LPA11,KW11K

LPA/KW11K DIAG TEST
CRLPGB0

AH-B038B-MC

FICHE 1 OF 1

JAN 1980
digital
MADE IN USA

Product Code: AC-9037B-MC

Product Name: CRLPG80 LPA/KW11-K DIAGNOSTIC

Date Revised: JULY 1979

Maintainer: DIAGNOSTIC ENGINEERING

Table of Contents

1.0 ABSTRACT

****

"B" VERSION OF THE PROGRAM FIXES LS210 THRU LS230 PROGRAM BUGS AND SEVERAL DOCUMENTATION ERRORS.

This program allows the user to check out or debug the KW11K, DUAL REAL TIME CLOCK. The logic test is self contained and needs no external maintenance hardware or operator intervention.

Five special tests are included within this program to allow the user to check out and debug the external I/O signals. To run these tests a jumper wires is needed in order to loop output to an input.

THIS PROGRAM IS A MODIFIED VERSION OF "MD-11-DZKWK-A". IT WAS MODIFIED TO ENABLE THE OPERATOR TO CHECK OUT THE KW11K OPTION WHEN IT IS ON THE LPA11-KX I/O BUS. NO RECABLING IS NEEDED. SOME TEST DONE IN THE ORIGINAL DIAGNOSTIC SUCH AS ARBRITRATION TEST, WERE DELETED AS THEY COULD NOT BE CHECKED. IF THIS DIAGNOSTIC DOESN'T FIND A SUSPECTED PROBLEM, YOU MAY HAVE TO RUN "MD-11-DZKWK-A". YOU SHOULD RUN "MD-11-DRLPA" BEFORE RUNNING THIS DIAGNOSTIC. PLEASE READ SECTION 10.

2.0 REQUIREMENTS

2.1 Equipment

1. PDP11 FAMILY COMPUTER with 16K of memory or more and I/O facilities (switch register or TTY).
2. KW11K under test installed in the LPA-11KX.
3. For external I/O signal tests a loopback wire (Jumper) is needed. Jumpers are 30 AWG jumper type 915.
4. LPA11-KX

2.2 Storage

This program occupies and uses only the lower 16K of memory.

3.0 LOADING PROCEDURE

3.1 Method

Standards procedure for normal binary tapes should be followed.

1. Absolute loader must be in memory.
2. Place binary tape in reader.
3. Load address *7500 (* determined by location of loader).
4. Press "Start" (program will be loaded into memory).

The program can also be loaded by XXDP, ACT, or APT.

3.2 Non-Standard Address, Vector, or Priority; or Use of Software Switch Register

This program is set to test a KW11K with a standard address, vector, and priority. If any of these are different on the KW11K you are testing, change the corresponding location in memory before starting this test.

LOCATION	TAG	CURRENT CONTENTS	COMMENTS
1254	$BASE:	170404	;;Base address of equipment ;; under test
1250	$VECT1:	000344	;;INTERRUPT Vector #1
1252	$PRIOR:	000006	;;Bus priority - 1,#2
176	$SWREG:	000000	;Manual SWR.
	$TPFLG:	.BYTE 0	;;"Terminal Available" ; Flag (Bit<0:7>-0=Yes)

NOTE

If no hardware Switch Register exists, you may set any bit in "SWREG" as you would have set it in the SWR.

4.0 STARTING PROCEDURE

4.1 Control Switch Settings

Starting at memory locations 200, 204, 210, 214, 220, 224, 230 or 234, set all switches as desired. See Section 5.1.

4.2 Starting Addresses

200	Start address for logic test.
204	Restart address for logic test.
210	Start address for "STP2 OUT", "SCHMITT TRIG 1" tests.
214	Start address for "STP1 OUT", "SCHMITT TRIG 2" tests.
220	Start address for "SCHMITT TRIG 3 IN", "ST3 OUT" tests.
224	Starting address for "A EVENT OUT" test.
230	Starting address for "B EVENT OUT" test.
234	Starting address for "USER LINK" loop.

4.3 Program AND/OR Operator Action

1. Load program into core.
2. Set switch register to starting address.
3. Load address.
4. Set switches to deisred settings - see section 5.1.
5. IF starting a special I/O signal test:
   MAKE WIRE LOOP CONNECTION.
6. Press Start.

5.0 OPERATING PROCEDURE

5.1 Switch Register Function

Switch use

15	Halt on error
14	Loop on test
13	Inhibit error typeout (all tests)
13	Inhibit "*" typeout (special I/O signal tests)
11	Inhibit iterations (short pass)
10	Bell on error
9	Loop on error
8	Loop on test in SWR <7:0>

5.2 Scope Loops

If an error occurs and the user wishes to scope the error, he (or she) should set SW15-1 to halt on error, then when the program halts on error, SW15=0, set SW14=1. To loop on current test, set SW13=1 to inhibit error printout, and press continue on the CPU's console.

NOTE

For each test in the listing, you will find a test description. In each description a probable SYNC Point is listed. These Points are listed AS A GUIDE in order for you to SYNC your scope to the Signals being generated.

5.3 Program AND/OR Operator Action

5.3.1 Logic Test

The first pass through the program will be made with iterations inhibited. Successive passes will enable iterations if SWR11=0. 'END PASS" is printed out at the end of a pass.

If not inhibited by APT, the program will look for more KW11Ks to exercise, one pass will exercise all KW11Ks.

5.3.2 Special I/O Signal Tests

There are no "Short Passes". Each pass will iterate 65,324 times. A "*" is typed at the end of a pass unless SWR13 1.

6.0 ERRORS

6.1 Error Printout

Printout varies with the error detected. The error PC typed out is he actual location of the error call.

A halt at location '$TYPE'+10 when running with no terminal indicates an error has occurred. To find out the number of the error, examine location '$TSTNM'. This is the item number of the error. To find out what the error typout would have been GOTO to the error pointer table beginning at location '$ERRTB'.

6.1.1 Example

If we examined location '$TSTNM' and found a 5 (101) we go to location '$ERRTB' and look through the error pointer table until we found item 5. The information would look like:

```
;ITEM 5

     EM5                ;CLOCK B SR DATA ERROR
     DH5                ;ERRPC   BSR     WAS     S/B
     DT5                ;$ERRPC,BSR,$BDDAT,$GDDAT
     DF0                ;ALL NUMBERS ARE IN OCTAL FORM
```

To find out the information specified by DT5 (SERRPC,BSR,$GDADR,$BDADR) follow these steps:

1. Look uphe address of the label (i.e., $ERRPC) in the symbol table which follows the listing.

2. Put this address in the witch register and depress the load address switch on the processor's console.

3. Now depress the Examine switch.

4. The data displayed in the data lights is the information that would have been printed for his label if you had a input/output terminal.

6.2 Non-Standard Error HALTS

A HALT MAY OCCUR IF THE PROGRAM DETECTS AN LPA11-KX ERROR. CHECK THE COMMENTS IN THE LISTING OPPOSITE THE PC HALT.

7.0 RESTRICTIONS
****

7.1

Jumper W2 must be installed if not jumpered on module.
JUMPER W3, W4, AND W5 MUST BE INSTALLED TO RUN SIGNAL TESTS.

7.2

Logic Test must be run before any special I/O Signal Test.

8.0 MISCELLANEOUS
****

8.1

After a power failure occurs, program execution will continue at the point where the power failure occured after the program types "POWER".

8.2

This program is chainable under XXDP, ACT, or APT.

8.3 Execution Time

8.3.1 Logic Test

90 SECONDS iterations inhibited - no errors.

375 SECONDS with iterations - no errors.

8.3.2 Special I/O Signal Tests

1.0 Minutes No errors, SW13 0.

Execution times are approximate, as the various PDP-11 CPU's have varied instruction execution times.
Times quoted were taken from a run on a PDP-11/34.

8.4 USER LINK TO I/O DEVICE

A SPECIAL USER LINK HAS BEEN PROVIDED IN ORDER FOR THE OPERATOR TO EXAMINE OR MODIFY LOCATIONS ON THE LPA11-KX I/O BUS. (NOTE: THIS CANNOT BE DONE DIRECTLY.)

PROCEDURE:

1) START THE PROCESSOR AT LOCATION 234

2) THE DIALOG TO EXAMINE A LOCATION IS AS FOLLOWS:

```
E OR D              'E''
DEVICE ADDRS=       'OCTAL ADDRS''
XXXXXX
```

WHERE XXXXXX IS THE CONTENTS OF THE SPECIFIED LOC.

3) THE DIALOG TO MODIFY A LOCATION IS AS FOLLOWS:

```
E OR D              'D''
DATA=               'DATA TO BE DEPOSITED''
```

4) THE PROGRAM WILL STAY IN THIS LOOP UNTIL THE OPERATOR IS FINISHED. AT THIS TIME THE PROCESSOR SHOULD BE HALTED.

NOTE: THE OPERATORS RESPONSE IS ENCLOSED IN QUOTES.

9.0 PROGRAM DESCRIPTION
****

9.1 Logic Tests

A complete description of each test is included withing the listing before each test.

9.2 Special Externa I/O Signal Tests

9.2.1 LS210 "STP2 OUT" to "SCHMITT RIG 1 IN" Tests

This is a special section devoted for testing and providing scope loop capabilities for "STP2 OUT" L and "SCHMITT TRIG 1" IN.

When you load and start at location 210, program control is transferred here. "STP2 OUT" L pulses are generated by "LD STAT A HI" + "BD10" H (Main. STP2).

Pin V ("STP2 OUT") is wired to pin LL (SCHMITT TRIG1) for this test. "STP2 OUT" pulses are received as "SCHMITT TRIG 1" pulses which set clock A's status register bit 15. If an error is detected, normal error reporting technique. and error switch register options are used. An "*" is typed after each 65,324 loops through the test. SW13=1 will inhibit this feature.

You must wire pins V and LL of J1 together.
Jumper W3 mist be INSTALLED.

Logic test (L + S 200) should be run first.

9.2.2 LS214 "STP1 OUT" to "SCHMITT RIG 2" H Tests

This is a special test section devoted for testing and providing scope loop capabilities for "STP2 OUT" and "SCHMITT TRIG2" IN.

When you load and start at location 214, program control is transferred here. "STP1 OUT" L pulses are generated by "LD STAT A HI" + "BD12" H (mIn S). Pin DD ("STP1 OUT") is wired to pin BB ("SCHMITT TRIG 2") for this test. "STP1 OUT" pulses are received as "SCHMITT RIG 2" pulses which will clear clock A's count register if mode 3 is selected. If an error is detected, normal error reporting technique. and error switch register options are used. An "*" is typed after each 65,324 loops through the test. SW13-1 will inhibit this feature.

You must wire pins DD and BB of J1 together.
Jumper W4 must be installed.

Logic tests (L + S at 200) should be run first.

9.2.3 LS220 "SCHMITT TRIG 3" in, "ST3 OUT" Tests

This is a special section devoted for testing and providing scope LOOPS CAPABILITIES FOR "SCHMITT TRIG 3" AND "ST3 OUT".

When you load and start at location 220, program control is transferred here. "STP1" pulses are generated by "LD STAT A H," + "BD12" H (main STP1). Pin dd ("STP1 OUT") is wired to pin T ("SCHMITT RIG 3"). "SCHMITT TRIG 3" pulses give us "ST3 OUT" pulses. Pin L ("ST3 OUT") is wired to pin BB ("SCHMITT TRIG2"), and "SCHMITT TRIG 2" will set clock A's status register bit 7.

If an error is detected, normal error reporting technique. and error switch register options are used. An "*" is typed after each 65,324 loops through the test. SW13=1 will inhibit this feature.

You must wire pins DD to T of J1 together, as well as pins L to BB of J1 together. Jumpers W4 and W5 must also be installed.

Tests LS210 and LS214 should be run first.

9.2.4 LS224 "A EVENT OUT" Test

This is a special section devoted for testing and providing scope loop capabilities for "A EVENT OUT".

When you load and start at location 224, program control is transferred here. "A EVENT OUT" pulses are generated by clock A overflows. Pin VV ("A EVENT OUT") is wired to pin BB ("SCHMITT TRIG 2"). "SCHMITT TRIG 2" pulses will set clock A's CSR bit 7. If an error is detected, normal erroporting technique. and error switch register options are used. An "*" is typed after each 65,324 loops through the test. SW13=1 will inhibit this feature.

You must wire pins VV and BB of J1 together.
Jumper W4 must also be installed.

Test LS210 should be run first.

9.2.5 LS230 "B EVENT OUT" Test

This is a special section devoted for testing and providing scope loop capabilities for "B EVENT OUT".

When you load and start at location 230, program control is transferred here. "B EVENT OUT" pulses are generated by clock B overflows. Pin TT ("B EVENT OUT") is wired to pin BB ("SCHMITT TRIG 2"). "SCHMITT TRIG 2" pulses will set clock A's CSR bit 7. And error switch register options are used. An "*" is typed after each 65,324 loops through the test. SW13-1 will inhibit this feature.

You must wire pins TT and BB of J1 together.
Jumper W4 must be installed also.

Test LS210 should be run first.

10.0 LPA11 (SYSTEM) DIAGNOSTIC SUMMARY

DIAGNOSTICS FOR THE LPA11 ARE WRITTEN AT THREE LEVELS: (1) TOTAL PDP-11 SYSTEM, (2) LPA11 SYSTEM; AND, (3) LPA11 OPTIONS.

LEVEL 1, IS DESIGNED TO ISOLATE A FAILURE TO THE LPA11 SYSTEM. ALL OPTIONS ON THE PDP-11 ARE EXERCISED.

LEVEL 2 DIAGNOSTICS ISOLATE A FAILURE TO THE INDIVIDUAL OPTION WITHIN THE LPA11. THE LEVEL 2 DIAGNOSTIC IS MD-11-DRLPA. WHEN THE USER RUNS DRLPA HE CAN GENERALLY TELL WHICH OPTION DIAGNOSTIC (LEVEL 3) TO RUN NEXT. M8254 AND M8200-YC ERRORS MAY "LOOK" ALIKE AND DRLPA MAY NOT BE ABLE TO DISTINGUISH BETWEEN THEM. ARBITRATION ERRORS WILL NOT BE DETECTED BY THIS DIAGNOSTIC.

LEVEL THREE DIAGNOSTICS AID IN DETERMINING IF THE ERROR WAS IN FACT ON THE OPTION THE DRLPA SPECIFIED. THE USER MAY "LOOP" ON THE ERROR. WITHIN LEVEL THREE, THERE ARE TWO GROUPS OF DIAGNOSTICS. THE FIRST GROUP REQUIRES NO "EXTRA" WORK BY THE USER IN ORDER TO RUN. GROUP "A" DIAGNOSTICS DO NOT CHECK ARBITRATION, AND REQUIRE EXTRA TIME FOR EXECUTION. THE SECOND GROUP (GROUP "B") REQUIRES THAT THE USER RECONFIGURE THE PDP-11 SYSTEM. THIS RECONFIGURATION INVOLVES CABLING THE UNIBUS TO THE LPA'S I/O BUS.

THE DIAGNOSTIC FOR THE M8254 FALLS INTO THE GROUP "B" CATAGORY.

THE LPA11-K DIAGNOSTIC KIT WILL INCLUDE:

OPTION	GROUP	DIAG. #	DIAG. TITLE
LPA11-KX	LEVEL 2	MD-11-DRLPA	LPA11-K SYSTEM DIAG.
M8254	"B"	MD-11-DRM8A	M8254 (JPSM) DIAG.
AA11-K	A	MD-11-DRLPB	AA11-K DIAG.
	B	MD-11-DZAAC	AA11-K DIAG.
AR11	A	MD-11-DRLPC	LPA/AR11 DIAG. #1
	A	MD-11-DRLPD	LPA/AR11 DIAG. #2
	A	MD-11-DRLPE	LPA/AR11 DIAG. #3
	B	MD-11-DZARA	AR11 DIAG. #1
	B	MD-11-DZARB	AR11 DIAG. #2
	B	MD-11-DZARC	AR11 DIAG. #3
DR11-K	A	MD-11-CRLPF	LPA/DR11-K DIAG.
	B	MD-11-DZDRC	DR11-K DIAG.
KW11-K	A	MD-11-CRLPG	LPA/KW11-K DIAG.
	B	MD-11-DZKWK	KW11-K DIAG.
LPS11	A	MD-11-DRLPH	LPA/LPS11 DIAG. #1
	A	MD-11-DRLPI	LPA/LPS11 DIAG. #2
	A	MD-11-DRLPJ	LPA/LPS11 DIAG. #3
	B	MD-11-DZLPC	LPS11 DIAG. #1
	B	MD-11-DZLPD	LPS11 DIAG. #2
	B	MD-11-DZLPI	LPS11 DIAG. #3
AD11-K	A	MD-11-CRLPK	LPA/AD11-K DIAG.
	B	MD-11-DZADL	AD11-K DIAG.
M8200-YC	B	MD-11-DZLPL	LPA/M8200-YC BASIC MICRO-CPU R/W TEST
	B	MD-11-DZLPM	LPA/M8200-YC JMP+ROM READ TEST

THIS IS A HISTORY FILE OF CRLPG-B

THE ''A'' VERSION HAD SEVERAL PROBLEMS WHEN DEALING WITH LS204
AND LS210 THRU LS230. THE FIRST WAS THAT LS204 FAILED TO CLEAR
A LOCATION ''.DVLS'', WHICH INDICATED THE MICRO-CODE WAS LOADED
AND RUNNING <AFTER A RESTART IS IS NOT RUNNING DUE TO ''INIT''>.
THIS PROBLEM IS ALSO COMMON TO LS210 THRU LS230.
THE SECOND PROBLEM IS THAT LS210 HAD A DESIGN FLAW <REALLY ONLY A
ONE LINE TYPING REVERSAL> JAN 1979 R. SHOOP

HISTORY FILE OF DRLPG-A

PRODUCT CODE: MAINDEC-11-DZKWK-A
PRODUCT NAME: KW11-K DIAGNOSTIC TEST
DATE: FEBRUARY 1976
MAINTAINER. DIANOSTIC GROUP

PRODUCT CODE: MAINDEC-11-DRLPG-A
PRODUCT NAME: LPA/KW11-K DIAGNOSTIC TEST
DATE: JANUARY 1978
MAINTAINER: DIAGNOSTIC GROUP

REASON FOR DEVELOPMENT:

1) TO ENABLE THE OPERATOR TO CHECK OUT THE KW11-K OPTION
   WHEN IT IS ON THE LPA11-KX I/O BUS.

CHANGES MADE:

1) TOOK OUT CERTAIN TESTS FROM ORIGINAL DIAGNOSTIC (I.E.
   INTERRUPTS,TIME DEPENDENT CODE).
2) REPLACED DIRECT LINKS TO DEVICE WITH MACRO CALLS TO THE
   KMC-11 MICRO CODE. KMC-11 MICRO CODE (FILE:DRLPX2) HANDLES
   DIRECT COMMUNICATIONS WITH THE DEVICE.

FILE: DRLPA.MAC
      CONTAINS MACRO LINKS BETWEEN PDP-11 CODE AND KMC-11
      MICRO CODE. FILE: DRLPX2 NEEDS TO BE ASSEMBLED WITH
      DRLPG (SEE .CTL FILE).

FILE: DRLPX2
      MICRO CODE FILE THAT GETS LOADED INTO THE KMC-11
      VIA ROUTINES IN DRLPA.MAC.
      DRLPX2.P11 IS ASSEMBLED WITH MACY11 (ONLY) AS ANY OTHER
      .P11 FILE. THE RESULTS OF ITS ASSEMBLY IS A .OBJ
      MODULE AS WAS THE RESULT OF THE ASSEMBLY OF THE
      DIAGNOSTIC .P11 FILE. BOTH .OBJ FILES GET LINKED
      WITH LNKX11 (ONLY).

FILE: DRLPG.CTL
      THIS FILE EXPLAINS SEQUENCE OF ASSEMBLES AND LINKS.
      IT IS IN TOPS-20 FORMAT.

```
                .REM    [

                                LPA.MAC

WELCOME, THIS DIAGNOSTIC IS ONE IN A SERIES OF DIAGNOSTIC
DESIGNED IN ORDER TO AID YOU IN TESTING THE LPA-11XX OPTION.
I HOPE THAT YOU HAVE READ THE DOCUMENTATION SECTION OF THIS
DIAGNOSTIC.  IF YOU HAVE,YOU KNOW ABOUT ALL OF THE DIAGNOSTICS
THAT ARE AVAILIBLE FOR TESTING THE LPA SYSTEM.

                                GOOD LUCK .

[
.GLOBL  DRLPX2
```

```
2942                                    .REM    .
2943
2944                                    THIS IS A LIST OF TESTS DELETED FROM THIS DIAGNOSTIC.
2945                                    THESE TEST COULD NOT BE DONE THROUGH THE LPA-11.
2946
2947                                    TEST THE LOW BYTE OPERATION OF CLOCK A'S STATUS REGISTER
2948                                    TEST THE HIGH BYTE OPERATION OF A'S STATUS REGISTER
2949                                    TEST THE LOW BYTE OPERATION OF B'S STATUS REGISTER
2950                                    TEST THE HIGH BYTE OPERATION OF B'STATUS REGISTER
2951                                    TEST THAT INIT CLEARS STATUS REGISTER A
2952                                    TEST THAT INIT CLEARS BUFFER REGISTER A
2953                                    TEST THAT INIT CLEARS STATUS REGISTER B
2954                                    TEST THAT INIT CLEARS BUFFER REGISTER B
2955                                    TEST THAT A'S COUNT REGISTER IS CLEARED BY INIT
2956                                    TEST THAT CLOCK A WILL INTR. AND TO THE RIGHT VECTOR
2957                                    TEST THAT CLOCK A WILL INTR. WHEN CPU PSW = CLK INTR LEV -1
2958                                    TEST THAT CLOCK A WILL NOT INTR. WHEN CPU PSW - CLK INTR LEVEL
2959                                    TEST THAT ST1 WILL CAUSE CLOCK A TO INTER.
2960                                    TEST THAT CLOCK A OVERFLOW WILL CAUSE AN INTR.
2961                                    TEST THAT A CLOCK A COUNTER BUFFER WILL CAUSES AN INTR.
2962                                    TEST THAT CLOCK B WILL INTR. AND TO THE RIGHT VECTOR
2963                                    TEST THAT CLOCK B WILL INTR. WHEN CPU PSW=CLK INTR LEV -1
2964                                    TEST THAT CLOCK B WILL NOT INTR. WHEN CPU PSW=CLK INTR LEVEL
2965                                    TEST THAT A CLOCKB OVERFLOW WILL CAUSE AN INTR.
2966                                    TEST CLOCK A'S REPEATIBILITY AT 1MHZ RATE
2967                                    TEST CLOCK A'S REPEATIBILITY AT 100KHZ RATE
2968                                    TEST CLOCK A'S REPEATIBILITY AT 10KHZ RATE
2969                                    TEST CLOCK A'S REPEATIBILITY AT 1KHZ RATE
2970                                    TEST CLOCK A'S REPEATIBILITY AT 100HZ RATE
2971                                    TEST CLOCK B'S REPEATIBILITY AT 1MHZ RATE
2972                                    TEST CLOCK B'S REPEATIBILITY AT 100KHZ RATE
2973                                    TEST CLOCK B'S REPEATIBILITY AT 10KHZ RATE
2974                                    TEST CLOCK B'S REPEATIBILITY AT 1KHZ RATE
2975                                    TEST CLOCK B'S REPEATIBILITY AT 100HZ RATE
2976                                    TEST THAT ''INIT'' CLEARS B'S 100KHZ DIVIDE BY 10 CHIPS
2977                                            .
2978
2979
2980                                    .TITLE  LPA-KW11K DIAGNOSTIC MD-11-CRLPG-B
 (1)                                    ;*COPYRIGHT (C) 1978
 (1)                                    ;*DIGITAL EQUIPMENT CORP.
 (1)                                    ;*MAYNARD, MASS. 0175-
 (1)                                    ;*
 (1)                                    ;*PROGRAM BY EDWARD C. BADGER REV B BY R. SHOOP
 (1)                                    ;*
 (1)                                    ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
 (1)                                    ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
 (1)                                    ;*
 (1)        000001                      $TN 1
2981                                    .SBTTL  OPERATIONAL SWITCH SETTINGS
 (1)                                    ;*
 (1)                                    ;*      SWITCH                  USE
 (1)                                    ;*      ------          ----------------------
 (1)                                    ;*        15            HALT ON ERROR
 (1)                                    ;*        14            LOOP ON TEST
 (1)                                    ;*        13            INHIBIT ERROR TYPEOUTS
```

```
                                ;*          11              INHIBIT ITERATIONS
                                ;*          10              BELL ON ERROR
                                ;*           9              LOOP ON ERROR
                                ;*           8              LOOP ON TEST IN SWR<7:0>
                                .SBTTL  TRAP CATCHER

                                        .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                                        .=174
000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                .=200
000200  000137  001730                  JMP     @#START         ;GO TO STARTING ADDRESS OF PROGRAM

000204  000137  002436                  JMP     @#RSTART        ;GO TO RESTART ADDRESS.
000210  000137  023410                  JMP     @#LS210         ;GO TO SPECIAL TEST #1.
000214  000137  023606                  JMP     @#LS214         ;GO TO SPECIAL TEST #2.
000220  000137  024022                  JMP     @#LS220         ;GO TO SPECIAL TEST #3.
000224  000137  024206                  JMP     @#LS224         ;GO TO SPECIAL TEST #4.
000230  000137  024414                  JMP     @#LS230         ;GO TO SPECIAL TEST #5.

                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

```
(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
(1)
(1)                             ;*''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
(1)
(1)                             ;*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
(1)                             .EQUIV  BIT07,BIT7
```

```
  (1)                                   .EQUIV  BIT06,BIT6
  (1)                                   .EQUIV  BIT05,BIT5
  (1)                                   .EQUIV  BIT04,BIT4
  (1)                                   .EQUIV  BIT03,BIT3
  (1)                                   .EQUIV  BIT02,BIT2
  (1)                                   .EQUIV  BIT01,BIT1
  (1)                                   .EQUIV  BIT00,BIT0
  (1)
  (1)                                   ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
  (1)           000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
  (1)           000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
  (1)           000014                  TBITVEC=14              ;;''T'' BIT
  (1)           000014                  TRTVEC= 14              ;;TRACE TRAP
  (1)           000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
  (1)           000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
  (1)           000024                  PWRVEC= 24              ;;POWER FAIL
  (1)           000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
  (1)           000034                  TRAPVEC=34              ;;''TRAP'' TRAP
  (1)           000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
  (1)           000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
  (1)           000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
2994
2995            170404                          ABASE=  170404
2996            000344                          AVECT1= 344
2997            000006                          APRIOR= 6
2998
2999
3022
3026
3032
3033
3045
3046
3056
3057                                    .SBTTL  ACT11 HOOKS
  (1)
  (2)                                   ;;******************************************************************
  (1)                                   ;HOOKS REQUIRED BY ACT11
  (1)           000234                          $SVPC=.                 ;SAVE PC
  (1)           000046                          .=46
  (1)  000046   023356                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
  (1)           000052                          .=52
  (1)  000052   000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
  (1)           000234                          .=$SVPC                 ;; RESTORE PC
3058            001000                          .=1000
3059                                    .SBTTL  APT PARAMETER BLOCK
  (1)
  (2)                                   ;;******************************************************************
  (1)                                   ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
  (2)                                   ;;******************************************************************
  (1)           001000                          .$X=.   ;;SAVE CURRENT LOCATION
  (1)           000024                          .-24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
  (1)  000024   000200                          200     ;;FOR APT START UP
  (1)           000044                          .-44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
  (1)  000044   001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
  (1)           001000                          . .$X   ;;RESET LOCATION COUNTER
```

```
   (2)                          ;;*****************************************************************
   (1)                          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
   (1)                          ;INTERFACE SPEC.
   (1)
   (1)  001000                  $APTHD:
   (1)  001000  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
   (1)  001002  001202          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
   (1)  001004  000002          $TSTM:  .WORD   2       ;;RUN TIM OF LONGEST TEST
   (1)  001006  000120          $PASTM: .WORD   120     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
   (1)  001010  000120          $UNITM: .WORD   120     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
   (1)  001012  000052                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
 3060
```

```
3061                                    .SBTTL  COMMON TAGS
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
 (1)                                    ;*USED IN THE PROGRAM.
 (1)
 (1)            001100                          .=1100
 (1)  001100                            $CMTAG:                         ;;START OF COMMON TAGS
 (1)  001100    000000                          .WORD   0
 (1)  001102       000                  $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
 (1)  001103       000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
 (1)  001104    000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
 (1)  001106    000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
 (1)  001110    000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
 (1)  001112    000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
 (1)  001114       000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
 (1)  001115       001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
 (1)  001116    000000                  $FRRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
 (1)  001120    000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
 (1)  001122    000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
 (1)  001124    000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
 (1)  001126    000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
 (1)  001130    000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
 (1)  001132    000000                          .WORD   0
 (1)  001134       000                  $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
 (1)  001135       000                  $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
 (1)  001136    000000                          .WORD   0
 (1)  001140    177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
 (1)  001142    177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
 (1)  001144    177560                  $TKS:   177560                  ;;TTY KBD STATUS
 (1)  001146    177562                  $TKB:   177562                  ;;TTY KBD BUFFER
 (1)  001150    177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
 (1)  001152    177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
 (1)  001154       000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
 (1)  001155       002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
 (1)  001156       012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
 (1)  001157       000                  $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
 (1)  001160    000000                  $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
 (1)                                                                    ;;WHICH  ($REG0) WAS OBTAINED
 (3)  001162    000000                  $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
 (3)  001164    000000                  $TMP0:  .WORD   0               ;;USER DEFINED
 (1)  001166    000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
 (1)  001170    000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
 (1)  001172    177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
 (1)  001176       077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
 (1)  001177       015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
 (1)  001200    000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
 (2)                                    ;;**********************************************************************
 (2)                                    .SBTTL  APT MAILBOX-ETABLE
 (2)
 (3)                                    ;;**********************************************************************
 (2)                                    .EVEN
 (2)  001202                            $MAIL:                          ;;APT MAILBOX
 (2)  001202    000000                  $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
 (2)  001204    000000                  $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
 (2)  001206    000000                  $TESTN: .WORD   ATESTN          ;;TEST NUMBER
```

```
(2)  001210  000000          $PASS:   .WORD   APASS    ;;PASS COUNT
(2)  001212  000000          $DEVCT:  .WORD   ADEVCT   ;;DEVICE COUNT
(2)  001214  000000          $UNIT:   .WORD   AUNIT    ;;I/O UNIT NUMBER
(2)  001216  000000          $MSGAD:  .WORD   AMSGAD   ;;MESSAGE ADDRESS
(2)  001220  000000          $MSGLG:  .WORD   AMSGLG   ;;MESSAGE LENGTH
(2)  001222                  $ETABLE:                  ;;APT ENVIRONMENT TABLE
(2)  001222      000         $ENV:    .BYTE   AENV     ;;ENVIRONMENT BYTE
(2)  001223      000         $ENVM:   .BYTE   AENVM    ;;ENVIRONMENT MODE BITS
(2)  001224  000000          $SWREG:  .WORD   ASWREG   ;;APT SWITCH REGISTER
(2)  001226  000000          $USWR:   .WORD   AUSWR    ;;USER SWITCHES
(2)  001230  000000          $CPUOP:  .WORD   ACPUOP   ;;CPU TYPE,OPTIONS
(2)                          ;*                        BITS 15-11=CPU TYPE
(2)                          ;*                                11/04=01,11/05=02,11/20 03,11/40-04,11/45=05
(2)                          ;*                                11/70=06,PDQ=07,Q=10
(2)                          ;*                        BIT 10=REAL TIME CLOCK
(2)                          ;*                        BIT  9=FLOATING POINT PROCESSOR
(2)                          ;*                        BIT  8=MEMORY MANAGEMENT
(2)  001232      000         $MAMS1:  .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
(2)  001233      000         $MTYP1:  .BYTE   AMTYP1   ;;MEM. TYPE,BLK#1
(2)                          ;*                        MEM.TYPE BYTE   --  (HIGH BYTE)
(2)                          ;*                                900 NSEC CORE=001
(2)                          ;*                                300 NSEC BIPOLAR=002
(2)                          ;*                                500 NSEC MOS=003
(2)  001234  000000          $MADR1:  .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
(2)                          ;*                          MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(2)  001236      000         $MAMS2:  .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
(2)  001237      000         $MTYP2:  .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
(2)  001240  000000          $MADR2:  .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
(2)  001242      000         $MAMS3:  .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
(2)  001243      000         $MTYP3:  .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
(2)  001244  000000          $MADR3:  .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
(2)  001246      000         $MAMS4:  .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
(2)  001247      000         $MTYP4:  .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
(2)  001250  000000          $MADR4:  .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
(2)  001252  000344          $VECT1:  .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001254  000000          $VECT2:  .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001256  170404          $BASE:   .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001260  000000          $DEVM:   .WORD   ADEVM    ;;DEVICE MAP
(2)  001262  000000          $CDW1:   .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
(2)  001264  000000          $CDW2:   .WORD   ACDW2    ;;CONTROLLER DESCRIPTION WORD#2
(2)  001266  000000          $DDW0:   .WORD   ADDW0    ;;DEVICE DESCRIPTOR WORD#0
(2)  001270  000000          $DDW1:   .WORD   ADDW1    ;;DEVICE DESCRIPTOR WORD#1
(2)  001272  000000          $DDW2:   .WORD   ADDW2    ;;DEVICE DESCRIPTOR WORD#2
(2)  001274  000000          $DDW3:   .WORD   ADDW3    ;;DEVICE DESCRIPTOR WORD#3
(2)  001276  000000          $DDW4:   .WORD   ADDW4    ;;DEVICE DESCRIPTOR WORD#4
(2)  001300  000000          $DDW5:   .WORD   ADDW5    ;;DEVICE DESCRIPTOR WORD#5
(2)  001302  000000          $DDW6:   .WORD   ADDW6    ;;DEVICE DESCRIPTOR WORD#6
(2)  001304  000000          $DDW7:   .WORD   ADDW7    ;;DEVICE DESCRIPTOR WORD#7
(2)  001306  000000          $DDW8:   .WORD   ADDW8    ;;DEVICE DESCRIPTOR WORD#8
(2)  001310  000000          $DDW9:   .WORD   ADDW9    ;;DEVICE DESCRIPTOR WORD#9
(2)  001312  000000          $DDW10:  .WORD   ADDW10   ;;DEVICE DESCRIPTOR WORD#10
(2)  001314  000000          $DDW11:  .WORD   ADDW11   ;;DEVICE DESCRIPTOR WORD#11
(2)  001316  000000          $DDW12:  .WORD   ADDW12   ;;DEVICE DESCRIPTOR WORD#12
(2)  001320  000000          $DDW13:  .WORD   ADDW13   ;;DEVICE DESCRIPTOR WORD#13
(2)  001322  000000          $DDW14:  .WORD   ADDW14   ;;DEVICE DESCRIPTOR WORD#14
(2)  001324  000000          $DDW15:  .WORD   ADDW15   ;;DEVICE DESCRIPTOR WORD#15
```

```
(2)
(2)
(2)  001326                          $ETEND:
(2)
(3)
(3)  001326  170404                  ASR:    170404          ;/CLOCK A STATUS REGISTER.
(3)  001330  170406                  ABR:    170406          ;/CLOCK A BUFFER REGISTER.
(3)  001332  170430                  ACR:    170430          ;/CLOCK A COUNT REGISTER.
(3)
(3)  001334  170432                  BSR:    170432          ;/CLOCK B STATUS REGISTER.
(3)  001336  170434                  BBR:    170434          ;/CLOCK B BUFFER REGISTER.
(3)  001340  170436                  BCR:    170436          ;/CLOCK B COUNT REGISTER.
(3)
(3)  001342  000344                  AVECT:  344             ;/CLOCK A INTR. VECTOR ADDR.
(3)  001344  000346                  AVECP2: 346             ;/CLOCK A INTR. STATUS WORD.
(3)
(3)  001346  000364                  BVECT:  364             ;/CLOCK B INTR. VECTOR ADDR.
(3)  001350  000366                  BVECT2: 366             ;/CLOCK B INTR. STATUS WORD.
(3)
(3)  001352  000006                  APRITY: 6               ;/PRIORITY LEVEL OF CLOCK A.
(3)  001354  000006                  BPRITY. 6               ;/PRIORITY LEVEL OF CLOCK B.
(3)  001356  000000                  $TMDAT: 0
(3)
(3)
```

```
  (1)                                   .SBTTL  ERROR POINTER TABLE
  (1)
  (1)                                   ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
  (1)                                   ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
  (1)                                   ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
  (1)                                   ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
  (1)                                   ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
  (1)
  (1)                                   ;*      EM              ;:POINTS TO THE ERROR MESSAGE
  (1)                                   ;*      DH              ;:POINTS TO THE DATA HEADER
  (1)                                   ;*      DT              ;:POINTS TO THE DATA
  (1)                                   ;*      DF              ;:POINTS TO THE DATA FORMAT
  (1)
  (1)
  (1)  001360                           $ERRTB:
3066
3067                                    ;ITEM   1
3068
3069  001360  030140                            EM1                     ;CLOCK A SR FUNCTION ERROR
3070  001362  031025                            DH1                     ;ERRPC   ASR     WAS     S/B
3071  001364  031640                            DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
3072  001366  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
  (1)
3073
3074                                    ;ITEM   2
3075
3076  001370  030173                            EM2                     ;CLOCKA SR DATA ERROR
3077  001372  031025                            DH1                     ;ERRPC   ASR     WAS     S/B
3078  001374  031640                            DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
3079  001376  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
  (1)
3080
3081                                    ;ITEM   3
3082
3083  001400  030222                            EM3                     ;CLOCKA BR DATA ERROR
3084  001402  031063                            DH3                     ;ERRPC   ABR     WAS     S/B
3085  001404  031652                            D 3                     ;$ERRPC,ABR,$BDDAT,$GDDAT
3086  001406  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
  (1)
3087
3088                                    ;ITEM   4
3089
3090  001410  030251                            EM4                     ;CLOCKA CR DATA ERROR
3091  001412  031121                            DH4                     ;ERRPC   ACR     WAS     S/B
3092  001414  031664                            DT4                     ;$ERRPC,ACR,$BDDAT,$GDDAT
3093  001416  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
  (1)
3094
3095                                    ;ITEM   5
3096
3097  001420  030300                            EM5                     ;CLOCK B SR DATA ERROR
3098  001422  031157                            DH5                     ;ERRPC   BSR     WAS     S/B
3099  001424  031676                            DT5                     ;$ERRPC,BSR,$BDDAT,$GDDAT
3100  001426  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
  (1)
3101
```

```
3102                                    ;ITEM   6
3103
3104  001430  030327                            EM6                     ;CLOCK B BR DATA ERROR
3105  001432  031215                            DH6                     ;ERRPC    BBR     WAS     S/B
3106  001434  031710                            DT6                     ;$ERRPC,BBR,$BDDAT,$GDDAT
3107  001436  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3108
3109                                    ;ITEM   7
3110
3111  001440  030356                            EM7                     ;CLOCK B CR DATA ERROR
3112  001442  031253                            DH7                     ;ERRPC    BCR     WAS     S/B
3113  001444  031722                            DT7                     ;$ERRPC,BCR,$BDDAT,$GDDAT
3114  001446  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3115
3116                                    ;ITEM   10
3117
3118  001450  030405                            EM10                    ;DUAL ADDRESS ERROR
3119  001452  031311                            DH10                    ;ERROR    GOOD    BAD     GOOD    DATA READ FROM
3120                                                                    ;  PC     ADDR    ADDR    DATA    DUAL ADDRESS
3121  001454  031734                            DT10                    ;$ERRPC,$GDADR,$BDADR,$GDDAT,$BDDAT
3122  001456  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3123
3124                                    ;ITEM   11
3125
3126  001460  030432                            EM11                    ;CLOCK A COUNT ERROR
3127  001462  031121                            DH4                     ;ERRPC    ACR    WAS    S/B
3128  001464  031664                            DT4                     ;$ERRPC, ACR, $BDDAT, $GDDAT
3129  001466  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3130
3131                                    ;ITEM   12
3132
3133  001470  030461                            EM12                    ;CLOCK A COUNT FUNCTION ERROR
3134  001472  031450                            DH12                    ;ERRPC    ASR
3135  001474  031750                            DT12                    ;ERRPC, ASR
3136  001476  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3137
3138                                    ;ITEM   13
3139
3140  001500  032042                            DF0                     ;ERROR 13 DOES NOT EXSIST.
3141  001502  032042                            DF0                     ;IT WOULD BE BAD LUCK.
3142  001504  032042                            DF0
3143  001506  032042                            DF0
3144
3145                                    ;ITEM   14
3146
3147  001510  030521                            EM14                    ;CLOCK B COUNT FUNCTION ERROR
3148  001512  031467                            DH14                    ;ERRPC    BSR
3149  001514  031756                            DT14                    ;$ERRPC, BSR
3150  001516  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3151
```

```
3152                                    ;ITEM   15
3153
3154  001520  030561                            EM15                    ;CLOCK B COUNT ERROR
3155  001522  031253                            DH7                     ;ERRPC   CSR   WAS   S/B
3156  001524  031722                            DT7                     ;$ERRPC, BCR, $BDDAT, $GDDAT
3157  001526  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3158
3159                                    ;ITEM   16
3160
3161  001530  030610                            EM16                    ;CLOCK A INTERRUPT ERROR
3162  001532  031450                            DH12                    ;ERRPC   ASR
3163  001534  031750                            DT12                    ;$ERRPC, ASR
3164  001536  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3165
3166                                    ;ITEM   17
3167
3168  001540  030643                            EM17                    ;CLOCK B INTERRUPT ERROR
3169  001542  031467                            DH14                    ;ERRPC   BSR
3170  001544  031756                            DT14                    ;$ERRPC, BSR
3171  001546  032042                            DF0
3172
3173                                    ;ITEM   20
3174
3175  001550  030676                            EM20                    ;CLOCK A REPEATABILITY ERROR
3176  001552  031505                            DH20                    ;ERROR   ASR   2ND CNT 1ST CNT
3177  001554  031640                            DT1                     ;$ERRPC, ASR, $BDDAT, $GDDAT
3178  001556  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3179
3180                                    ;ITEM   21
3181
3182  001560  030432                            EM11                    ;CLOCK A COUNT ERROR
3183  001562  031121                            DH4                     ;ERROR   ASR   2ND CNT 1ST CNT
3184  001564  031764                            DT21                    ;$ERRPC, ASR, $BDDAT, $GDDAT
3185  001566  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3186
3187                                    ;ITEM   22
3188
3189  001570  030432                            EM11                    ;CLOCK A COUNT ERROR
3190  001572  031121                            DH4                     ;ERRPC   ASR   WAS   S/B
3191  001574  031776                            DT22                    ;$ERRPC, ACR, $BDDAT, $TMP0
3192  001576  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3193
3194                                    ;ITEM   23
3195
3196  001600  030735                            EM23                    ;CLOCK B REPEATABILITY ERROR
3197  001602  031547                            DH23                    ;ERROR   ASR   2NDCNT  1STCNT
3198  001604  031640                            DT1                     ;$ERRPC, ASR, $BDDAT, $GDDAT
3199  001606  032042                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3200
3201                                    ;ITEM   24
```

```
3202
3203  001610  030561                    EM15                    ;CLOCK B COUNT ERROR
3204  001612  031253                    DH7                     ;ERRPC   BCR   WAS    S/B
3205  001614  032010                    DT24                    ;$ERRPC, BCR, $GDDAT, $TMPO
3206  001616  032042                    DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3207
3208                            ;ITEM   25
3209
3210  001620  030561                    EM15                    ;CLOCK B COUNT ERROR
3211  001622  031253                    DH7                     ;ERRPC   BCR     WAS      S/B
3212  001624  032022                    DT25                    ;$ERRPC,BCR,$BDDAT,$TMPO
3213  001626  032042                    DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3214
3215                            ;ITEM   26
3216
3217  001630  030774                    EM26    ;CLOCK ADDRESSING ERRROR
3218  001632  031611                    DH26    ;ERRPC   CLOCK ADDR.
3219  001634  032034                    DT26    ;$ERRPC,$TMPO
3220  001636  032042                    DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
 (1)
3221
3222
3223                                    ;
 (1)                                    ;ADDRESS OF KMC-11 OF LPA-11    THE ADDR FOR KMAD0 MAY BE
 (1)                                    ;                               CHANGED BY THE USER TO REFLECT
 (1)                                    ;                               A DIFFERENT KMC-11 ADDR. THE
 (1)                                    ;                               REST OF THE ADDRESSES WILL
 (1)                                    ;                               BE CHANGED BY THE PROGRAM.
 (1)                                    ;
 (1)                                    ;
 (1)
 (1)  001640                    LPCI:
 (1)  001640  170460            KMAD0:  .WORD   170460          ;BASE KMC ADDR. MAY BE PATCHED BY USER.
 (1)
 (1)  001642                    LPMR:
 (1)  001642  170461            KMAD1:  .WORD   170460+1                ;>DO NOT        <;KMC-CSR ADDR
 (1)  001644                    LPCO:
 (1)  001644  170462            KMAD2:  .WORD   170460+2                ;>PATCH         <;
 (1)  001646                    LPSO:
 (1)  001646  170463            KMAD3:  .WORD   170460+3                ;>THIS AREA     <
 (1)  001650                    LPADL:
 (1)  001650  170464            KMAD4:  .WORD   170460+4                ;
 (1)  001652                    LPADH:
 (1)  001652  170465            KMAD5:  .WORD   170460+5                ;>DO NOT        <
 (1)  001654                    LPMS1:
 (1)  001654  170466            KMAD6:  .WORD   170460+6                ;>PATCH         <
 (1)  001656                    LPMS2:
 (1)  001656  170467            KMAD7:  .WORD   170460+7                ;>THIS AREA     <
 (1)
 (1)  001660  000344            VECTOR: .WORD   AVECT1&777      ;BASE VECTOR OF KMC
 (1)  001662  000350            VECTPS: .WORD   4+AVECT1&777    ;VECOTR ADDR.+2
 (1)
 (1)  001664  000004            VERSN:  .WORD   4               ;CURRENT VERSION NUMBER OF MICROCODE.
 (1)
 (1)  001666  000000            .DVLS:  .WORD   0               ;/DEVICE LIST OF I/O ADDR. DEFINED
```

```
   (1)  001670  000020                .BLKW   16.              ;/BY INIT.
   (1)
  3224                                .SBTTL  PROGRAM START
```

```
                                        START:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001730  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001734  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001736  022706  001140                  CMP     #SWR,R6 ;;DONE?
001742  001374                          BNE     .-6             ;;LOOP BACK IF NO
001744  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001750  012737  026010  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001756  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001764  012737  025242  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001772  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002000  012737  030054  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002006  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002014  012737  027676  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
002022  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
002030  005037  001166                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
002034  005037  001170                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002040  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002046  012737  002046  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002054  012737  002054  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002062  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002066  012737  002122  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002074  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002102  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002110  022777  177777  177022          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002116  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT  -1
002120  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002122  012716  002130          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002126  000002                          RTI
002130  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002136  012737  000174  001142          MOV     #DISPREG,DISPLAY
002144  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002150  005037  001210                  CLR     $PASS           ;;CLEAR PASS COUNT
002154  132737  000200  001223          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002162  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002164  012737  001224  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002172                          67$:
002172  005737  000042                  TST     @#42            ;IF RUNNING UNDER ACT-
002176  001015                          BNE     10$             ;NO TYPEOUT.

