S                    ;;BR IF NO
(1) 012656 042703 177770          BIC     #177770,R3            ;;GET RID OF JUNK
(1) 012662 001002                    BNE     4S                    ;;TEST FOR 0
(1) 012664 005704                    TST     R4                    ;;SUPPRESS THIS 0?
(1) 012666 001403                    BEQ     5S                    ;;BR IF YES
(1) 012670 005204                    4S:   INC     R4                ;;DON'T SUPPRESS ANYMORE 0'S
(1) 012672 052703 000060          BIS     #'0,R3                ;;MAKE THIS DIGIT ASCII
(1) 012676 052703 000040          5S:   BIS     #' ',R3           ;;MAKE ASCII IF NOT ALREADY
(1) 012702 110337 012746          MOVB     R3,R5                ;;SAVE FOR TYPING
(1) 012706 104400 012746          TYPE    R5                    ;;GO TYPE THIS DIGIT
(1) 012712 105337 012750          7S:   DECB    SOCNT            ;;COUNT BY 1
(1) 012716 003347                    BGT     2S                    ;;BR IF MORE TO DO
(1) 012720 002402                    BLT     6S                    ;;BR IF DONE
(1) 012722 005204                    INC     R4                    ;;INSURE LAST DIGIT ISN'T A BLANK
(1) 012724 000744                    BR      2S                    ;;GO DO THE LAST DIGIT
(1) 012726 012605                    6S:   MOV     (SP)+,R5           ;;RESTORE R5
(1) 012730 012604                    MOV     (SP)+,R4           ;;RESTORE R4
(1) 012732 012603                    MOV     (SP)+,R3           ;;RESTORE R3
(1) 012734 016666 000002 000004  MOV     2(SP),4(SP)          ;;SET THE STACK FOR RETURNING
(1) 012742 012616                    MOV     (SP)+,(SP)

```



```

(1) 013114 105726          9S:  TSTB  (SP)+      ;; WAS THE LSD THE FIRST NON-ZERO?
(1) 013116 100003          BPL  9S          ;; BR IF NO
(1) 013120 116663 177777 177776  MOVB -1(SP),-2(R3) ;; YES--SET THE SIGN FOR TYPING
(1) 013126 105013          CLRB (R3)       ;; SET THE TERMINATOR
(3) 013130 012605          MOV  (SP)+,R5   ;; POP STACK INTO R5
(3) 013132 012603          MOV  (SP)+,R3   ;; POP STACK INTO R3
(3) 013134 012602          MOV  (SP)+,R2   ;; POP STACK INTO R2
(3) 013136 012601          MOV  (SP)+,R1   ;; POP STACK INTO R1
(3) 013140 012600          MOV  (SP)+,R0   ;; POP STACK INTO R0
(1) 013142 104400 013170  TYPE  SDBLK      ;; NOW TYPE THE NUMBER
(1) 013146 016666 000002 000004  MOV  2(SP),4(SP) ;; ADJUST THE STACK
(1) 013154 012616          MOV  (SP)+,(SP)
(1) 013156 000002          RTI              ;; RETURN TO USER
(1) 013160 023420          SOTBL: 10000.
(1) 013162 001750          1000.
(1) 013164 000144          100.
(1) 013166 000012          10.
(1) 013170 000004          SDBLK: .BLKW 4
5765 ;*****
(1) .SBTTL TTY INPUT ROUTINE
(1) ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1) ;;CALL:
(1) ;; RDCHR          ;; INPUT A SINGLE CHARACTER FROM THE TTY
(1) ;; RETURN HERE   ;; CHARACTER IS ON THE STACK
(1)
(1)
(1) 013200 011646          SRDCHR: MOV  (SP)-,(SP) ;; PUSH DOWN THE PC
(1) 013202 016666 000004 000002  MOV  4(SP),2(SP) ;; SAVE THE PS
(1) 013210 105777 165726 1S:  TSTB  2STKS ;; WAIT FOR
(1) 013214 100375          BPL  1S         ;; A CHARACTER
(1) 013216 117766 165722 000004  MOVB 2STKB,4(SP) ;; READ THE TTY
(1) 013224 042766 177600 000004  BIC  8+C(177),4(SP) ;; GET RID OF JUNK IF ANY
(1) 013232 000002          RTI              ;; GO BACK TO USER
(2) ;*****
(1) ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1) ;;CALL:
(1) ;; RDLIN          ;; INPUT A STRING FROM THE TTY
(1) ;; RETURN HERE   ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1) ;;              ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1) 013234 010346          SRDLIN: MOV  R3,-(SP) ;; SAVE R3
(1) 013236 012703 013342 1S:  MOV  8$TTYIN,R3 ;; GET ADDRESS
(1) 013242 022703 013352 2S:  CMP  8$TTYIN+8.,R3 ;; BUFFER FULL?