002200  104401  002206                  TYPE    ,69$            ;;TYPE ASCIZ STRING
002204  000412                          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <15><12><12>#MD-11-CRLPG-B#<15><12>
002232                          68$:

002232  013737  001256  001326  10$:    MOV     $BASE,ASR
002240  012737  000001  001212          MOV     #1,$DEVCT
002246  005037  001210                  CLR     $PASS
```

```
3238                                    ;
 (1)                                    ;THIS SECTION OF CODE HANDLES INITIALIZING LPA-11 FUNCTIONS
 (1)                                    ;
 (1)
 (1)  002252  010046                    MOV     R0,-(SP)
 (1)  002254  010146                    MOV     R1,-(SP)
 (1)  002256  013700  001640            MOV     KMAD0,R0        ;GET KMC-11 ADDRESS.
 (1)  002262  012701  001642            MOV     #KMAD1,R1       ;GET ADDR. OF ADDR. LIST.
 (1)
 (1)  002266  005200            70$:    INC     R0              ;UPDATE ADDR.
 (1)  002270  010021                    MOV     R0,(1)+         ;WRITE ADDR.
 (1)  002272  020127  001660            CMP     R1,#KMAD7+2     ;DONE ALL ADDRESSES?
 (1)  002276  001373                    BNE     70$             ;NO - DO NEXT ADDR.
 (1)  002300  005037  001666            CLR     .DVLS           ;CLR ADDR. LIST.
 (1)  002304  012601                    MOV     (SP)+,R1
 (1)  002306  012600                    MOV     (SP)+,R0
3239
3240  002310                    LOOP:
3241  002310  005000                    CLR     R0
3242  002312  005200            1$:     INC     R0              ;DELAY SOME TIME SO THAT FIRST RESET
3243  002314  001376                    BNE     1$              ;INSTR. WON'T CLOBBER TYPEOUT.
3244  002316  013700  001326            MOV     ASR,R0          ;NOW WE'RE GONNA FIX
3245  002322  062700  000002            ADD     #2,R0           ;ALL CLOCK ADDRESSES BASED ON ASR.
3246  002326  010037  001330            MOV     R0,ABR
3247  002332  062700  000022            ADD     #22,R0
3248  002336  010037  001332            MOV     R0,ACR
3249  002342  062700  000002            ADD     #2,R0
3250  002346  010037  001334            MOV     R0,BSR
3251  002352  062700  000002            ADD     #2,R0
3252  002356  010037  001336            MOV     R0,BBR
3253  002362  062700  000002            ADD     #2,R0
3254  002366  010037  001340            MOV     R0,BCR
3255
3256  002372  013700  001342            MOV     AVECT,R0        ;NOW FIX VECTOR ADDRESSES
3257  002376  062700  000002            ADD     #2,R0           ;BASED ON AVECT.
3258  002402  010037  001344            MOV     R0,AVECP2
3259  002406  062700  000016            ADD     #16,R0
3260  002412  010037  001346            MOV     R0,BVECT
3261  002416  062700  000002            ADD     #2,R0
3262  002422  010037  001350            MOV     R0,BVECT2
3263
3264  002426  013737  001352  001354    MOV     APRITY,BPRITY   ;FIX CLK B'S PRIORITY BASED ON A'S.
3265  002434  000402                    BR      SSTART
3266  002436  005037  001666    RSTART: CLR     .DVLS
3267  002442  012706  001100    SSTART: MOV     #STACK,SP
3268  002446  012746  000340            MOV     #340,-(SP)      ;SET PROCESSOR PRIORITY TO 7.
3269  002452  012746  002460            MOV     #1$,-(SP)
3270  002456  000002                    RTI
3271  002460                    1$:
3272
3273                            .SBTTL  *
3274                            .SBTTL  * PHASE 1 CLOCKS A+B BASIC LOGIC TESTS.
3275                            .SBTTL  *
```

```
3302
 (1)
 (5)                                    ;;*****************************************************************
 (4)                                    ;*TEST 1        *TEST THE ADDRESSABILITY OF CLOCK ADDRESS
 (5)                                    ;*
 (5)                                    ;*'BUS A17'':''A04''='DEVICE'' H; 'DEVICE'' H +''TP0'' H='DEV ENABLE'' H
 (5)                                    ;*'DEV ENABLE'' H+''TP1'' H='DEV ENB 2'' H
 (5)                                    ;*
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'BUS A17''
 (6)                                    ;*
 (5)                                    ;* CLOCK ADDRESS  TEST. SCOPE FOR 'DEV ENB 2'' H AND WORK BACK
 (5)                                    ;*
 (4)                                    ;*****************************************************************
 (3)  002460  000240                    TST1:   NOP
 (2)  002462  012737  000050  001166            MOV     #50,$TIMES      ;;DO 50 ITERATIONS
 (2)  002470  012737  002476  001106            MOV     #1$,$LPADR      ;;SET SCOPE LOOP ADDRESS
 (1)  002476  112737  000001  001102  1$:       MOVB    #1,$TSTNM
 (1)  002504  112737  000001  001206            MOVB    #1,$TESTN
 (1)
 (2)
 (2)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
 (2)
 (2)                                    ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
 (2)
 (2)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (2)
 (2)                                    ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
 (2)
 (2)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 (1)
3325
3326                                    ;;*****************************************************************
 (3)                                    ;*TEST 2        *TEST THAT CLOCK A BUFFER CAN BE WRITTEN INTO
 (4)                                    ;*
 (4)                                    ;*FOR LOADING DATA:
 (4)                                    ;*
 (4)                                    ;*(WE KNOW WE CAN ADDR. KW11), 'BUS A01'' L + ''A02'' H + ''A03'' H
 (4)                                    ;*+'BUS C1'' L='LD BUFF A'' L
 (4)                                    ;*'LD BUF A'' L + BUFFERED DATA LOADS INTO MUX LATCH (NOTE WE KNOW
 (4)                                    ;*BY NOW 'TP1'' L SHOULD BE GOOD).
 (4)                                    ;*
 (4)                                    ;* FOR READING DATA:
 (4)                                    ;*
 (4)                                    ;*BUS A01 L + (DATA IN H + EV ENABLE(1) H)-RD BUFF AL
 (4)                                    ;*[BA01H*(DEPENDING ON WHICH DATA BITS READ) RD BUF AL]+BUFF A00:15
 (4)                                    ;* +[DEV ENABLE*DATA IN L]=BUS DATA
 (4)                                    ;*
 (4)                                    ;*SINCE WE WONT LOOK FOR ANY SPECIFIC DATA BIT FAILURE,
 (4)                                    ;*JUST THAT WE CAN WRITE INTO BUFFER + READ BACK,
 (4)                                    ;*IF FAILED, KEY ON 'LD BUFF A L'' AND 'RD BUFF A L''
 (5)                                    ;*
 (5)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'BUS A17'' (2 OCCURANCES PER LOOP)
```

```
 (5)                                    ;*
 (4)                                    ;*
 (4)
 (3)                                    ;;*****************************************************************
 (2)  002572  000004                    TST2:   SCOPE
 (1)  002574  012737  000050  001166            MOV     #50,$TIMES      ;;DO 50 ITERATIONS
3327
3328                                                                    ;WRITE INTO PRESET BUFFER.
3329                                                                    ;SET $GDDAT FOR ERROR TYPEOUT.
3330                                                                    ;READ THE PRESET BUFFER.
3331  002602  012737  001416  001124            MOV     #1416,$GDDAT
3332
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
3333
 (1)                                    ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
3334  002630  005737  001126                    TST     $BDDAT
3335  002634  001001                            BNE     1$              ;IF ANY DATA WAS READ BACK, WE WILL
3336                                                                    ;BR PAST THE ERROR CALL.
3337
3338
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3339  002636  104003                            ERROR   3               ;UNABLE TO LOAD AND READ BACK
3340                                                                    ;FROM THE BUFFER REGISTER CLOCK A.
3341
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3342  002640                            1$:
```

```
                                ;;*********************************************************************
                                ;*TEST 3         *TEST THAT CLOCK A BUFFER CAN BE WRITTEN TO A ZERO
                                ;*
                                ;*THE LAST TEST WROTE 1'S INTO CLOCK A'S BUFFER. IN THIS
                                ;*TEST WE TRY TO WRITE ALL ZEROS.
                                ;*
                                ;*SIGNALS - SAME AS LAST TEST. SUSPECT F/F OR DATA GATE STUCK OPEN
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
                                ;*
                                ;*
                                ;;*********************************************************************
002640  000004                  TST3:   SCOPE
002642  012737  000050  001166          MOV     #50,$TIMES          ;;DO 50 ITERATIONS

002650  005037  001124                  CLR     $GDDAT              ;INDICATE WE EXPECT 0'S.
                                                                    ;CLEAR BUFFER REGISTER.
                                                                    ;READ CLOCK A'S BUFFER REGISTER

                                ;*      MOV     $GDDAT,@ABR         ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT         ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
002674  005737  001126                  TST     $BDDAT
002700  001401                          BEQ     1$                  ;SHOULD BE CLEAR - IF CLEAR - NEXT TEST.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

002702  104003                          ERROR   3                   ;CLEAR INTR. FAILED TO CLEAR CLOCK A'S
                                                                    ;BUFFER REGISTER.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

002704                          1$:


                                ;;*********************************************************************
                                ;*TEST 4         *TEST THAT CLOCK A'S STATUS CAN BE WRITTEN AND READ
                                ;*
                                ;*NOW THAT WE CAN WRITE INTO THE BUFFER REGISTER, WE'RE GOING TO TRY
                                ;*WRITING INTO THE STATUS REGISTER AND READ IT BACK.
                                ;*
                                ;*NEW SIGNALS: ['BA03'' L + 'BA02'' L + 'BA01'' L] 'LD STAT A'' !
                                ;*      'DATA OUT LO'' L + 'DATA OUT HI'' L + 'LD STATA'' L   'LD STAT A HI'' H
                                ;*      + 'LD STATA LO H''
                                ;*FOR READ BACK: 'RD STATA'' L
                                ;*
                                ;*NO ATTEMPT MADE TO VERIFY THAT CORRECT DATA CAME BACK, BUT
                                ;*JUST THAT SOME DATA CAME BACK.
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEV ENABLE (1)'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
```

```
 (3)                                    ;;*****************************************************************
 (2) 002704  000004                     TST4:   SCOPE
 (1) 002706  012737  000050  001166             MOV     #50,$TIMES      ;;DO 50 ITERATIONS
3389
3390 002714  012737  001416  001124             MOV     #1416,$GDDAT    ;LOAD $GDDAT WITH S/B.
3391                                                                    ;LOAD STATUS REGISTER.
3392                                                                    ;READ BACK STATUS REGISTER
3393
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
3394
 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
3395 002742  005737  001126                     TST     $BDDAT
3396 002746  001001                             BNE     1$              ;IF ANY BITS RETURNED - NO ERROR.
3397
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3398 002750  104002                           ERROR   2               ;ERROR-UNABLE TO LOAD AND READ BACK
3399                                                                    ;STATUS REGISTER OF CLOCK A.
3400
3401
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3402 002752                            1$:                             ;CLEAR CLOCKA'S STATUS REG.
3403 002752  005037  001124                     CLR     $GDDAT
3404
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
3405
3413
3414                                    ;;*****************************************************************
 (3)                                    ;*TEST 5         *TEST THAT CLOCK B'S STATUS REGISTER CAN BE WROTE/READ
 (4)                                    ;*
 (4)                                    ;*NEW SIGNALS: ['BA03'' H + 'BA02'' L + 'BA01'' L]='LD STATB'' L*'RD STAT B'' L
 (5)                                    ;*
 (5)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEV ENABLE (1)'' 2 OCCURANCES PER PASS
 (5)                                    ;*
 (4)                                    ;*
 (4)
 (3)                                    ;;*****************************************************************
 (2) 002766  000004                     TST5:   SCOPE
 (1) 002770  012737  000050  001166             MOV     #50,$TIMES      ;;DO 50 ITERATIONS
3415
3416 002776  012737  001016  001124             MOV     #1016,$GDDAT    ;USE 1016 AS PATTERN, PUT IN $GDDAT.
3417                                                                    ;LOAD B'S STAT REG.
3418                                                                    ;READ BACK THE STATUS REG.
3419
 (1)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
3420
 (1)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
3421 003024  005737  001126                     TST     $BDDAT
3422 003030  001001                             BNE     1$              ;IF ANY BITS CAME BACK, SUBTEST OK.
3423
3424
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3425 003032  104005                           ERROR   5               ;ERROR-COULD NOT WRITE/READ BACK
3426                                                                    ;CLOCK B'S STATUS REGISTER.
```

```
3427
3428
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

3429  003034                                1$:
3430
3431
3438
3439                                        ;;*****************************************************************
 (3)                                        ;*TEST 6          *TEST THAT CLOCK B'S BUFFER REGISTER CAN BE WROTE/READ
 (4)                                        ;*
 (4)                                        ;*NEW SIGNALS: ['BA03'' H + 'BA02'' L + 'BA01'' H]='LD BUFF B' L*'RD BUFF B'' L
 (5)                                        ;*
 (5)                                        ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEV ENABLE (1)'' 2 OCCURANCES PER PASS
 (5)                                        ;*
 (4)                                        ;*
 (3)                                        ;*****************************************************************
 (2)  003034  000004                        TST6:   SCOPE
 (1)  003036  012737  000050  001166                MOV     #50,$TIMES      ;;DO 50 ITERATIONS
3440
3441  003044  012737  000370  001124                MOV     #370,$GDDAT     ;USE PATTERN '370', PUT IN $GDDAT.
3442                                                                        ;WRITE INTO CLOCK B'S BUFFER REGISTER.
3443                                                                        ;READ IT BACK.
3444
 (1)                                        ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
3445
 (1)                                        ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
3446  003072  005737  001126                        TST     $BDDAT
3447  003076  001001                                BNE     1$              ;IF ANY BITS COME BACK-SUBTEST OK.
3448
3449
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

3450  003100  104006                                ERROR   6               ;ERROR-FAILED TO WRITE/READ CLOCKB'S
3451                                                                        ;BUFFER REGISTER.
3452
3453
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

3454  003102                                1$:
3455
3463
```

```
 3562                                   ;/#
  (6)                                   ;;*****************************************************************
  (5)                                   ;*TEST 7        *TEST THAT CLOCK A STATUS REGISTER BIT 15 CAN BE SET AND CLEARED
  (6)                                   ;*
  (6)                                   ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
  (6)                                   ;*F/FS OR GATES
  (7)                                   ;*
  (7)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
  (7)                                   ;*
  (6)                                   ;*
  (5)                                   ;;*****************************************************************
  (4)  003102  000004                   TST7:   SCOPE
  (3)  003104  012737  000100  001166           MOV     #100,$TIMES     ;;DO 100 ITERATIONS
  (2)                                                                   ;/CLEAR THE STATUS REGISTER.
  (2)                                                                   ;/SET BIT 15.
  (2)                                                                   ;/SET FOR ERROR TYPEOUT S/B.
  (2)                                                                    ;/READ THE STATUS REGISTER.
  (2)  003112  012737  100000  001124           MOV     #BIT15,$GDDAT
  (3)
  (3)                                   ;*       MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (3)
  (3)                                   ;*       MOV     @ASR,$BDDAT     ; READ DEVICE REG ASR,PUT DATA IN $BDDAT.
  (2)  003140  023737  001124  001126           CMP     $GDDAT,$BDDAT   ./DID BIT 15 AND ONLY BIT 15 SET?
  (2)  003146  001402                           BEQ     1$              ./IF SO-LETS TRY CLEARING IT.
  (3)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

  (2)  003150  104002                           ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
  (2)                                                                   ;/BIT 15 FAILED TO BIT SET.
  (3)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

  (2)  003152  000416                           BR      2$              ;/BR TO END SUBTEST.
  (2)  003154                           1$:                             ;/TRY CLEARING BIT 15.
  (2)  003154  005037  001124                   CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
  (2)                                                                   ;/NOW READ IT BACK.
  (3)
  (3)                                   ;*       MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (3)
  (3)                                   ;*       MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
  (2)  003200  005737  001126                   TST     $BDDAT
  (2)  003204  001401                           BEQ     2$              ;/IF ZERO-NO ERROR.
  (3)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

  (2)  003206  104002                           ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
  (2)                                                                   ;/BIT 15 FAILED TO CLEAR.
  (3)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

  (2)  003210                           2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 10        *TEST THAT CLOCK A STATUS REGISTER BIT 14 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
003210  000004                  TST10:  SCOPE
003212  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                                ;/CLEAR THE STATUS REGISTER.
                                                                ;/SET BIT 14.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE STATUS REGISTER.
003220  012737  040000  001124          MOV     #BIT14,$GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
003246  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
003254  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

003256  104002                          ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
                                                                ;/BIT 14 FAILED TO BIT SET.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

003260  000416                          BR      2$              ;/BR TO END SUBTEST.
003262                          1$:                             ;/TRY CLEARING BIT 14.
003262  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
003306  005737  001126                  TST     $BDDAT
003312  001401                          BEQ     2$              ;/IF ZERO-NO ERROR.

        ;;;$$$$$$$$$$$$$$$$$!'$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

003314  104002                          ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
                                                                ;/BIT 14 FAILED TO CLEAR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

003316                          2$:
```

```
(2)                                             ;/#
(6)                                     ;;*********************** ****************************************
(5)                                     ;*TEST 11        *TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED
(6)                                     ;*
(6)                                     ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                     ;*F/FS OR GATES
(7)                                     ;*
(7)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                     ;*
(6)                                     ;*
(5)                                     ;;*****************************************************************
(4)  003316  000004                     TST11:  SCOPE
(3)  003320  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                     ;/CLEAR THE STATUS REGISTER.
(2)                                                                     ;/SET BIT 13.
(2)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                      ;/READ THE STATUS REGISTER.
(2)  003326  012737  020000  001124             MOV     #BIT13,$GDDAT
(3)
(3)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003354  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
(2)  003362  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003364  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 13 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003366  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  003370                             1$:                             ;/TRY CLEARING BIT 13.
(2)  003370  005037  001124                     CLR     $GDDAT           ;/CLEAR S/3 FOR TYPEOUT IF ANY.
(2)                                                                     ;/NOW READ IT BACK.
(3)
(3)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003414  005737  001126                     TST     $BDDAT
(2)  003420  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003422  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                     ;/BIT 13 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003424                             2$:
```

```
(2)                                             ;/#
(6)                                     ;;**********************************************************************
(5)                                     ;*TEST 12        *TEST THAT CLOCK A STATUS REGISTER BIT 9 CAN BE SET AND CLEARED
(6)                                     ;*
(6)                                     ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                     ;*F/FS OR GATES
(7)                                     ;*
(7)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                     ;*
(6)                                     ;*
(5)                                     ;;**********************************************************************
(4)  003424  000004                     TST12:  SCOPE
(3)  003426  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                     ;/CLEAR THE STATUS REGISTER.
(2)                                                                     ;/SET BIT 9.
(2)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                      ;/READ THE STATUS REGISTER.
(2)  003434  012737  001000  001124             MOV     #BIT9,$GDDAT
(3)
(3)                                     .*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003462  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 9 AND ONLY BIT 9 SET?
(2)  003470  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003472  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 9 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003474  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  003476                          1$:                                ;/TRY CLEARING BIT 9.
(2)  003476  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                     ;/NOW READ IT BACK.
(3)
(3)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003522  005737  001126                     TST     $BDDAT
(2)  003526  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003530  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                     ;/BIT 9 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003532                          2$:
```

```
(2)                                             ;/#
(6)                                             ;;*****************************************************************
(5)                                             ;*TEST 13        *TEST THAT CLOCK A STATUS REGISTER BIT 8 CAN BE SET AND CLEARED
(6)                                             ;*
(6)                                             ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                             ;*F/FS OR GATES
(7)                                             ;*
(7)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                             ;*
(6)                                             ;*
(5)                                             ;;*****************************************************************
(4)  003532  000004                             TST13:  SCOPE
(3)  003534  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                                             ;/SET BIT 8.
(2)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                              ;/READ THE STATUS REGISTER.
(2)  003542  012737  000400  001124                     MOV     #BIT8,$GDDAT
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003570  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 8 AND ONLY BIT 8 SET?
(2)  003576  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003600  104002                                     ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                             ;/BIT 8 FAILED TO BIT SET.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003602  000416                                     BR      2$              ;/BR TO END SUBTEST.
(2)  003604                                     1$:                             ;/TRY CLEARING BIT 8.
(2)  003604  005037  001124                             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                             ;/NOW READ IT BACK.
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003630  005737  001126                             TST     $BDDAT
(2)  003634  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003636  104002                                     ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                             ;/BIT 8 FAILED TO CLEAR.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003640                                     2$:
```

```
(2)                                     ;/#
(6)                                     ;;*****************************************************************
(5)                                     ;*TEST 14        *TEST THAT CLOCK A STATUS REGISTER BIT 7 CAN BE SET AND CLEARED
(6)                                     ;*
(6)                                     ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                     ;*F/FS OR GATES
(7)                                     ;*
(7)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                     ;*
(6)                                     ;*
(5)                                     ;;*****************************************************************
(4)  003640  000004                     TST14:  SCOPE
(3)  003642  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                     ;/CLEAR THE STATUS REGISTER.
(2)                                                                     ;/SET BIT 7.
(2)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                      ;/READ THE STATUS REGISTER.
(2)  003650  012737  000200  001124             MOV     #BIT7,$GDDAT
(3)
(3)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003676  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(2)  003704  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003706  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 7 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003710  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  003712                             1$:                             ;/TRY CLEARING BIT 7.
(2)  003712  005037  001124                     CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                     ;/NOW READ IT BACK.
(3)
(3)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  003736  005737  001126                     TST     $BDDAT
(2)  003742  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003744  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                     ;/BIT 7 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  003746                             2$:
```

```
(2)                                             ;/#
(6)                                             ;;*****************************************************************
(5)                                             ;*TEST 15       *TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED
(6)                                             ;*
(6)                                             ;*CLOCK A STATUS FEGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                             ;*F/FS OR GATES
(7)                                             ;*
(7)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                             ;*
(6)                                             ;*
(5)                                             ;;*****************************************************************
(4)  003746  000004                             TST15:  SCOPE
(3)  003750  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                                             ;/SET BIT 6.
(2)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                              ;/READ THE STATUS REGISTER.
(2)  003756  012737  000100  001124                     MOV     #BIT6,$GDDAT
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004004  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
(2)  004012  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004014  104C02                                     ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                             ;/BIT 6 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004016  000416                                     BR      2$              ;/BR TO END SUBTEST.
(2)  004020                                     1$:                             ;/TRY CLEARING BIT 6.
(2)  004020  005037  001124                             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                             ;/NOW READ IT BACK.
(2)
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004044  005737  001126                             TST     $BDDAT
(2)  004050  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004052  104002                                     ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                             ;/BIT 6 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004054                                     2$
```

```
(2)                                             ;/#
(6)                                             ;;*****************************************************************
(5)                                             ;*TEST 16        *TEST THAT CLOCK A STATUS REGISTER BIT 5 CAN BE SET AND CLEARED
(6)                                             ;*
(6)                                             ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                             ;*F/FS OR GATES
(7)                                             ;*
(7)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                             ;*
(6)                                             ;*
(5)                                             ;;*****************************************************************
(4)  004054  000004                             TST16:  SCOPE
(3)  004056  012737  000100  001166                    MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                                             ;/SET BIT 5.
(2)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                              ;/READ THE STATUS REGISTER.
(2)  004064  012737  000040  001124                    MOV     #BIT5,$GDDAT
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004112  023737  001124  001126                    CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(2)  004120  001402                                    BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004122  104002                                    ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                             ;/BIT 5 FAILED TO BIT SET.
(3)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004124  000416                                    BR      2$              ;/BR TO END SUBTEST.
(2)  004126                                 1$:                                ;/TRY CLEARING BIT 5.
(2)  004126  005037  001124                            CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                             ;/NOW READ IT BACK.
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004152  005737  001126                            TST     $BDDAT
(2)  004156  001401                                    BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004160  104002                                    ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                             ;/BIT 5 FAILED TO CLEAR.
(3)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004162                                 2$:
```

```
(2)                                             ;/#
(6)                                     ;;*******************************************************************
(5)                                     ;*TEST 17        *TEST THAT CLOCK A STATUS REGISTER BIT 3 CAN BE SET AND CLEARED
(6)                                     ;*
(6)                                     ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAI !RE-SUSPECT INDIVIDUAL
(6)                                     ;*F/FS OR GATES
(7)                                     ;*
(7)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                     ;*
(6)                                     ;*
(5)                                     ;;*******************************************************************
(4)  004162  000004                     TST17:  SCOPE
(3)  004164  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                     ;/CLEAR THE STATUS REGISTER.
(2)                                                                     ;/SET BIT 3.
(2)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                      ;/READ THE STATUS REGISTER.
(2)  004172  012737  000010  001124             MOV     #BIT3,$GDDAT
(3)
(3)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004220  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
(2)  004226  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004230  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                     ;/BIT 3 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004232  000416                             BR      2$              ;/BR TO END SUBTEST.
(2)  004234                             1$:                             ;/TRY CLEARING BIT 3.
(2)  004234  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                     ;/NOW READ IT BACK.
(3)
(3)                                     ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                     ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004260  005737  001126                     TST     $BDDAT
(2)  004264  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004266  104002                             ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                     ;/BIT 3 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004270                             2$:
```

```
(2)                                             ;/#
(6)                                             ;;*****************************************************************
(5)                                             ;*TEST 20       *TEST THAT CLOCK A STATUS REGISTER BIT 2 CAN BE SET AND CLEARED
(6)                                             ;*
(6)                                             ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                             ;*F/FS OR GATES
(7)                                             ;*
(7)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                             ;*
(6)                                             ;*
(5)                                             ;;*****************************************************************
(4)  004270  000004                             TST20:  SCOPE
(3)  004272  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                                             ;/SET BIT 2.
(2)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                              ;/READ THE STATUS REGISTER.
(2)  004300  012737  000004  001124                     MOV     #BIT2,$GDDAT
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004326  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(2)  004334  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004336  104002                                     ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                             ;/BIT 2 FAILED TO BIT SET.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004340  000416                                     BR      2$              ;/BR TO END SUBTEST.
(2)  004342                                     1$:                             ;/TRY CLEARING BIT 2.
(2)  004342  005037  001124                             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                             ;/NOW READ IT BACK.
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004366  005737  001126                             TST     $BDDAT
(2)  004372  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR!
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004374  104002                                     ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                             ;/BIT 2 FAILED TO CLEAR.
(3)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004376                                     2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 21        *TEST THAT CLOCK A STATUS REGISTER BIT 1 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
004376  000004                TST21:  SCOPE
004400  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                                ;/CLEAR THE STATUS REGISTER.
                                                                ;/SET BIT 1.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE STATUS REGISTER.
004406  012737  000002  001124          MOV     #BIT1,$GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
004434  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
004442  001402                        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

004444  104002                        ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
                                                                ;/BIT 1 FAILED TO BIT SET.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

004446  000416                        BR      2$              ;/BR TO END SUBTEST.
004450                          1$:                           ;/TRY CLEARING BIT 1.
004450  005037  001124                CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
004474  005737  001126                TST     $BDDAT
004500  001401                        BEQ     2$              ;/IF ZERO-NO ERROR!

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

004502  104002                        ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
                                                                ;/BIT 1 FAILED TO CLEAR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

004504                          2$:
```

```
(2)                                             ;/#
(6)                                             ;;*****************************************************************
(5)                                             ;*TEST 22        *TEST THAT CLOCK A STATUS REGISTER BIT 0 CAN BE SET AND CLEARED
(6)                                             ;*
(6)                                             ;*CLOCK A STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(6)                                             ;*F/FS OR GATES
(7)                                             ;*
(7)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(7)                                             ;*
(6)                                             ;*
(5)                                             ;*****************************************************************
(4)  004504  000004                             TST22:  SCOPE
(3)  004506  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(2)                                                                             ;/CLEAR THE STATUS REGISTER.
(2)                                                                             ;/SET BIT 0.
(2)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(2)                                                                              ;/READ THE STATUS REGISTER.
(2)  004514  012737  000001  001124                     MOV     #BIT0,$GDDAT
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004542  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(2)  004550  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004552  104002                                     ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER.
(2)                                                                             ;/BIT 0 FAILED TO BIT SET.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004554  000416                                     BR      2$              ;/BR TO END SUBTEST.
(2)  004556                                     1$:                             ;/TRY CLEARING BIT 0.
(2)  004556  005037  001124                             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(2)                                                                             ;/NOW READ IT BACK.
(3)
(3)                                             ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(3)
(3)                                             ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
(2)  004602  005737  001126                             TST     $BDDAT
(2)  004606  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004610  104002                                     ERROR   2               ;/ERROR-CLOCK A STATUS REGISTER.
(2)                                                                             ;/BIT 0 FAILED TO CLEAR.
(3)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(2)  004612                                     2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 23        *TEST THAT CLOCK A BUFFER REGISTER BIT 0 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
004612  000004                  TST23:  SCOPE
004614  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                                ;/CLEAR THE BUFFER REGISTER.
                                                                ;/SET BIT 0.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE BUFFER REGISTER.
004622  012737  000001  001124          MOV     #BIT0,$GDDAT

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
004650  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
004656  001402                        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

004660  104003                        ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                                ;/BIT 0 FAILED TO BIT SET.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

004662  000416                        BR      2$              ;/BR TO END SUBTEST.
004664                          1$:                           ;/TRY CLEARING BIT 0.
004664  005037  001124                CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
004710  005737  001126                TST     $BDDAT
004714  001401                        BEQ     2$              ;/IF ZERO-NO ERROR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

004716  104003                        ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                                ;/BIT 0 FAILED TO CLEAR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

004720                          2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 24        *TEST THAT CLOCK A BUFFER REGISTER BIT 1 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
004720  000004                TST24:  SCOPE
004722  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                                ;/CLEAR THE BUFFER REGISTER.
                                                                ;/SET BIT 1.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE BUFFER REGISTER.
00473C  012737  000002  001124          MOV     #BIT1,$GDDAT

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
004756  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
004764  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

004766  104003                          ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                                ;/BIT 1 FAILED TO BIT SET.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

004770  000416                          BR      2$              ;/BR TO END SUBTEST.
004772                          1$:                             ;/TRY CLEARING BIT 1.
004772  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005016  005737  001126                  TST     $BDDAT
005022  001401                          BEQ     2$              ;/IF ZERO-NO ERROR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

005024  104003                          ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                                ;/BIT 1 FAILED TO CLEAR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

005026                          2$:
```

```
(3)                                             ;/#
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 25       *TEST THAT CLOCK A BUFFER REGISTER BIT 2 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*S/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  005026  000004                             TST25:  SCOPE
(4)  005030  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                                             ;/SET BIT 2.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE BUFFER REGISTER.
(3)  005036  012737  000004  001124                     MOV     #BIT2,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005064  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(3)  005072  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005074  104003                                     ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                             ;/BIT 2 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005076  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  005100                                 1$:                                 ;/TRY CLEARING BIT 2.
(3)  005100  005037  001124                             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005124  005737  001126                             TST     $BDDAT
(3)  005130  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005132  104003                                     ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                             ;/BIT 2 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005134                                 2$:
```

```
                                        ;/#
                                ;;*****************************************************************
                                ;*TEST 26        *TEST THAT CLOCK A BUFFER REGISTER BIT 3 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;*****************************************************************
005134  000004                TST26:  SCOPE
005136  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                                ;/CLEAR THE BUFFER REGISTER.
                                                                ;/SET BIT 3.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE BUFFER REGISTER.
005144  012737  000010  001124          MOV     #BIT3,$GDDAT

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005172  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
005200  001402                          BEQ     1$              ;/IF SO-LETS TRY  LEARING IT.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

005202  104003                          ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                                ;/BIT 3 FAILED TO BIT SET.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

005204  000416                          BR      2$              ;/BR TO END SUBTEST.
005206                          1$:                             ;/TRY CLEARING BIT 3.
005206  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005232  005737  001126                  TST     $BDDAT
005236  001401                          BEQ     2$              ;/IF ZERO-NO ERROR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

005240  104003                          ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTFR.
                                                                ;/BIT 3 FAILED TO CLEAR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

005242                          2$:
```

```
(3)                                             ;/#
(7)                                     ;;**********************************************************************
(6)                                     ;*TEST 27        *TEST THAT CLOCK A BUFFER REGISTER BIT 4 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;**********************************************************************
(5)  005242  000004                     TST27:  SCOPE
(4)  005244  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 4.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  005252  012737  000020  001124             MOV     #BIT4,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005300  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
(3)  005306  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005310  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 4 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005312  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  005314                             1$:                             ;/TRY CLEARING BIT 4.
(3)  005314  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005340  005737  001126                     TST     $BDDAT
(3)  005344  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005346  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 4 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005350                             2$:
```

```
                                        ;/*
                                ;;*****************************************************************
                                ;*TEST 30        *TEST THAT CLOCK A BUFFER REGISTER BIT 5 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;*****************************************************************
005350  000004                 TST30:  SCOPE
005352  012737  000100  001166         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                                ;/CLEAR THE BUFFER REGISTER.
                                                                ;/SET BIT 5.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE BUFFER REGISTER.
005360  012737  000040  001124         MOV     #BIT5,$GDDAT
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005406  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
005414  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005416  104003                         ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                                ;/BIT 5 FAILED TO BIT SET.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005420  000416                         BR      2$              ;/BR TO END SUBTEST.
005422                          1$:                            ;/TRY CLEARING BIT 5.
005422  005037  001124                 CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
005446  005737  001126                 TST     $BDDAT
005452  001401                         BEQ     2$              ;/IF ZERO-NO ERROR.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005454  104003                         ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                                ;/BIT 5 FAILED TO CLEAR.
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
005456                          2$:
```

```
(3)                                             ;/*
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 31       *TEST THAT CLOCK A BUFFER REGISTER BIT 6 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;*****************************************************************
(5)  005456  000004                     TST31:  SCOPE
(4)  005460  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 6.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  005466  012737  000100  001124             MOV     #BIT6,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005514  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
(5)  005522  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005524  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 6 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005526  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  005530                             1$:                             ;/TRY CLEARING BIT 6.
(3)  005530  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005554  005737  001126                     TST     $BDDAT
(3)  005560  001401                             BEQ     2$              ;/IF ZERO-NO ERROR
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005562  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 6 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005564                             2$:
```

```
(3)                                             ;/#
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 32        *TEST THAT CLOCK A BUFFER REGISTER BIT 7 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  005564  000004                             TST32:  SCOPE
(4)  005566  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                                             ;/SET BIT 7.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE BUFFER REGISTER.
(3)  005574  012737  000200  001124                     MOV     #BIT7,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005622  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(3)  005630  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005632  104003                                     ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                             ;/BIT 7 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005634  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  005636                                     1$:                             ;/TRY CLEARING BIT 7.
(3)  005636  005037  001124                             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005662  005737  001126                             TST     $BDDAT
(3)  005666  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005670  104003                                     ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                             ;/BIT 7 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005672                                     2$:
```

```
(3)                                             ;/#
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 33       *TEST THAT CLOCK A BUFFER REGISTER BIT 8 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  005672  000004                             TST33:  SCOPE
(4)  005674  012737  000100  001166                    MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                                             ;/SET BIT 8.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE BUFFER REGISTER.
(3)  005702  012737  000400  001124                    MOV     #BIT8,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005730  023737  001124  001126                    CMP     $GDDAT,$BDDAT   ;/DID BIT 8 AND ONLY BIT 8 SET?
(3)  005736  001402                                    BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005740  104003                                    ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                             ;/BIT 8 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005742  000416                                    BR      2$              ;/BR TO END SUBTEST.
(3)  005744                                     1$:                             ;/TRY CLEARING BIT 8.
(3)  005744  005037  001124                            CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  005770  005737  001126                            TST     $BDDAT
(3)  005774  001401                                    BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  005776  104003                                    ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                             ;/BIT 8 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006000                                     2$:
```

```
(3)                                             ;/#
(7)                                     ;;**********************************************************************
(6)                                     ;*TEST 34        *TEST THAT CLOCK A BUFFER REGISTER BIT 9 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;**********************************************************************
(5)  006000  000004                     TST34:  SCOPE
(4)  006002  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 9.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  006010  012737  001000  001124             MOV     #BIT9,$GDDAT
(4)
(4)                                     ;*       MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*       MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006036  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 9 AND ONLY BIT 9 SET?
(3)  006044  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006046  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 9 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006050  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006052                             1$:                             ;/TRY CLEARING BIT 9.
(3)  006052  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*       MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*       MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006076  005737  001126                     TST     $BDDAT
(3)  006102  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006104  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 9 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006106                             2$:
```

```
(3)                                             ;/#
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 35       *TEST THAT CLOCK A BUFFER REGISTER BIT 10 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;*****************************************************************
(5)  006106  000004                     TST35:  SCOPE
(4)  006110  012737  000100  001166             MOV     #100,$TIMES     ;,DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 10.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  006116  012737  002000  001124             MOV     #BIT10,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006144  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 10 AND ONLY BIT 10 SET?
(3)  006152  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006154  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 10 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006156  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006160                             1$:                             ;/TRY CLEARING BIT 10.
(3)  006160  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006204  005737  001126                     TST     $BDDAT
(3)  006210  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006212  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 10 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006214                             2$:
```

```
(3)                                             ;/*
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 36       *TEST THAT CLOCK A BUFFER REGISTER BIT 11 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  006214  000004                             TST36:  SCOPE
(4)  006216  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                                             ;/SET BIT 11.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE BUFFER REGISTER.
(3)  006224  012737  004000  001124                     MOV     #BIT11,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006252  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
(3)  006260  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006262  104003                                     ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                             ;/BIT 11 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006264  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  006266                                     1$:                             ;/TRY CLEARING BIT 11.
(3)  006266  005037  001124                             CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                             ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006312  005737  001126                             TST     $BDDAT
(3)  006316  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006320  104003                                     ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                             ;/BIT 11 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>` ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006322                                     2$:
```

```
(3)                                             ;/#
(7)                                     ;;**********************************************************************
(6)                                     ;*TEST 37        *TEST THAT CLOCK A BUFFER REGISTER BIT 12 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;**********************************************************************
(5)  006322  000004                     TST37:  SCOPE
(4)  006324  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 12.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  006332  012737  010000  001124             MOV     #BIT12,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@^BR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006360  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 12 AND ONLY BIT 12 SET?
(3)  006366  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006370  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 12 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006372  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006374                             1$:                             ;/TRY CLEARING BIT 12.
(3)  006374  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006420  005737  001126                     TST     $BDDAT
(3)  006424  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006426  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 12 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006430                             2$:
```

```
(3)                                         ;/#
(7)                                         ;;*****************************************************************
(6)                                         ;*TEST 40       *TEST THAT CLOCK A BUFFER REGISTER BIT 13 CAN BE SET AND CLEARED
(7)                                         ;*
(7)                                         ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                         ;*F/FS OR GATES
(8)                                         ;*
(8)                                         ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                         ;*
(7)                                         ;*
(6)                                         ;;*****************************************************************
(5)  006430  000004                         TST40:  SCOPE
(4)  006432  012737  000100  001166                 MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                         ;/CLEAR THE BUFFER REGISTER.
(3)                                                                         ;/SET BIT 13.
(3)                                                                         ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                          ;/READ THE BUFFER REGISTER.
(3)  006440  012737  020000  001124                 MOV     #BIT13,$GDDAT
(4)
(4)                                         ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                         ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006466  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
(3)  006474  001402                                 BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                    ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006476  104003                                 ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                         ;/BIT 13 FAILED TO BIT SET.
(4)
                    ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006500  000416                                 BR      2$              ;/BR TO END SUBTEST.
(3)  006502                                 1$:                             ;/TRY CLEARING BIT 13.
(3)  006502  005037  001124                         CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                         ;/NOW READ IT BACK.
(4)
(4)                                         ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                         ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006526  005737  001126                         TST     $BDDAT
(3)  006532  001401                                 BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
                    ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006534  104003                                 ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                         ;/BIT 13 FAILED TO CLEAR.
(4)
                    ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006536                                 2$:
```

```
(3)                                     ;/#
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 41       *TEST THAT CLOCK A BUFFER REGISTER BIT 14 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;*****************************************************************
(5)  006536  000004                     TST41:  SCOPE
(4)  006540  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 14.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  006546  012737  040000  001124             MOV     #BIT14,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006574  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
(3)  006602  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006604  104003                             ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
(3)                                                                     ;/BIT 14 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006606  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  006610                          1$:                                ;/TRY CLEARING BIT 14.
(3)  006610  005037  001124                     CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(4)
(4)                                     ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(3)  006634  005737  001126                     TST     $BDDAT
(3)  006640  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006642  104003                             ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
(3)                                                                     ;/BIT 14 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  006644                          2$:
```

```
                                        ;/#
                                ;;*****************************************************************
                                ;*TEST 42        *TEST THAT CLOCK A BUFFER REGISTER BIT 15 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK A BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;*****************************************************************
006644  000004                  ^ST42:  SCOPE
006646  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
                                                                ;/CLEAR THE BUFFER REGISTER.
                                                                ;/SET BIT 15.
                                                                ;/SET FOR ERROR TYPEOUT S/B.
                                                                 ;/READ THE BUFFER REGISTER.
006654  012737  100000  001124          MOV     #BIT15,$GDDAT

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
006702  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 15 AND ONLY BIT 15 SET?
006710  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

006712  104003                          ERROR   3               ;/ERROR CLOCK AS BUFFER REGISTER.
                                                                ;/BIT 15 FAILED TO BIT SET.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

006714  000416                          BR      2$              ;/BR TO END SUBTEST.
006716                          1$:                             ;/TRY CLEARING BIT 15.
006716  005037  001124                  CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
                                                                ;/NOW READ IT BACK.

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

                                ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
006742  005737  001126                  TST     $BDDAT
006746  001401                          BEQ     2$              ;/IF ZERO-NO ERROR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

006750  104003                          ERROR   3               ;/ERROR-CLOCK A BUFFER REGISTER.
                                                                ;/BIT 15 FAILED TO CLEAR.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

006752                          2$:
```

```
(3)                                             ;/#
(7)                                     ;;**********************************************************************
(6)                                     ;*TEST 43        *TEST THAT CLOCK B STATUS REGISTER BIT 11 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;**********************************************************************
(5)  006752  000004                     TST43:  SCOPE
(4)  006754  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 11.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  006762  012737  004000  001124             MOV     #BIT11,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007010  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
(3)  007016  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007020  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 11 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007022  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007024                             1$:                             ;/TRY CLEARING BIT 11.
(3)  007024  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007050  005737  001126                     TST     $BDDAT
(3)  007054  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007056  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 11 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007060                             2$:
(2)
```

```
(3)                                             ;/*
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 44       *TEST THAT CLOCK B STATUS REGISTER BIT 7 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  007060  000004                             TST44:  SCOPE
(4)  007062  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                                             ;/SET BIT 7.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE STATUS REGISTER.
(3)  007070  012737  000200  001124                     MOV     #BIT7,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007116  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(3)  007124  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007126  104005                                     ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                             ;/BIT 7 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007130  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  007132                                     1$:                             ;/TRY CLEARING BIT 7.
(3)  007132  005037  001124                             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007156  005737  001126                             TST     $BDDAT
(3)  007162  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007164  104005                                     ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                             ;/BIT 7 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007166                                     2$:
(2)
```

```
(3)                                             ;/#
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 45       *TEST THAT CLOCK B STATUS REGISTER BIT 6 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OU'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;*****************************************************************
(5)  007166  000004                     TST45:  SCOPE
(4)  007170  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 6.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  007176  012737  000100  001124             MOV     #BIT6,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007224  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
(3)  007232  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007234  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 6 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007236  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007240                          1$:                                ;/TRY CLEARING BIT 6.
(3)  007240  005037  001124                     CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007264  005737  001126                     TST     $BDDAT
(3)  007270  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007272  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 6 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007274                          2$:
(2)
```

```
(3)                                             ;/#
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 46        *TEST THAT CLOCK B STATUS REGISTER BIT 5 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  007274  000004                             TST46:  SCOPE
(4)  007276  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                                             ;/SET BIT 5.
(3)                                                                             ;/SET FOR EPROR TYPEOUT S/B.
(3)                                                                              ;/READ THE STATUS REGISTER.
(3)  007304  012737  000040  001124                     MOV     #BIT5,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007332  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(3)  007340  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007342  104005                                     ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                             ;/BIT 5 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007344  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  007346                                     1$:                             ;/TRY CLEARING BIT 5.
(3)  007346  005037  001124                             CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007372  005737  001126                             TST     $BDDAT
(3)  007376  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007400  104005                                     ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                             ;/BIT 5 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007402                                     2$:
(2)
```

```
(3)                                             ;/*
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 47       *TEST THAT CLOCK B STATUS REGISTER BIT 4 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  007402  000004                             TST47:  SCOPE
(4)  007404  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                                             ;/SET BIT 4.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE STATUS REGISTER.
(3)  007412  012737  000020  001124                     MOV     #BIT4,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007440  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
(3)  007446  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007450  104005                                     ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                             ;/BIT 4 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007452  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  007454                                 1$:                                 ;/TRY CLEARING BIT 4.
(3)  007454  005037  001124                             CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007500  005737  001126                             TST     $BDDAT
(3)  007504  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007506  104005                                     ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                             ;/BIT 4 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007510                                 2$:
(2)
```

```
(3)                                             ;/#
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 50        *TEST THAT CLOCK B STATUS REGISTER BIT 3 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  007510  000004                             TST50:  SCOPE
(4)  007512  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE STATUS REGISTER.
(3)                                                                             ;/SET BIT 3.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE STATUS REGISTER.
(3)  007520  012737  000010  001124                     MOV     #BIT3,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007546  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
(3)  007554  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007556  104005                                     ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                             ;/BIT 3 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007560  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  007562                                     1$:                             ;/TRY CLEARING BIT 3.
(3)  007562  005037  001124                             CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                             ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007606  005737  001126                             TST     $BDDAT
(3)  007612  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007614  104005                                     ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                             ;/BIT 3 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007616                                     2$:
(2)
```

```
(3)                                             ;/#
(7)                                     ;;**********************************************************************
(6)                                     ;*TEST 51        *TEST THAT CLOCK B STATUS REGISTER BIT 2 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;**********************************************************************
(5)  007616  000004                     TST51:  SCOPE
(4)  007620  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 2.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  007626  012737  000004  001124             MOV     #BIT2,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007654  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(3)  007662  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007664  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 2 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007666  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007670                          1$:                                ;/TRY CLEARING BIT 2.
(3)  007670  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007714  005737  001126                     TST     $BDDAT
(3)  007720  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007722  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 2 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007724                          2$:
(2)
```

```
(3)                                             ;/#
(7)                                     ;;**********************************************************************
(6)                                     ;*TEST 52        *TEST THAT CLOCK B STATUS REGISTER BIT 1 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;**********************************************************************
(5)  007724  000004                     TST52:  SCOPE
(4)  007726  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 1.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  007734  012737  000002  001124             MOV     #BIT1,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  007762  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
(3)  007770  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007772  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 1 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  007774  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  007776                             1$:                             ;/TRY CLEARING BIT 1.
(3)  007776  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  010022  005737  001126                     TST     $BDDAT
(3)  010026  001401                             BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010030  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 1 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010032                             2$:
(2)
```

```
(3)                                             ;/#
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 53        *TEST THAT CLOCK B STATUS REGISTER BIT 0 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK B STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;*****************************************************************
(5)  010032  000004                     TST53:  SCOPE
(4)  010034  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE STATUS REGISTER.
(3)                                                                     ;/SET BIT 0.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE STATUS REGISTER.
(3)  010042  012737  000001  001124             MOV     #BIT0,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  010070  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(3)  010076  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010100  104005                             ERROR   5               ;/ERROR CLOCK BS STATUS REGISTER.
(3)                                                                     ;/BIT 0 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010102  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  010104                          1$:                                ;/TRY CLEARING BIT 0.
(3)  010104  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
(4)
(4)                                     ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(3)  010130  005737  001126                     TST     $BDDAT
(3)  010134  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010136  104005                             ERROR   5               ;/ERROR-CLOCK B STATUS REGISTER.
(3)                                                                     ;/BIT 0 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010140                          2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 54        *TEST THAT CLOCK B BUFFER REGISTER BIT 0 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
(5)  010140  000004             TST54:  SCOPE
(4)  010142  012737  000100  001166     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 0.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3)  010150  012737  000001  001124     MOV     #BIT0,$GDDAT
(4)
(4)                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010176  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(3)  010204  001402                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010206  104006                     ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 0 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010210  000416                     BR      2$              ;/BR TO END SUBTEST.
(3)  010212                     1$:                             ;/TRY CLEARING BIT 0.
(3)  010212  005037  001124             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                             ;/NOW READ IT BACK.
(4)
(4)                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010236  005737  001126             TST     $BDDAT
(3)  010242  001401                     BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010244  104006                     ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 0 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010246                     2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 55        *TEST THAT CLOCK B BUFFER REGISTER BIT 1 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
(5) 010246  000004              TST55:  SCOPE
(4) 010250  012737  000100  001166      MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 1.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3) 010256  012737  000002  001124      MOV     #BIT1,$GDDAT
(4)
(4)                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3) 010304  023737  001124  001126      CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
(3) 010312  001402                      BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010314  104006                      ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 1 FAILED TO BIT SET.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010316  000416                      BR      2$              ;/BR TO END SUBTEST.
(3) 010320                      1$:                             ;/TRY CLEARING BIT 1.
(3) 010320  005037  001124              CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                             ;/NOW READ IT BACK.
(4)
(4)                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3) 010344  005737  001126              TST     $BDDAT
(3) 010350  001401                      BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010352  104006                      ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 1 FAILED TO CLEAR.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010354                      2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 56        *TEST THAT CLOCK B BUFFER REGISTER BIT 2 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
(5) 010354  000004              TST56:  SCOPE
(4) 010356  012737  000100  001166      MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                             ;/SET BIT 2.
(3)                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                              ;/READ THE BUFFER REGISTER.
(3) 010364  012737  000004  001124      MOV     #BIT2,$GDDAT
(4)
(4)                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3) 010412  023737  001124  001126      CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(3) 010420  001402                      BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010422  104006                      ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                             ;/BIT 2 FAILED TO BIT SET.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010424  000416                      BR      2$              ;/BR TO END SUBTEST.
(3) 010426                      1$:                             ;/TRY CLEARING BIT 2.
(3) 010426  005037  001124              CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                             ;/NOW READ IT BACK.
(4)
(4)                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3) 010452  005737  001126              TST     $BDDAT
(3) 010456  001401                      BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010460  104006                      ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                             ;/BIT 2 FAILED TO CLEAR.
(4)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3) 010462                      2$:
```

```
(3)                                             ;/#
(7)                                             ;;*****************************************************************
(6)                                             ;*TEST 57        *TEST THAT CLOCK B BUFFER REGISTER BIT 3 CAN BE SET AND CLEARED
(7)                                             ;*
(7)                                             ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                             ;*F/FS OR GATES
(8)                                             ;*
(8)                                             ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                             ;*
(7)                                             ;*
(6)                                             ;;*****************************************************************
(5)  010462  000004                             TST57:  SCOPE
(4)  010464  012737  000100  001166                     MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                             ;/CLEAR THE BUFFER REGISTER.
(3)                                                                             ;/SET BIT 3.
(3)                                                                             ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                              ;/READ THE BUFFER REGISTER.
(3)  010472  012737  000010  001124                     MOV     #BIT3,$GDDAT
(4)
(4)                                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010520  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
(3)  010526  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010530  104006                                     ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                                             ;/BIT 3 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010532  000416                                     BR      2$              ;/BR TO END SUBTEST.
(3)  010534                                 1$:                                 ;/TRY CLEARING BIT 3.
(3)  010534  005037  001124                             CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                             ;/NOW READ IT BACK.
(4)
(4)                                             ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                             ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010560  005737  001126                             TST     $BDDAT
(3)  010564  001401                                     BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010566  104006                                     ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                                             ;/BIT 3 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010570                                 2$:
```

```
(3)                                             ;/#
(7)                                     ;;**********************************************************************
(6)                                     ;*TEST 60        *TEST THAT CLOCK B BUFFER REGISTER BIT 4 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;**********************************************************************
(5)  010570  000004                     TST60:  SCOPE
(4)  010572  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 4.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  010600  012737  000020  001124             MOV     #BIT4,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                     ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010626  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
(3)  010634  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010636  104006                             ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                                     ;/BIT 4 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010640  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  010642                             1$:                             ;/TRY CLEARING BIT 4.
(3)  010642  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                     ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010666  005737  001126                     TST     $BDDAT
(3)  010672  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010674  104006                             ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                                     ;/BIT 4 FAILED TO C_EAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010676                             2$:
```

```
(3)                                         ;/#
(7)                                         ;;*****************************************************************
(6)                                         ;*TEST 61       *TEST THAT CLOCK B BUFFER REGISTER BIT 5 CAN BE SET AND CLEARED
(7)                                         ;*
(7)                                         ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                         ;*F/FS OR GATES
(8)                                         ;*
(8)                                         ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                         ;*
(7)                                         ;*
(6)                                         ;;*****************************************************************
(5)  010676  000004                         TST61:  SCOPE
(4)  010700  012737  000100  001166                 MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                         ;/CLEAR THE BUFFER REGISTER.
(3)                                                                         ;/SET BIT 5.
(3)                                                                         ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                          ;/READ THE BUFFER REGISTER.
(3)  010706  012737  000040  001124                 MOV     #BIT5,$GDDAT
(4)
(4)                                         ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                         ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010734  023737  001124  001126                 CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(3)  010742  001402                                 BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010744  104006                                 ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                                         ;/BIT 5 FAILED TO BIT SET.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  010746  000416                                 BR      2$              ;/BR TO END SUBTEST.
(3)  010750                                 1$:                             ;/TRY CLEARING BIT 5.
(3)  010750  005037  001124                         CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                         ;/NOW READ IT BACK.
(4)
(4)                                         ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                         ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  010774  005737  001126                         TST     $BDDAT
(3)  011000  001401                                 BEQ     2$              ;/IF ZERO-NO ERROR!
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011002  104006                                 ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                                         ;/BIT 5 FAILED TO CLEAR.
(4)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011004                                 2$:
```

```
(3)                                         ;/#
(7)                                 ;;**********************************************************************
(6)                                 ;*TEST 62       *TEST THAT CLOCK B BUFFER REGISTER BIT 6 CAN BE SET AND CLEARED
(7)                                 ;*
(7)                                 ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                 ;*F/FS OR GATES
(8)                                 ;*
(8)                                 ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                 ;*
(7)                                 ;*
(6)                                 ;;**********************************************************************
(5)  011004  000004                 TST62:  SCOPE
(4)  011006  012737  000100  001166         MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                 ;/CLEAR THE BUFFER REGISTER.
(3)                                                                 ;/SET BIT 6.
(3)                                                                 ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                  ;/READ THE BUFFER REGISTER.
(3)  011014  012737  000100  001124         MOV     #BIT6,$GDDAT
(4)
(4)                                 ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                 ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  011042  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
(3)  011050  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011052  104006                         ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                                 ;/BIT 6 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011054  000416                         BR      2$              ;/BR TO END SUBTEST.
(3)  011056                         1$:                             ;/TRY CLEARING BIT 6.
(3)  011056  005037  001124                 CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                 ;/NOW READ IT BACK.
(4)
(4)                                 ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                 ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR PUT DATA IN $BDDAT.
(3)  011102  005737  001126                 TST     $BDDAT
(3)  011106  001401                         BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011110  104006                         ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                                 ;/BIT 6 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011112                         2$.
```

```
(3)                                             ;/#
(7)                                     ;;*****************************************************************
(6)                                     ;*TEST 63        *TEST THAT CLOCK B BUFFER REGISTER BIT 7 CAN BE SET AND CLEARED
(7)                                     ;*
(7)                                     ;*CLOCK B BUFFER REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(7)                                     ;*F/FS OR GATES
(8)                                     ;*
(8)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
(8)                                     ;*
(7)                                     ;*
(6)                                     ;;*****************************************************************
(5)  011112  000004                     TST63:  SCOPE
(4)  011114  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(3)                                                                     ;/CLEAR THE BUFFER REGISTER.
(3)                                                                     ;/SET BIT 7.
(3)                                                                     ;/SET FOR ERROR TYPEOUT S/B.
(3)                                                                      ;/READ THE BUFFER REGISTER.
(3)  011122  012737  000200  001124             MOV     #BIT7,$GDDAT
(4)
(4)                                     ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                     ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  011150  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 7 AND ONLY BIT 7 SET?
(3)  011156  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011160  104006                             ERROR   6               ;/ERROR CLOCK BS BUFFER REGISTER.
(3)                                                                     ;/BIT 7 FAILED TO BIT SET.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011162  000416                             BR      2$              ;/BR TO END SUBTEST.
(3)  011164                             1$:                             ;/TRY CLEARING BIT 7.
(3)  011164  005037  001124                     CLR     $GDDAT           ;/CLEAR S/B FOR TYPEOUT IF ANY.
(3)                                                                     ;/NOW READ IT BACK.
(4)
(4)                                     ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
(4)
(4)                                     ;*      MOV     @BBR,$BDDAT     ;/READ DEVICE REG BBR,PUT DATA IN $BDDAT.
(3)  011210  005737  001126                     TST     $BDDAT
(3)  011214  001401                             BEQ     2$              ;/IF ZERO-NO ERROR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011216  104006                             ERROR   6               ;/ERROR-CLOCK B BUFFER REGISTER.
(3)                                                                     ;/BIT 7 FAILED TO CLEAR.
(4)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

(3)  011220                             2$:
3563                                    .SBTTL  *
3564                                    .SBTTL  * PHASE 2 ADVANCED BASIC LOGIC TESTS
3565                                    .SBTTL  *
3566
3572
```

```
                                ;;*****************************************************************
                                ;*TEST 64        *TEST THAT CLOCK A'S COUNT REGISTER IS CLEAR
                                ;*
                                ;*CLOCK A COUNT REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;*****************************************************************
011220  000004                TST64:  SCOPE
011222  012737  000002  001166          MOV     #2,$TIMES       ;;DO 2 ITERATIONS

                                                                ;SELECT MODE 0.
                                                                ;CLEAR THE BUFFER REGISTER. BUFFER
                                                                ;REGISTER WILL BE TRANSFERRED TO
                                                                ;COUNT REGISTER SINCE THIS IS MODE 0.
011230  005037  001124                  CLR     $GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR

011254  005037  001124                  CLR     $GDDAT          ;EXPECT TO READ BACK ALL 0'S.
                                                                ;READ THE COUNT REGISTER - IT SHOULD BE CLEAR.

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
011270  005737  001126                  TST     $BDDAT
011274  001401                          BEQ     1$              ;BR IF YES TO NEXT TES
                                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
011276  104004                          ERROR   4               ;ERROR - CLOCK A'S COUNTER REGISTER
                                                                ;NOT CLEAR.
                                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
011300                          1$:
                                ;;*****************************************************************
                                ;*TEST 65        *TEST CLOCK A'S COUNT REGISTER WITH 125252 PATTERN
                                ;*
                                ;*CLOCK A COUNT REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:. 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;*****************************************************************
011300  000004                TST65:  SCOPE
```

```
  (1)  011302  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
3597
3598                                                                    ;SELECT MODE 0.
3599                                                                    ;LOAD THE BUFFER REGISTER WITH
3600                                                                    ;PATTERN 125252. IT WILL BE
3601                                                                    ;TRANSFERRED TO THE COUNT REGISTER
3602                                                                    ;SINCE THIS IS MODE 0.
3603  011310  005037  001124                  CLR     $GDDAT
3604
  (1)                                 ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
3605
3606  011324  012737  125252  001124          MOV     #125252,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
3607                                                                    ;NEED OF ERROR TYPEOUT.
3608                                                                    ;READ THE COUNT REGISTER
3609
  (1)                                 ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
3610
  (1)                                 ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
3611
3612  011352  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
3613                                                                    ;COME THROUGH?
3614  011360  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST.
3615
3616
          ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3617  011362  104004                          ERROR   4               ;DATA ERROR CLOCK A   PATTERN "125252"
3618                                                                    ;FAILED TO TRANSFER PROPERLY BETWEEN
3619                                                                    ;BUFFER AND COUNT REGISTERS.
3620
3621
          ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3622  011364                          1$:
3623
3624                                 ;;**********************************************************************
  (3)                                ;*TEST 66        *TEST CLOCK A'S COUNT REGISTER WITH 052525 PATTERN
  (4)
  (5)                                ;*
  (5)                                ;*CLOCK A COUNT REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
  (5)                                ;*F/FS OR GATES
  (6)                                ;*
  (6)                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
  (6)                                ;*
  (5)                                ;*
  (4)
  (3)                                ;;**********************************************************************
  (2)  011364  000004                 T66:    SCOPE
  (1)  011366  012737  000050  001166          MOV     #50,$TIMES      ;;DO 50 ITERATIONS
3625
3626                                                                    ;SELECT MODE 0.
3627  011374  005037  001124                  CLR     $GDDAT
3628
  (1)                                 ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
3629                                                                    ;LOAD THE BUFFER REGISTER WITH
3630                                                                    ;PATTERN 052525. IT WILL BE
```

```
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.
011410  012737  052525  001124          MOV     #052525,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
                                                        ;NEED OF ERROR TYPEOUT.
                                                        ;READ THE COUNT REGISTER
                              ;*        MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                              ;*        MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
011436  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
                                                        ;COME THROUGH?
011444  001401                        BEQ     1$              ;BR IF YES TO NEXT TEST.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

011446  104004                        ERROR   4               ;DATA ERROR CLOCK A   PATTERN '052525''
                                                        ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                        ;BUFFER AND COUNT REGISTERS.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

011450                          1$:

                                ;;**********************************************************************
                                ;*TEST 67        *TEST THAT CLOCK B'S COUNT REGISTER IS CLEAR
                                ;*
                                ;*CLOCK B COUNT REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*
                                ;;**********************************************************************
011450  000004                  TST67:  SCOPE
011452  012737  000002  001166          MOV     #2,$TIMES       ;;DO 2 ITERATIONS

                                                        ;SELECT MODE 0.
                                                        ;CLEAR THE BUFFER REGISTER. BUFFER
                                                        ;REGISTER WILL BE TRANSFERRED TO
                                                        ;COUNT REGISTER SINCE THIS IS MODE 0.
011460  005037  001124                  CLR     $GDDAT
                              ;*        MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
                              ;*        MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
011504  005037  001124                  CLR     $GDDAT          ;EXPECT TO READ BACK ALL 0'S.
                                                        ;READ THE COUNT REGISTER - IT SHOULD BE CLEAR.
```

```
 (1)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
3671  011520  005737  001126                    TST     $BDDAT
3672  011524  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST
3673
3674
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3675  011526  104007                          ERROR   7               ;ERROR - CLOCK B'S COUNTER REGISTER
3676                                                                  ;NOT CLEAR.
3677
3678
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3679  011530                                1$:

3680
3681
3682                                    ;;*****************************************************************
 (3)                                    ;*TEST 70        *TEST CLOCK B'S COUNT REGISTER WITH 125 PATTERN
 (4)
 (5)                                    ;*
 (5)                                    ;*CLOCK B COUNT REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
 (5)                                    ;*F/FS OR GATES
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
 (6)                                    ;*
 (5)                                    ;*
 (4)
 (3)                                    ;;*****************************************************************
 (2)  011530  000004                    TST70:  SCOPE
 (1)  011532  012737  000100  001166            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
3683
3684                                                                    ;SELECT MODE 0.
3685  011540  005037  001124                   CLR     $GDDAT
3686
 (1)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
3687                                                                    ;LOAD THE BUFFER REGISTER WITH
3688                                                                    ;PATTERN 125. IT WILL BE
3689                                                                    ;TRANSFERRED TO THE COUNT REGISTER
3690                                                                    ;SINCE THIS IS MODE 0.
3691
3692  011554  012737  000125  001124            MOV     #125,$GDDAT     ;SET EXPECTED TO PATTERN IN CASE OF
3693
 (1)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
3694                                                                    ;NEED OF ERROR TYPEOUT.
3695                                                                    ;READ THE COUNT REGISTER
3696
 (1)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
3697
3698  011602  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
3699                                                                    ;COME THROUGH?
3700  011610  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST.
3701
3702
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

3703  011612  104007                          ERROR   7               ;DATA ERROR CLOCK B - PATTERN ''125''
```

```
                                                        ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                        ;BUFFER AND COUNT REGISTERS.


        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

011614                          1$:

                                ;;*****************************************************************
                                ;*TEST 71        *TEST CLOCK B'S COUNT REGISTER WITH 252 PATTERN

                                ;*
                                ;*CLOCK B COUNT REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 OCCURANCES PER PASS
                                ;*
                                ;*

                                ;;*****************************************************************
011614  000004                  TST71:  SCOPE
011616  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

                                                        ;SELECT MODE 0.
011624  005037  001124          CLR     $GDDAT
                                ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
                                                        ;LOAD THE BUFFER REGISTER WITH
                                                        ;PATTERN 252. IT WILL BE
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.
011640  012737  000252  001124          MOV     #252,$GDDAT     ;SET EXPECTED TO PATTERN IN CASE OF
                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
                                                        ;NEED OF ERROR TYPEOUT.
                                                        ;READ THE COUNT REGISTER
                                ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
011666  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
                                                        ;COME THROUGH?
011674  001401                  BEQ     1$              ;BR IF YES TO NEXT TEST.


        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

011676  104007                  ERROR   7               ;DATA ERROR CLOCK B - PATTERN ''252''
                                                        ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                        ;BUFFER AND COUNT REGISTERS.


        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

011700                          1$:
```

```
                                ;;*****************************************************************
                                ;*TEST 72        *TEST THE SETTING OF MAINTENANCE STP1 IN CLOCK A BIT 15 TO SET
                                ;*
                                ;*NEW SIGNALS    GENERATION OF ''STP1'' BY 'LD STAT A'' H + 'BD 12'' H
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 PER PASS
                                ;*
                                ;*
                                ;;*****************************************************************
011700  000004                 TST72:  SCOPE

011702  005037  001124                 CLR     $GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
                                                                ;MAKE SURE THE STATUS REGISTER IS CLEAR.
                                                                ;SET MAINTENANCE STP1.
011726  012737  010000  001124         MOV     #BIT12,$GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                                                ;DID BIT15 (STP' FLAG) SET?

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
011754  005737  001126                 TST     $BDDAT
011760  100401                         BMI     1$              ;BR IF YES - NEXT TEST

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

011762  104001                         ERROR   1               ;ERROR - MAINTENANCE STP1 (BIT12)
                                                                ;DID NOT SET BIT15 (STP1 FLAG) CLOCK A.

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

011764                          1$:


                                ;;*****************************************************************
                                ;*TEST 73        *TEST THAT BIT00 IN CLOCK A STATUS REG. WILL SET WHEN BIT13 AND MAIN. ST
                                ;*
                                ;*NFW SIGNALS    ''STP1'' H + ''ST1 ENB CNTR (1)'' H DIRECT SETTING
                                ;*              'ST1 FLAG''
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'DEVICE OUT'' 2 PER PASS
                                ;*
                                ;*
                                ;;*****************************************************************
011764  000004                 TST73:  SCOPE

                                                                ;SET ''ST1 ENB COUNTER'' IN CLK A'S STATUS REG.
                                                                ;GENERATE A MAINTENANCE ST1.
                                                                ;DID BIT00 (ENABL CNTR A) SET?
```

```
 3787  011766  012737  020000  001124          MOV     #BIT13,$GDDAT
 3788
  (1)                            ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 3789
  (1)                            ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
 3790  012014  052737  010000  001124          BIS     #BIT12,$GDDAT
 3791
  (1)                            ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 3792
  (1)                            ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
 3793  012042  032737  000001  001126          BIT     #BIT00,$BDDAT
 3794  012050  001001                          BNE     1$              ;BR IF YES - NEXT TEST.
 3795
 3796
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 3797  012052  104001                          ERROR   1               ;ERROR - BIT00 OF CLOCK A'S STATUS REGISTER
 3798                                                                  ;FAILED TO SET WHEN BIT13 WAS SET
 3799                                                                  ;AND A MAINTENANCE ST1 GENERATED.
 3800
 3801
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 3802  012054                          1$:                             ;LEAVE SUBTEST WITH CLOCK CLEAR.
 3803  012054  005037  001124                  CLR     $GDDAT
 3804
  (1)                            ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 3805
 3817
 3818                            ;;**********************************************************************
  (3)                            ;*TEST 74       *TEST THAT CLOCK A WILL .NCREMENT - MODE 0 - RATE STP1 FIRST COUNT TEST
  (4)                            ;*
  (4)                            ;*COUNT TEST - THIS IS THE VERY FIST TIME THAT THE COUNTER
  (4)                            ;*HAS BEEN ASKED TO INCREMENT! WHAT WE ARE GOING TO DO IS
  (4)                            ;*CLEAR THE BUFFER, SELECT MODE 0, RATE OF STP1.
  (4)                            ;*NEXT WILL GENERATE THE STP1 THROUGH MAINTENANCE MODE (WE'VE
  (4)                            ;*DONE THIS BEFORE IN A PREVIOUS TEST TO SEE IF THE FLAG WOULD SET)
  (4)                            ;*AND SEE IF THE COUNTER HAS INCREMENTED.
  (5)                            ;*
  (5)                            ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (5)                            ;*
  (4)                            ;*
  (3)                            ;;**********************************************************************
  (2)  012070  000004            TST74:  SCOPE
 3819
 3820                                                                  ;CLEAR CLOCK A.
 3821                                                                  ;CLEAR BUFFER REGISTER.
 3822  012072  005037  001124                  CLR     $GDDAT
 3823
  (1)                            ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 3824
  (1)                            ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
 3825                                                                  ;SELECT: MODE0! RATE ''STP1'' AND ENABLE
 3826                                                                  ;CLOCK A TO COUNT.
 3827                                                                  ;NOW GENERATE A MAINTENANCE STP1 - A'
 3828                                                                  ;THIS TIME THE CLOCK SHOULD COUNT ONCE.
```

```
 3829  012116  012737  000015  001124          MOV     #15,$GDDAT
 3830
  (1)                            ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 3831
  (1)                            ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
 3832  012144  052737  010000  001124          BIS     #BIT12,$GDDAT
 3833
  (1)                            ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 3834  012162  012737  000001  001124          MOV     #1,$GDDAT       ;FOR ERROR TYPEOUT (IF NEEDED) SET THE #1.
 3835                                                                  ;READ THE COUNT REGISTER - SHOULD=1.
 3836
  (1)                            ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 3837  012200  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;DID THE COUNTER COUNT ONCE?
 3838  012206  001401                          BEQ     1$              ;IF YES - BR NEXT TEST.
 3839
 3840
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 3841  012210  104011                          ERROR   11              ;ERROR - COUNT A FAILED TO COUNT
 3842                                                                  ;ONCE, WHEN ENABLED, MODE 0
 3843                                                                  ;RATE ''STP1'' SEE ABOVE COMMENTS
 3844                                                                  ;FOR COMPLETE DESCRIPTION AND LIST
 3845                                                                  ;OF EVENTS.
 3846
 3847
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 3848  012212                          1$:
 3849
 3866
 3867                           ;;*********************************************************************
  (3)                           ;*TEST 75       *TEST THE ABILITY OF CLOCK A TO COUNT FROM ZERO TO OVERFLOW USING M STP1
  (4)                           ;*
  (4)                           ;*IN THIS TEST WE'LL COUNT THE COUNTER THROUGH EACH STEP
  (4)                           ;*FROM ZERO TO OVERFLOW USING MAINTENANCE ''STP1'' COUNTS.
  (4)                           ;*IT IS KNOWN THAT THE COUNTER WILL INCREMENT ONCE USING
  (4)                           ;*THE MAINTENANCE STP1 AND THAT THE COUNTER F/FS WILL
  (4)                           ;*PASS ALL DATA BITS.
  (4)                           ;*UNKNOW IS THE ABILITY OF THE F/F'S TO PROPAGATE THEIR
  (4)                           ;*OVERFLOWS.
  (4)                           ;*
  (4)                           ;*IF IT IS DESIRED TO START THIS TEST AT A VALUE OTHER THAN
  (4)                           ;*ZERO, CHANGE THE SECOND INSTR. OF THIS TEST TO A VALUE TO BE
  (4)                           ;*LOADED INTO THE BUFFER TO THAT VALUE DESIRED.
  (5)                           ;*
  (5)                           ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (5)                           ;*
  (4)                           ;*
  (3)                           ;*********************************************************************
  (2)  012212  000004           TST75:  SCOPE
  (1)  012214  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
 3868
 3869  012222  005037  001124                  CLR     $GDDAT  ;START THE COUNTER FROM ZERO.
 3870                                                                  ;NOTE: A VALUE OTHER THAN ZERO MAY BE
 3871                                                                  ;PATCHED IN HERE IN ORDER THAT A COUNT
 3872                                                                  ;MAY BE STARTED HIGHER.
```

```
3873  012226                                1$:                                 ;DISABLE CLOCK A.
3874  012226  005037  001356                      CLR     $TMDAT
3875
 (1)                                        ;*     MOV     $TMDAT,@ASR         ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
3876
 (1)                                        ;*     MOV     $GDDAT,@ABR         ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
3877                                                                            ;LOAD THE COUNTER BUFFER WITH VALUE.
3878                                                                            ;"1$" IS THE LOOP BACK POINT ON
3879                                                                            ;"LOOP ON TEST" (SW14=1) FEATURE. NOWMAL
3880                                                                            ;LOOP BACK POINT WILL BE "2$".
3881
3882  012252  012737  000015  001356            MOV     #15,$TMDAT
3883
 (1)                                        ;*     MOV     $TMDAT,@ASR         ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
3884                                                                            ;ENABLE COUNTER TO COUNT. SELECTED:
3885                                                                            ;MODE 0, RATE "STP1"
3886  012270                                2$:
 (1)
 (1)                                        ;*     MOV     @ASR,$TMDAT         ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
3887  012300  052737  010000  001356            BIS     #BIT12,$TMDAT       ;GENERATE A MAINTENANCE "STP1". THIS
3888
 (1)                                        ;*     MOV     $TMDAT,@ASR         ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
3889                                                                            ;ONE VALUE.
3890  012316  005237  001124                      INC     $GDDAT              ;$GDDAT IS USED TO KEEP TRACK OF THE
3891                                                                            ;VALUE THE CLOCK SHOULD COUNT TO.
3892
3893
 (1)                                        ;*     MOV     @ACR,$BDDAT         ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
3894
3895  012332  023737  001126  001124            CMP     $BDDAT,$GDDAT       ;DID COUNT OCCUR CORRECTLY?
3896  012340  001402                            BEQ     3$                  ;IF YES - BR TO "3$".
3897
3898
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3899  012342  104011                          ERROR   11                      ;ERROR - CLOCK A FAILED TO UPCOUNT
3900                                                                          ;CORRECTLY USING MODE 0 - RATE "STP1".
3901
3902
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

3903  012344  000403                            BR      4$
3904  012346  005737  001124          3$:     TST     $GDDAT              ;DID WE ACHIEVE OVERFLOW TO ZERO YET?
3905  012352  001410                            BEQ     5$                  ;IF YES - BR NEXT TEST
3906
3907  012354  032777  040000  166556  4$:     BIT     #BIT14,@SWR         ;LOOP CURRENT COUNT?
3908  012362  001742                            BEQ     2$                  ;BR IF NO TO NEXT COUNT UPDATE.
3909  012364  162737  000001  001124            SUB     #1,$GDDAT           ;IF YES - DECREMENT EXPECTED AND RELOAD.
3910  012372  000715                            BR      1$                  ;GOTO RELOAD POINT.
3911
3912  012374                                5$:                             ;END SUBTEST
3913
3914                                        .SBTTL  *
3915                                        .SBTTL  * PHASE 3 CLOCK A COUNT FUNCTION TESTS
3916                                        .SBTTL  *
3917
```

```
                                        ;;*********************************************************************
                                        ;*TEST 76        *TEST THAT CLOCK A OVERFLOW WILL OCCUR
                                        ;*
                                        ;*NOW WE'LL TRY AND GENERATE ''A OVERFLOW L''
                                        ;*WHICH SETS BD05. WE'LL DO IT BY PRESETTING THE BUFFER
                                        ;*AT 177777 AND GENERATE A MAINTENANCE STP1. WE ALREADY
                                        ;*KNOW MAINTENANCE STP1'S WILL ADVANCE THE COUNTER.
                                        ;*ALSO SEE IF 'MODE'' FLG GETS SET.
                                        ;*
                                        ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
                                        ;*
                                        ;*********************************************************************
012374  000004                        TST76:  SCOPE
                                                                        ;MAKE SURE CLOCK A CLEAR.
012376  005037  001124                        CLR     $GDDAT
                                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
012412  012737  177777  001124                MOV     #177777,$GDDAT  ;PRESET BUFFER TO ALL ONES.
                                        ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                                                        ;SELECT MODE 0; RATE STP4; GO.
                                                                        ;GENERATE A MAINTENANCE STP1.
012430  012737  000015  001124                MOV     #15,$GDDAT
                                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                        ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
012456  052737  010000  001124                BIS     #BIT12,$GDDAT
                                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
012474                                  1$:                             ;DID OVERFLOW BIT SET?
                                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
012504  032737  000040  001126                BIT     #BIT05,$BDDAT
012512  001002                              BNE     2$              ;BR IF YES TO '2$''.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
012514  104012                              ERROR   12              ;ERROR BIT05 OF CSR CLOCK A FAILED
                                                                        ;TO SET ON OVERFLOW.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
012516  000411                              BR      3$
012520                                  2$:                             ;DID BIT07 - 'MODE'' FLG SET?
                                        ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
012530  032737  000200  001126                BIT     #BIT07,$BDDAT
                                                                        ;IT SHOULD SET WHEN BIT05 SETS.
012536  001001                              BNE     3$              ;BR IF YES TO 3$.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
3959  012540  104012                          ERROR   12              ;OVERFLOW FAILED TO SET CLOCK A'S
3960                                                                  ;CSR BIT07 ''MODE'' FLG. IT
3961                                                                  ;HAD; HOWEVER SET BIT05 ''OVERFLOW''
3962                                                                  ;FLAG.
3963
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

3964  012542                          3$:
3974                                  ;;*****************************************************************
 (3)                                  ;*TEST 77       *TEST IN CLOCK A THAT OVERFLOW IN MODE 0 CAUSE CLEARING OF ''ENB CNTR'' F/
 (4)                                  ;*
 (4)                                  ;*WE'RE GOING TO SEE IF ''A PELOAD'' H AND ''MODE 0'' H CLEAR
 (4)                                  ;*''ENB CNTR A'' F/F.  ''A RELOAD'' GENERATED ON OVERFLOW AND
 (4)                                  ;*''MODE 0'' GENERATED BY ''MODE A0 (0)'' AND ''MODE A0 (1)''
 (4)                                  ;*ALSO SEE IF COUNTER REGISTER GETS RELOADED
 (5)                                  ;*
 (5)                                  ;* PROBABLE SYNC POINT FOR THIS TEST:: ''LD BUFF A''
 (5)                                  ;*
 (4)     .                            ;*
 (3)                                  ;*****************************************************************
 (2)  012542  000004                  TST77:  SCOPE
3975
3976                                                                  ;MAKE SURE CLOCK A IS CLEAR.
3977  012544  005037  001124                 CLR     $GDDAT
3978
 (1)                                  ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
3979                                                                  ;PRESET BUFFER TO ALL ONES.
3980                                                                  ;SELECT MODE 0, RATE ''STP4''; GO.
3981                                                                  ;GENERATE A MAINTENANCE STP1.
3982                                                                  ;THIS SHOULD CAUSE AN OVERFLOW
3983                                                                  ;OVERFLOW WILL GENERATE ''A RELOAD'' H
3984                                                                  ;COMBINE THIS WITH MODE 0 AND WE
3985  012560  012737  177777  001124          MOV     #177777,$GDDAT
3986
 (1)                                  ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
3987  012576  012737  000015  001124          MOV     #15,$GDDAT
3988
 (1)                                  ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
3989
 (1)                                  ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
3990  012624  052737  010000  001124          BIS     #BIT12,$GDDAT
3991
 (1)                                  ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
3992                                                                  ;SHOULD CLEAR ''ENB CNTR A'' F/F.
3993
3994
 (1)                                  ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
3995  012652  032737  000001  001356          BIT     #BIT00,$TMDAT   ;DID BIT00 ''ENB CNTR A'' F/F CLEAR?
3996  012660  001402                          BEQ     1$              ;BR IF YES - NEXT TEST.
3997
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

3998  012662  104012                          ERROR   12              ;''ENB CNTR A'' F/F NOT CLEARED
3999                                                                  ;ON OVERFLOW.
4000
```

```
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 4001  012664  000411                          BR      2$
 4002
 4003  012666                          1$:
  (1)
  (1)                                   ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 4004  012676  022737  177777  001126           CMP     #177777,$BDDAT  ;DID COUNTER GET RELOADED AFTER
 4005                                                                    ;THE OVERFLOW OCCURRED?
 4006  012704  001401                          BEQ     2$              ;BR IF YES - NEXT TEST.
 4007
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 4008  012706  104012                          ERROR   12              ;ERROR CLOCK A - COUNTER DID NOT GET RELOAD FROM
 4009                                                                    ;BUFFER AFTER OVERFLOW. ''A RELOAD'' H
 4010                                                                    ;DID GET GENERATED ON WE WOULD
 4011                                                                    ;HAVE GOT PREVIOUS ERROR; THEREFORE PROBLEM
 4012                                                                    ;MUST EXIST WITH COMBINING ''A RELOAD'' H
 4013                                                                    ;WITH 'MODE A1 (0)'' H TO GENERATE A
 4014                                                                    ;COUNTER LOAD.
 4015
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 4016  012710                          2$:
 4064
```

```
                                        ;/#

                                ;;*****************************************************************
                                ;*TEST 100       *TEST THE ABILITY OF CLOCK A TO COUNT AT 1MHZ RATE PART 1
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
                                ;*IN RATE: 1MHZ          PART 1
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
                                ;*
                                ;*
                                ;;*****************************************************************
012710  000004                  TST100: SCOPE
012712  012737  000020  001166          MOV     #20,$TIMES      ;;DO 20 ITERATIONS

                                                                ;/MAKE SURE CLOCK IS CLEAR.
                                                                ;/CLEAR THE BUFFER.
                                                                ;/SELECT: MODE 0, RATE 1MHZ ; GO.
012720  005037  001124                  CLR     $GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
012744  012737  000003  001124          MOV     #^!2,$GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
012762  005000                  1$:     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
012764  005200                          INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
012766  001376                          BNE     1$

                                                                ;/DID COUNTER INCREMENT AT ALL?