(1) 013246 101405          BLOS 4S        ;; BR IF YES
(1) 013250 104405          RDCHR          ;; GO READ ONE CHARACTER FROM THE TTY
(1) 013252 112613          MOVB (SP)+,(R3) ;; GET CHARACTER
(1) 013254 122713 000177  CMPB 8177,(R3) ;; IS IT A RUBOUT
(1) 013260 001003          BNE 3S        ;; SKIP IF NOT
(1) 013262 104400 001170 4S:  TYPE  SQUES ;; TYPE A '?'
(1) 013266 000763          BR 1S         ;; CLEAR THE BUFFER AND LOOP
(1) 013270 111337 013340 3S:  MOVB (R3),9S   ;; ECHO THE CHARACTER
(1) 013274 104400 013340  TYPE  9S
(1) 013300 122723 000015  CMPB 815,(R3)+ ;; CHECK FOR RETURN
(1) 013304 001356          BNE 2S        ;; LOOP IF NOT RETURN

```

```

(1) 013306 105063 177777          CLRB    -1(R3)          ;; CLEAR RETURN (THE 15)
(1) 013312 104400 001172          TYPE    SLF            ;; TYPE A LINE FEED
(1) 013316 012603                   MOV     (SP)+,R3       ;; RESTORE R3
(1) 013320 011646                   MOV     (SP),-(SP)     ;; ADJUST THE STACK AND PUT ADDRESS OF THE
(1) 013322 016666 000004 000002   MOV     4(SP),2(SP)    ;; FIRST ASCII CHARACTER ON IT
(1) 013330 012766 013342 000004   MOV     @STTYIN,4(SP)
(1) 013336 000002                   RTI                    ;; RETURN
(1) 013340 000          9S:      .BYTE    0              ;; STORAGE FOR ASCII CHAR. TO TYPE
(1) 013341 000          .BYTE    0              ;; TERMINATOR
(1) 013342 000010   STTYIN: .BLKB    8.      ;; RESERVE 8 BYTES FOR TTY INPUT
5766 ;*****
(1)                                     .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)                                     ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                     ;*CHANGE IT TO BINARY.
(1)                                     ;*CALL:
(1)                                     ;*      RDOCT          ;; READ AN OCTAL NUMBER
(1)                                     ;*      RETURN HERE  ;; LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                     ;*                                     ;; HIGH ORDER BITS ARE IN SHIOCT
(1) 013352 011646 000004 000002   SRDOCT: MOV     (SP),-(SP) ;; PROVIDE SPACE FOR THE
(1) 013354 016666                   MOV     4(SP),2(SP)    ;; INPUT NUMBER
(3) 013362 010046                   MOV     R0,-(SP)       ;; PUSH R0 ON STACK
(3) 013364 010146                   MOV     R1,-(SP)       ;; PUSH R1 ON STACK
(3) 013366 010246                   MOV     R2,-(SP)       ;; PUSH R2 ON STACK
(1) 013370 104406   1S:      RDLIN          ;; READ AN ASCII LINE
(1) 013372 012600                   MOV     (SP)+,R0       ;; GET ADDRESS OF 1ST CHARACTER
(1) 013374 005001                   CLR     R1              ;; CLEAR DATA WORD
(1) 013376 005002                   CLR     R2
(1) 013400 112046   2S:      MOVSB    (R0)+,-(SP)  ;; PICKUP THIS CHARACTER
(1) 013402 001412                   BEQ     3S              ;; IF ZERO GET OUT
(1) 013404 006301                   ASL     R1              ;; #2
(1) 013406 006102                   ROL     R2
(1) 013410 006301                   ASL     R1              ;; #4
(1) 013412 006102                   ROL     R2
(1) 013414 006301                   ASL     R1              ;; #8