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
013000  005737  001126                  TST     $BDDAT
013004  001011                          BNE     2$              ;/IF YES - BR NEXT TEST.

                                                                ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.

                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
013016  032737  000040  001126          BIT     #BIT05,$BDDAT

                                                                ;/AT HIGH RATE - SO WE'LL SEE IF 'OVERFLOW''
                                                                ;/F/F HAD SET.
013024  001001                          BNE     2$              ;/BR IF YES NEXT TEST.

                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

013026  104012                          ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
                                                                ;/COUNT RATE: 1MHZ.

                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

013030                          2$:
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*TEST 101       *TEST THE ABILITY OF CLOCK A TO COUNT AT 100KHZ RATE PART 1
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
                                ;*IN RATE: 100KHZ                PART 1
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
                                ;*
                                ;*
                                ;;**********************************************************************
013030  000004                TST101: SCOPE
013032  012737  000020  001166        MOV     #20,$TIMES      ;;DO 20 ITERATIONS
                                                                ;/MAKE SURE CLOCK IS CLEAR.
                                                                ;/CLEAR THE BUFFER.
                                                                ;/SELECT: MODE 0, RATE 100KHZ ; GO.
013040  005037  001124                CLR     $GDDAT
                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
013064  012737  000005  001124        MOV     #1!4,$GDDAT
                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
013102  005000                        CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
013104  005200                1$:     INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
013106  001376                        BNE     1$
                                                                ;/DID COUNTER INCREMENT AT ALL?
                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
013120  005737  001126                TST     $BDDAT
013124  001011                        BNE     2$              ;/IF YES - BR NEXT TEST.
                                                                ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.
                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
013136  032737  000040  001126        BIT     #BIT05,$BDDAT
                                                                ;/AT HIGH RATE - SO WE'LL SEE IF 'OVFRFLOW''
                                                                ;/F/F HAD SET.
013144  001001                        BNE     2$              ;/BR IF YES NEXT TEST.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
013146  104012                        ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
                                                                ;/COUNT RATE: 100KHZ.
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
013150                          2$:
```

```
 4067                                           ;/#
  (1)
  (5)                                           ;;*****************************************************************
  (4)                                           ;*TEST 102      *TEST THE ABILITY OF CLOCK A TO COUNT AT 10KHZ RATE PART 1
  (5)                                           ;*
  (5)                                           ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
  (5)                                           ;*IN RATE: 10KHZ                PART 1
  (6)                                           ;*
  (6)                                           ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (6)                                           ;*
  (5)                                           ;*
  (4)                                           ;*****************************************************************
  (3)  013150  000004                           TST102: SCOPE
  (2)  013152  012737  000020  001166                   MOV     #20,$TIMES      ;;DO 20 ITERATIONS
  (1)
  (1)                                                                           ;/MAKE SURE CLOCK IS CLEAR.
  (1)                                                                           ;/CLEAR THE BUFFER.
  (1)                                                                           ;/SELECT: MODE 0, RATE 10KHZ ; GO.
  (1)  013160  005037  001124                           CLR     $GDDAT
  (2)
  (2)                                           ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                                           ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
  (1)  013204  012737  000007  001124                   MOV     #1!6,$GDDAT
  (2)
  (2)                                           ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (1)  013222  005000                                   CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
  (1)  013224  005200                           1$:     INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
  (1)  013226  001376                                   BNE     1$
  (1)                                                                           ;/DID COUNTER INCREMENT AT ALL?
  (2)
  (2)                                           ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
  (1)  013240  005737  001126                           TST     $BDDAT
  (1)  013244  001011                                   BNE     2$              ;/IF YES - BR NEXT TEST.
  (1)
  (1)                                                                           ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.
  (2)
  (2)                                           ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
  (1)  013256  032737  000040  001126                   BIT     #BIT05,$BDDAT
  (1)                                                                           ;/AT HIGH RATE - SO WE'LL SEE IF 'OVERFLOW''
  (1)                                                                           ;/F/F HAD SET.
  (1)  013264  001001                                   BNE     2$              ;/BR IF YES NEXT TEST.
  (2)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  013266  104012                                   ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
  (1)                                                                           ;/COUNT RATE: 10KHZ.
  (2)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  013270                                   2$:
```

```
 4068
  (1)
  (5)                                   ;/#
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 103      *TEST THE ABILITY OF CLOCK A TO COUNT AT 1KHZ RATE PART 1
  (5)                                   ;*
  (5)                                   ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
  (5)                                   ;*IN RATE: 1KHZ          PART 1
  (6)                                   ;*
  (6)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (6)                                   ;*
  (5)                                   ;*
  (4)                                   ;;*****************************************************************
  (3)  013270  000004                   TST103: SCOPE
  (2)  013272  012737  000020  001166           MOV     #20,$TIMES      ;;DO 20 ITERATIONS
  (1)
  (1)                                           ;/MAKE SURE CLOCK IS CLEAR.
  (1)                                           ;/CLEAR THE BUFFER.
  (1)                                           ;/SELECT: MODE 0, RATE 1KHZ ; GO.
  (1)  013300  005037  001124           CLR     $GDDAT
  (2)
  (2)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                                   ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
  (1)  013324  012737  000011  001124           MOV     #1!10,$GDDAT
  (2)
  (2)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (1)  013342  005000                           CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
  (1)  013344  005200                   1$:     INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
  (1)  013346  001376                           BNE     1$
  (1)                                           ;/DID COUNTER INCREMENT AT ALL?
  (2)
  (2)                                   ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
  (1)  013360  005737  001126           TST     $BDDAT
  (1)  013364  001011                   BNE     2$              ;/IF YES - BR NEXT TEST.
  (1)
  (1)                                           ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.
  (2)
  (2)                                   ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
  (1)  013376  032737  000040  001126           BIT     #BIT05,$BDDAT
  (1)                                           ;/AT HIGH RATE - SO WE'LL SEE IF 'OVERFLOW''
  (1)                                           ;/F/F HAD SET.
  (1)  013404  001001                           BNE     2$              ;/BR IF YES NEXT TEST.
  (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
  (1)  013406  104012                           ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
  (1)                                           ;/COUNT RATE: 1KHZ.
  (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  013410                           2$:
```

```
 4069
  (1)
  (5)                                   ;;*********************************************************************
  (4)                                   ;*TEST 104      *TEST THE ABILITY OF CLOCK A TO COUNT AT 100HZ RATE PART 1
  (5)                                   ;*
  (5)                                   ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
  (5)                                   ;*IN RATE: 100HZ                PART 1
  (6)                                   ;*
  (6)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (6)                                   ;*
  (5)                                   ;*
  (4)                                   ;;*********************************************************************
  (3)  013410  000004                   TST104: SCOPE
  (2)  013412  012737  000020  001166           MOV     #20,$TIMES      ;;DO 20 ITERATIONS
  (1)
  (1)                                                                   ;/MAKE SURE CLOCK IS CLEAR.
  (1)                                                                   ;/CLEAR THE BUFFER.
  (1)                                                                   ;/SELECT: MODE 0, RATE 100HZ ; GO.
  (1)  013420  005037  001124           CLR     $GDDAT
  (2)
  (2)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                                   ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
  (1)  013444  012737  000013  001124           MOV     #1!12,$GDDAT
  (2)
  (2)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (1)  013462  005000                   1$:     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
  (1)  013464  005200                           INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
  (1)  013466  001376                           BNE     1$
  (1)                                                                   ;/DID COUNTER INCREMENT AT ALL?
  (2)
  (2)                                   ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
  (1)  013500  005737  001126                   TST     $BDDAT
  (1)  013504  001011                           BNE     2$              ;/IF YES - BR NEXT TEST.
  (1)
  (1)                                                                   ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.
  (2)
  (2)                                   ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
  (1)  013516  032737  000040  001126           BIT     #BIT05,$BDDAT
  (1)                                                                   ;/AT HIGH RATE - SO WE'LL SEE IF 'OVERFLOW''
  (1)                                                                   ;/F/F HAD SET.
  (1)  013524  001001                           BNE     2$              ;/BR IF YES NEXT TEST.
  (2)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  013526  104012                           ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
  (1)                                                                   ;/COUNT RATE: 100HZ.
  (2)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  013530                           2$:
```

```
4070                                    ;/#
 (1)
 (5)                                    ;;*********************************************************************
 (4)                                    ;*TEST 105      *TEST THE ABILITY OF CLOCK A TO COUNT AT LINE-FREQ RATE PART 1
 (5)                                    ;*
 (5)                                    ;*THIS TEST IS DESIGNED TO TEST CLOCK A'S ABILITY TO COUNT
 (5)                                    ;*IN RATE: LINE-FREQ            PART 1
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (6)                                    ;*
 (5)                                    ;*
 (4)                                    ;;*********************************************************************
 (3)  013530  000004                    TST105: SCOPE
 (2)  013532  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                                                    ;/MAKE SURE CLOCK IS CLEAR.
 (1)                                                                    ;/CLEAR THE BUFFER.
 (1)                                                                    ;/SELECT: MODE 0, RATE LINE-FREQ ; GO.
 (1)  013540  005037  001124            CLR     $GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
 (1)  013564  012737  000017  001124            MOV     #1!16,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (1)  013602  005000                    1$:     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY: THIS DELAY
 (1)  013604  005200                            INC     R0              ;/WILL AMOUNT TO 369MS ON A PDP-11/20
 (1)  013606  001376                            BNE     1$
 (1)                                                                    ;/DID COUNTER INCREMENT AT ALL?
 (2)
 (2)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 (1)  013620  005737  001126                    TST     $BDDAT
 (1)  013624  001011                            BNE     2$              ;/IF YES - BR NEXT TEST.
 (1)
 (1)                                                                    ;/COUNTER MAY HAVE HAD TIME TO OVERFLOW.
 (2)
 (2)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
 (1)  013636  032737  000040  001126            BIT     #BIT05,$BDDAT
 (1)                                                                    ;/AT HIGH RATE - SO WE'LL SEE IF 'OVERFLOW''
 (1)                                                                    ;/F/F HAD SET.
 (1)  013644  001001                            BNE     2$              ;/BR IF YES NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  013646  104012                            ERROR   12              ;/ERROR CLOCK A COUNTER FAILED TO
 (1)                                                                    ;/COUNT RATE: LINE-FREQ.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  013650                            2$:
4071
4080
4081                                    ;;*********************************************************************
 (3)                                    ;*TEST 106      *TEST THAT CLOCK A DOESN'T COUNT WHEN NO RATE IS SELECTED
 (4)                                    ;*
```

```
 (4)                                    ;*WE KNOW THAT CLOCK A COUNTS AT ALL
 (4)                                    ;*ITS RATES, SO LETS BE SURE IT DOESNT
 (4)                                    ;*COUNT WHEN NO RATE IS SELECTED. ON
 (4)                                    ;*ERROR SUSPECT DUAL RATE SELECTION.
 (5)                                    ;*
 (5)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (5)                                    ;*
 (3)                                    ;;*****************************************************************
 (2)  013650  000004                    TST106: SCOPE
 (1)  013652  012737  000050  001166            MOV     #50,$TIMES      ;;DO 50 ITERATIONS
4082
4083                                                                    ;CLEAR CLOCK A.
4084                                                                    ;ZERO ITS BUFFER.
4085                                                                    ;TELL THE CLOCK TO GO!
4086  013660  005037  001124                   CLR     $GDDAT
4087
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4088
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
4089  013704  005237  001124                   INC     $GDDAT
4090
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4091                                                                    ;NO RATE SELECTED.
4092  013720  005000                    1$:     CLR     R0
4093  013722  005200                            INC     R0
4094  013724  001376                            BNE     1$
4095                                                                    ;ANY COUNT OCCUR?
4096
 (1)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
4097  013736  005737  001126                   TST     $BDDAT
4098  013742  001010                            BNE     2$              ;IF COUNTER HAS SOMETHING IN IT REPORT ERROR
4099
4100
 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
4101  013754  032737  000040  001126            BIT     #BIT05,$BDDAT   ;IF THE OVERFLOW BIT SET THEN IT MUST
4102                                                                    ;HAVE COUNTED.
4103  013762  001401                            BEQ     3$
4104
4105
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

4106  013764  104012                    2$:     ERROR   12              ;AH HA! ERROR CLOCK A COUNTED WHEN
4107                                                                    ;ENABLED-BUT-NO-RATE SELECTED
4108                                                                    ;BETTER FIND OUT HOW CAUSED SIGNAL:
4109                                                                    ;''CLOCK A'' L.
4110
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

4111  013766                            3$:
4112
4122
4123                                    ;;*****************************************************************
 (3)                                    ;*TEST 107      *TEST THAT CLOCK A'S COUNT REG ISN'T LOADED WHEN CLOCK A IS ENABLED
 (4)                                    ;*
 (4)                                    ;*IN THIS TEST WE'LL FIND OUT IF F/F ''ENB CNTR A'' WHEN SET,
 (4)                                    ;*INHIBITS LOADING OF CLOCK A'S COUNT REGISTER
```

```
                                ;*THE LOADING OF THE COUNT REGISTER IS A FUNCTION OF:
                                ;*'ENB CNTR (0)'' H + 'BUFFER LOAD'' H
                                ;*
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'RD STAT B''
                                ;*
                                ;;*****************************************************************
                                T107: SCOPE
                                                                ;GENERATE A SYNC PULSE
                                                                ;CLEAR CLOCK A'S STATUS REG.
                                                                ;CLEAR CLOCK A'S BUFFER REG. NOTE
                                                                ;THIS WILL ALSO CAUSE ZEROS TO BE
                                                                ;LOADED INTO THE COUNT REG.

                                ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
014000  005737  001126          1$:     TST     $BDDAT
014004  005037  001124                  CLR     $GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
014030  005237  001124                  INC     $GDDAT          ;SET THE ENABLE F/F. THIS

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                                                ;FROM BEING LOADED WHEN THE
                                                                ;BUFFER IS LOADED.
014044  012737  177777  001124          MOV     #177777,$GDDAT  ;NOW LOAD THE BUFFER REG.

                                ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
                                                                ;TO THE COUNT REGISTER.
                                                                ;DID ANY BITS GET TRANSFERRED TO
                                                                ;THE COUNT REGISTER?

                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
014072  005737  001126          2$:     TST     $BDDAT
014076  001401                  3$:     BEQ     1$              ;BR IF NO - NEXT TEST.

                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

014100  104012                          ERROR   12              ;ERROR CLOCK A BUFFER TO COUNT REG TRANSFER
                                                                ;OCCURRED EVEN THOUGH THE 'ENB CNTR A'' F/F
                                                                ;WAS SET.

                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

014102                          1$:


                                ;;*****************************************************************
                                ;*TEST 110      *TEST THAT CLOCK A IN MODE 1 DOES NOT CLEAR ENABLE ON OVERFLOW
                                ;*
                                ;*NOW W='RE GOING TO SEE IF ''A OVERFLOW'' H WHEN GENERATED
                                ;*BY CAUSING AN OVERFLOW DOFSN'T CLEAR THE 'ENB CNTR A'' F/F
                                ;*WHEN IN MODE 1. IF F/F GETS CLEARED SUSPECT SIGNAL 'MODE 0'' H.
                                ;*
```

```
 (5)                                    ;*
 (5)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (5)                                    ;*
 (3)                                    ;;*****************************************************************
 (2)  014102  000004                    TST110: SCOPE
4164
4165                                                                ;MAKE SURE CLOCK A IS CLEAR.
4166                                                                ;SET BUFFER + COUNT REG. TO -1 FROM OVERFLOW
4167                                                                ;SET: MODE 1; RATE STP1; GO.
4168                                                                ;CAUSE A MAINTENANCE STP4. CLOCK 1
4169                                                                ;SHOULD OVERFLOW - BUT THIS OVERFLOW SHOULD
4170                                                                ;NOT CLEAR 'ENB CNTR A'' F/F.
4171                                                                ;DID BIT00, 'ENB CNTR A'' F/F GET CLEARED?
4172  014104  005037  001124                       CLR     $GDDAT
4173
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4174  014120  012737  177777  001124              MOV     #177777,$GDDAT
4175
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
4176  014136  012737  000415  001124              MOV     #415,$GDDAT
4177
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4178
 (1)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
4179  014164  052737  010000  001356              BIS     #BIT12,$TMDAT
4180
 (1)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
4181
 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
4182  014212  032737  000001  001126              BIT     #BIT00,$BDDAT
4183  014220  001001                             BNE     1$              ;BR IF NO TO NEXT TEST.
4184
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4185  014222  104012                             ERROR   12              ;ERROR MODE 1 OPERATION 'ENB CNTR A'' F/F
4186                                                                    ;WAS CLEARED ON OVERFLOW.
4187
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4188  014224                        1$:
4189
4201
4202                                    ;;*****************************************************************
 (3)                                    ;*TEST 111      *TEST THAT A CLOCK A 'BUFFER TO COUNT REG'' DOESN'T TAKE PLACE ON A MODE
 (4)                                    ;*
 (4)                                    ;*THIS WILL BE THE FIRST TIMF WE'VE DONE A MODE 2 OPERATION
 (4)                                    ;*ON CLOCK A. THE FUNCTION OF THIS TEST WILL BE TO MAKE
 (4)                                    ;*SURE A BUFFER TO COUNT REGISTER DOESN'T TAKE PLACE ON OVERFLOW
 (4)                                    ;*THE COMBO THAT GAVE US BUFFER TO COUNT REG ON OVERFLOW BEFORE
 (4)                                    ;*IS ['MODE A1 (0)'' H + ''A RELOAD'' H]. WE'LL STILL BE GENERATING
 (5)                                    ;*
 (5)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (5)                                    ;*
 (4)                                    ;*''A RELOAD'' H BUT SHOULD NOT GET.'MODE A1 (0)'' H AS THIS IS MODE 2.
 (4)                                    ;*
 (3)                                    ;;*****************************************************************
```

```
  (2)  014224  000004                    TST111: SCOPE
4203
4204                                                          ;MAKE SURE CLOCK A IS CLEAR.
4205                                                          ;SET BUFFER + COUNT REG TO -1 FROM OVERFLOW.
4206                                                          ;SET: MODE 2; RATE STP1; GO.
4207                                                          ;CAUSE A MAINTENANCE STP1 - CLOCK ONCE.
4208                                                          ;THIS SHOULD CAUSE AN OVERFLOW AND LEAVE
4209                                                          ;THE COUNT REGISTER CLEAR AS A
4210                                                          ;BUFFER TO COUNT REG. SHOULDN'T OCCUR.
4211                                                          ;IS THE COUNT REG. CLEAR?
4212  014226  005037  001124                      CLR     $GDDAT
4213
  (1)                                  ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4214  014242  012737  177777  001124            MOV     #177777,$GDDAT
4215
  (1)                                  ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
4216  014260  012737  001015  001124            MOV     #1015,$GDDAT
4217
  (1)                                  ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4218
  (1)                                  ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
4219  014306  052737  010000  001124            BIS     #BIT12,$GDDAT
4220
  (1)                                  ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4221
  (1)                                  ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
4222  014334  005737  001126                    TST     $BDDAT
4223  014340  001401                            BEQ     1$              ;BR IF YES TO NEXT TEST.
4224
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

4225  014342  104012                      ERROR   12                    ;ERROR THE CONTENTS OF THE BUFFER REG.
4226                                                          ;GOT TRANSFERRED TO THE COUNT REG.
4227                                                          ;(CLOCK A) ON OVERFLOW DOING A
4228                                                          ;MODE 2 OPERATION.
4229
4230                                                          ;THERE'S AN OUTSIDE CHANCE THAT THE
4231                                                          ;COUNT REG. NEVER GOES TO ZERO ON
4232                                                          ;AN OVERFLOW - THIS IS THE FIRST TIME
4233                                                          ;THAT WE WERE ABLE TO LOOK AT IT ON
4234                                                          ;OVERFLOW BECAUSE MODES 0 + 1 CAUSED
4235                                                          ;THAT AUTOMATIC BUFFER TO COUNT REG.
4236
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

4237  014344                        1$:
4238
4250
4251                                ;;*************************************************************************
  (3)                               ;*TEST 112      *TEST THAT CLOCK A MODE 2 + MAINTENANCE ST2 SET MODE FLG
  (4)                               ;*
  (4)                               ;*NOW WE'LL SEE IF CAN GENERATE A ''CNTR TO BUFF'' H SIGNAL.
  (4)                               ;*TO DETECT IT, WE'RE GOING TO DEPEND ON IT SETTING THE MODE FLAG,
  (4)                               ;*CLOCK A CSR BIT07. ['MODE A1 (0)'' H + ''ST2 (1)'' H] + ''TP0'' L=''CNTR TO BUFF'' H.
  (4)                               ;*BEING IN MODE 2, SHOULD GIVE US 'MODE A1 (1)'' H. WELL GET ST2 (1) H
  (4)                               ;*BY GENERATING A MAINTENANCE ''ST2''. TP0 COMES FROM THE INTERNAL
```

```
 (4)                                ;*CLOCK PAGE.
 (5)                                ;*
 (5)                                ;* PROBABLE SYNC POIN" FOR THIS TEST:: 'RD STAT B''
 (5)                                ;*
 (4)                                ;*
 (3)                                ;;*****************************************************************
 (2)  014344  000004                TST112: SCOPE
4252
4253                                                        ;GENERATE A SYNC PULSE
4254
 (1)                                ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
4255  014356  005737  001126                TST     $BDDAT
4256                                                        ;MAKE SURE CLOCK A'S STAT REG IS CLEAR.
4257  014362  005037  001124                CLR     $GDDAT
4258
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4259                                                        ;SET MODE 2.
4260                                                        ;GENERATE MAINTENANCE ST2.
4261                                                        ;THE COMBO OF MODE 2 + ST2 SHOULD
4262                                                        ;GET ''CNTR TO BUFF'' H WHICH SHOULD
4263                                                        ;SET 'MODE FLG'' F/F.
4264  014376  012737  001000  001124        MOV     #1000,$GDDAT
4265
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4266
 (1)                                ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
4267  014424  052737  002000  001124        BIS     #BIT10,$GDDAT
4268
 (1)                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4269                                                        ;DID IT SET?
4270
 (1)                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
4271  014452  032737  000200  001126        BIT     #BIT07,$BDDAT
4272  014460  001001                      BNE     1$              ;IF YES-BR TO NEXT TEST.
4273
                   ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

4274  014462  104012                      ERROR   12              ;ERROR - MODE 2 + ST2 DID NOT SET MODE FL.
4275
                   ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$'$$$$$$$$$

4276  014464                       1$:
4277
4332
 (5)                                ;;*****************************************************************
 (4)                                ;*TEST 113       *TEST THAT PATTERN 052525 CAN BE XFERRED BETWEEN A'S COUNT-BUFFER REGS
 (5)                                ;*
 (5)                                ;*NOW WE'LL SHOT THE WORKS - WE KNOW FROM THE PREVIOUS TEST WE
 (5)                                ;*CAN GENERATE ''CNTR TO BUFF'' H FROM MODE 2 + MAINTENANCE ST2, NOW
 (5)                                ;*WE WELL TRY AND GENERATE A TRANSFER BETWEEN THE COUNTER AND BUFFER
 (5)                                ;*USING A CB PAT PATTERN.
 (5)                                ;*IF NO DATA PATTERN GETS TRANSFERRED, SUSPECT SIG 'LD BUFFER''
 (5)                                ;*TO BE STUCK LOW AT THE MUX INPUT FOR THE BUFFER OR
 (5)                                ;*''CNTR TO BUFF'' H NOT GETTING THROUGH TO THE LOAD INPUTS OF THE
 (5)                                ;*BUFFER REGISTER.
 (5)                                ;*IF JUST ONE OR A FEW BITS GETS MESSED UP ON THE XFERR-
```

```
(5)                                 ;*SUSPECT THE RESPECTIVE MUX OR ETCH BETWEEN THE COUNT REG
(5)                                 ;*AND MUX. GOOD LUCK.
(6)                                 ;*
(6)                                 ;* PROBABLE SYNC POINT FOR THIS TEST:: 'RD STAT B''
(6)                                 ;*
(5)                                 ;*
(4)                                 ;;*****************************************************************
(3)  014464  000004                 TST113: SCOPE
(1)
(1)                                                             ;GENERATE A SYNC PULSE
(2)
(2)                                 ;*      MOV     @#BSR,$BDDAT    ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(1)  014476  005737  001126                 TST     $BDDAT
(1)                                                             ;MAKE SURE CLOCK A IS CLEAR.
(1)                                                             ;PUT PATTERN 052525 INTO BUFFER REG.
(1)                                                             ;IT SHOULD GET XFERRED TO COUNT REG.
(1)                                                             ;SELECT: MODE 2, ENABLE.
(1)                                                             ;NOW GENERATE A MAINTENANCE ST2.
(1)  014502  005037  001124                 CLR     $GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@#ASR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  014516  012737  052525  001124         MOV     #052525,$GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@#ABR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(1)  014534  012737  001001  001124         MOV     #1001,$GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@#ASR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  014552  005037  001124                 CLR     $GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@#ABR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(2)
(2)                                 ;*      MOV     @#ASR,$GDDAT    ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
(1)  014576  052737  002000  001124         BIS     #BIT10,$GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@#ASR    ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  014614  012737  052525  001124         MOV     #052525,$GDDAT  ;RECORD $GDDAT (PATTERN) IN CASE WE
(1)                                                             ;NEED TO TYPE OUT AN ERROR.
(1)                                                             ;NOW READ BACK THE BUFFER REG.
(2)
(2)                                 ;*      MOV     @#ABR,$BDDAT    ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(1)  014632  023737  001126  001124         CMP     $BDDAT,$GDDAT   ;WAS THE TRANSFER SUCCESSFUL?
(1)  014640  001401                         BEQ     1$              ;IF YES THEN BR TO NEXT TEST.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  014642  104012                         ERROR   12              ;ERROR FAILED TO XFERR 052525 PATTERN
(1)                                                             ;CORRECTLY FROM COUNT TO BUFFER REG.
(1)                                                             ;SEE INIT. COMMENT AS TO WHY
(1)                                                             ;IT MIGHT HAVE GONE SOUR.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  014644                         1$:
(1)          000011                         P P+1
4333
(5)                                 ;;*****************************************************************
```

```
(4)                                 ;*TEST 114      *TEST THAT PATTERN 125252 CAN BE XFERRED BETWEEN A'S COUNT-BUFFER REGS
(5)                                 ;*
(5)                                 ;*NOW WE'LL SHOT THE WORKS - WE KNOW FROM THE PREVIOUS TEST WE
(5)                                 ;*CAN GENERATE ''CNTR TO BUFF'' H FROM MODE 2 + MAINTENANCE ST2, NOW
(5)                                 ;*WE WELL TRY AND GENERATE A TRANSFER BETWEEN THE COUNTER AND BUFFER
(5)                                 ;*USING A CB PAT PATTERN.
(5)                                 ;*IF NO DATA PATTERN GETS TRANSFERRED, SUSPECT SIG 'LD BUFFER''
(5)                                 ;*TO BE STUCK LOW AT THE MUX INPUT FOR THE BUFFER OR
(5)                                 ;*''CNTR TO BUFF'' H NOT GETTING THROUGH TO THE LOAD INPUTS OF THE
(5)                                 ;*BUFFER REGISTER.
(5)                                 ;*IF JUST ONE OR A FEW BITS GETS MESSED UP ON THE XFERR-
(5)                                 ;*SUSPECT THE RESPECTIVE MUX OR ETCH BETWEEN THE COUNT REG
(5)                                 ;*AND MUX. GOOD LUCK.
(6)                                 ;*
(6)                                 ;* PROBABLE SYNC POINT FOR THIS TEST:: 'RD STAT B''
(6)                                 ;*
(5)                                 ;*
(4)                                 ;;*****************************************************************
(3)  014644  000004                 T$T114: SCOPE
(1)
(1)                                                             ;GENERATE A SYNC PULSE
(2)
(2)                                 ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
(1)  014656  005737  001126                 TST     $BDDAT
(1)                                                             ;MAKE SURE CLOCK A IS CLEAR.
(1)                                                             ;PUT PATTERN 125252 INTO BUFFER REG.
(1)                                                             ;IT SHOULD GET XFERRED TO COUNT REG.
(1)                                                             ;SELECT: MODE 2, ENABLE.
(1)                                                             ;NOW GENERATE A MAINTENANCE ST2.
(1)  014662  005037  001124                 CLR     $GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  014676  012737  125252  001124         MOV     #125252,$GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(1)  014714  012737  001001  001124         MOV     #1001,$GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  014732  005037  001124                 CLR     $GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
(2)
(2)                                 ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
(1)  014756  052737  002000  001124         BIS     #BIT10,$GDDAT
(2)
(2)                                 ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
(1)  014774  012737  125252  001124         MOV     #125252,$GDDAT  ;RECORD $GDDAT (PATTERN) IN CASE WE
(1)                                                             ;NEED TO TYPE OUT AN ERROR.
(1)                                                             ;NOW READ BACK THE BUFFER REG.
(2)
(2)                                 ;*      MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
(1)  015012  023737  001126  001124         CMP     $BDDAT,$GDDAT   ;WAS THE TRANSFER SUCCESSFUL?
(1)  015020  001401                         BEQ     1$              ;IF YES THEN BR TO NEXT TEST.
(2)
                    ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
  (1)  015022  104012                          ERROR   12              ;ERROR FAILED TO XFERP 125252 PATTERN
  (1)                                                                  ;CORRECTLY FROM COUNT TO BUFFER REG.
  (1)                                                                  ;SEE INIT. COMMENT AS TO WHY
  (1)                                                                  ;IT MIGHT HAVE GONE SOUR.
  (2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  015024                          1$:
  (1)            000012                        P=P+1
 4334
 4341
 4377
  (5)                                  ;;*****************************************************************
  (4)                                  ;*TEST 115      *TEST THAT A'S COUNT REGISTER ISN'T CLEARED IN MODE 1 WHEN STP2 IS GENER
  (5)                                  ;*
  (5)                                  ;*THIS TEST IS DESIGNED TO MAKE SURE THAT WE DON'T CLEAR THE COUNT
  (5)                                  ;*REGISTER IN MODE 1 WHEN AN STP2 IS GENERATED.
  (6)                                  ;*
  (6)                                  ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (6)                                  ;*
  (5)                                  ;*
  (4)                                  ;*****************************************************************
  (3)  015024  000004                  TST115: SCOPE
  (1)
  (1)                                                                  ;MAKE SURE CLOCK A IS CLEAR.
  (1)                                                                  ;LOAD ALL ONES INTO BUFFER COUNT REGS.
  (1)                                                                  ;SET MODE 1.
  (1)                                                                  ;GENERATE A MAINTENANCE STP2.
  (1)                                                                  ;SEE IF IT CLEARED COUNT REG.
  (1)  015026  005037  001124                  CLR     $GDDAT
  (2)
  (2)                               ;*         MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (1)  015042  012737  177777  001124          MOV     #177777,$GDDAT
  (2)
  (2)                               ;*         MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
  (1)  015060  012737  000200  001124          MOV     #200,$GDDAT
  (2)
  (2)                               ;*         MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                               ;*         MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
  (1)  015106  052737  002000  001124          BIS     #BIT10,$GDDAT
  (2)
  (2)                               ;*         MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                               ;*         MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
  (1)  015134  005737  001126                  TST     $BDDAT
  (1)  015140  001001                          BNE     1$              ;BR IF NO - NEXT TEST.
  (2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  015142  104012                          ERROR   12              ;ERROR CLOCK A MODE 1 + STP2 CLEARED COUNT REG.
  (1)                                                                  ;TO SCOPE FIND OUT WHAT IS GENERATING
  (1)                                                                  ;SIGNAL ON CLR INPUT OF COUNT REG.
  (2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
  (1)  015144                          1$:
 4378
  (5)                                  ;;*****************************************************************
  (4)                                  ;*TEST 116      *TEST THAT A'S COUNT REGISTER ISN'T CLEARED IN MODE 2 WHEN STP2 IS GENER
  (5)                                  ;*
  (5)                                  ;*THIS TEST IS DESIGNED TO MAKE SURE THAT WE DON'T CLEAR THE COUNT
  (5)                                  ;*REGISTER IN MODE 2 WHEN AN STP2 IS GENERATED.
  (6)                                  ;*
  (6)                                  ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (6)                                  ;*
  (5)                                  ;*
  (4)                                  ;;*****************************************************************
  (3)  015144  000004                  TST16: SCOPE
  (1)
  (1)                                                                 ;MAKE SURE CLOCK A IS CLEAR.
  (1)                                                                 ;LOAD ALL ONES INTO BUFFER COUNT REGS.
  (1)                                                                 ;SET MODE 2.
  (1)                                                                 ;GENERATE A MAINTENANCE STP2.
  (1)                                                                 ;SEE IF IT CLEARED COUNT REG.
  (1)  015146  005037  001124                CLR     $GDDAT
  (2)
  (2)                               ;*       MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (1)  015162  012737  177777  001124        MOV     #177777,$GDDAT
  (2)
  (2)                               ;*       MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
  (1)  015200  012737  001000  001124        MOV     #1000,$GDDAT
  (2)
  (2)                               ;*       MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                               ;*       MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
  (1)  015226  052737  002000  001124        BIS     #BIT10,$GDDAT
  (2)
  (2)                               ;*       MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                               ;*       MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                               ;*       MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
  (1)  015254  005737  001126                TST     $BDDAT
  (1)  015260  001001                        BNE     1$              ;BR IF NO - NEXT TEST.
  (2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  015262  104012                        ERROR   12              ;ERROR CLOCK A MODE 2 + STP2 CLEARED COUNT REG.
  (1)                                                                ;TO SCOPE FIND OUT WHAT IS GENERATING
  (1)                                                                ;SIGNAL ON CLR INPUT OF COUNT REG.
  (2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  015264                          1$:
 4379
 4390
 4391                                  ;;*****************************************************************
  (3)                                  ;*TEST 117      *TEST THAT MODE 3 + ''STP2'' CLEARS A'S COUNT REGISTER.
  (4)                                  ;*
  (4)                                  ;*IN THIS TEST WE'LL DO A MODE 3 FOR THE FIRST TIME.
  (4)                                  ;*MODE 3 + ''STP2'' SHOULD CLEAR THE COUNT REGISTER.
  (4)                                  ;*WE KNOW FROM A PREVIOUS TEST THAT ''INIT'' L WAS ABLE TO
  (4)                                  ;*CLEAR THE REG. AND ALSO THAT WE CAN GENERATE AN ''STP2'' M.
```

```
 (4)                                    ;*'MODE A0 (1)'' H + 'MODE A1 (1)'' H + ''STP2'' H = CLEARING SIGNAL.
 (5)                                    ;*
 (5)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (5)                                    ;*
 (4)                                    ;*
 (3)                                    ;;*****************************************************************
 (2)  015264  000004                    TST117: SCOPE
4392
4393                                                                ;CLEAR CLOCK A + SELECT MODE 3.
4394                                                                ;LOAD ALL ONE'S TO BUFFER + COUNTER REGS.
4395                                                                ;GENERATE A MAINTENANCE ''STP2''.
4396                                                                ;THIS SHOULD CLEAR THE COUNT REG.
4397
4398                                                                ;IS COUNT REG. CLEAR?
4399  015266  012737  001400  001124          MOV     #1400,$GDDAT
4400
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4401  015304  012737  177777  001124          MOV     #177777,$GDDAT
4402
 (1)                                    ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
4403
 (1)                                    ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
4404  015332  052737  002000  001124          BIS     #BIT10,$GDDAT
4405
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4406
 (1)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
4407  015360  005737  001126                  TST     $BDDAT
4408  015364  001401                          BEQ     1$              ;BR IF YES - NEXT TEST.
4409
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

4410  015366  104012                    ERROR   12                      ;ERROR-CLOCK A-COUNT REGISTER FAILED TO
4411                                                                    ;CLEAR IN MODE 3 ON ''STP2'' PULSE.
4412
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$

4413  015370                    1$:
4414
4434
4435                                    ;;*****************************************************************
 (3)                                    ;*TEST 120       *TEST THE AUTODECREMENT FEATURE OF CLOCK A'S BUFFER
 (4)                                    ;*
 (4)                                    ;*THIS IS GOING TO BE A BIGGIE-WITH TWO PARTS.
 (4)                                    ;*PART1:         WE'RE LOADING THE BUFFER WITH ALL ONES; MODE 0; RATE STP1,
 (4)                                    ;*               WITH ''AUTO INC (1)'' SET (FOR FIRST TIME!). NOW WE'LL GENERATE
 (4)                                    ;*               THE THING TO WATCH FOR IS THE EXPLOSIVE COMBO OF
 (4)                                    ;*               'MODE A0 (0)'' H + 'MODE A1 (0)'' H (MODE 0) + ''AUTO INC (1)'' H +
 (4)                                    ;*               ''A OVERFLOW'' ALL FEEDING THE DOWN COUNT SIDE OF THE 74193 CHIP.
 (4)                                    ;*               THE RESULTS SHOULD BE 177776.
 (4)                                    ;*PART2:         IF PART 1 WAS SUCCESSFUL WE'LL LOOK TO SEE IF
 (4)                                    ;*               THE COUNTER GOT LOADED WITH THE NEW VALUE 177776.
 (4)                                    ;*               THE HANG-UP HERE IS THAT ''A OVERFLOW'' FEEDS A ONE SHOT
 (4)                                    ;*               THAT GENERATES ''A RELOAD'' H. WE KNOW THAT WE CAN
 (4)                                    ;*               GET ''A RELOAD'' H BUT THE BIG TEST HERE IS SEEING IF THAT
 (4)                                    ;*               ONE SHOT SPITS OUT ''A RELOAD'' H TOO SOON TO CATCH THE BUFFER
```

```
 (4)                            ;*              WITH ITS PANTS DOWN AS ITS DECREMENTING ITSELF.
 (4)                            ;*
 (5)                            ;*
 (5)                            ;* PROBABLE SYNC POINT FOR THIS TEST.: 'LD BUFF A''
 (5)                            ;*
 (3)                            ;;*****************************************************************
 (2) 015370  000004             TST120: SCOPE
4436
4437                                                            ;MAKE SURE CLOCK B IS CLEAR.
4438                                                            ;MAKE SURE CLOCK A IS CLEAR.
4439                                                            ;LOAD BUFFER + COUNT REGS.
4440                                                            ;SET AUTO INCREMENT MODE
4441                                                            ;ENABLE COUNTER, MODE 0, RATE STP1
4442                                                            ;ZAP. A MAINTENANCE STP1 HAS BEEN MADE.
4443                                                            ;STOP HERE AND LOOK WHAT JUST HAPPENED.
4444                                                            ;1. STP1 CLOCKED THE COUNTER (SET 177777).
4445                                                            ;2. COUNTER OVERFLOWED - ''A OVERFLOW'' L
4446                                                            ;3. MODE 0 + ''A OVERFLOW'' L + ''A AUTO INC (1)'' H
4447                                                            ;   DOWN COUNTED THE BUFFER.
4448                                                            ;4. ''A OVERFLOW'' L GOES THROUGH ONE SHOT DELAY
4449                                                            ;   TO GIVE ''A RELOAD'' H
4450                                                            ;5. ''A RELOAD'' H CAUSES A BUFFER TO COUNTER
4451                                                            ;   RELOAD.
4452 015372  005037  001124                CLR     $GDDAT
4453
 (1)                            ;*         MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
4454
 (1)                            ;*         MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4455 015416  012737  177777  001124        MOV     #177777,$GDDAT
4456
 (1)                            ;*         MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
4457 015434  012737  000020  001124        MOV     #20,$GDDAT
4458
 (1)                            ;*         MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
4459 015452  012737  000015  001124        MOV     #15,$GDDAT
4460
 (1)                            ;*         MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4461
 (1)                            ;*         MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
4462 015500  052737  010000  001124        BIS     #BIT12,$GDDAT
4463
 (1)                            ;*         MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4464
4465                                        ,PART 1 DID BUFFER GET DECREMENTED?
4466
4467 015516  012737  177776  001124        MOV     #177776,$GDDAT  ;SET FOR ERROR TYPEOUT  F ANY.
4468                                                                ;READ THE RESULTS OF THE BUFFER.
4469
 (1)                            ;*         MOV     @ABR,$BDDAT     ;/READ DEVICE REG ABR,PUT DATA IN $BDDAT.
4470 015534  023727  001126  177776        CMP     $BDDAT,#177776  ;DID BUFFER DECREMENT TO 177776?
4471 015542  001402                        BEQ     1$              ;BR IF YES TO PART 2
4472
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4473 015544  104012                        ERROR   12              ;ERROR-CLOCK A. AFTER AN OVERFLOW
4474                                                                ;WITH AUTO INC ENABLED MODE 0, THE
```

```
                                                        ;BUFFER REGISTER FAILED TO DOWN COUNT
                                                        ;SEE ABOVE COMMENTS FOR SEQUENCE
                                                        ;OF EVENTS.
015546  000411                  BR      2$              ;IF ERROR ONLY LOOP ON PART 1.

        :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

015550                  1$:     ;PART 2 DID NEW COUNT GET TRANSFERRED TO COUNT REGISTER?

                                                        ;READ THE COUNT REGISTER

                        ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
015560  023727  001126  177776        CMP     $BDDAT,#177776  ;DID NEW VALUE OF THE BUFFER
                                                        ;REGISTER GET PROPERLY LOADED INTO
                                                        ;THE COUNT REGISTER?
015566  001401                  BEQ     2$              ;BR IF YES - NEXT TEST.

        :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

015570  104012                  ERROR   12              ;ERROR CLOCK A. NEW CONTENTS OF
                                                        ;BUFFER REGISTER FAILED TO BE PROPERLY
                                                        ;LOADED INTO COUNT REGISTER AFTER
                                                        ;AUTO DECREMENT.
                                                        ;SEE ABOVE COMMENTS FOR SEQUENCE OF EVENTS.

        :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

015572                  2$:                             ;CLEAR AUTO INC OPTION.
015572  005037  001124          CLR     $GDDAT

                        ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR

                        ;;**********************************************************************
                        ;*TEST 121      *TEST THAT CLOCK A'S 1 MHZ CLK CAN BE DISABLED
                        ;*
                        ;*FOR THIS TEST, WE'LL TRY DISABLING THE 1 MHZ CLOCK. TO DO THIS,
                        ;*WE'LL SET THE 'DISABLE OSC 1 MHZ', BIT 11 IN CLOCK B SR. THEN
                        ;*COUNTED OR OVERFLOWED, IF SO ERROR.
                        ;*THE UNKNOWN THING HERE IS BIT 11 SETTING THE DISABLE F/F THAT
                        ;*GATES WITH THE 1 MHZ FREG TO PRODUCE 'A 1 MHZ CLK' L
                        ;*
                        ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT B'
                        ;*
                        ;;**********************************************************************
000004                  TST121: SCOPE
012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

                                                        ;CLEAR CLOCK A.
                                                        ;SET THE 'DISABLE OSC 1 MHZ' F/F.
                                                        ;CLEAR THE BUFFER + COUNT REGS.
                                                        ;START CLOCK: RATE 1 MHZ, MODE 0, GO.
015616  005037  001124          CLR     $GDDAT

                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
015632  012737  004000  001124          MOV     #BIT11,$GDDAT
```

```
  (1)                                   ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
4518  015650  005037  001124                    CLR     $GDDAT
4519
  (1)                                   ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
4520  015664  012737  000003  001124            MOV     #3,$GDDAT
4521
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4522  015702  005000                            CLR     R0              ;SHORT DELAY. WE ALLOW THIS DELAY TO
4523  015704  105200                    1$:     INCB    R0              ;OCCUR. IF THE 1 MHZ CLOCK IS DISABLED
4524  015706  100376                            BPL     1$              ;NO CLOCKING OF CLOCK A WILL OCCUR.
4525                                                                    ;SEE SOMETHING IN THE COUNT REGISTERS
4526
  (1)                                   ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
4527  015720  005737  001126                    TST     $BDDAT          ;DOES COUNT REG HAVE ANYTHING IN IT?
4528  015724  001007                            BNE     2$              ;YES - REPORT ERROR!
4529
  (1)                                   ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
4530  015736  105737  001126                    TSTB    $BDDAT
4531  015742  100001                            BPL     3$              ;NO - BR NEXT TEST - NO ERROR.
4532
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

4533  015744  104012                    2$:     ERROR   12              ;ERROR - UNABLE TO DISABLE 1 MHZ CLK
4534                                                                    ;USING "DISABLE OSC 1 MHZ" F/F
4535
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

4536                                                                    ;CLEAR CLOCK A.
4537  015746  005037  001124            3$:     CLR     $GDDAT
4538
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
```

```
                                        .SBTTL  *
                                        .SBTTL  * PHASE 4 CLOCK B COUNT FUNCTION TESTS
                                        .SBTTL  *


                                ;;**********************************************************************
                                ;*TEST 122      *TEST THAT CLOCK B WILL COUNT ONCE FIRST CLOCK B COUNT
                                ;*
                                ;*VERY FIRST TIME COUNTING WITH CLOCK B.
                                ;*WHAT WE'LL TRY AND DO IS COUNT IT FROM 0 TO 1.
                                ;*WE DO HAVE A PROBLEM HERE HOWEVER. WE CAN'T READ CLOCK B AS IT
                                ;*COUNTS AND THERE IS NO NICE WAY OF GENERATING A ''CLOCK B'' L PULSE.
                                ;*SO WE'RE GOING TO HAVE TO DO A COUPLE TRICKY THINGS: (1) DISABLE
                                ;*CLOCK A'S 1 MHZ CLOCK (WE DID THAT IN LAST TEST) SO THAT WE CAN
                                ;*GENERATE A MAINTENANCE 1 MHZ PULSE THROUGH CLOCK A (NEVER
                                ;*DID THAT BEFORE); (2) SET CLOCK B 'FEED B TO A'' BIT 05 (NEVER DID THAT
                                ;*BEFORE EITHER) SO THAT WE CAN ROUTE ''A 1 MHZ'' H TO MAKE
                                ;*''B 1 MHZ'' L TO GIVE US ''CLOCK B'' L
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT B''
                                ;*
                                ;*
                                ;;**********************************************************************
016762  000004                  TST122: SCOPE

                                                        ;CLEAR CLOCK B AND DISABLE 1MHZ OSC.
016764  012737  004000  001124          MOV     #BIT11,$GDDAT

                                ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
                                                        ;CLEAR B'S BUFFER + COUNT REGS.
                                                        ;SELECT 'FEED B TO A'', RATE 1 MHZ, GO.
016002  005037  001124                  CLR     $GDDAT

                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
                                                        ;CLOCK A'S 1 MHZ CLOCK.
                                                        ;GENERATE A MAINTENANCE 1 MHZ PULSE..
                                                        ;DID COUNTER COUNT?

                                ;*      MOV     @BSR,$GDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $GDDAT.
016026  052737  000043  001124          BIS     #BIT5!BIT1!BIT0,$GDDAT

                                ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR

                                ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
016054  052737  004000  001124          BIS     #BIT11,$GDDAT

                                ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR

                                ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
016102  005737  001126                  TST     $BDDAT
016106  001001                          BNE     1$              ;BR IF YES TO NEXT TEST.

                        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

016110  104014                          ERROR   14              ;ERROR - CLOCK B. FAILED TO ADVANCE
                                                        ;COUNT REGISTER.
```

```
                                                        ;TO SCOPE: CHECK LOAD + CLEAR INPUTS TO
                                                        ;COUNT REG TO SEE IF NOT BEING HELD LOW.
                                                        ;IF OK SEE IF WE'RE GETTING ANY ''CLOCK B'' L
                                                        ;PULSES - WE SHOULD GET ONE EACH LOOP OF
                                                        ;THIS SUBTEST. ''CLOCK B'' L COMES FROM
                                                        ;THE B CLOCK TIMING SECTION. THE RATE IS
                                                        ;1 MHZ. 'B 1 MHZ'' L IS GENERATED BY
                                                        ;''A 1 MHZ'' H + 'FEED B TO A (1)'' H.
                                                        ;THE ''A 1 MHZ'' IS GENERATED BY 'MAINT
                                                        ;SIMUL 1 MHZ'' IN CLOCK A.

            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

4595  016112                        1$:
4596
4610
4611                                ;;**********************************************************************
 (3)                                ;*TEST 123      *TEST THE ABILITY IF CLOCK B TO COUNT FROM ZERO TO OVERFLOW
 (4)                                ;*
 (4)                                ;*LAST TEST WE FOUND OUT WE COULD ADVANCE CLOCK A'S COUNTER,
 (4)                                ;*SO IN THIS TEST WE'RE GOING TO GO FROM 0-377 COUNTING.
 (4)                                ;*WE'LL USE THE SAME PROCEDURE AS LAST TEST TO GENERATE ''CLOCK B'' L.
 (4)                                ;*UNKNOWN IS THE ABILITY OF THE F/F'S TO PROPAGATE THEIR OVERFLOWS.
 (4)                                ;*
 (4)                                ;*IF IT IS DESIRED TO START THIS TEST AT A VALUE OTHER THAN 0, CHANGE
 (4)                                ;*THE SECOND INSTR. OF THIS TEST TO A VALUE TO BE LOADED INTO THE BUFFER
 (4)                                ;*TO THAT VALUE DESIRED.
 (4)                                ;*
 (5)                                ;*
 (5)                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF B''
 (5)                                ;*
 (3)                                ;;**********************************************************************
 (2)  016112  000004                TST123: SCOPE
 (1)  016114  012737  000001  001166        MOV     #1,$TIMES       ;;DO 1 ITERATION
4612
4613  016122  005037  001124              CLR     $GDDAT  ;START THE COUNTER FROM ZERO.
4614                                                        ;NOTE: A VALUE OTHER THAN ZERO MAY BE
4615                                                        ;PATCHED IN HERE IN ORDER THAT A COUNT
4616                                                        ;MAY BE STARTED HIGHER.
4617  016126  005037  001356        1$:   CLR     $TMDAT
4618
 (1)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
4619                                                        ;CLEAR CLOCK B, DISABLE 1 MHZ CLOCK A.
4620  016142  012737  004000  001356        MOV     #BIT11,$TMDAT
4621
 (1)                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
4622                                                        ;LOAD VALUE INTO COUNT + BUFFER REGS.
4623                                                        ;''1$'' IS THE LOOP BACK POINT ON
4624                                                        ;'LOOP ON TEST'' (SW14=1) FEATURE. NORMAL
4625                                                        ;LOOP BACK POINT WILL BE '2$''.
4626
4627
 (1)                                ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
4628                                                        ;SELECT: 'DISABLE OSC 1 MHZ''; 'FEED B TO A'';
4629                                                        ;RATE 1 MHZ.
4630                                                        ;GO. ENABL MUST BE SET AFTER 'FEED B TO A''
```

```
4631  016170  012737  004042  001356          MOV     #4042,$TMDAT
4632
 (1)                                  ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
4633  016206  005237  001356                    INC     $TMDAT
4634
 (1)                                  ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
4635
4636  016222                          2$:                             ;GENERATE A CLOCK PULSE.
4637                                                                  ;SHOULD CAUSE THE CLOCK TO
4638                                                                  ;INCREMENT ONCE.
4639
 (1)                                  ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
4640  016232  052737  004000  001356          BIS     #BIT11,$TMDAT
4641
 (1)                                  ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
4642  016250  005237  001124                    INC     $GDDAT          ;$GDDAT IS USED TO KEEP TRACK OF THE
4643                                                                  ;VALUE THE CLOCK SHOULD COUNT TO.
4644
4645
 (1)                                  ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
4646
4647  016264  123737  001126  001124          CMPB    $BDDAT,$GDDAT   ;DID THE COUNT OCCUR CORRECTLY?
4648  016272  001402                          BEQ     3$              ;IF YES - BR '3$'.
4649
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4650  016274  104015                          ERROR   15              ;ERROR CLOCK B FAILED TO UPCOUNT
4651                                                                  ;CORRECTLY - SEE COMMENTS AT SUB-TEST
4652                                                                  ;HEADING FOR MORE DETAILS.
4653
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4654  016276  000403                          BR      4$
4655
4656  016300  105737  001124          3$:     TSTB    $GDDAT          ;DID WE ACHIEVE OVERFLOW YET?
4657  016304  001410                          BEQ     5$              ;
4658
4659  016306  032777  040000  162624  4$:     BIT     #BIT14,@SWR     ;LOOP ON CURRENT COUNT?
4660  016314  001742                          BEQ     2$              ;BR IF NO TO NEXT COUNT UPDATE.
4661  016316  162737  000001  001124          SUB     #1,$GDDAT       ;IF YES DECREMENT EXPECTED AND RELOAD.
4662  016324  000700                          BR      1$              ;GO TO RELOAD POINT
4663
4664  016326                          5$:                             ;END SUBTEST.
4665
4676
4677                                    ;;**************************************************************************
 (3)                                    ;*TEST 124       *TEST THAT CLOCK B CAN GENERATE AN OVERFLOW
 (4)                                    ;*
 (4)                                    ;*NOW WE'LL TRY AND GENERATE 'B OVERFLOW' L WHICH SETS BIT 07.
 (4)                                    ;*WE'LL DO IT BY PRESETTING THE BUFFER TO 377 AND GENERATING A ''CLOCK B'' L
 (4)                                    ;*WE ALREADY KNOW WE CAN ADVANCE THE COUNTER, WHAT WE
 (4)                                    ;*WANT TO SEE IS 'B OVERFLOW' L COME OVER AND DIRECT SET
 (4)                                    ;*'OVERFLOW FL B'' F/F (BIT 07 IN CSR).
 (5)                                    ;*
 (5)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF B''
 (5)                                    ;*
```

```
 (4)                                    ;*
 (3)                                    ;**************************************************************
 (2)  016326  000004                    TST124: SCOPE
4678
4679                                                        ;CLEAR CLOCK A.
4680                                                        ;CLEAR CLOCK B.
4681                                                        ;SET COUNT AND BUFFER REGS.
4682                                                        ;SELECT 'DISABLE OSC 1 MHZ'; 'FEED B TO A';
4683                                                        ;RATE 1 MHZ.
4684                                                        ;GO. ENABL MUST BE SET AFTER 'FEED B TO A''
4685                                                        ;GENERATE A CLOCK PULSE.
4686
4687                                                        ;DID OVERFLOW FLAG SET?
4688  016330  005037  001124               CLR     $GDDAT
4689
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4690
 (1)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
4691  016354  012737  000377  001124          MOV     #377,$GDDAT
4692
 (1)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
4693  016372  012737  004042  001124          MOV     #4042,$GDDAT
4694
 (1)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
4695  016410  005237  001124                 INC     $GDDAT
4696
 (1)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
4697
 (1)                                    ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
4698  016434  052737  004000  001124          BIS     #BIT11,$GDDAT
4699
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4700
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4701  016462  105737  001126                 TSTB    $BDDAT
4702  016466  100402                        BMI     1$              ;BR IF YES TO '1$''.
4703
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4704  016470  104014                        ERROR   14              ;ERROR CLOCK B 'B OVERFL FLAG'' (CSR BIT7)
4705                                                        ;FAILED TO SET ON OVERFLOW.
4706
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4707  016472  000411                        BR      2$
4708  016474                        1$:                             ;WAS COUNTER RELOAD AFTER OVERFLOW
4709
 (1)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
4710  016504  022737  000377  001126          CMP     #377,$BDDAT
4711
4712  016512  001401                        BEQ     2$              ;BR IF YES TO NEXT TEST.
4713
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4714  016514  104014                        ERROR   14              ;ERROR CLOCK B COUNT REG. FAILED
4715                                                        ;TO BE RELOADED AFTER OVERFLOW.
```

```
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

                                        2$:
                                        ;;**********************************************************************
                                        ;*TEST 125      *TEST THE INIT. ABILITY OF CLOCK B'S COUNT REG.
                                        ;*
                                        ;*ALL WE'RE GOING TO DO HERE IS TEST THE INITIALIZE INPUTS
                                        ;*TO THE 74193 ICS THAT FORM THE COUNT REGISTER.
                                        ;*
                                        ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A''
                                        ;*
                                        ;*
                                        ;;**********************************************************************
016516  000004                          TST125: SCOPE
016520  012737  000001  001166                  MOV     #1,$TIMES       ;;DO 1 ITERATION

                                                                        ;CLEAR CLOCK A.
                                                                        ;CLEAR CLOCK B.
                                                                        ;LOAD ALL ONES TO BUFFER + COUNT REGS.
016526  005037  001124                          CLR     $GDDAT
                                        ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
                                        ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
016552  012737  000377  001124          MOV     #377,$GDDAT
                                        ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
016570  004737  033676                  JSR     PC,$RESET                       ;DO SYSTEM INIT - GENERATES ''INIT'' H.
016574  005037  001124                  CLR     $GDDAT          ;FIX $GDDAT FOR ERROR TYPEOUT, IF NEEDED.
                                                                ;READ B'S COUNT REGISTER, INIT
                                        ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
016610  005737  001126                  TST     $BDDAT
                                                                ;SHOULD HAVE MADE IT AL ZEROS.
016614  001401                          BEQ     1$              ;BR IF YES - NEXT TEST.

                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

016616  104007                          ERROR   7               ;ERROR CLOCK B - INIT FAILED TO CLEAR
                                                                ;COUNT REG.

                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

016620                                  1$:
```

```
4752                                    ;;**********************************************************************
 (3)                                    ;*TEST 126      *TEST THAT CLOCK B DOESN'T COUNT WHEN NO RATE IS SELECTED
 (3)                                    ;;**********************************************************************
 (2) 016620  000004                     TST126: SCOPE
 (1) 016622  012737  000100  001166             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
4753
4754                                                                    ;CLEAR CLOCK A.
4755                                                                    ;CLEAR CLOCK B.
4756                                                                    ;ZEROS TO BUFFER + COUNT REGS.
4757                                                                    ;ENABLE COUNTER TO COUNT - RATE - 0
4758                                                                    ;(NO RATE).
4759 016630  005037  001124                     CLR     $GDDAT
4760
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4761
 (1)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
4762
 (1)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
4763 016664  005237  001124                     INC     $GDDAT
4764
 (1)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
4765 016700  005000                             CLR     R0              ;DELAY FOR ANY COUNT THAT
4766 016702  105200                     1$:     INCB    R0              ;COULD FALSELY OCCUR.
4767 016704  001376                             BNE     1$              ;THIS DELAY APPROX. 369 MS ON A
4768                                                                    ;PDP 11/20.
4769
4770                                                                    ;DID ANY COUNT OCCUR?
4771
 (1)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
4772 016716  005737  001126                     TST     $BDDAT
4773 016722  001401                             BEQ     2$              ;IF NO BR TO NEXT TEST.
4774
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4775 016724  104014                             ERROR   14              ;ERROR - CLOCK B COUNTED WHEN ENABLED BUT
4776                                                                    ;NO RATE SELECTED. BETTER FIND OUT
4777                                                                    ;WHATS GENERATING ''CLOCK B'' L PULSES.
4778
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4779 016726                             2$:
4780
4831
```

```
4832
 (1)
 (5)                                    ;;*********************************************************************
 (4)                                    ;*TEST 127      *TEST THE ABILITY OF CLOCK B TO COUNT AT 1MHZ PART 1
 (5)                                    ;*
 (5)                                    ;*THIS TEST IS DESIGNED TO TEST CLOCK B'S ABILITY TO COUN
 (5)                                    ;*IN RATE: 1MHZ PART1
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A''
 (6)                                    ;*
 (5)                                    ;*
 (4)                                    ;;*********************************************************************
 (3)  016726  000004                    TST127: SCOPE
 (2)  016730  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                                                    ;/CLEAR CLOCK A.
 (1)  016736  005037  001124            CLR     $GDDAT
 (1)                                                                    ;/CLEAR CLOCK B.
 (1)                                                                    ;/CLEAR THE BUFFER + COUNT REGS.
 (1)                                                                    ;/SELECT: RATE: 1MHZ ; GO.
 (2)
 (2)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 (1)  016772  012737  000003  001124            MOV     #1!2,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  017010  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  017012  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  017014  001376                            BNE     1$              ;/ON A PDP-11/20.
 (1)
 (1)                                                                    ;/DID COUNTER COUNT AT ALL?
 (2)
 (2)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 (1)  017026  005737  001126                    TST     $BDDAT
 (1)  017032  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  017044  105737  001126                    TSTB    $BDDAT
 (1)                                                                    ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                                    ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  017050  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017052  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                                    ;/AT 1MHZ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017054                            2$:
```

```
 4833                                   ;/#
  (1)
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 130      *TEST THE ABILITY OF CLOCK B TO COUNT AT 100KHZ PART 1
  (5)                                   ;*
  (5)                                   ;*THIS TEST IS DESIGNED TO TEST CLOCK B'S ABILITY TO COUNT
  (5)                                   ;*IN RATE: 100KHZ       PART1
  (6)                                   ;*
  (6)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A''
  (6)                                   ;*
  (5)                                   ;*
  (4)                                   ;;*****************************************************************
  (3)  017054  000004                   TST130: SCOPE
  (2)  017056  012737  000020  001166           MOV     #20,$TIMES      ;;DO 20 ITERATIONS
  (1)
  (1)                                                                   ;/CLEAR CLOCK A.
  (1)  017064  005037  001124           CLR     $GDDAT
  (1)                                                                   ;/CLEAR CLOCK B.
  (1)                                                                   ;/CLEAR THE BUFFER + COUNT REGS.
  (1)                                                                   ;/SELECT: RATE: 100KHZ ; GO.
  (2)
  (2)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
  (2)
  (2)                                   ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
  (2)
  (2)                                   ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
  (1)  017120  012737  000005  001124           MOV     #1!4,$GDDAT
  (2)
  (2)                                   ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
  (1)
  (1)  017136  005000                           CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
  (1)  017140  005200                   1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
  (1)  017142  001376                           BNE     1$              ;/ON A PDP-11/20.
  (1)
  (1)                                                                   ;/DID COUNTER COUNT AT ALL?
  (2)
  (2)                                   ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
  (1)  017154  005737  001126                   TST     $BDDAT
  (1)  017160  001010                           BNE     2$              ;/BR IF YES - NEXT TEST.
  (1)
  (2)
  (2)                                   ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
  (1)  017172  105737  001126                   TSTB    $BDDAT
  (1)                                                                   ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
  (1)                                                                   ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
  (1)  017176  100401                           BMI     2$              ;/BR IF SET - NEXT TEST.
  (2)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  017200  104014                           ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
  (1)                                                                   ;/AT 100KHZ RATE.
  (2)
               ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  017202                           2$:
```

```
4834
 (1)
 (5)                                    ;;*****************************************************************
 (4)                                    ;*TEST 131      *TEST THE ABILITY OF CLOCK B TO COUNT AT 10KHZ PART 1
 (5)                                    ;*
 (5)                                    ;*THIS TEST IS DESIGNED TO TEST CLOCK B'S ABILITY TO COUNT
 (5)                                    ;*IN RATE: 10KHZ        PART1
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A''
 (6)                                    ;*
 (5)                                    ;*
 (4)                                    ;;*****************************************************************
 (3)  017202  000004                    TST131: SCOPE
 (2)  017204  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                            ;/CLEAR CLOCK A.
 (1)  017212  005037  001124            CLR     $GDDAT
 (1)                                            ;/CLEAR CLOCK B.
 (1)                                            ;/CLEAR THE BUFFER + COUNT REGS.
 (1)                                            ;/SELECT: RATE: 10KHZ ; GO.
 (2)
 (2)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 (1)  017246  012737  000007  001124            MOV     #1.6,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  017264  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  017266  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  017270  001376                            BNE     1$              ;/ON A PDP-11/20.
 (1)
 (1)                                            ;/DID COUNTER COUNT AT ALL?
 (2)
 (2)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 (1)  017302  005737  001126                    TST     $BDDAT
 (1)  017306  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  017320  105737  001126                    TSTB    $BDDAT
 (1)                                            ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                            ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  017324  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017326  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                            ;/AT 10KHZ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017330                            2$:
```

```
4835                                    ;/#
 (1)
 (5)                                    ;;*****************************************************************
 (4)                                    ;*TEST 132      *TEST THE ABILITY OF CLOCK B TO COUNT AT 1KHZ PART 1
 (5)                                    ;*
 (5)                                    ;*THIS TEST IS DESIGNED TO TEST CLOCK B'S ABILITY TO COUNT
 (5)                                    ;*IN RATE: 1KHZ PART1
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A''
 (6)                                    ;*
 (5)                                    ;*
 (4)                                    ;;*****************************************************************
 (3)  017330  000004                    TST132: SCOPE
 (2)  017332  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                            ;/CLEAR CLOCK A.
 (1)  017340  005037  001124            CLR     $GDDAT
 (1)                                            ;/CLEAR CLOCK B.
 (1)                                            ;/CLEAR THE BUFFER + COUNT REGS.
 (1)                                            ;/SELECT: RATE: 1KHZ ; GO.
 (2)
 (2)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 (1)  017374  012737  000011  001124            MOV     #1.10,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  017412  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  017414  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  017416  001376                            BNE     1$              ;/ON A PDP-11/20.
 (1)
 (1)                                            ;/DID COUNTER COUNT AT ALL?
 (2)
 (2)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 (1)  017430  005737  001126                    TST     $BDDAT
 (1)  017434  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  017446  105737  001126                    TSTB    $BDDAT
 (1)                                            ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                            ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  017452  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017454  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                            ;/AT 1KHZ RATE.
 (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017456                            2$:
```

```
4836
 (1)
 (5)                                    ;;*********************************************************************
 (4)                                    ;*TEST 133      *TEST THE ABILITY OF CLOCK B TO COUNT AT 100HZ PART 1
 (5)                                    ;*
 (5)                                    ;*THIS TEST IS DESIGNED TO TEST CLOCK B'S ABILITY TO COUNT
 (5)                                    ;*IN RATE: 100HZ        PART1
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A''
 (6)                                    ;*
 (5)                                    ;*
 (4)                                    ;;*********************************************************************
 (3)  017456  000004                    TST133: SCOPE
 (2)  017460  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                            ;/CLEAR CLOCK A.
 (1)  017466  005037  001124            CLR     $GDDAT
 (1)                                            ;/CLEAR CLOCK B.
 (1)                                            ;/CLEAR THE BUFFER + COUNT REGS.
 (1)                                            ;/SELECT: RATE: 100HZ ; GO.
 (2)
 (2)                                    ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 (1)  017522  012737  000011  001124            MOV     #1!11,$GDDAT
 (2)
 (2)                                    ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  017540  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  017542  005200                    1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  017544  001376                            BNE     1$              ;/ON A PDP-11/20.
 (1)
 (1)                                            ;/DID COUNTER COUNT AT ALL?
 (2)
 (2)                                    ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 (1)  017556  005737  001126                    TST     $BDDAT
 (1)  017562  001010                            BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                    ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  017574  105737  001126                    TSTB    $BDDAT
 (1)                                            ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                            ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  017600  100401                            BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017602  104014                            ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                            ;/AT 100HZ RATE.
 (2)
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017604                            2$:
```

```
4837                                            ;/#
 (1)
 (5)                                            ;;*****************************************************************
 (4)                                            ;*TEST 134      *TEST THE ABILITY OF CLOCK B TO COUNT AT LINE-FREQ PART 1
 (5)                                            ;*
 (5)                                            ;*THIS TEST IS DESIGNED TO TEST CLOCK B'S ABILITY TO COUNT
 (5)                                            ;*IN RATE: LINE-FREQ    PART1
 (6)                                            ;*
 (6)                                            ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT A''
 (6)                                            ;*
 (5)                                            ;*
 (4)                                            ;*****************************************************************
 (3)  017604  000004                            TST134: SCOPE
 (2)  017606  012737  000020  001166                    MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                                                            ;/CLEAR CLOCK A.
 (1)  017614  005037  001124                            CLR     $GDDAT
 (1)                                                                            ;/CLEAR CLOCK B.
 (1)                                                                            ;/CLEAR THE BUFFER + COUNT REGS.
 (1)                                                                            ;/SELECT: RATE: LINE-FREQ ; GO.
 (2)
 (2)                                            ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 (2)
 (2)                                            ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (2)
 (2)                                            ;*      MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 (1)  017650  012737  000017  001124                    MOV     #1.16,$GDDAT
 (2)
 (2)                                            ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (2)                                            ;*      MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 (1)
 (1)  017666  005000                                    CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY.
 (1)  017670  005200                            1$:     INC     R0              ;/THIS DELAY WILL AMOUNT TO APP. 269 MS.
 (1)  017672  001376                                    BNE     1$              ;/ON A PDP-11/20.
 (1)
 (1)                                                                            ;/DID COUNTER COUNT AT ALL?
 (2)
 (2)                                            ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
 (1)  017704  005737  001126                            TST     $BDDAT
 (1)  017710  001010                                    BNE     2$              ;/BR IF YES - NEXT TEST.
 (1)
 (2)
 (2)                                            ;*      MOV     @BSR,$BDDAT     ;/READ DEVICE REG BSR,PUT DATA IN $BDDAT.
 (1)  017722  105737  001126                            TSTB    $BDDAT
 (1)                                                                            ;/COUNT TO OVERFLOW - SO WE'LL SEE IF
 (1)                                                                            ;/THE OVERFLOW F/F SET BEFORE WE CRY WOLF.
 (1)  017726  100401                                    BMI     2$              ;/BR IF SET - NEXT TEST.
 (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017730  104014                                    ERROR   14              ;/ERROR CLOCK B - COUNTER FAILED TO COUNT
 (1)                                                                            ;/AT LINE-FREQ RATE.
 (2)
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 (1)  017732                                    2$:
4838
```

```
 4847
 4848                                ;;*****************************************************************
  (3)                                ;*TEST 135      *TEST THE 'FEED B TO A'' 24 BIT COUNTER FEATURE OF CLOCKS A + B
  (4)                                ;*
  (4)                                ;*WE'RE GOING TO TEST CLOCKS A+B AS A 24 BIT COUNTER; THAT IS;
  (4)                                ;*WE'RE GOINT TO TAKE THE OVERFLOW FROM CLOCK B AND FEED IT INTO
  (4)                                ;*CLOCK A.
  (5)                                ;*
  (5)                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF B''
  (5)                                ;*
  (4)                                ;*
  (3)                                ;;*****************************************************************
  (2)  017732  000004                TST135: SCOPE
 4849
 4850                                                        ;CLEAR CLOCK A.
 4851                                                        ;CLEAR CLOCK B.
 4852                                                        ;CLEAR A'S BUFFER + COUNT REGISTERS.
 4853                                                        ;PRESET B'S BUFFER + COUNT TO -1 FROM
 4854                                                        ;OVERFLOW.
 4855                                                        ;SELECT: 'DISABLE OSC 1 MHZ''; RATE 1 MHZ;
 4856                                                        ;'FEED B TO A''
 4857                                                        ;SET ENABL. MUST BE SET AFTER 'FEED B TO A''.
 4858                                                        ;ENABLE CLOCK A; MODE 0; RATE 0.
 4859                                                        ;SIMULATE A 1 MHZ PULSE - THIS PULSE
 4860                                                        ;WILL CLOCK CLOCK B'S COUNTER
 4861                                                        ;REGISTER. AN OVERFLOW WILL
 4862                                                        ;OCCUR - THAT OVERFLOW SHOULD
 4863                                                        ;CLOCK CLOCK A'S COUNT REGISTER.
 4864                                                        ;DID CLOCK A'S COUNT REG GET CLOCKED?
 4865  017734  005037  001124                   CLR     $GDDAT
 4866
  (1)                                ;*        MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 4867
  (1)                                ;*        MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 4868
  (1)                                ;*        MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
 4869  017770  012737  000377  001124        MOV     #377,$GDDAT
 4870
  (1)                                ;*        MOV     $GDDAT,@BBR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BBR
 4871  020006  012737  004042  001124        MOV     #4042,$GDDAT
 4872
  (1)                                ;*        MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 4873  020024  005237  001124               INC     $GDDAT
 4874
  (1)                                ;*        MOV     $GDDAT,@BSR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG BSR
 4875  020040  012737  000001  001124        MOV     #1,$GDDAT
 4876
  (1)                                ;*        MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 4877
  (1)                                ;*        MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
 4878  020066  052737  004000  001124        BIS     #BIT11,$GDDAT
 4879
  (1)                                ;*        MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 4880
  (1)                                ;*        MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 4881  020114  005737  001126               TST     $BDDAT
```

```
4882
4883  020120  001001                          BNE     1$              ;IF YES THEN BR NEXT TEST.
4884
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4885  020122  104014                          ERROR   14              ;ERROR - UNABLE TO CLOCK CLOCK A'S
4886                                                                  ;COUNT REGISTER WITH THE OVERFLOW
4887                                                                  ;FROM CLOCK B.
4888                                                                  ;THE MOST LOGICAL PLACE TO START
4889                                                                  ;WOULD BE !A CLOCK TIMING''.
4890                                                                  ;''FEED B TO A (1)'' H + ''B OVERFLOW'' H
4891                                                                  ;SHOULD BE COMING TOGETHER GOING
4892                                                                  ;INTO THE MUX - BEGING SELECTED AND
4893                                                                  ;COMING OUT OF THE MUX TO BECOME
4894                                                                  ;''CLOCK A'' L.
4895
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
4896  020124                          1$:
4897                                  .SBTTL  *
4898                                  .SBTTL  * PHASE 6 CLOCK A+B ADVANCE TESTING
4899                                  .SBTTL  *
4900
```

```
 4918                                   ;/#
 4919
 4920                                   ;;*****************************************************************
  (3)                                   ;*TEST 136      *TEST THAT THE TRAILING EDGE OF STP1 WILL INCR. COUNTER
  (4)                                   ;*
  (4)                                   ;*IN THIS TEST WE'LL SEE IF WE CATCH THE FIRST COUNT
  (4)                                   ;*AFTER STP1 COMES IN AND SETS THE ENABLE F/F ('ST1 ENB COUNTER'
  (4)                                   ;*SET).
  (4)                                   ;*WHAT WE SHOULD SEE IS THE LEADING EDGE OF ST1 COME
  (4)                                   ;*IN AND SET THE ENB F/F AND THE TRAILING EDGE TRIGGER
  (4)                                   ;*A 'CLOCK A' PULSE TO INCREMENT THE COUNTER.
  (4)                                   ;*WE KNOW FROM A PREVIOUS TEST THAT AN STP1 WILL
  (4)                                   ;*COME IN AND SET 'ENABL CNTR A' F/F AND THAT THE
  (4)                                   ;*COUNTER WILL INCREMENT, SO WHATS HAPPENING IS WE'RE ACCUALLY
  (4)                                   ;*LOOKING AT THE TRAILING EDGE OF STP1 TO SEE IF ITS DOING
  (4)                                   ;*THE INCREMENTING
  (4)                                   ;*
  (5)                                   ;*
  (5)                                   ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
  (5)                                   ;*
  (3)                                   ;;*****************************************************************
  (2) 020124  000004                    TST136: SCOPE
 4921
 4922                                                                ;CLR CLK A, SET 'ST1 ENB COUNTER'.;RATE:STP1.
 4923                                                                ;CLR COUNT + BUFFER REGS.
 4924                                                                ;GENERATE A MAINTENANCE ST1.
 4925                                                                ;THE LEADING EDGE SHOULD CAUSE
 4926                                                                ;'.ENABL CNTR A' F/F TO SET (CSR BIT 00).
 4927                                                                ;THE TRAILING EDGE SHOULD CLOCK
 4928                                                                ;THE COUNTER ONCE.
 4929 020126  012737  000001  001356          MOV     #BIT0,$TMDAT
 4930
  (1)                                   ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 4931 020144  005037  001124                  CLR     $GDDAT
 4932
  (1)                                   ;*      MOV     $GDDAT,@ABR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ABR
 4933
  (1)                                   ;*      MOV     @ASR,$GDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $GDDAT.
 4934 020170  052737  010000  001124          BIS     #BIT12,$GDDAT
 4935
  (1)                                   ;*      MOV     $GDDAT,@ASR     ;/ PUT DATA FROM $GDDAT TO DEVICE REG ASR
 4936
 4937 020206  012737  000001  001124          MOV     #1,$GDDAT       ;SET S/B FOR ERROR TYPEOUT IF NEEDED.
 4938                                                                ;READ THE COUNT REGISTER.
 4939
  (1)                                   ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 4940 020224  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;DID THE COUNT REG COUNT ONCE?
 4941 020232  001401                         BEQ     1$              ;BR IF YES TO NEXT TEST.
 4942
 4943
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 4944
 4945 020234  104012                         ERROR   12              ;ST1 FAILED TO COUNT
 4946                                                                ;CLOCK A'S COUNT REG. AFTER SETTING
 4947                                                                ;'ENABL CNTR A' - SEE TEST HEADING
```

```
 4948                                                          ;COMMENTS
 4949
              ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$
 4950
 4951  020236                          1$:
 4952
 5041
```

```
5042
 (1)
 (5)                                ;;*****************************************************************
 (4)                                ;*TEST 137      *TEST CLOCK A'S 100KHZ DIVIDER
 (5)                                ;*
 (5)                                ;+IN THIS TEST WE'LL SEE IF THE 100KHZ DIVIDER WILL DIVIDE 1MHZ
 (5)                                ;+BY 10 TO GIVE US A 100KHZ CLK L PULSE.
 (5)                                ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
 (5)                                ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100KHZ
 (5)                                ;*PULSES.
 (5)                                ;*THEN WE'LL GENERATE 9 MORE 1MHZ PULSES AND MAKE
 (5)                                ;*SURE THAT WE DON'T GET ANOTHER 100KHZ PULSE.
 (5)                                ;*
 (6)                                ;*
 (6)                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (6)                                ;*
 (4)                                ;;*****************************************************************
 (3)  020236  000004                TST137: SCOPE
 (2)  020240  012737  000020  001166        MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                                                ;/CLEAR CLOCK B.
 (1)                                                                ;/CLEAR CLOCK A.
 (1)                                                                ;/CLEAR A'S BUFFER + COUNT REGS.
 (1)                                                                ;/DISABLE THE 1MHZ OSC.
 (1)                                                                ;/ENABLE CNTR, RATE: 100KHZ ;MODE.
 (1)
 (1)  020246  005037  001356                CLR     $TMDAT
 (2)
 (2)                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
 (1)  020302  012737  004000  001356        MOV     #BIT11,$TMDAT
 (2)
 (2)                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  020330  052737  000405  001356        BIS     #401!4,$TMDAT
 (2)
 (2)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (1)
 (1)                                                                ;/SET TO GENFRATE ON 1MHZ PULSES
 (1)  020346  012700  177766                MOV     #-10.,R0
 (1)
 (1)  020352                        1$:                             ;/GENFRATE 1 1MHZ PULSE
 (1)                                                                ;/HAS COUNTER ADVANCED ANY?
 (2)
 (2)                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  020362  052737  004000  001356        BIS     #BIT11,$TMDAT
 (2)
 (2)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 (1)  020410  005737  001126                TST     $BDDAT
 (1)  020414  001002                        BNE     10$             ;/IF SO EXIT THIS LOOP.
```

```
(1)                                                     ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                                     ;/OSC.,THE DIVIDER COULD HAVE
(1)                                                     ;/AND COUNT LEFT IN IT.
(1)                                                     ;/AFTER THIS LOOP ,WE SHOULD BE SUNK.
(1)  020416  005200                     INC     R0      ;/DONE 10. 1MHZ PULSES?
(1)  020420  001354                     BNE     1$      ;/IF NOT - DO ANOTHER.
(1)
(1)  020422  012737  000001  001124  10$: MOV    #1,$GDDAT       ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                                     ;/READ THE COUNTER.
(2)
(2)                                 ;*  MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)
(1)  020440  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  020446  001402                     BEQ     2$              ;/IF YES - NEXT CHECK.
(2)
          ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020450  104012                     ERROR   12              ;/ERROR - CLOCK A - 100KHZ - PULSE
(1)                                                     ;/NOT GENERATED WHEN 10 1MHZ PULSES
(2)
          ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020452  000430                     BR      4$
(1)  020454  012700  000011         2$: MOV     #9.,R0  ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  020460                         3$:                 ;/GENERATE 9 1MHZ PULSES
(2)
(2)                                 ;*  MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  020470  052737  004000  001356      BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  020506  005300                     DEC     R0              ;/INORDER TO CHECK TO SEE THAT
(1)  020510  001363                     BNE     3$              ;/WE DON'T GENERATE ANOTHER 100KHZ PULSE.
(1)
(1)                                                     ;/READ THE COUNTER
(2)
(2)                                 ;*  MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)  020522  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100KHZ PULSE GENERATED?
(1)  020530  001401                     BEQ     4$              ;/NO-GO TO NEXT TEST!
(1)
(2)
          ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020532  104012                     ERROR   12              ;/ERROR CLOCK A WE SEEM TO HAVE
(1)                                                     ;/GENERATED A SECOND 100KHZ PULSE
(1)                                                     ;/ON ONLY 9 1MHZ PULSES.
(2)
          ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020534                         4$:
```