(1) 013416 006102                   ROL     R2
(1) 013420 042716 177770   BIC     @C7(SP)        ;; STRIP THE ASCII JUNK
(1) 013424 062601                   ADD     (SP)+,R1       ;; ADD IN THIS DIGIT
(1) 013426 000764                   BR      2S              ;; LOOP
(1) 013430 005726   3S:      TST     (SP)+          ;; CLEAN TERMINATOR FROM STACK
(1) 013432 010166 000012   MOV     R1,12(SP)     ;; SAVE THE RESULT
(1) 013436 010237 013452   MOV     R2,SHIOCT
(3) 013442 012602                   MOV     (SP)+,R2       ;; POP STACK INTO R2
(3) 013444 012601                   MOV     (SP)+,R1       ;; POP STACK INTO R1
(3) 013446 012600                   MOV     (SP)+,R0       ;; POP STACK INTO R0
(1) 013450 000002                   RTI                    ;; RETURN
(1) 013452 000000   SHIOCT: .WORD    0      ;; HIGH ORDER BITS GO HERE
  
```


5770	013516			
5778	013516	000400	000440	000167
5779	013534	102507	016005	000002
5780	013552	001100	120527	000101
5781	013570	126560	173064	000002
5782	013602	000	201	202
5783	013612	210	011	012
5784	013622	012700	175610	112710
5785	013640	004767	000142	020527
5786	013656	005705	001366	004767
5787	013674	004767	000174	010501
5788	013712	000240	003407	060205
5789	013730	000164	000733	004767
5790	013746	000772	105703	001403
5791	013764	000144	000723	004767
5792	014002	000126	010107	004767
5793	014020	042716	007760	004767
5794	014036	006305	042705	170017
5795	014054	032710	100200	001622
5796	014072	000207	004767	177706
5797	014110	105005	150405	000207
5798	014126	000102	000402	112705
5799	014144	110560	000006	000207
5800				
5801	014152	240	037	015
5802	014160	005004	012700	177500
5803	014176	012702	000375	112103
5804	014214	105202	100772	116012
5805	014232	005002	120312	001377
5806				

ROMMAP:

.WORD	000400,000440,000167,000432,132710,100200,001775
.WORD	102507,016005,000002,032705,177560,001406,005703
.WORD	001100,120527,000101,001079,000207,042705,177760
.WORD	126560,173064,000002,001066,000207
.BYTE	000,201,202,003,204,005,006,207
.BYTE	210,011,012,213,014,215,216,017
.WORD	012700,175610,112710,000006,12706,077776,005003
.WORD	004767,000142,020527,040001,001372,004767,00130
.WORD	005705,001366,004767,000206,010502,162702,000004
.WORD	004767,000174,010501,005702,001431,160705,022705
.WORD	000240,003407,060205,022705,177704,003003,004767
.WORD	000164,000733,004767,000046,005702,100402,110521
.WORD	000772,105703,001403,004767,000144,000720,004767
.WORD	000144,000723,004767,000012,105703,001366,004767
.WORD	000126,010107,004767,177514,010546,111666,000001
.WORD	042716,007760,004767,177476,006305,006305,006305
.WORD	006305,042705,170017,052605,060503,005302,000207
.WORD	032710,100200,001622,100734,016005,000002,100731
.WORD	000207,004767,177706,010504,004767,177700,000305
.WORD	105005,150405,000207,112705,000117,000405,112705
.WORD	000102,000402,112705,000015,105760,000004,100375
.WORD	110560,000006,000207
.BYTE	240,037,015,005,024,224
.WORD	005004,012700,177500,000005,010410,012701,173434
.WORD	012702,000375,112103,112110,100407,130310,001776
.WORD	105202,100772,116012,000002,000771,005710,100756
.WORD	005002,120312,001377,00112

SNO6	=	000100	47168			
SNO7	=	000200	47168			