```
5043
 (1)
 (5)                                    ;:*****************************************************************
 (4)                                    ;*TEST 140      *TEST CLOCK A'S 10KHZ DIVIDER
 (5)                                    ;*
 (5)                                    ;+IN THIS TEST WE'LL SEE IF THE 10KHZ DIVIDER WILL DIVIDE 100KHZ
 (5)                                    ;+BY 10 TO GIVE US A 10KHZ CLK L PULSE.
 (5)                                    ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
 (5)                                    ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 10KHZ
 (5)                                    ;*PULSES.
 (5)                                    ;*THEN WE'LL GENERATE 9 MORE 100KHZ PULSES AND MAKE
 (5)                                    ;*SURE THAT WE DON'T GET ANOTHER 10KHZ PULSE.
 (5)                                    ;*
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (6)                                    ;*
 (4)                                    ;:*****************************************************************
 (3)  020534  000004                    TST140: SCOPE
 (2)  020536  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                                                    ;/CLEAR CLOCK B.
 (1)                                                                    ;/CLEAR CLOCK A.
 (1)                                                                    ;/CLEAR A'S BUFFER + COUNT REGS.
 (1)                                                                    ;/DISABLE THE 1MHZ OSC.
 (1)                                                                    ;/ENABLE CNTR, RATE: 10KHZ ;MODE.
 (1)
 (1)  020544  005037  001356            ;*      CLR     $TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
 (1)  020600  012737  004000  001356            MOV     #BIT11,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  020626  052737  000407  001356            BIS     #401!6,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (1)
 (1)                                                                    ;/SET TO GENERATE ON 1MHZ PULSES
 (1)  020644  012700  177634                    MOV     #-100.,R0
 (1)
 (1)  020650                            1$:                             ;/GENERATE 1 1MHZ PULSE
 (1)                                                                    ;/HAS COUNTER ADVANCED ANY?
 (2)
 (2)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  020660  052737  004000  001356            BIS     #BIT11,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 (1)  020706  005737  001126                    TST     $BDDAT
 (1)  020712  001002                            BNE     10$             ;/IF SO EXIT THIS LOOP.
```

```
(1)                                                     ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                                     ;/OSC.,THE DIVIDER COULD HAVE
(1)                                                     ;/AND COUNT LEFT IN IT.
(1)                                                     ;/AFTER THIS LOOP ,WE SHOULD BE SUNK.
(1)  020714  005200                     INC     R0      ;/DONE 100. 1MHZ PULSES?
(1)  020716  001354                     BNE     1$      ;/IF NOT - DO ANOTHER.
(1)
(1)  020720  012737  000001  001124  10$: MOV    #1,$GDDAT       ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                                     ;/READ THE COUNTER.
(2)
(2)                                 ;*  MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)
(1)  020736  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  020744  001402                     BEQ     2$              ;/IF YES - NEXT CHECK.
(2)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020746  104012                     ERROR   12              ;/ERROR - CLOCK A - 10KHZ - PULSE
(1)                                                     ;/NOT GENERATED WHEN 10 100KHZ PULSES
(2)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  020750  000430                     BR      4$
(1)  020752  012700  000143          2$: MOV    #99.,R0 ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  020756                          3$:                        ·/GENERATE 9 100KHZ PULSES
(2)
(2)                                 ;*  MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  020766  052737  004000  001356      BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  021004  005300                     DEC     R0              ;/INORDER TO CHECK TO SEE THAT
(1)  021006  001363                     BNE     3$              ;/WE DON'T GENERATE ANOTHER 10KHZ PULSE.
(1)
(1)                                                     ;/READ THE COUNTER
(2)
(2)                                 ;*  MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)  021020  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 10KHZ PULSE GENERATED?
(1)  021026  001401                     BEQ     4$              ;/NO-GO TO NEXT TEST.
(1)
(2)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021030  104012                     ERROR   12              ;/ERROR CLOCK A WE SEEM TO HAVE
(1)                                                     ;/GENERATED A SECOND 10KHZ PULSE
(1)                                                     ;/ON ONLY 9 100KHZ PULSES.
(2)
        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021032                          4$:
```

```
5044                                    ;/#
 (1)
 (5)                                    ;;*****************************************************************
 (4)                                    ;*TEST 141      *TEST CLOCK A'S 1KHZ DIVIDER
 (5)                                    ;*
 (5)                                    ;+IN THIS TEST WE'LL SEE IF THE 1KHZ DIVIDER WILL DIVIDE 10KHZ
 (5)                                    ;+BY 10 TO GIVE US A 1KHZ CLK L PULSE.
 (5)                                    ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
 (5)                                    ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 1KHZ
 (5)                                    ;*PULSES.
 (5)                                    ;*THEN WE'LL GENERATE 9 MORE 10KHZ PULSES AND MAKE
 (5)                                    ;*SURE THAT WE DON'T GET ANOTHER 1KHZ PULSE.
 (5)                                    ;*
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (6)                                    ;*
 (4)                                    ;;*****************************************************************
 (3)  021032  000004                    TST141: SCOPE
 (2)  021034  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                                                    ;/CLEAR CLOCK B.
 (1)                                                                    ;/CLEAR CLOCK A.
 (1)                                                                    ;/CLEAR A'S BUFFER + COUNT REGS.
 (1)                                                                    ;/DISABLE THE 1MHZ OSC.
 (1)                                                                    ;/ENABLE CNTR, RATE: 1KHZ ;MODE.
 (1)
 (1)  021042  005037  001356            CLR     $TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
 (1)  021076  012737  004000  001356            MOV     #BIT11,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  021124  052737  000411  001356            BIS     #401!10,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (1)
 (1)                                                                    ;/SET TO GENERATE ON 1MHZ PULSES
 (1)  021142  012700  176030            MOV     #-1000.,R0
 (1)
 (1)  021146                            1$:                             ;/GENERATE 1 1MHZ PULSE
 (1)                                                                    ;/HAS COUNTER ADVANCED ANY?
 (2)
 (2)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  021156  052737  004000  001356            BIS     #BIT11,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 (1)  021204  005737  001126            TST     $BDDAT
 (1)  021210  001002                    BNE     10$             ;/IF SO EXIT THIS LOOP.
```

```
(1)                                                         ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                                         ;/OSC.,THE DIVIDER COULD HAVE
(1)                                                         ;/AND COUNT LEFT IN IT.
(1)                                                         ;/AFTER THIS LOOP ,WE SHOULD BE SUNK.
(1)  021212  005200                      INC     R0         ;/DONE 1000. 1MHZ PULSES?
(1)  021214  001354                      BNE     1$         ;/IF NOT - DO ANOTHER.
(1)
(1)  021216  012737  000001  001124  10$: MOV     #1,$GDDAT  ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                                         ;/READ THE COUNTER.
(2)
(2)                                ;*    MOV     @ACR,$BDDAT ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)
(1)  021234  023737  001126  001124       CMP     $BDDAT,$GDDAT ;/DID THE COUNTER ADVANCE ONCE?
(1)  021242  001402                      BEQ     2$         ;/IF YES - NEXT CHECK.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021244  104012                      ERROR   12         ;/ERROR - CLOCK A - 1KHZ - PULSE
(1)                                                         ;/NOT GENERATED WHEN 10 10KHZ PULSES
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021246  000430                      BR      4$
(1)  021250  012700  001747          2$: MOV     #999.,R0   ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  021254                          3$:                    ;/GENERATE 9 10KHZ PULSES
(2)
(2)                                ;*    MOV     @ASR,$TMDAT ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(2)
(2)                                ;*    MOV     @ASR,$TMDAT ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021264  052737  004000  001356       BIS     #BIT11,$TMDAT
(2)
(2)                                ;*    MOV     $TMDAT,@ASR ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  021302  005300                      DEC     R0         ;/INORDER TO CHECK TO SEE THAT
(1)  021304  001363                      BNE     3$         ;/WE DON'T GENERATE ANOTHER 1KHZ PULSE.
(1)
(1)                                                         ;/READ THE COUNTER
(2)
(2)                                ;*    MOV     @ACR,$BDDAT ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)  021316  023737  001126  001124       CMP     $BDDAT,$GDDAT ;/WAS ANOTHER 1KHZ PULSE GENERATED?
(1)  021324  001401                      BEQ     4$         ;/NO-GO TO NEXT TEST
(1)
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021326  104012                      ERROR   12         ;/ERROR CLOCK A WE SEEM TO HAVE
(1)                                                         ;/GENERATED A SECOND 1KHZ PULSE
(1)                                                         ;/ON ONLY 9 10KHZ PULSES.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021330                          4$:
```

```
5045                                    ;/*
 (1)
 (5)                                    ;;*****************************************************************
 (4)                                    ;*TEST 142      *TEST CLOCK A'S 100HZ DIVIDER
 (5)                                    ;*
 (5)                                    ;+IN THIS TEST WE'LL SEE IF THE 100HZ DIVIDER WILL DIVIDE 1KHZ
 (5)                                    ;+BY 10 TO GIVE US A 100HZ CLK L PULSE.
 (5)                                    ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
 (5)                                    ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100HZ
 (5)                                    ;*PULSES.
 (5)                                    ;*THEN WE'LL GENERATE 9 MORE 1KHZ PULSES AND MAKE
 (5)                                    ;*SURE THAT WE DON'T GET ANOTHER 100HZ PULSE.
 (5)                                    ;*
 (6)                                    ;*
 (6)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
 (6)                                    ;*
 (4)                                    ;;*****************************************************************
 (3)  021330  000004                    TST142: SCOPE
 (2)  021332  012737  000020  001166            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 (1)
 (1)                                            ;/CLEAR CLOCK B.
 (1)                                            ;/CLEAR CLOCK A.
 (1)                                            ;/CLEAR A'S BUFFER + COUNT REGS.
 (1)                                            ;/DISABLE THE 1MHZ OSC.
 (1)                                            ;/ENABLE CNTR, RATE: 100HZ ;MODE.
 (1)
 (1)  021340  005037  001356            CLR     $TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
 (1)  021374  012737  004000  001356            MOV     #BIT11,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (2)
 (2)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  021422  052737  000413  001356            BIS     #401!12,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (1)
 (1)                                            ;/SET TO GENERATE ON 1MHZ PULSES
 (1)  021440  012700  154360            MOV     #-10000.,R0
 (1)
 (1)  021444                            1$:                             ;/GENERATE 1 1MHZ PULSE
 (1)                                            ;/HAS COUNTER ADVANCED ANY?
 (2)
 (2)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
 (1)  021454  052737  004000  001356            BIS     #BIT11,$TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
 (1)  021502  005737  001126            TST     $BDDAT
 (1)  021506  001002                    BNE     10$             ;/IF SO EXIT THIS LOOP.
```

```
(1)                                                     ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                                     ;/OSC.,THE DIVIDER COULD HAVE
(1)                                                     ;/AND COUNT LEFT IN IT.
(1)                                                     ;/AFTER THIS LOOP ,WE SHOULD BE SUNK.
(1)  021510  005200                     INC     R0      ;/DONE 10000. 1MHZ PULSES?
(1)  021512  001354                     BNE     1$      ;/IF NOT - DO ANOTHER.
(1)
(1)  021514  012737  000001  001124  10$: MOV    #1,$GDDAT       ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                                     ;/READ THE COUNTER.
(2)
(2)                                 ;*  MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)
(1)  021532  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  021540  001402                     BEQ     2$              ;/IF YES - NEXT CHECK.
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021542  104012                     ERROR   ^2              ;/ERROR - CLOCK A - 100HZ - PULSE
(1)                                                     ;/NOT GENERATED WHEN 10 1KHZ PULSES
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021544  000430                     BR      4$
(1)  021546  012700  023417          2$: MOV    #9999.,R0       ;/GET THE NUMBER OF '1 MHZ' H PULSES
(1)
(1)  021552                          3$:                        ;/GENERATE 9 1KHZ PULSES
(2)
(2)                                 ;*  MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021562  052737  004000  001356      BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  021600  005300                     DEC     R0              ;/INORDER TO CHECK TO SEE THAT
(1)  021602  001363                     BNE     3$              ;/WE DON'T GENERATE ANOTHER 100HZ PULSE.
(1)
(1)                                                     ;/READ THE COUNTER
(2)
(2)                                 ;*  MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
(1)  021614  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100HZ PULSE GENERATED?
(1)  021622  001401                     BEQ     4$              ;/NO-GO TO NEXT TEST!
(1)
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021624  104012                     ERROR   12              ;/ERROR CLOCK A WE SEEM TO HAVE
(1)                                                     ;/GENERATED A SECOND 100HZ PULSE
(1)                                                     ;/ON ONLY 9 1KHZ PULSES.
(2)
            ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  021626                          4$:
5046
5145
5146
(5)                                 ;;*******************************************************************
(4)                                 ;*TEST 143       *TEST CLOCK B'S 100KHZ DIVIDER
```

```
(5)                                     ;*
(5)                                     ;*IN THIS TEST WE'LL SEE IF THE 100HZ DIVIDER WILL DIVIDE 1KHZ
(5)                                     ;*BY 10 TO GIVE US A 100HZ CLK L PULSE.
(5)                                     ;*TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
(5)                                     ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100HZ
(5)                                     ;*PULSES.
(5)                                     ;*THEN WE'LL GENERATE 9 MORE 1KHZ PULSES AND MAKE
(5)                                     ;*SURE THAT WE DON'T GET ANOTHER 100HZ PULSE.
(5)                                     ;*
(6)                                     ;*
(6)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
(6)                                     ;*
(4)                                     ;;*****************************************************************
(3)  021626  000004                     TST143: SCOPE
(2)  021630  012737  000020  001166             MOV     #20,$TIMES      ;;DO 20 ITERATIONS
(1)
(1)                                                                     ;/CLEAR CLOCK B.
(1)                                                                     ;/CLEAR CLOCK A.
(1)
(1)                                                                     ;/CLEAR B'S BUFFER + COUNT REGS.
(1)                                                                     ;/DISABLE THE 1MHZ OSC.
(1)                                                                     ;/RATE: 100KHZ
(1)                                                                     ;/ENABLE CLOCK B.
(1)  021636  005037  001356             CLR     $TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@#BSR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                                     ;*      MOV     $TMDAT,@#ASR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                                     ;*      MOV     $TMDAT,@#BBR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG BBR
(2)
(2)                                     ;*      MOV     $TMDAT,@#BBR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG BBR
(1)  021672  012737  004040  001356             MOV     #BIT11!BIT5,$TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@#BSR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                                     ;*      MOV     @#BSR,$TMDAT    ;/READ DEVICE REG BSR,PUT DATA IN $TMDAT.
(1)  021720  052737  000004  001356             BIS     #4,$TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@#BSR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  021736  005237  001356             INC     $TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@#BSR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  021752  012700  177766             MOV     #-10.,R0        ;/SET TO GENERATE 10. 1MHZ PULSES
(1)
(1)  021756                             1$:                             ;/GENERATE 1 1MHZ PULSE
(1)                                                                     ;/HAS THE COUNTER ADVANCED?
(2)
(2)                                     ;*      MOV     @#ASR,$TMDAT    ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  021766  052737  004000  001356             BIS     #BIT11,$TMDAT
(2)
(2)                                     ;*      MOV     $TMDAT,@#ASR    ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                                     ;*      MOV     @#BCR,$BDDAT    ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022014  005737  001126             TST     $BDDAT
(1)  022020  001002                     BNE     10$             ;/EXIT LOOP IF SO.
(1)                                                             ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                                             ;/      OSC., THE DIVIDER COULD HAVE
```

```
(1)                                                     ;/      HAD ANY COUNT IN IT.
(1)                                                     ;/AFTER THIS LOOP, WE SHOULD BE SUNK.
(1)
(1)  022022  005200                     INC     R0      ;/DONE 10. 1MHZ PULSES?
(1)  022024  001354                     BNE     1$      ;/IF NOT - DO ANOTHER.
(1)
(1)  022026  012737  000001  001124  10$: MOV    #1,$GDDAT       ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                                     ;/READ THE COUNTER
(2)
(2)                                 ;*  MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)
(1)  022044  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  022052  001402                     BEQ     2$      ;/IF YES - NEXT CHECK
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022054  104015                     ERROR   15      ;/ERROR - CLOCK B - 100KHZ - PULSE
(1)                                                     ;/NOT GENERATED WHEN 10 1MHZ PULSE
(1)                                                     ;/WERE GENERATED.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022056  000430                     BR      4$
(1)
(1)  022060  012700  177767         2$: MOV    #-9.,R0 ;/GET THE NUMBER OF "1 MHZ" H PULSES
(1)                                                     ;/NEED TO GIVE 9 1MHZ PULSES
(1)  022064                          3$:
(2)
(2)                                 ;*  MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  022074  052737  004000  001356      BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  022112  005200                     INC     R0      ;/IN ORDER TO CHECK TO SEE THAT
(1)  022114  001363                     BNE     3$      ;/WE DON'T GENERATE ANOTHER 100KHZ PULSE
(1)
(2)
(2)                                 ;*  MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022126  023737  001126  001124      CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100KHZ PULSE GENERATED?
(1)  022134  001401                     BEQ     4$      ;/NO - GO TO NEXT CHECK.
(1)
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022136  104015                     ERROR   15      ;/ERROR CLOCK B WE SEEM TO HAVE
(1)                                                     ;/GENERATED A SECOND 100KHZ PULSE
(1)                                                     ;/ON ONLY 9 1MHZ PULSES.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022140                          4$:
5147
(5)                                 ;;**************************************************************
(4)                                 ;*TEST 144       *TEST CLOCK B'S 10KHZ DIVIDER
(5)                                 ;*
(5)                                 ;*IN THIS TEST WE'LL SEE IF THE 100HZ DIVIDER WILL DIVIDE 1KHZ
```

```
(5)                                     ;+BY 10 TO GIVE US A 100HZ CLK L PULSE.
(5)                                     ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
(5)                                     ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100HZ
(5)                                     ;*PULSES.
(5)                                     ;*THEN WE'LL GENERATE 9 MORE 1KHZ PULSES AND MAKE
(5)                                     ;*SURE THAT WE DON'T GET ANOTHER 100HZ PULSE.
(5)                                     ;*
(6)                                     ;*
(6)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
(6)                                     ;*
(4)                                     ;;*****************************************************************
(3)  022140  000004                     TST144: SCOPE
(2)  022142  012737  000020  001166             MOV     #20,$TIMES      ;;DO 20 ITERATIONS
(1)
(1)                                             ;/CLEAR CLOCK B.
(1)                                             ;/CLEAR CLOCK A.
(1)
(1)                                             ;/CLEAR B'S BUFFER + COUNT REGS.
(1)                                             ;/DISABLE THE 1MHZ OSC.
(1)                                             ;/RATE: 10KHZ
(1)                                             ;/ENABLE CLOCK B.
(1)  022150  005037  001356             CLR     $TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                             ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                             ;*      MOV     $TMDAT,@BBR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BBR
(1)  022204  012737  004040  001356       MOV     #BIT11.BIT5,$TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                             ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                             ;*      MOV     @BSR,$TMDAT     ;/READ DEVICE REG BSR,PUT DATA IN $TMDAT.
(1)  022232  052737  000006  001356       BIS     #6,$TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  022250  005237  001356             INC     $TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  022264  012700  177634             MOV     #-100.,R0       ;/SET TO GENERATE 100. 1MHZ PULSES
(1)
(1)  022270                     1$:                             ;/GENERATE 1 1MHZ PULSE
(1)                                             ;/HAS THE COUNTER ADVANCED?
(2)
(2)                             ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  022300  052737  004000  001356       BIS     #BIT11,$TMDAT
(2)
(2)                             ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                             ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022326  005737  001126             TST     $BDDAT
(1)  022332  001002                     BNE     10$             ;/EXIT LOOP IF SO.
(1)                                             ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                             ;/      OSC., THE DIVIDER COULD HAVE
(1)                                             ;/      HAD ANY COUNT IN IT.
(1)                                             ;/AFTER THIS LOOP, WE SHOULD BE SUNK.
```

```
(1)
(1)  022334  005200                          INC     R0              ;/DONE 100. 1MHZ PULSES?
(1)  022336  001354                          BNE     1$              ;/IF NOT - DO ANOTHER.
(1)
(1)  022340  012737  000001  001124  10$:    MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                                                  ;/READ THE COUNTER
(2)
(2)                                 ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)
(1)  022356  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  022364  001402                          BEQ     2$              ;/IF YES - NEXT CHECK
(2)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022366  104015                          ERROR   15              ;/ERROR - CLOCK B - 10KHZ - PULSE
(1)                                                                  ;/NOT GENERATED WHEN 10 100KHZ PULSE
(1)                                                                  ;/WERE GENERATED.
(2)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022370  000430                          BR      4$
(1)
(1)  022372  012700  177635          2$:     MOV     #-99.,R0        ;/GET THE NUMBER OF ''1 MHZ'' H PULSES
(1)                                                                  ;/NEED TO GIVE 9 100KHZ PULSES
(1)  022376                          3$:
(2)
(2)                                 ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  022406  052737  004000  001356          BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  022424  005200                          INC     R0              ;/IN ORDER TO CHECK TO SEE THAT
(1)  022426  001363                          BNE     3$              ;/WE DON'T GENERATE ANOTHER 10KHZ PULSE
(1)
(2)
(2)                                 ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022440  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 10KHZ PULSE GENERATED?
(1)  022446  001401                          BEQ     4$              ;/NO - GO TO NEXT CHECK.
(1)
(2)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022450  104015                          ERROR   15              ;/ERROR CLOCK B WE SEEM TO HAVE
(1)                                                                  ;/GENERATED A SECOND 10KHZ PULSE
(1)                                                                  ;/ON ONLY 9 100KHZ PULSES.
(2)
             :::$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022452                          4$:
5'48
(5)                                 ;;*********************************************************************
(4)                                 ;*TEST 145       *TEST CLOCK B'S 1KHZ DIVIDER
(5)                                 ;*
(5)                                 ;+IN THIS TEST WE'LL SEE IF THE 100HZ DIVIDER WILL DIVIDE 1KHZ
(5)                                 ;+BY 10 TO GIVE US A 100HZ CLK L PULSE.
(5)                                 ;+TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
```

```
(5)                                     ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100HZ
(5)                                     ;*PULSES.
(5)                                     ;*THEN WE'LL GENERATE 9 MORE 1KHZ PULSES AND MAKE
(5)                                     ;*SURE THAT WE DON'T GET ANOTHER 100HZ PULSE.
(5)                                     ;*
(6)                                     ;*
(6)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A''
(6)                                     ;*
(4)                                     ;;*****************************************************************
(3)  022452  000004                     TST145: SCOPE
(2)  022454  012737  000020  001166             MOV     #20,$TIMES      ;;DO 20 ITERATIONS
(1)
(1)                                                                     ;/CLEAR CLOCK B.
(1)                                                                     ;/CLEAR CLOCK A.
(1)
(1)                                                                     ;/CLEAR B'S BUFFER + COUNT REGS.
(1)                                                                     ;/DISABLE THE 1MHZ OSC.
(1)                                                                     ;/RATE: 1KHZ
(1)                                                                     ;/ENABLE CLOCK B.
(1)  022462  005037  001356             CLR     $TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                                 ;*  MOV     $TMDAT,@BBR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BBR
(1)  02251C  012737  004040  001356      MOV     #BIT11!BIT5,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                                 ;*  MOV     @BSR,$TMDAT     ;/READ DEVICE REG BSR,PUT DATA IN $TMDAT.
(2)
(2)                                 ;*  MOV     @BSR,$TMDAT     ;/READ DEVICE REG BSR,PUT DATA IN $TMDAT.
(1)  022544  052737  000010  001356      BIS     #10,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  022562  005237  001356             INC     $TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  022576  012700  176030             MOV     #-1000.,R0      ;/SET TO GENERATE 1000. 1MHZ PULSES
(1)
(1)  022602                         1$:                         ;/GENERATE 1 1MHZ PULSE
(1)                                                             ;/HAS THE COUNTER ADVANCED?
(2)
(2)                                 ;*  MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  022612  052737  004000  001356      BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                                 ;*  MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022640  005737  001126             TST     $BDDAT
(1)  022644  001002                     BNE     10$             ;/EXIT LOOP IF SO.
(1)                                                             ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                                             ;/      OSC., THE DIVIDER COULD HAVE
(1)                                                             ;/      HAD ANY COUNT IN IT.
(1)                                                             ;/AFTER THIS LOOP, WE SHOULD BE SUNK.
(1)
(1)  022646  005200                     INC     R0              ;/DONE 1000. 1MHZ PULSES?
```

```
(1)  022650  001354                          BNE     1$              ;/IF NOT - DO ANOTHER.
(1)
(1)  022652  012737  000001  001124  10$:    MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                                                  ;/READ THE COUNTER
(2)
(2)                                 ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)
(1)  022670  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  022676  001402                          BEQ     2$              ;/IF YES - NEXT CHECK
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022700  104015                          ERROR   15              ;/ERROR - CLOCK B - 1KHZ - PULSE
(1)                                                                  ;/NOT GENERATED WHEN 10 10KHZ PULSE
(1)                                                                  ;/WERE GENERATED.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022702  000430                          BR      4$
(1)
(1)  022704  012700  176031          2$:     MOV     #-999.,R0       ;/GET THE NUMBER OF ''1 MHZ'' H PULSES
(1)                                                                  ;/NEED TO GIVE 9 10KHZ PULSES
(1)  022710                          3$:
(2)
(2)                                 ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  022720  052737  004000  001356          BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  022736  005200                          INC     R0              ;/IN ORDER TO CHECK TO SEE THAT
(1)  022740  001363                          BNE     3$              ;/WE DON'T GENERATE ANOTHER 1KHZ PULSE
(1)
(2)
(2)                                 ;*      MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  022752  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 1KHZ PULSE GENERATED?
(1)  022760  001401                          BEQ     4$              ;/NO - GO TO NEXT CHECK.
(1)
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022762  104015                          ERROR   15              ;/ERROR CLOCK B WE SEEM TO HAVE
(1)                                                                  ;/GENERATED A SECOND 1KHZ PULSE
(1)                                                                  ;/ON ONLY 9 10KHZ PULSES.
(2)
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  022764                          4$:
5149
(5)                                  ;;*********************************************************************
(4)                                  ;*TEST 146       *TEST CLOCK B'S 100HZ DIVIDER
(5)                                  ;*
(5)                                  ;*IN THIS TEST WE'LL SEE IF THE 100HZ DIVIDER WILL DIVIDE 1KHZ
(5)                                  ;*BY 10 TO GIVE US A 100HZ CLK L PULSE.
(5)                                  ;*TO DO THIS, WE'LL DISABLE THE REGULAR 1MHZ CLK PULSE AND
(5)                                  ;*PULSES THAT IN TURN SHOULD DIVIDE BY TEN TO GIVE US ONE 100HZ
(5)                                  ;*PULSES.
```

```
(5)                                     ;*THEN WE'LL GENERATE 9 MORE 1KHZ PULSES AND MAKE
(5)                                     ;*SURE THAT WE DON'T GET ANOTHER 100HZ PULSE.
(5)                                     ;*
(6)                                     ;*
(6)                                     ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF A'
(6)                                     ;*
(4)                                     ;;*****************************************************************
(3)  022764  000004                     TST146: SCOPE
(2)  022766  012737  000020  001166             MOV     #20,$TIMES      ;;DO 20 ITERATIONS
(1)
(1)                                                                     ;/CLEAR CLOCK B.
(1)                                                                     ;/CLEAR CLOCK A.
(1)
(1)                                                                     ;/CLEAR B'S BUFFER + COUNT REGS.
(1)                                                                     ;/DISABLE THE 1MHZ OSC.
(1)                                                                     ;/RATE: 100HZ
(1)                                                                     ;/ENABLE CLOCK B.
(1)  022774  005037  001356             CLR     $TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                                 ;*  MOV     $TMDAT,@BBR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BBR
(1)  023030  012737  004040  001356      MOV     #BIT11.BIT5,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(2)
(2)                                 ;*  MOV     @BSR,$TMDAT     ;/READ DEVICE REG BSR,PUT DATA IN $TMDAT.
(1)  023056  052737  000012  001356      BIS     #12,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  023074  005237  001356             INC     $TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
(1)  023110  012700  154360             MOV     #-10000.,R0     ;/SET TO GENERATE 10000. 1MHZ PULSES
(1)
(1)  023114                         1$:                         ;/GENERATE 1 1MHZ PULSE
(1)                                                             ;/HAS THE COUNTER ADVANCED?
(2)
(2)                                 ;*  MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  023124  052737  004000  001356      BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*  MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(2)
(2)                                 ;*  MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)  023152  005737  001126             TST     $BDDAT
(1)  023156  001002                     BNE     10$             ;/EXIT LOOP IF SO.
(1)                                                             ;/NOTE: WHEN WE DISABLED THE 1 MHZ.
(1)                                                             ;/      OSC., THE DIVIDER COULD HAVE
(1)                                                             ;/      HAD ANY COUNT IN IT.
(1)                                                             ;/AFTER THIS LOOP, WE SHOULD BE SUNK.
(1)
(1)  023160  005200                     INC     R0              ;/DONE 10000. 1MHZ PULSES?
(1)  023162  001354                     BNE     1$              ;/IF NOT - DO ANOTHER.
(1)
```

```
(1)  023164  012737  000001  001124  10$:    MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT IF NEEDED.
(1)
(1)                                           ;/READ THE COUNTER
(2)
(2)                                 ;*        MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(1)
(1)  023202  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/DID THE COUNTER ADVANCE ONCE?
(1)  023210  001402                         BEQ     2$              ;/IF YES - NEXT CHECK
(2)
           ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  023212  104015                         ERROR   15              ;/ERROR - CLOCK B - 100HZ - PULSE
(1)                                                                 ;/NOT GENERATED WHEN 10 1KHZ PULSE
(1)                                                                 ;/WERE GENERATED.
(2)
           :.:$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  023214  000430                         BR      4$
(1)
(1)  023216  012700  154361          2$:    MOV     #-9999.,R0      ;/GET THE NUMBER OF ''1 MHZ'' H PULSES
(1)                                                                 ;/NEED TO GIVE 9 1KHZ PULSES
(1)  023222                          3$:
(2)
(2)                                 ;*        MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
(1)  023232  052737  004000  001356          BIS     #BIT11,$TMDAT
(2)
(2)                                 ;*        MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
(1)  023250  005200                         INC     R0              ;/IN ORDER TO CHECK TO SEE THAT
(1)  023252  001363                         BNE     3$              ;/WE DON'T GENERATE ANOTHER 100HZ PULSE
(1)
(2)
(2)                                 ;*        MOV     @BCR,$BDDAT     ;/READ DEVICE REG BCR,PUT DATA IN $BDDAT.
(2)
(1)  023264  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;/WAS ANOTHER 100HZ PULSE GENERATED?
(1)  023272  001401                         BEQ     4$              ;/NO - GO TO NEXT CHECK.
(1)
(2)
           :.:$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  023274  104015                         ERROR   15              ;/ERROR CLOCK B WE SEEM TO HAVE
(1)                                                                 ;/GENERATED A SECOND 100HZ PULSE
(1)                                                                 ;/ON ONLY 9 1KHZ PULSES.
(2)
           .:;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  023276                          4$:
5150
5176
5177                                          .SBTTL
5178
5179                                .SBTTL  END OF PASS ROUTINE
(1)
(2)                                 ;;**************************************************************************
(1)                                 ;*INCREMENT THE PASS NUMBER ($PASS)
(1)                                 ;*TYPE 'END PASS''
(1)                                 ;*IF THERES A MONITOR GO TO IT
(1)                                 ;*IF THERE ISN'T JUMP TO LOOP
```

```
  (1)                                           ;*IF IT IS DESIRED TO HAVE A BELL INDICATE THE 'END OF PASS' LOCATION
  (1)                                           ;*$ENDMG CAN BE CHANGED TO 7.
  (1)
  (1)  023276                                   $EOP:
  (2)  023276  000240                                   NOP
  (1)  023300  005037  001102                           CLR     $TSTNM          ;;ZERO THE TEST NUMBER
  (1)  023304  005037  001166                           CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (1)  023310  005237  001210                           INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (1)  023314  042737  100000  0012'0                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (1)  023322  005327                                   DEC     (PC)+           ;;LOOP?
  (1)  023324  000001                           $EOPCT: .WORD   1
  (1)  023326  003017                                   BGT     $DOAGN          ;;YES
  (1)  023330  012737                                   MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (1)  023332  000001                           $ENDCT: .WORD   1
  (1)  023334  023324                                   $EOPCT
  (1)  023336  104401  023375                           TYPE    ,$ENDMG         ;;TYPE 'END PASS'
  (1)  023342  104401  023372                           TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
  (1)  023346  013700  000042                   $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
  (1)  023352  001405                                   BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  (1)  023354  000005                                   RESET                   ;;CLEAR THE WORLD
  (1)  023356  004710                           $FNDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  (1)  023360  000240                                   NOP                     ;;SAVE ROOM
  (1)  023362  000240                                   NOP                     ;;FOR
  (1)  023364  000240                                   NOP                     ;;ACT11
  (1)  023366                                   $DOAGN:
  (1)  023366  000137                                   JMP     @(PC)+          ;;RETURN
  (1)  023370  002310                           $RTNAD: .WORD   LOOP
  (1)  023372     377     377     000           $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
  (1)  023375     015  042412  042116           $ENDMG: .ASCIZ  <15><12>/END PASS/
  (1)  023402  050040  051501  000123
5180                                                    .SBTTL  *
5181                                                    .SBTTL  * SPECIAL I/O SIGNAL TESTS
5182                                                    .SBTTL  *
5183
5199
5217
5218                                            .SBTTL  *       "STP2 OUT" TO "SCHMITT TRIG 1 IN" TESTS
5219                                                    ;*
5220                                                    ;* THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
5221                                                    ;* PROVIDING SCOPE LOOP CAPABILITIES FOR "STP2 OUT" L
5222                                                    ;* AND "SCHMITT TRIG 1" IN.
5223                                                    ;*
5224                                                    ;* WHEN YOU LOAD AND START AT LOCATION 210, PROGRAM
5225                                                    ;* CONTROL IS TRANSFERRED HERE. "STP2 OUT" L PULSES ARE
5226                                                    ;* GENERATED BY "LO STAT A HI" H + "BD10" H (MAIN. STP2).
5227                                                    ;* PIN V ("STP2 OUT") IS WIRED TO PIN LL (SCHMITT TRIG1) FOR
5228                                                    ;* THIS TEST. "STP2 OUT" PULSES ARE RECEIVED AS "SCHMITT TRIG 1"
5229                                                    ;* PULSES WHICH SET CLOCK A'S STATUS REGISTER BIT 15.
5230                                                    ;* IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING TECHNIC.
5231                                                    ;* AND ERROR SWITCH REGISTER OPTIONS ARE USED.
5232                                                    ;* AN "*" IS TYPED AFTER EACH 65,324 LOOPS THROUGH THE
5233                                                    ;* TEST. SW13=1 WILL INHIBIT THIS FEATURE.
5234                                            ;*
  (1)                                           ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT B"
  (1)                                           ;*
5235                                                    ;*
```

```
                                        ;* YOU MUST WIRE PINS V AND LL OF J1 TOGETHER AND INSTALL JUMPER W3
                                        ;*
                                        ;* LOGIC TEST (L * S 200) SHOULD BE RUN FIRST.
                                        ;*

023410  005037  001666        LS210:  CLR     .DVLS

023414  005037  001104        1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
023420  012737  177704  001210        MOV     #-60.,$PASS     ;/SET PASS COUNT.
                                                              ;/NOTE: PASS COUNT USED ONLY
                                                              ;/      TO DETECT 60 PASSES SO
                                                              ;/      IT CAN GENERATE A CRLF.
                                                              ;/      AFTER CRLF IT WILL BE ZEROED.
                                                              ;/CLEAR CLOCK A.
023426                          2$:                           ;/CLEAR CLOCK B.
023426  005037  001356                CLR     $TMDAT

                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR

                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR

                                                              ;GENERATE AN "STP2 OUT" PULSE
                                                              ;AT THIS POINT YOU SHOULD
                                                              ;SEE AN OUTPUT AT PIN V.
                                                              ;PIN V SHOULD BE WIRED TO
                                                              ;PIN LL FOR "SCHMITT TRIG 1" IN.

                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR

                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
023472  052737  002000  001356        BIS     #BIT10,$TMDAT   ;GENERATE STP2

                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
023510  000240                      NOP
023512  000240                      NOP

                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
023524  032737  100000  001356        BIT     #BIT15,$TMDAT   ;IS ST1 FLAG SET <BIT 15>
023532  001001                      BNE     3$              ;BR IF YES TO 3$:

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

023534  104000                      ERROR                   ;ERROR "SCHMITT TRIG 1" IN NOT
                                                            ;RECEIVED. HAVE YOU WIRED IT RIGHT?

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

023536                          3$:

023536  032777  020000  155374        BIT     #BIT13,@SWR     ;/INHIBIT "*" TYPEOUT?
023544  001330                      BNE     2$              ;/YES - IGNORE ANY UPDATES.
```

```
   (1)  023546  005237  001104            INC     $ICNT           ;/UPDATE COUNT.
   (1)  023552  001325                    BNE     2$              ;/IF NOT DONE 65,324 TIMES,
   (1)                                                            ;/DO IT AGAIN.
   (1)
   (2)  023554  104401  023562            TYPE    ,65$            ;;TYPE ASCIZ STRING
   (2)  023560  000401                    BR      64$             ;;GET OVER THE ASCIZ
   (2)                             ;;65$: .ASCIZ  #*#
   (2)  023564                     64$:
   (1)
   (1)  023564  005237  001210            INC     $PASS           ;/DONE 60 PASSES?
   (1)  023570  100716                    BMI     2$              ;/NO - NO NEED FOR CR,LF.
   (2)  023572  104401  023600            TYPE    ,67$            ;;TYPE ASCIZ STRING
   (2)  023576  000402                    BR      66$             ;;GET OVER THE ASCIZ
   (2)                             ;;67$: .ASCIZ  <15><12>##
   (2)  023604                     66$:
   (1)  023604  000703                    BR      1$
   (1)
  5267
  5268                             .SBTTL ;*      ''STP1 OUT'' TO ''SCHMITT TRIG 2'' H TESTS
  5269                                    ;*
  5270                                    ;* THIS IS A SPECIAL TEST SECTION DEVOTED FOR TESTING AND
  5271                                    ;* PROVIDING SCOPE LOOP CAPABILITIES FOR ''STP1 OUT'' AND
  5272                                    ;* ''SCHMITT TRIG2'' IN.
  5273                                    ;*
  5274                                    ;* WHEN YOU LOAD AND START AT LOCATION 214, PROGRAM
  5275                                    ;* CONTROL IS TRANSFERRED HERE. ''STP1 OUT'' L PULSES ARE
  5276                                    ;* GENERATED BY ''LD STAT A HI'' + ''BD12'' H (MAIN ST1).
  5277                                    ;* PIN DD (''STP1 OUT'') IS WIRED TO PIN BB (''SCHMITT
  5278                                    ;* TRIG 2'') FOR THIS TEST. ''STP1 OUT'' PULSES ARE RECEIVED AS
  5279                                    ;* ''SCHMITT TRIG 2'' PULSES WHICH WILL CLEAR CLOCK A'S
  5280                                    ;* COUNT REGISTER IF MODE 3 IS SELECTED.
  5281                                    ;* IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING TECHNIC.
  5282                                    ;* AND ERROR SWITCH REGISTER OPTIONS ARE USED.
  5283                                    ;* AN ''*'' IS TYPED AFTER EACH 65,324 LOOPS THROUGH
  5284                                    ;* THE TEST. SW13=1 WILL INHIBIT THIS FEATURE.
  5285                                    ;*
  5286                                ;*
   (1)                              ;* PROBABLE SYNC POINT FOR THIS TEST:: ''LD STAT B''
   (1)                              ;*
  5287                                    ;* YOU MUST WIRE PINS DD AND BB OF J1 TOGETHER AND INSTALL JUMPER W4.
  5288                                    ;*
  5289                                    ;* LOGIC TESTS (L + S AT 200) SHOULD BE RUN FIRST.
  5290                                    ;*
  5291
  5292  023606  005037  001666    LS214: CLR     .DVLS
  5293
   (1)  023612  005037  001104    1$:    CLR     $ICNT           ;/CLEAR ITERATION COUNT
   (1)  023616  012737  177704  001210   MOV     #-60.,$PASS     ;/SET PASS COUNT.
   (1)                                                           ;/NOTE: PASS COUNT USED ONLY
   (1)                                                           ;/      TO DETECT 60 PASSES SO
   (1)                                                           ;/      IT CAN GENERATE A CRLF.
   (1)                                                           ;/      AFTER CRLF IT WILL BE ZEROED.
   (1                                                            ;/CLEAR CLOCK A.
   (1)  023624                     2$:                           ;/CLEAR CLOCK B.
   (1)  023624  005037  001356            CLR     $TMDAT
   (2)
```

```
 (2)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (1)
5294                                                                ;LOAD ALL ONES TO CLK A'S BUFFER + COUNT REG.
5295                                                                ;SELECT MODE 3.
5296                                                                ;GENERATE A ''STP1 OUT'' PULSE.
5297                                                                ;AT THIS POINT WE SHOULD SEE AN
5298                                                                ;OUTPUT AT PIN DD.  PIN DD SHOULD
5299                                                                ;BE WIRED TO PIN BB FOR
5300                                                                ;''SCHMITT TRIG 2'' IN. THIS SHOULD
5301                                                                ;CAUSE AN ''STP2'' WHICH WILL CLEAR
5302                                                                ;CLOCK A'S COUNT REGISTER.
5303  023650  012737  177777  001356          MOV     #-1,$TMDAT
5304
 (1)                                ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR
5305  023666  012737  001400  001356          MOV     #BIT8.BIT9,$TMDAT
5306
 (1)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
5307
 (1)                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
5308  023714  052737  010000  001356          BIS     #BIT12,$TMDAT
5309
 (1)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
5310                                                                ;WAS THE COUNT REGISTER CLEARED?
5311
 (1)                                ;*      MOV     @ACR,$BDDAT     ;/READ DEVICE REG ACR,PUT DATA IN $BDDAT.
5312  023742  005737  001126                  TST     $BDDAT
5313
5314  023746  001401                          BEQ     3$              ;IF YES BR 3$.
5315
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

5316  023750  104000                          ERROR                   ;ERROR ''SCHMITT TRIG 2'' IN NOT
5317                                                                ;RECEIVED. HAVE YOU WIRED IT RIGHT?
5318
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

5319  023752                          3$:
5320
 (1)  023752  032777  020000  155160          BIT     #BIT13,@SWR     ;/INHIBIT ''*'' TYPEOUT?
 (1)  023760  001321                          BNE     2$              ;/YES - IGNORE ANY UPDATES.
 (1)
 (1)  023762  005237  001104                  INC     $ICNT           ;/UPDATE COUNT.
 (1)  023766  001316                          BNE     2$              ;/IF NOT DONE 65,324 TIMES,
 (1)                                                                ;/DO IT AGAIN.
 (1)
 (2)  023770  104401  023776                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 (2)  023774  000401                          BR      64$             ;;GET OVER THE ASCIZ
 (2)                                ;;65$:  .ASCIZ  #*#
 (2)  024000                        64$:
 (1)
 (1)  024000  005237  001210                  INC     $PASS           ;/DONE 60 PASSES?
 (1)  024004  100707                          BMI     2$              ;/NO - NO NEED FOR CR,LF.
 (2)  024006  104401  024014                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 (2)  024012  000402                          BR      66$             ;;GET OVER THE ASCIZ
```

```
 (2)
 (2)  024020
 (1)  024020  000674
 (1)
5321
5322                                    .SBTTL  *       "SCHMITT TRIG 3" IN, "ST3 OUT" TESTS
5323                                            ;*
5324                                            ;*THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
5325                                            ;*PROVIDING SCOPE LOOP CAPABILITIES FOR "SCHMITT TRIG 3"
5326                                            ;*AND "ST3 OUT".
5327                                            ;*
5328                                            ;*WHEN YOU LOAD AND START AT LOCATION 220, PROGRAM
5329                                            ;*CONTROL IS TRANSFERRED HERE. "STP1" PULSES ARE GENERATED
5330                                            ;*BY "LD STAT A H," + "BD12" H (MAIN STP1). PIN DD ("STP1 OUT")
5331                                            ;*IS WIRED TO PIN T ("SCHMITT TRIG 3"), "SCHMITT TRIG 3" PULSES
5332                                            ;*GIVE US "ST3 OUT" PULSES. PIN L ("ST3 OUT") IS WIRED
5333                                            ;*TO PIN BB ("SCHMITT TRIG2"), AND "SCHMITT TRIG 2" WILL SET
5334                                            ;*CLOCK A'S STATUS REGISTER BIT 7.
5335                                            ;*IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING TECHNIC.
5336                                            ;*AND ERROR SWITCH REGISTER OPTIONS ARE USED.
5337                                            ;*AN "*" IS TYPED AFTER EACH 65,324 LOOPS THROUGH THE
5338                                            ;*TEST. SW13=1 WILL INHIBIT THIS FEATURE.
5339                                            ;*
5340                                    ;*
 (1)                                    ;* PROBABLE SYNC POINT FOR THIS TEST:: "LD STAT B"
 (1)                                    ;*
5341                                            ;*YOU MUST WIRE PINS DD TO T OF J1 TOGETHER, AS WELL AS
5342                                            ;*PINS L TO BB OF J1 TOGETHER AND INSTALL JUMPERS W4 AND W5.
5343                                            ;*
5344                                            ;*TESTS LS210 AND LS214 SHOULD BE RUN FIRST.
5345                                            ;*
5346
5347  024022  005037  001666          LS220:  CLR     .DVLS
5348
 (1)  024026  005037  001104          1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
 (1)  024032  012737  177704  00'210          MOV     #-60.,$PASS     ;/SET PASS COUNT.
 (1)                                                                  ;/NOTE: PASS COUNT USED ONLY
 (1)                                                                  ;/      TO DETECT 60 PASSES SO
 (1)                                                                  ;/      IT CAN GENERATE A CRLF.
 (1)                                                                  ;/      AFTER CRLF IT WILL BE ZEROED.
 (1)                                                                  ;/CLEAR CLOCK A.
 (1)  024040                          2$:                             ;/CLEAR CLOCK B.
 (1)  024040  005037  001356                  CLR     $TMDAT
 (2)
 (2)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                    ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (1)
5349                                                                    ;GENERATE A "STP2 OUT" PULSE.
5350  024064  005037  001356                  CLR     $TMDAT
5351
 (1)                                    ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
5352  024100  052737  011000  001356          BIS     #BIT12!BIT9,$TMDAT
5353
 (1)                                    ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
5354                                                                    ;AT THIS POINT YOU SHOULD SEE AN
```

```
                                                                ;OUTPUT AT PIN DD. PIN DD SHOULD BE
                                                                ;WIRED TO PIN T TO GIVE ''SCHMITT TRIG3''
                                                                ;INPUT WHICH IN TURN SHOULD GIVE
                                                                ;AN OUTPUT AT PIN L (''ST3''). PIN
                                                                ;L BEING WIRED TO PIN BB (''SCHMITT
                                                                ;TRIG2 IN'') SHOULD GIVE ''ST2'' WHICH
                                                                ;SETS CLOCK A'S CSR BIT 7.


                                                                ;DID BIT 7 SET?

 (1)                                    ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
5365  024126  105737  001126                    TSTB    $BDDAT
5366  024132  100401                          BMI     3$              ;IF YES, BR TO 3$.
5367
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

5368  024134  104000                          ERROR                   ;ERROR ''SCHMITT TRIG 2'' NOT RECEIVED
5369                                                                  ;HAVE YOU WIRED IT RIGHT?
5370
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

5371  024136                            3$:
5372
 (1)  024136  032777  020000  154774          BIT     #BIT13,@SWR     ;/INHIBIT ''*'' TYPEOUT?
 (1)  024144  001335                          BNE     2$              ;/YES - IGNORE ANY UPDATES.
 (1)
 (1)  024146  005237  001104                  INC     $ICNT           ;/UPDATE COUNT.
 (1)  024152  001332                          BNE     2$              ;/IF NOT DONE 65,324 TIMES,
 (1)                                                                  ;/DO IT AGAIN.
 (1)
 (2)  024154  104401  024162                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 (2)  024160  000401                          BR      64$             ;;GET OVER THE ASCIZ
 (2)                                ;;65$:    .ASCIZ  #*#
 (2)  024164                        64$:
 (1)
 (1)  024164  005237  001210                  INC     $PASS           ;/DONE 60 PASSES?
 (1)  024170  100723                          BMI     2$              ;/NO - NO NEED FOR CR,LF.
 (2)  024172  104401  024200                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 (2)  024176  000402                          BR      66$             ;;GET OVER THE ASCIZ
 (2)                                ;;67$:    .ASCIZ  <15><12>##
 (2)  024204                        66$:
 (1)  024204  000710                          BR      1$
 (1)
5373
5374                                .SBTTL  *       ''A EVENT OUT'' TEST
5375                                        ;*
5376                                        ;*THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
5377                                        ;*PROVIDING SCOPE LOOP CAPABILITIES FOR ''A EVENT OUT''.
5378                                        ;*
5379                                        ;*WHEN YOU LOAD AND START AT LOCATION 224, PROGRAM
5380                                        ;*CONTROL IS TRANSFERRED HERE. ''A EVENT OUT'' PULSES ARE
5381                                        ;*GENERATED BY CLOCK A OVERFLOWS. PIN VV
5382                                        ;*(''A EVENT OUT'') IS WIRED TO PIN BB (''SCHMITT TRIG 2'')
5383                                        ;*''SCHMITT TRIG 2'' PULSES WILL SET CLOCK A'S CSR BIT 7.
5384                                        ;*IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING
5385                                        ;*AND ERROR SWITCH REGISTER OPTIONS ARE USED.
```

```
                                        ;*AN "*" IS TYPED AFTER EACH 65,324 LOOPS THROUGH THE TEST.
                                        ;*SW13=1 WILL INHIBIT THIS FEATURE.
                                        ;*
                                ;*
                                ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD STAT B'
                                ;*
                                        ;*YOU MUST WIRE PINS VV AND BB OF J1 TOGETHER AND INSTALL JUMPER W4.
                                        ;*
                                        ;*TEST LS210 SHOULD BE RUN FIRST.
                                        ;*

024206  005037  001666          LS224:  CLR     .DVLS

024212  005037  001104          1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
024216  012737  177704  001210          MOV     #-60.,$PASS     ;/SET PASS COUNT.
                                                                ;/NOTE: PASS COUNT USED ONLY
                                                                ;/      TO DETECT 60 PASSES SO
                                                                ;/      IT CAN GENERATE A CRLF.
                                                                ;/      AFTER CRLF IT WILL BE ZEROED.
                                                                ;/CLEAR CLOCK A.
024224                          2$:                             ;/CLEAR CLOCK B.
024224  005037  001356                  CLR     $TMDAT

                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR

                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR

                                                                ;PRELOAD CLOCK A'S BUFFER + COUNT REGS.
024250  012737  177777  001356          MOV     #-1,$TMDAT

                                ;*      MOV     $TMDAT,@ABR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ABR

                                                                ;RATE: 1MHZ, ENABLE COUNTER
024266  012737  001003  001356          MOV     #1003,$TMDAT

                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR

024304                          3$:                             ;HAS AN OVERFLOW OCCURRED?

                                ;*      MOV     @ASR,$TMDAT     ;/READ DEVICE REG ASR,PUT DATA IN $TMDAT.
024314  032737  000040  001356          BIT     #BIT05,$TMDAT
024322  001770                          BEQ     3$              ;NO - THEN WAIT FOR IT.
                                                                ;OVERFLOW SHOULD GO OUT AS
                                                                ;"A EVENT OUT". YOU SHOULD SEE
                                                                ;AN OUTPUT AT PIN VV.
                                                                ;THIS IN TURN IS BROUGHT
                                                                ;IN AS "SCHMITT TRIG 2" IN TO
                                                                ;SET CLOCK A'S CSR BIT 7.


                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
024334  105737  001126                  TSTB    $BDDAT          ;DID BIT 7 SET
024340  100401                          BMI     4$              ;BR IF YES TO 4$

        ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
5420  024342  104000                          ERROR                   ;ERROR ''A EVENT OUT'' PULSE
5421                                                                  ;NOT DETECTED. HAVE YOU
5422                                                                  ;WIRED IT RIGHT?
5423
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$

5424  024344                          4$:
5425
 (1)  024344  032777  020000  154566           BIT     #BIT13,@SWR     ;/INHIBIT ''*'' TYPEOUT?
 (1)  024352  001324                          BNE     2$              ;/YES - IGNORE ANY UPDATES.
 (1)
 (1)  024354  005237  001104                  INC     $ICNT           ;/UPDATE COUNT.
 (1)  024360  001321                          BNE     2$              ;/IF NOT DONE 65,324 TIMES,
 (1)                                                                  ;/DO IT AGAIN.
 (1)
 (2)  024362  104401  024370                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 (2)  024366  000401                          BR      64$             ;;GET OVER THE ASCIZ
 (2)                                  ;;65$:  .ASCIZ  #*#
 (2)  024372                          64$:
 (1)
 (1)  024372  005237  001210                  INC     $PASS           ;/DONE 60 PASSES?
 (1)  024376  100712                          BMI     2$              ;/NO - NO NEED FOR CR,LF.
 (2)  024400  104401  024406                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 (2)  024404  000402                          BR      66$             ;;GET OVER THE ASCIZ
 (2)                                  ;;67$:  .ASCIZ  <15><12>##
 (2)  024412                          66$:
 (1)  024412  000677                          BR      1$
 (1)
5426
5427                                  .SBTTL  *       'B EVENT OUT'' TEST
5428                                          ;*
5429                                          ;*THIS IS A SPECIAL SECTION DEVOTED FOR TESTING AND
5430                                          ;*PROVIDING SCOPE LOOP CAPABILITIES FOR 'B EVENT OUT''.
5431                                          ;*
5432                                          ;*WHEN YOU LOAD AND START AT LOCATION 230, PROGRAM
5433                                          ;*CONTROL IS TRANSFERRED HERE. 'B EVENT OUT'' PULSES ARE
5434                                          ;*GENERATED BY CLOCK B OVERFLOWS. PIN TT
5435                                          ;*('B EVENT OUT'') IS WIRED TO PIN BB (''SCHMITT TRIG 2'')
5436                                          ;*''SCHMITT TRIG 2'' PULSES WILL SET CLOCK A'S CSR BIT 7.
5437                                          ;*IF AN ERROR IS DETECTED, NORMAL ERROR REPORTING TECHNIQ.
5438                                          ;*AND ERROR SWITCH REGISTER OPTIONS ARE USED.
5439                                          ;*AN ''*'' IS TYPED AFTER EACH 65,324 LOOPS THROUGH THE TEST.
5440                                          ;*SW13-1 WILL INHIBIT THIS FEATURE.
5441                                          ;*
5442                                  ;*
 (1)                                  ;* PROBABLE SYNC POINT FOR THIS TEST:: 'LD BUFF B''
 (1)                                  ;*
5443                                          ;*YOU MUST WIRE PINS TT AND BB OF J1 TOGETHER AND INSTALL JUMPER W4.
5444                                          ;*
5445                                          ;*TEST LS210 SHOULD BE RUN FIRST.
5446                                          ;*
5447
5448  024414  005037  001666          LS230:  CLR     .DVLS
5449
 (1)  024420  005037  001104          1$:     CLR     $ICNT           ;/CLEAR ITERATION COUNT
 (1)  024424  012737  177704  001210          MOV     #-60.,$PASS     ;/SET PASS COUNT.
```

```
 (1)                                                    ;/NOTE: PASS COUNT USED ONLY
 (1)                                                    ;/      TO DETECT 60 PASSES SO
 (1)                                                    ;/      IT CAN GENERATE A CRLF.
 (1)                                                    ;/      AFTER CRLF IT WILL BE ZEROED.
 (1)                                                    ;/CLEAR CLOCK A.
 (1) 024432                          2$:                ;/CLEAR CLOCK B.
 (1) 024432 005037 001356            CLR     $TMDAT
 (2)
 (2)                                 ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
 (2)
 (2)                                 ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
 (1)
5450                                                    ;PRELOAD CLOCK B'S BUFFER + COUNT REGS.
5451 024456 012737 177777 001356    MOV     #-1,$TMDAT
5452
 (1)                                ;*      MOV     $TMDAT,@BBR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BBR
5453 024474 012737 001000 001356    MOV     #1000,$TMDAT
5454
 (1)                                ;*      MOV     $TMDAT,@ASR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG ASR
5455
5456                                                    ;RATE: 1MHZ, ENABLE COUNTER
5457
5458                                                    ;HAS AN OVERFLOW OCCURRED?
5459 024512 012737 000003 001356    MOV     #3,$TMDAT
5460
 (1)                                ;*      MOV     $TMDAT,@BSR     ;/ PUT DATA FROM $TMDAT TO DEVICE REG BSR
5461 024530                         3$:
 (1)
 (1)                                ;*      MOV     @BSR,$TMDAT     ;/READ DEVICE REG BSR,PUT DATA IN $TMDAT.
5462 024540 032737 000200 001356    BIT     #BIT07,$TMDAT
5463 024546 001770                  BEQ     3$                      ;NO -THEN WAIT FOR IT.
5464                                                    ;OVERFLOW SHOULD GO OUT AS
5465                                                    ;'B EVENT OUT''. YOU SHOULD SEE
5466                                                    ;AN OUTPUT AT PIN TT.
5467                                                    ;THIS IN TURN IS BROUGHT
5468                                                    ;IN AS ''SCHMITT TRIG 2'' IN TO
5469                                                    ;SET CLOCK A'S CSR BIT 7.
5470
5471
 (1)                                ;*      MOV     @ASR,$BDDAT     ;/READ DEVICE REG ASR,PUT DATA IN $BDDAT.
5472 024560 105737 001126           TSTB    $BDDAT                  ;DID BIT 7 SET
5473 024564 100401                  BMI     4$                      ;BR IF YES TO 4$
5474
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$
5475 024566 104000                  ERRCR                           ;ERROR 'B EVENT OUT'' PULSE
5476                                                    ;NOT DETECTED. HAVE YOU
5477                                                    ;WIRED IT RIGHT?
5478
                ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$
5479 024570                         4$:
5480
 (1) 024570 032777 020000 154342    BIT     #BIT13,@SWR             ;/INHIBIT ''*'' TYPEOUT?
 (1) 024576 001315                  BNE     2$                      ;/YES - IGNORE ANY UPDATES.
 (1)
```

```
  (1)  024600  005237  001104                 INC     $ICNT           ;/UPDATE COUNT.
  (1)  024604  001312                         BNE     2$              ;/IF NOT DONE 65,324 TIMES,
  (1)                                                                 ;/DO IT AGAIN.
  (1)
  (2)  024606  104401  024614                 TYPE    ,65$            ;;TYPE ASCIZ STRING
  (2)  024612  000401                         BR      64$             ;;GET OVER THE ASCIZ
  (2)                                 ;;65$:  .ASCIZ  #*#
  (2)  024616                         64$:
  (1)
  (1)  024616  005237  001210                 INC     $PASS           ;/DONE 60 PASSES?
  (1)  024622  100703                         BMI     2$              ;/NO - NO NEED FOR CR,LF.
  (2)  024624  104401  024632                 TYPE    ,67$            ;;TYPE ASCIZ STRING
  (2)  024630  000402                         BR      66$             ;;GET OVER THE ASCIZ
  (2)                                 ;;67$:  .ASCIZ  <15><12>##
  (2)  024636                         66$:
  (1)  024636  000670                         BR      1$
  (1)
 5481
 5482                                 ;*ROUTINE TO HANDLE TRAPS TO LOC 4, 10 AND .
 5483                                 ;*INTERRUPTS TO WRONG VECTORS.
 5484                                 ;*.+2, IOTT(TRAPS) WERE PUT IN LOCATIONS 4-1000
 5485
 5486
 5487  024640                         IOTRD:
 5488  024640  011637  025010                 MOV     (R6),2$         ;GET WHERE WE TRAPPED TO.
 5489  024644  162737  000004  025010         SUB     #4,2$           ;=WHERE R6 RETURN 10-4
 5490  024652  104401  024660                 TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  024656  000412                         BR      64$             ;;GET OVER THE ASCIZ
  (1)                                 ;;65$:  .ASCIZ  <15><12>#ILLEGAL TRAP TO: #
  (1)  024704                         64$:
 5491
 5492  024704  013746  025010                 MOV     2$,-(SP)        ;;SAVE 2$ FOR TYPEOUT
  (1)  024710  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 5493
 5494  024712  104401  024720                 TYPE    ,67$            ;;TYPE ASCIZ STRING
  (1)  024716  000407                         BR      66$             ;;GET OVER THE ASCIZ
  (1)                                 ;;67$:  .ASCIZ  # FROM LOC.: #
  (1)  024736                         66$:
 5495
 5496  024736  062706  000004                 ADD     #4,R6           ;POINT TO WHERE WE TRAPPED FROM.
 5497
 5498  024742  011637  025012                 MOV     (R6),3$         ;PICK UP LOC
 5499  024746  162737  000002  025012         SUB     #2,3$           ;FROM REAL ADDR.
 5500  024754  013746  025012                 MOV     3$,-(SP)        ;;SAVE 3$ FOR TYPEOUT
  (1)  024760  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 5501
 5502  024762  023727  025010  000004         CMP     2$,#4           ;DID WE TRAP TO LOC 4?
 5503  024770  001405                         BEQ     1$              ;IF SO - DON'T RETURN!
 5504  024772  023727  025010  000010         CMP     2$,#10          ;DID WE TRAP TO LOC. 10?
 5505  025000  001401                         BEQ     1$              ;ID SO - DON'T RETURN.
 5506  025002  000002                         RTI                     ;TRY RETURNING.
 5507
 5508
                      ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$$

 5509  025004  000000                 1$:     HALT                    ;WE STOPPED HERE BECAUSE WE TRAPPED
```

```
 5510  025006  000776                          BR      1$              ;TO LOC 4 OR LOC 10.  THIS IS A
 5511                                                                  ;FATAL CONDITION THAT WE CAN NOT
 5512                                                                  ;RECOVER FROM.
 5513
 5514
             ;;;$$$$$$$$$$$$$$$$$$$$$$$$$$>> ERROR <<$$$$$$$$$$$$$$$$$$$$$$$$$$

 5515  025010  000000                    2$:     .WORD   0               ;USED BY IOTRP TO STORE WHERE WE TRAPPED TO.
 5516  025012  000000                    3$:     .WORD   0               ;USED BY IOTRP TO STORE WHERE WE TRAPPED FROM.
 5517                                            .SBTTL
 5518                                            .SBTTL  *SYSMAC ROUTINES
 5519                                            .SBTTL
 5520
 5521                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
  (1)
  (2)                                   ;;**********************************************************************
  (1)                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
  (1)                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
  (1)                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
  (1)                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
  (1)                                   ;*      .BYTE   M               ;;M=1 OR 0
  (1)                                   ;*                                      ;;1=TYPE LEADING ZEROS
  (1)                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
  (1)                                   ;*
  (1)                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
  (1)                                   ;*$TYPOS OR $TYPOC
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
  (1)                                   ;*
  (1)                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
  (1)
  (1)  025014  017646  000000           $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
  (1)  025020  116637  000001  025237           MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
  (1)  025026  112637  025241                   MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
  (1)  025032  062716  000002                   ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
  (1)  025036  000406                           BR      $TYPON
  (1)  025040  112737  000001  025237   $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
  (1)  025046  112737  000006  025241           MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
  (1)  025054  112737  000005  025236   $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
  (1)  025062  010346                           MOV     R3,-(SP)        ;;SAVE R3
  (1)  025064  010446                           MOV     R4,-(SP)        ;;SAVE R4
  (1)  025066  010546                           MOV     R5,-(SP)        ;;SAVE R5
  (1)  025070  113704  025241                   MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
  (1)  025074  005404                           NEG     R4
  (1)  025076  062704  000006                   ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
  (1)  025102  110437  025240                   MOVB    R4,$OMODE       ;;SAVE IT FOR USE
  (1)  025106  113704  025237                   MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
  (1)  025112  016605  000012                   MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
  (1)  025116  005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
```

```
(1)  025120  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  025122  000404                          BR      3$              ;;GO DO MSB
(1)  025124  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  025126  006105                          ROL     R5
(1)  025130  006105                          ROL     R5
(1)  025132  010503                          MOV     R5,R3
(1)  025134  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  025136  105337  025240                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  025142  100016                          BPL     7$              ;;BR IF NO
(1)  025144  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
(1)  025150  001002                          BNE     4$              ;;TEST FOR 0
(1)  025152  005704                          TST     R4              ;;SUPPRESS THIS 0?
(1)  025154  001403                          BEQ     5$              ;;BR IF YES
(1)  025156  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  025160  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  025164  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  025170  110337  025234                  MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  025174  104401  025234                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  025200  105337  025236          7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  025204  003347                          BGT     2$              ;;BR IF MORE TO DO
(1)  025206  002402                          BLT     6$              ;;BR IF DONE
(1)  025210  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  025212  000744                          BR      2$              ;;GO DO THE LAST DIGIT
(1)  025214  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  025216  012604                          MOV     (SP)+,R4        ;;RESTORE R4
(1)  025220  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  025222  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  025230  012616                          MOV     (SP)+,(SP)
(1)  025232  000002                          RTI                     ;;RETURN
(1)  025234     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  025235     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  025236     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  025237     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  025240  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
5522                                 .SBTTL  ERROR HANDLER ROUTINE
 (1)
 (2)                                 ;;**********************************************************************
 (1)                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 (1)                                 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 (1)                                 ;*AND GO TO $ERRTYP ON ERROR
 (1)                                 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 (1)                                 ;*SW15=1        HALT ON ERROR
 (1)                                 ;*SW13=1        INHIBIT ERROR TYPEOUTS
 (1)                                 ;*SW10=1        BELL ON ERROR
 (1)                                 ;*SW09-1        LOOP ON ERROR
 (1)                                 ;*CALL
 (1)                                 ;*      ERROR   N       ;;ERROR-EMT AND N-ERROR ITEM NUMBER
 (1)
 (1)  025242                         $ERROR:
 (1)  025242  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 (1)  025244  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
 (1)  025250  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 (1)  025252  013777  001102  153662          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
 (1)  025260  032777  002000  153652          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
 (1)  025266  001402                          BEQ     1$              ;;NO - SKIP
 (1)  025270  104401  001172                  TYPE    ,$BELL          ;;RING BELL
```

```
(1)  025274  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(1)  025300  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  025304  162737  000002  001116          SUB     #2,$ERRPC
(1)  025312  117737  153600  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  025320  032777  020000  153612          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  025326  001004                          BNE     20$             ;;SKIP TYPEOUTS
(1)  025330  004737  025430                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  025334  104401  001177                  TYPE    ,$CRLF
(1)  025340                          20$:
(1)  025340  122737  000001  001222          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  025346  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
(1)  025350  113737  001114  025362          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  025356  004737  027446                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(1)  025362     000                  21$:    .BYTE   0
(1)  025363     000                          .BYTE   0
(1)  025364  000777                  22$:    BR      22$             ;;APT ERROR LOOP
(1)  025366  005777  153546          2$:     TST     @SWR            ;;HALT ON ERROR
(1)  025372  100002                          BPL     3$              ;;SKIP IF CONTINUE
(1)  025374  000000                          HALT                    ;;HALT ON ERROR!
(1)  025376  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  025400  032777  001000  153532  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  025406  001402                          BEQ     4$              ;;BR IF NO
(1)  025410  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  025414  005737  001170          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  025420  001402                          BEQ     5$              ;;BR IF NONE
(1)  025422  013716  001170                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  025426                          5$:
(1)  025426  000002                          RTI                     ;;RETURN
5523                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
(1)                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE" ($ERRTB),
(1)                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(1)
(1)  025430                          $ERRTYP:
(1)  025430  104401  001177                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  025434  010046                          MOV     R0,-(SP)        ;;SAVE R0
(1)  025436  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  025440  153700  001114                  BISB    @#$ITEMB,R0
(1)  025444  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                  ;;TYPE THE PC OF THE ERROR
(2)  025446  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  025452  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  025454  000426                          BR      6$              ;;GET OUT
(1)  025456  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  025460  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
(1)  025462  006300                          ASL     R0
(1)  025464  006300                          ASL     R0
(1)  025466  062700  001360                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  025472  012037  025502                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  025476  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  025500  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  025502  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  025504  104401  001177                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
```

```
   (1)  025510  012037  025520          3$:     MOV     (R0)+,4$        ;;PICKUP 'DATA HEADER' POINTER
   (1)  025514  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
   (1)  025516  104401                          TYPE                    ;;TYPE THE 'DATA HEADER'
   (1)  025520  000000                  4$:     .WORD   0               ;;'DATA HEADER' POINTER GOES HERE
   (1)  025522  104401  001177                  TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
   (1)  025526  011000                  5$:     MOV     (R0),R0         ;;PICKUP 'DATA TABLE' POINTER
   (1)  025530  001004                          BNE     7$              ;;GO TYPE THE DATA
   (1)  025532  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
   (1)  025534  104401  001177                  TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
   (1)  025540  000207                          RTS     PC              ;;RETURN
   (1)  025542                          7$:
   (2)  025542  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
   (2)  025544  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
   (1)  025546  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
   (1)  025550  001770                          BEQ     6$              ;;BR IF NO
   (1)  025552  104401  025560                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
   (1)  025556  000771                          BR      7$              ;;LOOP
   (1)  025560  020040     000          8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
   (1)          025564                          .EVEN
  5524                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
   (1)
   (2)                          ;;**************************************************************************
   (1)                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
   (1)                          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
   (1)                          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
   (1)                          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
   (1)                          ;*REPLACED WITH SPACES.
   (1)                          ;*CALL:
   (1)                          ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
   (1)                          ;*      TYPDS                   ;;GO TO THE ROUTINE
   (1)
   (1)  025564                  $TYPDS:
   (3)  025564  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (3)  025566  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (3)  025570  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (3)  025572  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
   (3)  025574  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
   (1)  025576  012746  020200          MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
   (1)  025602  016605  000020          MOV     20(SP),R5       ;;GET THE INPUT NUMBER
   (1)  025606  100004                  BPL     1$              ;;BR IF INPUT IS POS.
   (1)  025610  005405                  NEG     R5              ;;MAKE THE BINARY NUMBER POS.
   (1)  025612  112766  000055  000001  MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
   (1)  025620  005000          1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
   (1)  025622  012703  026000          MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
   (1)  025626  112723  000040          MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
   (1)  025632  005002          2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
   (1)  025634  016001  025770          MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
   (1)  025640  160105          3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
   (1)  025642  002402                  BLT     4$              ;;BR IF DONE
   (1)  025644  005202                  INC     R2              ;;INCREASE THE BCD DIGIT BY 1
   (1)  025646  000774                  BR      3$
   (1)  025650  060105          4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
   (1)  025652  005702                  TST     R2              ;;CHECK IF BCD DIGIT=0
   (1)  025654  001002                  BNE     5$              ;;FALL THROUGH IF 0
   (1)  025656  105716                  TSTB    (SP)            ;;STILL DOING LEADING 0'S
   (1)  025660  100407                  BMI     7$              ;;BR IF YES
```

```
(1)  025662  106316                    5$:     ASLB    (SP)            ;;MSD?
(1)  025664  103003                            BCC     6$              ;;BR IF NO
(1)  025666  116663  000001  177777            MOVB    1(SP),-(R3)     ;;YES--SET THE SIGN
(1)  025674  052702  000060            6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  025700  052702  000040            7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  025704  110223                            MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  025706  005720                            TST     (R0)+           ;;JUST INCREMENTING
(1)  025710  020027  000010                    CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  025714  002746                            BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  025716  003002                            BGT     8$              ;;GO TO EXIT
(1)  025720  010502                            MOV     R5,R2           ;;GET THE LSD
(1)  025722  000764                            BR      6$              ;;GO CHANGE TO ASCII
(1)  025724  105726                    8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  025726  100003                            BPL     9$              ;;BR IF NO
(1)  025730  116663  177777  177776            MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  025736  105013                    9$:     CLRB    (R3)            ;;SET THE TERMINATOR
(3)  025740  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  025742  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  025744  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  025746  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  025750  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  025752  104401  026000                    TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  025756  016666  000002  000004            MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  025764  012616                            MOV     (SP)+,(SP)
(1)  025766  000002                            RTI                     ;;RETURN TO USER
(1)  025770  023420                    $DTBL:  10000.
(1)  025772  001750                            1000.
(1)  025774  000144                            100.
(1)  025776  000012                            10.
(1)  026000  000004                    $DBLK:  .BLKW   4
5525                                    .SBTTL  SCOPE HANDLER ROUTINE
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(1)                                     ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(1)                                     ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(1)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                     ;*SW14=1        LOOP ON TEST
(1)                                     ;*SW11=1        INHIBIT ITERATIONS
(1)                                     ;*SW09=1        LOOP ON ERROR
(1)                                     ;*SW08-1        LOOP ON TEST IN SWR<7:0>
(1)                                     ;*CALL
(1)                                     ;*      SCOPE           ;;SCOPE-IOT
(1)
(1)  026010                             $SCOPE:
(1)  026010  104407                             CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
(1)  026012  032777  040000  153120     1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  026020  001114                             BNE     $OVER           ;;YES IF SW14=1
(1)                                     ;#####START OF CODE FOR THE XOR TESTER#####
(1)  026022  000416                     $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                     ;;THIS INSTRUCTION TO A "NOP" (NOP 240)
(1)  026024  013746  000004                     MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  026030  012737  026050  000004             MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  026036  005737  177060                     TST     @#177060        ;;TIME OUT ON XOR?
(1)  026042  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  026046  000463                             BR      $SVLAD          ;;GO TO THE NEXT TEST
```

```
(1)  026050  022626                    5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  026052  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  026056  000423                            BR      7$              ;;LOOP ON THE PRESENT TEST
(1)  026060                            6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  026060  032777  000400  153052            BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
(1)  026066  001404                            BEQ     2$              ;;BR IF NO
(1)  026070  127737  153044  001102            CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
(1)  026076  001465                            BEQ     $OVER           ;;BR IF YES
(1)  026100  105737  001103            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  026104  001421                            BEQ     3$              ;;BR IF NO
(1)  026106  123737  001115  001103            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED
(1)  026114  101015                            BHI     3$              ;;BR IF NO
(1)  026116  032777  001000  153014            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(1)  026124  001404                            BEQ     4$              ;;BR IF NO
(1)  026126  013737  001110  001106    7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(1)  026134  000446                            BR      $OVER
(1)  026136  105037  001103            4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  026142  005037  001166                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  026146  000415                            BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  026150  032777  004000  152762    3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  026156  001011                            BNE     1$              ;;BR IF YES
(1)  026160  005737  001210                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  026164  001406                            BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  026166  005237  001104                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  026172  023737  001166  001104            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  026200  002024                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  026202  012737  000001  001104    1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  026210  013737  026266  001166            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  026216  105237  001102            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  026222  113737  001102  001206            MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(1)  026230  011637  001106                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  026234  011637  001110                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  026240  005037  001170                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  026244  112737  000001  001115            MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  026252  013777  001102  152662    $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  026260  013716  001106                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  026264  000002                            RTI                     ;;FIXES PS
(1)  026266  000012                    $MXCNT: 10.                     ;;MAX. NUMBER OF ITERATIONS
5526                                   .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)
(2)                                    ;;**********************************************************************
(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 $HIOCT
(1)
(1)  026270  011646                    $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
(1)  026272  016666  000004  000002            MOV     4(SP),2(SP)     ;;INPUT NUMBER
(3)  026300  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  026302  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  026304  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  026306  104411                    1$:     RDLIN                   ;;READ AN ASCIZ LINE
(1)  026310  012600                            MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)  026312  005001                            CLR     R1              ;;CLEAR DATA WORD
```

```
(1)  026314  005002                        CLR     R2
(1)  026316  112046                  2$:   MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)  026320  001412                        BEQ     3$              ;;IF ZERO GET OUT
(1)  026322  006301                        ASL     R1              ;;*2
(1)  026324  006102                        ROL     R2
(1)  026326  006301                        ASL     R1              ;;*4
(1)  026330  006102                        ROL     R2
(1)  026332  006301                        ASL     R1              ;;*8
(1)  026334  006102                        ROL     R2
(1)  026336  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
(1)  026342  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  026344  000764                        BR      2$              ;;LOOP
(1)  026346  005726                  3$:   TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  026350  010166  000012                MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  026354  010237  026370                MOV     R2,$HIOCT
(3)  026360  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  026362  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  026364  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  026366  000002                        RTI                     ;;RETURN
(1)  026370  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
5527                                 .SBTTL  TTY INPUT ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  .ENABL  LSB
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
(1)                                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
(1)                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
(1)                                  ;*WHEN OPERATING IN TTY FLAG MODE.
(1)  026372  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  026400  001074                        BNE     15$             ;;BRANCH IF NO
(1)  026402  105777  152536                TSTB    @$TKS           ;;CHAR THERE?
(1)  026406  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  026410  117746  152532                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  026414  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  026420  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  026424  001062                        BNE     15$             ;;NO, RETURN TO USER
(1)  026426  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  026434  001456                        BEQ     15$             ;;BRANCH IF YES
(1)
(1)  026436  104401  027117                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  026442  104401  027124          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  026446  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
(2)  026452  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  026454  104401  027135                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  026460  005046                  19$:  CLR     -(SP)           ;;CLEAR COUNTER
(1)  026462  005046                        CLR     -(SP)           ;;THE NEW SWR
(1)  026464  105777  152454          7$:   TSTB    @$TKS           ;;CHAR THERE?
(1)  026470  100375                        BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  026472  117746  152450                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  026476  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)
(1)
(1)
```

```
(1)  026502  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  026506  001005                          BNE     10$             ;;BRANCH IF NOT
(1)  026510  104401  027112                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
(1)  026514  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  026520  000757                          BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  026522  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
(1)  026526  001022                          BNE     16$             ;;BRANCH IF NO
(1)  026530  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  026534  001403                          BEQ     11$             ;;BRANCH IF YES
(1)  026536  016677  000002  152374          MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  026544  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
(1)  026550  104401  001177          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  026554  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  026562  001003                          BNE     15$             ;;BRANCH IF NOT
(1)  026564  012777  000100  152352          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  026572  000002          15$:    RTI                             ;;RETURN
(1)  026574  004737  027360          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  026600  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
(1)  026604  002420                          BLT     18$             ;;BRANCH IF YES
(1)  026606  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
(1)  026612  003015                          BGT     18$             ;;BRANCH IF YES
(1)  026614  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  026620  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  026624  001403                          BEQ     17$             ;;BRANCH IF YES
(1)  026626  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  026630  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
(1)  026632  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
(1)  026634  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  026640  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  026644  000707                          BR      7$              ;;GET THE NEXT ONE
(1)  026646  104401  001176          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  026652  000720                          BR      20$             ;;SIMULATE CONTROL-U
(1)                                  .DSABL  LSB
(1)
(1)
(2)                                  ;;******************************************************************
(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)                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                                  ;
(1)
(1)  026654  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  026656  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  026664  105777  152254          1$:     TSTB    @$TKS           ;;WAIT FOR
(1)  026670  100375                          BPL     1$              ;;A CHARACTER
(1)  026672  117766  152250  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  026700  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  026706  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  026714  001013                          BNE     3$              ;;BRANCH IF NO
(1)  026716  105777  152222          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  026722  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  026724  117746  152216                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
```

```
(1)  026730  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  026734  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  026740  001366                        BNE     2$              ;;IF NOT DISCARD IT
(1)  026742  000750                        BR      1$              ;;YES, RESUME
(1)  026744  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  026752  002407                        BLT     4$              ;;BRANCH IF YES
(1)  026754  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  026762  003003                        BGT     4$              ;;BRANCH IF YES
(1)  026764  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  026772  000002                4$:     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)  026774  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  026776  012703  027102        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  027002  022703  027112        2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
(1)  027006  101405                        BLOS    4$              ;;BR IF YES
(1)  027010  104410                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  027012  112613                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  027014  122713  000177        10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  027020  001003                        BNE     3$              ;;SKIP IF NOT
(1)  027022  104401  001176        4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  027026  000763                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  027030  111337  027100        3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  027034  104401  027100                TYPE    ,9$
(1)  027040  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  027044  001356                        BNE     2$              ;;LOOP IF NOT RETURN
(1)  027046  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE ^5)
(1)  027052  104401  001200                TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  027056  012603                        MOV     (SP)+,R3        ;;RESTORE R3
(1)  027060  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  027062  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  027070  012766  027102  000004        MOV     #$TTYIN,4(SP)
(1)  027076  000002                        RTI                     ;;RETURN
(1)  027100     000                9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  027101     000                        .BYTE   0               ;;TERMINATOR
(1)  027102  000010                $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  027112  052536  005015  000   $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
(1)  027117     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
(1)  027124  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  027132  020075     000
(1)  027135     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
(1)  027142  036440  000040
5528                                       .SBTTL  TYPE ROUTINE
(1)
(2)                                        ;;**************************************************************************
(1)                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1)                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1)                                        ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1)                                        ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1)                                        ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1)                                        ;*
```

```
(1)                                     ;*CALL:
(1)                                     ;*1) USING A TRAP INSTRUCTION
(1)                                     ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                     ;*OR
(1)                                     ;*      TYPE
(1)                                     ;*      MESADR
(1)                                     ;*
(1)
(1)  027146  105737  001157            $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  027152  100002                            BPL     1$              ;;BR IF YES
(1)  027154  000000                            HALT                    ;;HALT HERE IF NO TERMINAL
(1)  027156  000430                            BR      3$              ;;LEAVE
(1)  027160  010046                    1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  027162  017600  000002                    MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  027166  122737  000001  001222            CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  027174  001011                            BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(1)  027176  132737  000100  001223            BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(1)  027204  001405                            BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  027206  010037  027216                    MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  027212  004737  027436                    JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(1)  027216  000000                    61$:    .WORD   0               ;;MESSAGE ADDRESS
(1)  027220  132737  000040  001223    62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  027226  001003                            BNE     60$             ;;YES,SKIP TYPE OUT
(1)  027230  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  027232  001005                            BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  027234  005726                            TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  027236  012600                    60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  027240  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  027244  000002                            RTI                     ;;RETURN
(1)  027246  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  027252  001430                            BEQ     8$
(1)  027254  122716  000200                    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  027260  001006                            BNE     5$
(1)  027262  005726                            TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  027264  104401                            TYPE                    ;;TYPE A CR AND LF
(1)  027266  001177                            $CRLF
(1)  027270  105037  027424                    CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  027274  000755                            BR      2$              ;;GET NEXT CHARACTER
(1)  027276  004737  027360            5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  027302  123726  001156            6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  027306  001350                            BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  027310  013746  001154                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                             ;;AND THE NULL CHAR.
(1)  027314  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  027320  002770                            BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  027322  004737  027360                    JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  027326  105337  027424                    DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  027332  000770                            BR      7$              ;;LOOP
(1)
(1)                                     ;HORIZONTAL TAB PROCESSOR
(1)
(1)  027334  112716  000040            8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  027340  004737  027360            9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  027344  132737  000007  027424            BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  027352  001372                            BNE     9$              ;;TAB STOP
(1)  027354  005726                            TST     (SP)+           ;;POP SPACE OFF STACK
```

```
(1)  027356  000724                          BR      2$              ;;GET NEXT CHARACTER
(1)  027360  105777  151564          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  027364  100375                          BPL     $TYPEC
(1)  027366  116677  000002  151556          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  027374  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  027402  001003                          BNE     1$              ;;BRANCH IF NO
(1)  027404  105037  027424                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  027410  000406                          BR      $TYPEX          ;;EXIT
(1)  027412  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  027420  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
(1)  027422  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  027424  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  027426  000207          $TYPEX: RTS     PC
(1)
5529                         .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)                          ;;**************************************************************************
(1)  027430  112737  000001  027674  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(1)  027436  112737  000001  027672  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(1)  027444  000403                          BR      $ATYC
(1)  027446  112737  000001  027674  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(1)  027454                          $ATYC:
(3)  027454  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  027456  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(1)  027460  105737  027672                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
(1)  027464  001450                          BEQ     5$              ;;IF NOT:  BR
(1)  027466  122737  000001  001222          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
(1)  027474  001031                          BNE     3$              ;;IF NOT:  BR
(1)  027476  132737  000100  001223          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1)  027504  001425                          BEQ     3$              ;;IF NOT:  BR
(1)  027506  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
(1)  027512  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  027520  005737  001202          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(1)  027524  001375                          BNE     1$              ;;IF NOT:  WAIT
(1)  027526  010037  001216                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
(1)  027532  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
(1)  027534  001376                          BNE     2$
(1)  027536  163700  001216                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
(1)  027542  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
(1)  027544  010037  001220                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
(1)  027550  012737  000004  001202          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
(1)  027556  000413                          BR      5$
(1)  027560  017637  000004  027604  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(1)  027566  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(3)  027574  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(1)  027600  004737  027146                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
(1)  027604  000000                  4$:     .WORD   0
(1)  027606                          5$:
(1)  027606  105737  027674          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(1)  027612  001416                          BEQ     12$             ;;IF NOT:  BR
(1)  027614  005737  001222                  TST     $ENV            ;;RUNNING UNDER APT?
(1)  027620  001413                          BEQ     12$             ;;IF NOT:  BR
(1)  027622  005737  001202          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1)  027626  001375                          BNE     11$             ;;IF NOT:  WAIT
(1)  027630  017637  000004  001204          MOV     @4(SP),$FATAL   ;;GET ERROR #
(1)  027636  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
```