SNO8	=	000400	47168			
SNO9	=	001000	47168			
SNO10	=	000002	47168			
SNO11	=	002000	47168			
SNO12	=	004000	47168			
SNO13	=	010000	47168			
SNO14	=	020000	47168			
SNO15	=	040000	47168			
SNO16	=	100000	47168			
SNO17	=	000004	47168			
SNO18	=	000010	47168			
SNO19	=	000020	47168			
SNO20	=	000040	47168			
SNO21	=	000100	47168			
SNO22	=	000200	47168			
SNO23	=	000400	47168			
SNO24	=	001000	47168			
TBITVE	=	000014	47168			
TCALF	=	000200	57628			
TEST.T	=	015012	4923	5760	58718	
THT	=	000011	57628			
TKVEC	=	000060	47168			
TPVEC	=	000064	47168			
TRAPVE	=	000034	47168	4895#		
TRTVEC	=	000014	47168			
TSTFLG	=	001436	48378	4894#	4911	4913#
TST1	=	004470	4912	49358	5758	5883
TST10	=	005544	51788	5883		
TST11	=	005632	52178	5883		
TST12	=	005674	52178	5883		
TST13	=	005736	52178	5883		
TST14	=	006000	52178	5883		
TST15	=	006042	52178	5883		
TST16	=	006104	52178	5883		
TST17	=	006146	52178	5883		
TST2	=	004564	49618	5883		
TST20	=	006210	52178	5883		
TST21	=	006252	52178	5883		
TST22	=	006314	52178	5883		
TST23	=	006356	52178	5883		
TST24	=	006420	52178	5883		
TST25	=	006462	52178	5883		
TST26	=	006524	52178	5883		
TST27	=	006566	52178	5883		
TST3	=	004662	49878	5883		
TST30	=	006630	52178	5883		
TST31	=	006672	52258	5883		
TST32	=	006726	52428	5883		
TST33	=	006770	52648	5883		
TST34	=	007104	53038	5883		
TST35	=	007144	53208	5883		
TST36	=	007232	53618	5883		
TST37	=	007300	53618	5883		
TST4	=	004720	50078	5883		

.SAPTH	44208		
.SAPTY	45838		
.SASTA	44678		
.SCATC	4788	47028	4717
.SCHTA	5808	47028	4720
.SDB2D	37368		
.SDB2O	38608		
.SDIV	36318		
.SEOP	15848	47028	5758
.SERRO	20028	47028	5760
.SERRT	21958	47028	5761
.SMULT	35678		
.SPOWE	32808		
.SRAND	33438		
.SRDOE	29738		
.SRDOC	28818	47048	5766
.SREAO	26658	47048	5765
.SR2AZ	40048		
.SSAVE	30498		
.SSB2D	38218		
.SSB2O	39238		
.SSCOP	17948	47038	5759
.SSIZE	34058		
.SSUPR	39618		
.STRAP	31498	47038	5768
.STYPB	25808		
.STYPD	25028	47038	5764
.STYPE	22838	47038	5762
.STYPO	24058	47038	5763
.S4OCA	5098		

ADD	4925	4944	5110	5158	5166	5418	5457	5476	5515	5523	5555	5563	5637	5760	5761
ASL	5762	5763	5764	5766	5866										
ASLB	5761	5766	5768												
BCC	5764														
BCS	5696														
BEQ	4896	4912	4917	4973	4994	5050	5060	5107	5121	5149	5217	5231	5249	5276	5283
	5286	5292	5309	5361	5457	5568	5584	5657	5690	5758	5759	5760	5761	5762	5763
	5766	5826	5863												
BGE	5759	5762													
BGT	5758	5759	5763	5764											
BHI	4919	5759													
BIC	5056	5123	5217	5361	5382	5457	5479	5648	5662	5665	5686	5758	5763	5765	5766
	5813														
BIS	5101	5307	5380	5383	5386	5477	5480	5483	5520	5635	5649	5666	5760	5763	5764
BISB	5761														
BIT	5759	5760													
BITB	5762														
BLOS	5115	5765													
BLT	5762	5763	5764												
BMI	5392	5489	5528	5553	5561	5698	5764								
BNE	4895	4896	4946	4998	5016	5055	5063	5075	5103	5157	5162	5168	5187	5234	5329
	5423	5524	5556	5564	5571	5573	5594	5601	5603	5614	5638	5661	5684	5701	5759
	5760	5761	5762	5763	5764	5765	5815	5819							
BPL	5108	5112	5253	5575	5578	5588	5591	5597	5605	5608	5611	5760	5762	5763	5764
	5765														
BR	4895	4896	4947	5017	5064	5077	5105	5117	5146	5159	5164	5188	5294	5330	5389
	5399	5424	5486	5496	5526	5644	5759	5761	5762	5763	5764	5765	5766		
CLC	4921														
CLR	4895	4898	4902	4904	4905	4906	4913	4968	5058	5070	5098	5119	5142	5143	5227
	5244	5267	5306	5378	5390	5396	5474	5487	5493	5516	5519	5521	5532	5548	5549
	5559	5606	5609	5612	5634	5636	5758	5759	5761	5763	5764	5766			
CLRB	5759	5762	5764	5765											
CMP	4895	4896	4918	4971	4977	4993	4996	5049	5078	5114	5124	5147	5148	5156	5165
	5167	5217	5230	5248	5275	5282	5291	5361	5395	5457	5492	5531	5554	5562	5567
	5583	5593	5656	5660	5683	5689	5700	5759	5760	5764	5765	5814	5818	5862	
CMPB	5759	5762	5765												
COM	5251														
DEC	4945	4997	5015	5054	5062	5106	5120	5186	5328	5422	5572	5600	5602	5758	5761
DEC8	4972	5762	5763												
EXT	4716														
HALT	4717	5760	5762	5864											
INC	4894	4896	5074	5102	5233	5270	5582	5599	5613	5659	5758	5759	5760	5763	5764
INCB	5252	5271	5273	5287	5289	5391	5488	5527	5570	5759	5760	5762			
IOT	4716														
JMP	4717	4719	4927	5758	5760	5817									
JSR	5758	5760	5762												
MOV	4895	4897	4899	4900	4901	4903	4907	4908	4909	4916	4923	4926	4936	4937	4938
	4939	4940	4941	4949	4951	4952	4962	4963	4964	4965	4966	4967	4974	4978	4979
	4988	4989	4990	4991	4992	5008	5009	5010	5011	5012	5013	5019	5021	5022	5038
	5040	5041	5042	5043	5044	5045	5046	5047	5048	5052	5057	5065	5067	5068	5071
	5072	5079	5080	5092	5093	5094	5095	5096	5097	5100	5109	5118	5125	5126	5138
	5139	5140	5141	5153	5155	5169	5170	5179	5180	5181	5182	5183	5184	5190	5192
	5193	5217	5225	5226	5228	5229	5242	5243	5245	5246	5247	5265	5266	5268	5269
	5278	5281	5303	5304	5305	5308	5321	5322	5323	5324	5325	5326	5332	5334	5335
	5361	5374	5375	5376	5377	5379	5385	5393	5397	5398	5400	5401	5403	5414	5415

	5416	5417	5419	5420	5426	5428	5429	5457	5470	5471	5472	5473	5475	5482	5490
	5494	5495	5497	5498	5500	5512	5513	5514	5517	5518	5529	5533	5534	5545	5546
	5547	5566	5580	5581	5585	5586	5595	5624	5625	5626	5627	5628	5629	5630	5631
	5632	5633	5640	5641	5642	5643	5650	5663	5667	5678	5679	5681	5682	5685	5687
	5688	5758	5759	5760	5761	5762	5763	5764	5765	5766	5768	5812	5816	5821	5827
	5859	5861	5865	5867											
MOV8	4895	4920	5272	5274	5279	5280	5288	5290	5558	5576	5579	5589	5592	5598	5647
	5655	5759	5760	5761	5762	5763	5764	5765	5766	5768					
MTPS	4910	4976	5039	5073	5081	5099	5127	5361	5373	5404	5457	5469	5501	5511	5535
	5760	5862													
NEG	5113	5763	5764												
NOP	4969	5295	5381	5384	5387	5478	5481	5484	5654	5758					
RESET	5307	5758	5760												
ROL	4922	5693	5695	5763	5766										
RTI	4950	4975	5020	5069	5122	5191	5333	5394	5427	5491	5530	5651	5664	5668	5759
	5760	5762	5763	5764	5765	5766	5829	5868							
RTS	5761	5762	5768												
SEC	5692	5694													
SUB	5111	5680	5760	5764	5860										
TRAP	5768														
TST	4895	4911	4942	4943	5014	5059	5144	5145	5154	5160	5161	5185	5217	5327	5361
	5421	5457	5550	5551	5615	5697	5759	5760	5761	5762	5763	5764	5766	5768	5828
TSTB	5285	5552	5560	5574	5577	5587	5590	5596	5604	5607	5610	5759	5761	5762	5764
	5765	5825													
.ASCII	4720	4859	4860	4861	4862										
.ASCIZ	4720	4843	4844	4845	4846	4847	4848	4849	4850	4851	4852	4853	4854	4855	4856
	4857	4858	4864	4865	4866	4867	4868	4869	4870	4871	4872	4873	4874	4875	4876
	4896	5758	5761	5833	5834	5835	5836	5837							
.BLKB	5765														
.BLKW	5764														
.BYTE	4720	4887	5758	5763	5765	5782	5783	5801							
.ENABL	4	4701													
.END	5890														
.ENDC	4713	4715	4716	4717	4720	4893	4895	4896	4912	4935	4961	4987	5007	5038	5091
	5138	5178	5217	5225	5242	5264	5303	5320	5361	5372	5413	5457	5468	5510	5545
	5624	5644	5678	5758	5759	5760	5761	5762	5763	5764	5765	5766	5768		
.EQUIV	4716														
.EVEN	4877	4888	4896	5761	5838										
.IF	4713	4715	4716	4717	4720	4893	4895	4896	4912	4935	4961	4987	5007	5038	5091
	5138	5178	5217	5225	5242	5264	5303	5320	5361	5372	5413	5457	5468	5510	5545
	5624	5644	5678	5758	5759	5760	5761	5762	5763	5764	5765	5766	5768		
.IFF	4715	4716	4720	4895	4912	4935	4961	4987	5007	5038	5091	5138	5178	5217	5225
	5242	5264	5303	5320	5361	5372	5413	5457	5468	5510	5545	5624	5644	5678	5758
	5759	5760	5761	5762	5763	5764	5765	5766	5768						
.IFT	4896	5759	5760	5765	5766										
.IFTF	4896	5759	5760	5765	5766										
.IIF	4713	4715	4717	4720	4895	4896	4994	5050	5060	5112	5115	5217	5231	5249	5276
	5283	5292	5309	5361	5457	5758	5759	5760	5761	5762	5765	5768			
.IRP	4893	4935	4961	4987	5007	5038	5091	5138	5178	5195	5217	5225	5242	5264	5303
	5320	5337	5361	5372	5413	5431	5457	5468	5510	5545	5624	5678	5759	5760	5764
	5766	5773	5883	5886											
.LIST	2	4693	4697	4700	4716	4717	4720	4842	4891	4892	4893	4895	4896	4935	4961
	4987	5007	5038	5091	5138	5178	5217	5225	5242	5264	5303	5320	5361	5372	5413
	5457	5468	5510	5545	5624	5678	5758	5759	5760	5765	5768	5777	5809	5810	5832
	5840	5841	5875	5883											
.MACRO	39	81	166	301	478	509	580	737	783	797	825	918	944	975	1023

	1036	1079	1113	1146	1159	1180	1193	1226	1275	1321	1358	1405	1438	1468	1524
	1532	1584	1794	2002	2195	2283	2405	2502	2580	2772	2881	2973	3049	3149	3280
	3343	3405	3529	3567	3631	3736	3821	3860	3923	3961	4004	4102	4150	4420	4467
	4507	4583	4705	4709	4715	4720	4929	4955	4981	5000	5024	5083	5129	5171	5217
	5219	5236	5255	5297	5313	5361	5363	5406	5457	5459	5503	5537	5617	5670	5704
	5714	5742	5768												
.MCALL	4702	4703	4704	4716	4895										
.MLIST	1	3	4695	4696	4716	4717	4720	4840	4841	4890	4893	4895	4896	4935	4961
	4987	5007	5038	5091	5138	5178	5217	5225	5242	5264	5303	5320	5361	5372	5413
	5457	5468	5510	5545	5624	5678	5758	5759	5760	5765	5768	5771	5772	5807	5830
	5831	5839	5873	5883											
.PAGE	4720	4816	4889	5767	5769	5811									
.REPT	4717	4720	5876												
.SBTTL	4715	4716	4717	4720	4935	4961	4987	5007	5038	5091	5138	5178	5217	5225	5242
	5264	5303	5320	5361	5372	5413	5457	5468	5510	5545	5624	5678	5758	5759	5760
	5761	5762	5763	5764	5765	5766	5768								
.TITLE	4713														
.WORD	4717	4720	4879	4880	4881	5758	5761	5762	5763	5766	5775	5778	5779	5780	5781
	5784	5785	5786	5787	5788	5789	5790	5791	5792	5793	5794	5795	5796	5797	5798
	5799	5802	5803	5804	5805										

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DMQAA/CRF=SYSMAC.B1 DMQAA.P11
RUN-TIME: 38 46 4 SECONDS
RUN-TIME RATIO: 326/89=3.6
CORE USED: 29K (57 PAGES)