```
(1)  027644  005237  001202                 INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1)  027650  105037  027674       12$:      CLRB    $FFLG           ;;CLEAR FATAL FLAG
(1)  027654  105037  027673                 CLRB    $LFLG           ;;CLEAR LOG FLAG
(1)  027660  105037  027672                 CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(3)  027664  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  027666  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  027670  000207                         RTS     PC              ;;RETURN
(1)  027672     000         $MFLG:  .BYTE   0               ;;MESSG. FLAG
(1)  027673     000         $LFLG:  .BYTE   0               ;;LOG FLAG
(1)  027674     000         $FFLG:  .BYTE   0               ;;FATAL FLAG
(1)          027676                 .EVEN
(1)          000200         APTSIZE=200
(1)          000001         APTENV=001
(1)          000100         APTSPOOL=100
(1)          000040         APTCSUP=040
5530                        .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                         ;;**********************************************************************
(1)                         ;POWER DOWN ROUTINE
(1)  027676  012737  030036  000024 $PWRDN:  MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)  027704  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
(3)  027712  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  027714  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  027716  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  027720  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  027722  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  027724  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(3)  027726  017746  151206                 MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
(1)  027732  010637  030042                 MOV     SP,$SAVR6       ;;SAVE SP
(1)  027736  012737  027750  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)  027744  000000                         HALT
(1)  027746  000776                         BR      .-2             ;;HANG UP
(1)
(2)                         ;;**********************************************************************
(1)                         ;POWER UP ROUTINE
(1)  027750  012737  030036  000024 $PWRUP:  MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1)  027756  013706  030042                 MOV     $SAVR6,SP       ;;GET SP
(1)  027762  005037  030042                 CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
(1)  027766  005237  030042         1$:     INC     $SAVR6          ;;WAIT FOR THE INC
(1)  027772  001375                         BNE     1$              ;;OF  WORD
(3)  027774  012677  151140                 MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
(3)  030000  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  030002  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  030004  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  030006  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  030010  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  030012  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  030014  012737  027676  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  030022  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  030030  104401                         TYPE                    ;REPORT THE POWER FAILURE
(1)  030032  030044         $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
(1)  030034  000002                         RTI
(1)  030036  000000         $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
(1)  030040  000776                         BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  030042  000000         $SAVR6: 0                       ;;PUT THE SP HERE
(1)  030044  005015  047520  042527 $POWER: .ASCIZ  <15><12>'POWER'
```

```
  (1)  030052  000122
  (1)                                        .EVEN
 5531                                .SBTTL  TRAP DECODER
  (1)
  (2)                                ;;**********************************************************************
  (1)                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
  (1)                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
  (1)                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
  (1)                                ;*GO TO THAT ROUTINE.
  (1)
  (1)  030054  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
  (1)  030056  016600  000002               MOV     2(SP),R0        ;;GET TRAP ADDRESS
  (1)  030062  005740                        TST     -(R0)           ;;BACKUP BY 2
  (1)  030064  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
  (1)  030066  006300                        ASL     R0              ;;POSITION FOR INDEXING
  (1)  030070  016000  030110                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
  (1)  030074  000200                        RTS     R0              ;;GO TO ROUTINE
  (1)
  (1)
  (1)                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
  (1)
  (1)  030076  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
  (1)  030100  016666  000004  000002       MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
  (1)  030106  000002                        RTI                     ;;RESTORE THE PSW
  (1)
  (3)                                .SBTTL  TRAP TABLE
  (3)
  (3)                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
  (3)                                ;*BY THE 'TRAP' INSTRUCTION.
  (3)
  (3)                                ;       ROUTINE
  (3)                                ;       -------
  (3)  030110  030076                $TRPAD: .WORD   $TRAP2
  (3)  030112  027146                        $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
  (3)  030114  025040                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
  (3)  030116  025014                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
  (3)  030120  025054                        $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
  (3)  030122  025564                        $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
  (1)
  (3)  030124  026442                        $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
  (1)
  (3)  030126  026372                        $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
  (3)  030130  026654                        $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
  (3)  030132  026774                        $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
  (3)  030134  026270                        $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
 5532  030136  024640                        IOTRD   ;;CALL=IOTT     TRAP+13(104413)
 5533  030140  005015  046103  041517  EM1:     .ASCIZ  <15><12>/CLOCKA SR FUNCTION ERROR/
       030146  040513  051440  020122
       030154  052506  041516  044524
       030162  047117  042440  051122
       030170  051117     000
 5534  030173     015  041412  047514  EM2:     .ASCIZ  <15><12>/CLOCKA SR DATA ERROR/
       030200  045503  020101  051123
       030206  042040  052101  020101
       030214  051105  047522  000122
 5535  030222  005015  046103  041517  EM3:     .ASCIZ  <15><12>/CLOCKA BR DATA ERROR/
```

```
                030230  040513  041040  020122
                030236  040504  040524  042440
                030244  051122  051117     000
5536            030251     015  041412  047514  EM4:    .ASCIZ  <15><12>/CLOCKA CR DATA ERROR/
                030256  045503  020101  051103
                030264  042040  052101  020101
                030272  051105  047522  000122
5537            030300  005015  046103  041517  EM5:    .ASCIZ  <15><12>/CLOCKB SR DATA ERROR/
                030306  041113  051440  020122
                030314  040504  040524  042440
                030322  051122  051117     000
5538            030327     015  041412  047514  EM6:    .ASCIZ  <15><12>/CLOCKB BR DATA ERROR/
                030334  045503  020102  051102
                030342  042040  052101  020101
                030350  051105  047522  000122
5539            030356  005015  046103  041517  EM7:    .ASCIZ  <15><12>/CLOCKB CR DATA ERROR/
                030364  041113  041440  020122
                030372  040504  040524  042440
                030400  051122  051117     000
5540            030405     015  042012  040525  EM10:   .ASCIZ  <15><12>/DUAL ADDRESS ERROR/
                030412  020114  042101  051104
                030420  051505  020123  051105
                030426  047522  000122
5541            030432  005015  046103  041517  EM11:   .ASCIZ  <15><12>#CLOCK A COUNT ERROR #
                030440  020113  020101  047503
                030446  047125  020124  051105
                030454  047522  020122     000
5542            030461     015  041412  047514  EM12:   .ASCIZ  <15><12>#CLOCK A COUNT FUNCTION ERROR #
                030466  045503  040440  041440
                030474  052517  052116  043040
                030502  047125  052103  047511
                030510  020116  051105  047522
                030516  020122     000
5543            030521     015  041412  047514  EM14:   .ASCIZ  <15><12>#CLOCK B COUNT FUNCTION ERROR #
                030526  045503  041040  041440
                030534  052517  052116  043040
                030542  047125  052103  047511
                030550  020116  051105  047522
                030556  020122     000
5544            030561     015  041412  047514  EM15:   .ASCIZ  <15><12>#CLOCK B COUNT ERROR #
                030566  045503  041040  041440
                030574  052517  052116  042440
                030602  051122  051117  000040
5545            030610  005015  046103  041517  EM16:   .ASCIZ  <15><12>#CLOCK A INTERRUPT ERROR #
                030616  020113  020101  047111
                030624  042524  051122  050125
                030632  020124  051105  047522
                030640  020122     000
5546            030643     015  041412  047514  EM17:   .ASCIZ  <15><12>#CLOCK B INTERRUPT ERROR #
                030650  045503  041040  044440
                030656  052116  051105  052522
                030664  052120  042440  051122
                030672  051117  000040
5547            030676  005015  046103  041517  EM20:   .ASCIZ  <15><12>#CLOCK A REPEATABILITY ERROR #
                030704  020113  020101  042522
                030712  042520  052101  041101
```

```
        030720  046111  052111  020131
        030726  051105  047522  020122
        030734     000
 5548   030735     015  041412  047514  EM23:   .ASCIZ  <15><12>#CLOCK B REPEATABILITY ERROR #
        030742  045503  041040  051040
        030750  050105  040505  040524
        030756  044502  044514  054524
        030764  042440  051122  051117
        030772  000040
 5549   030774  005015  046103  041517  EM26:   .ASCIZ  <15><12>#CLOCK ADDRESSING ERROR#
        031002  020113  042101  051104
        031010  051505  044523  043516
        031016  042440  051122  051117
        031024     000
 5550
 5551   031025     015  042412  051122  DH1:    .ASCIZ  <15><12>#ERRPC   ASR     WAS     S/B#
        031032  041520  020040  040440
        031040  051123  020040  020040
        031046  053440  051501  020040
        031054  020040  051440  041057
        031062     000
 5552   031063     015  042412  051122  DH3:    .ASCIZ  <15><12>#ERRPC   ABR     WAS     S/B#
        031070  041520  020040  040440
        031076  051102  020040  020040
        031104  053440  051501  020040
        031112  020040  051440  041057
        031120     000
 5553   031121     015  042412  051122  DH4:    .ASCIZ  <15><12>#ERRPC   ACR     WAS     S/B#
        031126  041520  020040  040440
        031134  051103  020040  020040
        031142  053440  051501  020040
        031150  020040  051440  041057
        031156     000
 5554   031157     015  042412  051122  DH5:    .ASCIZ  <15><12>#ERRPC   BSR     WAS     S/B#
        031164  041520  020040  041040
        031172  051123  020040  020040
        031200  053440  051501  020040
        031206  020040  051440  041057
        031214     000
 5555   031215     015  042412  051122  DH6:    .ASCIZ  <15><12>#ERRPC   BBR     WAS     S/B#
        031222  041520  020040  041040
        031230  051102  020040  020040
        031236  053440  051501  020040
        031244  020040  051440  041057
        031252     000
 5556   031253     015  042412  051122  DH7:    .ASCIZ  <15><12>#ERRPC   BCR     WAS     S/B#
        031260  041520  020040  041040
        031266  051103  020040  020040
        031274  053440  051501  020040
        031302  020040  051440  041057
        031310     000
 5557   031311     015  042412  051122  DH10:   .ASCII  <15><12>/ERROR   GOOD    BAD     GOOD    DATA READ FROM/
        031316  051117  020040  043440
        031324  047517  020104  020040
        031332  041040  042101  020040
        031340  020040  043440  047517
```

```
                031346  020104  020040  042040
                031354  052101  020101  042522
                031362  042101  043040  047522
                031370     115
 5558           031371     015  020012  050040          .ASCIZ  <15><12>/  PC    ADDR    ADDR    DATA    DUAL ADDRESS/
                031376  020103  020040  040440
                031404  042104  020122  020040
                031412  040440  042104  020122
                031420  020040  042040  052101
                031426  020101  020040  042040
                031434  040525  020114  042101
                031442  051104  051505  000123
 5559           031450  005015  051105  050122  DH12.   .ASCIZ  <15><12>#ERRPC   ASR #
                031456  020103  020040  051501
                031464  020122     000
 5560           031467     015  042412  051122  DH14:   .ASCIZ  <15><12>#ERRPC   BSR#
                031474  041520  020040  041040
                031502  051123     000
 5561           031505     015  042412  051122  DH20:   .ASCIZ  <15><12>#ERRPC   ASR     2NDCNT  1STCNT #
                031512  041520  020040  040440
                031520  051123  020040  020040
                031526  031040  042116  047103
                031534  020124  030440  052123
                031542  047103  020124     000
 5562           031547     015  042412  051122  DH23:   .ASCIZ  <15><12>#ERRPC   BSR     2NDCNT  1STCNT #
                031554  041520  020040  041040
                031562  051123  020040  020040
                031570  031040  042116  047103
                031576  020124  030440  052123
                031604  047103  020124     000
 5563           031611     015  042412  051122  DH26:   .ASCIZ  <15><12>#ERRPC   CLOCK ADDR.#
                031616  041520  020040  041440
                031624  047514  045503  040440
                031632  042104  027122     000
 5564
 5565           031640                          .EVEN
 5566
 5567           031640  001116  001326  001126  DT1:    .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,0
                031646  001124  000000
 5568           031652  001116  001330  001126  DT3:    .WORD   $ERRPC,ABR,$BDDAT,$GDDAT,0
                031660  001124  000000
 5569           031664  001116  001332  001126  DT4:    .WORD   $ERRPC,ACR,$BDDAT,$GDDAT,0
                031672  001124  000000
 5570           031676  001116  001334  001126  DT5:    .WORD   $ERRPC,BSR,$BDDAT,$GDDAT,0
                031704  001124  000000
 5571           031710  001116  001336  001126  DT6:    .WORD   $ERRPC,BBR,$BDDAT,$GDDAT,0
                031716  001124  000000
 5572           031722  001116  001340  001126  DT7:    .WORD   $ERRPC,BCR,$BDDAT,$GDDAT,0
                031730  001124  000000
 5573           031734  001116  001120  001122  DT10:   .WORD   $ERRPC,$GDADR,$BDADR,$GDDAT,$BDDAT,0
                031742  001124  001126  000000
 5574           031750  001116  001326  000000  DT12:   .WORD   $ERRPC,ASR,0
 5575           031756  001116  001334  000000  DT14:   .WORD   $ERRPC,BSR,0
 5576           031764  001116  001332  001124  DT21:   .WORD   $ERRPC,ACR,$GDDAT,$TMP0,0
                031772  001164  000000
 5577           031776  001116  001332  001126  DT22:   .WORD   $ERRPC,ACR,$BDDAT,$TMP0,0
```

```
        032004  001164  000000
 5578   032010  001116  001340  001124  DT24:   .WORD   $ERRPC,BCR,$GDDAT,$TMP0,0
        032016  001164  000000
 5579   032022  001116  001340  001126  DT25:   .WORD   $ERRPC,BCR,$BDDAT,$TMP0,0
        032030  001164  000000
 5580   032034  001116  001164  000000  DT26:   .WORD   $ERRPC,$TMP0,0
 5581
 5582   032042  000000  000000          DF0.    .WORD   0,0
 5583
 5584
 5585
  (2)                                           ;*
  (2)                                           ;*THIS SUB CODE IS USED TO INITIALIZE THE LPA-11
  (2)                                           ;*FIRST WE WILL LOAD MICROCODE INTO KMC-11
  (2)                                           ;*NEXT WE WILL INIT BOTH UPROCESSORS
  (2)                                           ;*THEN WE WILL LOAD DEVICE TABLE IN SLAVE UP.
  (2)                                           ;*THE ORDER OF LOAD IS DETERMINED BY THE USER.
  (2)                                           ;*
  (2)                                           ;*      CALL=   JSR     R5,$LPAI
  (2)                                           ;*              .WORD   0               ;ADDR. OF DEVICE ADDRESS.
  (2)                                           ;* ROUTINES REQUIRED:   .LOADLP
  (2)                                           ;* PROGRAMS REQUIRED:   DRLPX2
  (2)                                           ;*
  (2)
  (2)                                           ;*              ;RETURNS WITH $AERR-1 IF SLAVE
  (2)                                           ;*              ;MICRO SAYS AN ADDR. DOES NOT EXSIST. IN THE LIST.
  (2)                                           ;*
  (2)   032046                          $LPAI:
  (2)   032046  013746  000004                  MOV     4,-(SP)
  (2)
  (2)   032052  000413                          BR      31$             ;FIELD DOES NOT HAVE A BUS SWITCH TO
  (2)                                           ;WORRY ABOUT,SO WE WILL UNCONDITIONALLY
  (2)                                           ;BRANCH ARROUD THE NEXT CODE THAT
  (2)                                           ;WORKS BASED ON A BUS SWITCH.
  (2)                                           ;CODE LEFT IN HERE FOR IN HOUSE
  (2)                                           ;PERSONAL WHO MAY PATCH THIS BRANCH
  (2)                                           ;INSTRUCTION TO A <NOP> OCTAL <240>
  (2)                                           ;IN ORDER TO RUN PROGRAM WITH A SWITCH.
  (2)
  (2)                                           ;NOTE THIS 'SWITCH' IS A PIECE OF INHOUSE
  (2)                                           ;TEST EQUIPMENT ONLY IT CONNECTS
  (2)                                           ;THE UNIBUS TO THE I/O BUS FOR
  (2)                                           ;CERTAIN TESTING.
  (2)   032054  012737  032100  000004          MOV     #30$,4
  (2)   032062  005237  170000                  INC     170000
  (3)   032066  104401  032074                  TYPE    ,65$            ;;TYPE ASCIZ STRING
  (3)   032072  000401                          BR      64$             ;;GET OVER THE ASCIZ
  (3)                                   ;;65$:  .ASCIZ  <7>##
  (3)   032076                          64$:
  (2)   032076  000401                          BR      31$
  (2)   032100  022626                  30$:    CMP     (SP)+,(SP)+
  (2)   032102  012637  000004          31$:    MOV     (SP)+,4         ;ALL THIS JUNK MUST BE REMOVED
  (2)   032106  005037  032724                  CLR     $AERR
  (2)   032112  004537  032726                  JSR     R5,$LOAD        ;LOAD MICRO-CODE..
  (2)   032116  000000G                         .WORD   DRLPX2          ;FILE 'DRLPX2.OBJ'
  (2)
```

```
(2)  032120  052777  040000  147512          BIS     #BIT14,@KMAD0   ;ISSUE KMC+DMC INIT.
(2)
(2)  032126                          1$:
(2)                                                                  ;'HANGS'' HERE THEN KMC-11 ERROR.
(2)  032126  010146                          MOV     R1,-(SP)
(2)  032130  005001                          CLR     R1
(2)  032132  005201                  2$:     INC     R1              ;STALL FOR DMC-UP
(2)  032134  001376                          BNE     2$
(2)  032136  012777  104000  147474          MOV     #BIT15.BIT11,@KMAD0     ;SET RUN, AND ENABLE ARBITRATION.
(2)  032144  105201                  25$:    INCB    R1
(2)  032146  001376                          BNE     25$
(2)
(2)  032150  032777  000040  147462          BIT     #BIT5,@KMAD0    ;SLAVE READY? (READING IPBM SR)
(2)  032156  001401                          BEQ     3$
(2)                                                                  ;FATAL LPA-11 ERROR SLAVE NOT READY.
(2)  032160  104000                          ERROR
(2)
(2)  032162  012777  000004  147454  3$:     MOV     #4,@KMAD2       ;READ FAST PATH
(2)  032170                          4$:
(3)  032170  004537  033636                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  032174  104000                          ERROR                   ;/TIME-OUT ERROR
(3)                                                                  ;/WE FAILED TO COMPLETE
(3)                                                                  ;/CURRENT OPERATION.
(3)                                                                  ;/CONTINUES IN THIS LOOP
(3)                                                                  ;/WOULD MAKE US 'HANG'' HERE
(3)
(3)  032176  000774                          BR              4$
(3)
(3)                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
(2)  032200  122777  000377  147436          CMPB    #377,@KMAD2     ;WAIT TILL KMC DONE COMMAND.
(2)  032206  001370                          BNE     4$
(2)  032210  122777  000377  147432          CMPB    #377,@KMAD4     ;IF FAST PATH=377 THEN ERROR.
(2)  032216  001001                          BNE     35$
(2)  032220  104000                          ERROR                   ;IPBM ERROR (SLAVE SIDE)
(2)                                                                  ;YOU MUST RUN IPBM DIAGNOSTIC.
(2)
(2)  032222  122777  000004  147420  35$:    CMPB    #4,@KMAD4       ;IS THIS THE CORRECT VERSION OF MICRO-CODE?
(2)  032230  001543                          BEQ     5$              ;YES-CONTINUE.
(2)  032232  005227  177777                  INC     #-1
(2)  032236  001140                          BNE     5$
(2)  032240  005227  177777                  INC     #-1
(2)  032244  001135                          BNE     5$
(3)  032246  104401  032254                  TYPE    ,67$            ;;TYPE ASCIZ STRING
(3)  032252  000440                          BR      66$             ;;GET OVER THE ASCIZ
(3)                                  ;;67$:  .ASCIZ  <200>'W A R N I N G  THIS PROGRAM WAS DESIGNED TO RUN WITH VERSION 4''
(3)  032354                          66$:
(3)  032354  104401  032362                  TYPE    ,69$            ;;TYPE ASCIZ STRING
(3)  032360  000430                          BR      68$             ;;GET OVER THE ASCIZ
(3)                                  ;;69$:  .ASCIZ  <200>'MICRO-CODE. ANOTHER VERSION CODE WAS DETECTED.''
(3)  032442                          68$:
(3)  032442  104401  032450                  TYPE    ,71$            ;;TYPE ASCIZ STRING
(3)  032446  000434                          BR      70$             ;;GET OVER THE ASCIZ
(3)                                  ;;71$:  .ASCIZ  <200>''THIS MAY OR MAYNOT CAUSE FALSE ERROR TO BE REPORTED.''<200><200>
(3)  032540                          70$:
(2)
```

```
(2)  032540  112737  177777  032672  5$:     MOVB    #0-1,11$        ;DAC CODE FOR SLAVE.
(2)  032546  012501                          MOV     (5)+,R1         ;GET NEXT DEVICE ADDR.
(2)  032550  021127  000000          6$:     CMP     (R1),#0         ;TERM REACHED?
(2)  032554  001444                          BEQ     10$
(2)  032556  105237  032672                  INCB    11$
(2)  032562  113777  032672  147060          MOVB    11$,@KMAD4      ;FIFO DATA
(2)  032570  004737  032674                  JSR     PC,20$          ;ISSUE SEND
(2)  032574  112177  147050                  MOVB    (R1)+,@KMAD4    ;SEND LOW BYTE OF DEVICE ADDR TO SLAVE.
(2)  032600  004737  032674                  JSR     PC,20$          ;ISSUE SEND
(2)  032604  112177  147040                  MOVB    (R1)+,@KMAD4    ;SEND HIGH BYTE OF DEVICE ADDR. TO SLAVE.
(2)  032610  004737  032674                  JSR     PC,20$
(2)
(2)  032614  032777  000002  147016  7$:     BIT     #BIT1,@KMAD0    ;WAIT FOR FIFO DATA
(2)  032622  001374                          BNE     7$              ;-1 NO DATA. =0 DATA.
(2)  032624  112777  000002  147012          MOVB    #2,@KMAD2       ;READ FIFO.
(2)
(2)  032632                          8$:
(3)  032632  004537  033636                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  032636  104000                          ERROR                   ;/TIME-OUT ERROR
(3)                                                                  ;/WE FAILED TO COMPLETE
(3)                                                                  ;/CURRENT OPERATION.
(3)                                                                  ;/CONTINUES IN THIS LOOP
(3)                                                                  ;/WOULD MAKE US "HANG" HERE
(3)
(3)  032640  000774                          BR              8$
(3)
(3)                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
(2)  032642  122777  000377  146774          CMPB    #377,@KMAD2     ;WAIT FOR READ.
(2)  032650  001370                          BNE     8$
(2)  032652  105777  146772                  TSTB    @KMAD4          ;WAS A ZERO RETURNED?
(2)  032656  001734                          BEQ     6$              ;YES GET NEXT ADDR.
(2)                                                                  ;SLAVE WILL RETURN CODE 0 IF
(2)  032660  005237  032724                  INC     $AERR           ;DEV PRESENT.   ELSE
(2)                                                                  ;EXIT $AERR=1 IF SLAVE GIVES ERROR.
(2)  032664  005041                          CLR     -(1)            ;GET RID OF REFERENCE TO BAD ADDR.
(2)  032666  012601                  10$:    MOV     (SP)+,R1
(2)  032670  000205                          RTS     R5              ;RETURN ALL ADDR. CHECKED.
(2)
(2)  032672  000000                  11$:    .WORD   0               ;HOLDS DAC CODE PLUS OFFSET
(2)                                                                  ;TO SLAVES ADDR. TABLE.
(2)
(2)  032674  112777  000003  146742  20$:     MOVB    #3,@KMAD2       ;ISSUE FIFO WRITE
(2)  032702                          21$:
(3)  032702  004537  033636                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  032706  104000                          ERROR                   ;/TIME-OUT ERROR
(3)                                                                  ;/WE FAILED TO COMPLETE
(3)                                                                  ;/CURRENT OPERATION.
(3)                                                                  ;/CONTINUES IN THIS LOOP
(3)                                                                  ;/WOULD MAKE US "HANG" HERE
(3)
(3)  032710  000774                          BR              21$
(3)
(3)                                                                  ;/RETURNS HERE-FROM-TIMED OUT.
(2)  032712  122777  000377  146724          CMPB    #377,@KMAD2     ;KMC CODE WILL RETURN A "377"
```

```
(2)  032720  001370                    BNE     21$             ;WHEN DONE COMMAND.
(2)  032722  000207                    RTS     PC
(2)
(2)  032724  000000            $AERR:  .WORD   0               ; 0 IF ADDR. LIST OK,-1 IF BAD.
(2)
(2)                                     ;*
(2)                                     ;*THIS SUB CODE USED TO LOAD MICRO-CODE INTO LPA-11.
(2)                                     ;*      CALL =  JSR     R5,$LOAD
(2)                                     ;*              .WORD   XX              ;ADDR. OF MICRO CODE.
(2)                                     ;*              ;RETURNS HERE
(2)                                     ;*      NOTE:   MICRO CODE FILE MUST END IN -1 DATA.
(2)                                     ;*
(2)
(2)  032726  010446            $LOAD:  MOV     R4,-(SP)        ;SAVE R4.
(2)  032730  010046                    MOV     R0,-(SP)        ;SAVE R0.
(2)  032732  012500            1$:     MOV     (5)+,R0         ;GET PROG. ADDR.
(2)  032734  005077  146700            CLR     @KMAD0          ;CLEAR CSR
(2)  032740  005077  146704            CLR     @KMAD4          ;CLEAR CRAM ADDR.
(2)  032744  052777  002000  146666  2$:     BIS     #2000,@KMAD0    ;SELECT CRAM.
(2)  032752  012077  146676            MOV     (0)+,@KMAD6     ;WRITE DATA.
(2)  032756  052777  020000  146654        BIS     #20000,@KMAD0   ;SET CRAM WRITE
(2)  032764  005077  146650            CLR     @KMAD0          ;DISABLE CRAM.
(2)  032770  005277  146654            INC     @KMAD4          ;UPDATE CRAM ADDR.
(2)  032774  021027  177777            CMP     (0),#-1         ;ALL DONE?
(2)  033000  001361                    BNE     2$              ;NO LOOP.
(2)  033002  005077  146642            CLR     @KMAD4          ;CLEAR CRAM ADDR.
(2)  033006  016500  177776            MOV     -2(5),R0        ;GET MICRO CODE ADDR.
(2)
(2)  033012  052777  002000  146620  3$:     BIS     #2000,@KMAD0    ;SELECT CRAM
(2)  033020  022077  146630            CMP     (R0)+,@KMAD6    ;DATA OK?
(2)  033024  001013                    BNE     5$              ;NO - REPORT AN ERROR.
(2)  033026  021027  177777            CMP     (0),#-1         ;ALL DONE?
(2)  033032  001405                    BEQ     4$              ;YES - EXIT
(2)  033034  005077  146600            CLR     @KMAD0          ;NO - DESELECT CRAM.
(2)  033040  005277  146604            INC     @KMAD4          ;UPDATE CRAM ADDR.
(2)  033044  000762                    BR      3$
(2)
(2)  033046  012600            4$:     MOV     (SP)+,R0        ;RESTORE R0
(2)  033050  012604                    MOV     (SP)+,R4        ;RESTORE R4
(2)  033052  000205                    RTS     R5              ;EXIT
(2)
(2)  033054                    5$:                             ;COME HERE ON LOAD ERROR
(2)  033054  005745                    TST     -(5)
(2)  033056  105204                    INCB    R4              ;UPDATE ERROR COUNTER.
(2)  033060  100324                    BPL     1$              ;IF NOT TOO MANY, TRY AGAIN.
(2)  033062  000000                    HALT                    ;MICRO CODE LOAD ERROR.
(2)                                                             ;KMC-11 FAULT. YOU COULD TRY
(2)  033064  000722                    BR      1$              ;TO PRESS CONTINUE TO GIVE IT
(2)                                                             ;ANOTHER CHANCE, BUT I DOUBT
(2)                                                             ;THAT THAT WOULD WORK. SINCE I'VE
(2)                                                             ;ALREADY GIVEN IT 177 (OCTAL) CHANCES
(2)                                                             ;TRY RUNNING THE KMC-11 DIAGNOSTIC.
(2)
(2)
(2)                                     ;
(2)                                     ;*THIS ROUTINE ISSUES A WRITE COMMAND TO THE LPA-11
```

```
(2)                                                 ;*
(2)                                                 ;*      CALL =  JSR     R5,$TLKW
(2)                                                 ;*              .WORD   0               ;OFFSET OF DEVICE ADDR.
(2)                                                 ;*              .WORD   0               ;DATA TO BE WRITTEN
(2)                                                 ;*
(2)  033066  010046                         $TLKW:  MOV     R0,-(SP)        ;SAVE R0
(2)  033070  012500                                 MOV     (5)+,R0         ;GET DEVICE OFFSET
(2)  033072  052700  000340                         BIS     #340,R0         ;ADD WRITE CODE.
(2)  033076  004737  033350                         JSR     PC,$LPW         ;WAIT FOR FAST PATH READY
(2)  033102  010037  033174                         MOV     R0,W1
(2)  033106  010077  146536                         MOV     R0,@KMAD4
(2)  033112  112777  000005  146524                 MOVB    #5,@KMAD2       ;ISSUE FAST PATH WRITE
(2)  033120  004737  033350                         JSR     PC,$LPW         ;WAIT FOR RDY
(2)  033124  011537  033176                         MOV     (5),W2
(2)  033130  112577  146514                         MOVB    (5)+,@KMAD4     ;WRITE LOW BYTE DATA.
(2)
(2)  033134  112777  000005  146502                 MOVB    #5,@KMAD2       ;FP WRITE
(2)  033142  004737  033350                         JSR     PC,$LPW
(2)  033146  111537  033200                         MOVB    (5),W3
(2)  033152  112577  146472                         MOVB    (5)+,@KMAD4     ;WRITE HIGH BYTE
(2)  033156  112777  000005  146460                 MOVB    #5,@KMAD2
(2)  033164  004737  033350                         JSR     PC,$LPW
(2)  033170  012600                                 MOV     (SP)+,R0
(2)  033172  000205                                 RTS     R5              ;EXIT DONE.
(2)  033174  000000                         W1:     0
(2)  033176  000000                         W2:     0
(2)  033200  000000                         W3:     0
(2)
(2)                                                 ;*
(2)                                                 ;*THIS ROUTINE ISSUES A READ COMMAND TO THE LPA-11
(2)                                                 ;*
(2)                                                 ;*      CALL -  JSR     R5,$TLKR
(2)                                                 ;*              .WORD   0               ;OFFSET OF DEVICE
(2)                                                 ;*              ;RETURNS HERE
(2)                                                 ;*DATA IN WORD $DATR
(2)                                                 ;*
(2)
(2)  033202  010046                         $TLKR:  MOV     R0,-(SP)        ;SAVE R0
(2)  033204  012500                                 MOV     (5)+,R0         ;GET OFFSET
(2)  033206  052700  000300                         BIS     #300,R0         ;ADD READ CODE
(2)  033212  004737  033350                         JSR     PC,$LPW         ;WAIT TILL READY
(2)  033216  110077  146426                         MOVB    R0,@KMAD4
(2)  033222  112777  000005  146414                 MOVB    #5,@KMAD2       ;ISSUE WRITE FP
(2)  033230  004737  033350                         JSR     PC,$LPW
(2)  033234  010037  033344                         MOV     R0,RD1
(2)  033240                                 '$:
(3)  033240  004537  033636                         JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  033244  104000                                 ERROR                   ;/TIME-OUT ERROR
(3)                                                                         ;/WE FAILED TO COMPLETE
(3)                                                                         ;/CURRENT OPERATION.
(3)                                                                         ;/CONTINUES IN THIS LOOP
(3)                                                                         ;/WOULD MAKE US 'HANG' HERE
(3)
(3)  033246  000774                                 BR              '$
(3)
```

```
(3)                                                     ;/RETURNS HERE-FROM-TIMED OUT.
(2)  033250  032777  000040  146362          BIT     #BIT5,@KMAD0    ;FAST PATH GOT DATA?
(2)  033256  001370                          BNE     1$
(2)  033260  112777  000004  146356          MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
(2)  033266  004737  033350                  JSR     PC,$LPW
(2)  033272  117737  146352  033346          MOVB    @KMAD4,$DATR    ;GET LOW BYTE
(2)  033300                          2$:
(3)  033300  004537  033636                  JSR     R5, $TOUT       ;-TOUT-CHECK FOR TIMEOUT
(3)
(3)  033304  104000                          ERROR                   ;/TIME-OUT ERROR
(3)                                                                  ;/WE FAILED TO COMPLETE
(3)                                                                  ;/CURRENT OPERATION.
(3)                                                                  ;/CONTINUES IN THIS LOOP
(3)                                                                  ;/WOULD MAKE US "HANG" HERE
(3)
(3)  033306  000774                          BR              2$
(3)
(3)                                                     ;/RETURNS HERE-FROM-TIMED OUT.
(2)  033310  032777  000040  146322          BIT     #BIT5,@KMAD0    ;FAST PATH READY?
(2)  033316  001370                          BNE     2$
(2)  033320  112777  000004  146316          MOVB    #4,@KMAD2       ;ISSUE FAST PATH READ
(2)  033326  004737  033350                  JSR     PC,$LPW
(2)  033332  117737  146312  033347          MOVB    @KMAD4,$DATR+1  ;SAVE HIGH BYTE
(2)  033340  012600                          MOV     (SP)+,R0
(2)  033342  000205                          RTS     R5
(2)  033344  000000                  RD1:    0
(2)  033346  000000                  $DATR:  .WORD   0
(2)                                          ;
(2)                                          ;THIS ROUTINE WAITS FOR KMC-CODE TO BECOME READY AS WELL
(2)                                          ;AS FAST PATH TO BE READ.
(2)                                          ;
(2)                                          ;       CALL =  JSR     PC,$LPW
(2)                                          ;
(2)                                          ;IT WILL TIME OUT IF TOO MUCH TIME IS TAKEN BY
(2)                                          ;THE MICRO-PROCESSORS AND REPORT AN ERROR, THEN HALT.
(2)                                          ;
(2)
(2)  033350  010146                  $LPW:   MOV     R1,-(SP)        ;SAVE R1
(2)  033352  005001                          CLR     R1
(2)  033354  122777  000377  146262  1$:     CMPB    #377,@KMAD2     ;FINISHED INSTRUCTION?
(2)  033362  001403                          BEQ     2$
(2)  033364  005201                          INC     R1              ;TIME OUT?
(2)  033366  001372                          BNE     1$
(2)  033370  000411                          BR      10$
(2)
(2)  033372  032777  000020  146240  2$:     BIT     #BIT4,@KMAD0    ;FAST PATH READ?
(2)  033400  001403                          BEQ     3$
(2)  033402  005201                          INC     R1              ;NO - TIME OUT?
(2)  033404  001372                          BNE     2$
(2)  033406  000402                          BR      10$             ;YES - REPORT AN ERROR
(2)
(2)  033410  012601                  3$:     MOV     (SP)+,R1        ;RESTORE R1
(2)  033412  000207                          RTS     PC              ;EXIT
(2)
(2)  033414                          10$:
(3)  033414  104401  033422                  TYPE    ,65$            ;;TYPE ASCIZ STRING
```

```
(3)  033420  000407                        BR      64$             ;;GET OVER THE ASCIZ
(3)                                ;;65$:   .ASCIZ  <200>#LPA-11 FAULT#
(3)  033440                        64$:
(2)
(2)  033440  000000                11$:    HALT                    ;LPA-11 FAULT RUN LPA-11
(2)  033442  000776                        BR      11$             ;DIAGNOSTICS.
(2)
(2)
(2)
(2)                                        ;*
(2)                                        ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE TO
(2)                                        ;*A DEVICE ADDRESS ON THE I/O BUSS FOR WRITE ONLY.
(2)                                        ;*
(2)                                        ;* FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN USED
(2)                                        ;* BEFORE. IF NOT WE HAVE TO INITIALIZE THE LPA WITH
(2)                                        ;* THAT ADDRESS.
(2)                                        ;* WHEN THE ADDR. IS KNOWN BY THE LPA, DO THE OUTPUT BY
(2)                                        ;* $TLKW
(2)                                        ;*
(2)
(2)  033444  010046                $OUTLP: MOV     R0,-(SP)        ;SAVE R0
(2)  033446  010146                        MOV     R1,-(SP)        ;SAVE R1
(2)
(2)  033450  012700  001666                MOV     #.DVLS,R0       ;PROGRAM DEFINED LIST.
(2)  033454  005001                        CLR     R1
(2)  033456  005710                1$:     TST     (0)             ;TERMINATOR REACHED?
(2)  033460  001421                        BEQ     10$             ;YES NEXT STEP.
(2)  033462  027520  000000                CMP     @(5),(0)+       ;MATCH WITH ADDR IN LIST?
(2)  033466  001402                        BEQ     2$
(2)  033470  005201                        INC     R1
(2)  033472  000771                        BR      1$
(2)
(2)  033474  010137  033512        2$:     MOV     R1,3$           ;SAVE OFFSET, DEVICE KNOWN.
(2)  033500  005725                        TST     (5)+
(2)  033502  013537  033514                MOV     @(5)+,4$        ;GET DATA TO BE WRITTEN
(2)  033506  004537  033066                JSR     R5,$TLKW        ;DO WRITE
(2)  033512  000000                3$:     .WORD   0               ;DEVICE OFFSET
(2)  033514  000000                4$:     .WORD   0               ;DATA TO BE WRITTEN.
(2)  033516  012601                        MOV     (SP)+,R1
(2)  033520  012600                        MOV     (SP)+,R0
(2)  033522  000205                        RTS     R5
(2)  033524  017520  000000        10$:    MOV     @(5),(0)+       ;SAVE ADDR.
(2)  033530  005010                        CLR     (0)
(2)  033532  004537  032046                JSR     R5,$LPAI
(2)  033536  001666                        .WORD   .DVLS
(2)  033540  000755                        BR      2$
(2)
(2)                                        ;*
(2)                                        ;*THIS ROUTINE PROVIDES THE LINKAGE FROM USER CODE
(2)                                        ;*TO A DEVICE ADDR. ON THE I/O BUSS FOR READ ONLY.
(2)                                        ;*
(2)                                        ;*FIRST WE WILL DETERMINE IF THE ADDRESS HAS BEEN
(2)                                        ;*USED BEFORE. IF NOT, WE HAVE TO INITIALIZE THE LPA
(2)                                        ;*WITH THE NEW ADDR.
(2)                                        ;*WHEN THE ADDR IS KNOWN WE CAN DO OUTPUT THROUGH
(2)                                        ;*$TLKR
```

```
(2)                                         ;*      CALL THROUGH    MOVEI   DATA,ADDR.
(2)                                         ;*              WHICH EQUALS:
(2)                                         ;*                      JSR     R5,$INLP
(2)                                         ;*                      .WORD   XX      ADDR OF DEVICE
(2)                                         ;*                      .WORD   YY      ADDR TO STORE READ DATA.
(2)
(2)  033542  010046          $INLP:  MOV     R0,-(SP)        ;SAVE R0
(2)  033544  010146                  MOV     R1,-(SP)        ;SAVE R1
(2)
(2)  033546  012700  001666          MOV     #.DVLS,R0       ;PROG DEFINED ADDR. LIST.
(2)  033552  005001                  CLR     R1
(2)  033554  005710          1$:     TST     (0)             ;EOL REACHED?
(2)  033556  001420                  BEQ     10$             ;YES - DEFINE NEW ADDR.
(2)
(2)  033560  027520  000000          CMP     @(5),(0)+       ;ADDR. MATCH?
(2)  033564  001402                  BEQ     2$
(2)  033566  005201                  INC     R1
(2)  033570  000771                  BR      1$
(2)
(2)  033572  010137  033604  2$:     MOV     R1,3$           ;SAVE LIST OFFSET
(2)  033576  005725                  TST     (5)+
(2)  033600  004537  033202          JSR     R5,$TLKR        ;GO READ DEVICE
(2)          033604          $OFS=.
(2)  033604  000000          3$:     .WORD   0               ;OFFSET OF DEVICE
(2)
(2)  033606  013735  033346          MOV     $DATR,@(5)+     ;STORE DATA.
(2)  033612  012601                  MOV     (SP)+,R1        ;RESTORE R1
(2)  033614  012600                  MOV     (SP)+,R0        ;RESTORE R2
(2)  033616  000205                  RTS     R5              ;EXIT
(2)
(2)  033620  017520  000000  10$:     MOV     @(5),(0)+
(2)  033624  005010                  CLR     (0)
(2)  033626  004537  032046          JSR     R5,$LPAI
(2)  033632  001666                  .WORD   .DVLS
(2)  033634  000756                  BR      2$
(2)                                         ;*
(2)                                         ;*$TOUT ROUTINE USED TO WATCH IF
(2)                                         ;*      WE'RE IN A LOOP TOO-LONG
(2)                                         ;*      CALL=   JSR R5, $TOUT
(2)                                         ;*              ERROR X     ;RETURNS HERE ON TIMEOUT
(2)                                         ;*              BR
(2)                                         ;*              ;RETURNS HERE NO ERROR
(2)                                         ;*
(2)
(2)  033636  020537  033672  $TOUT:  CMP     R5,$SAD         ;SAME ADDR?
(2)  033642  001405                  BEQ     1$
(2)  033644  010537  033672          MOV     R5,$SAD         ;NO-SAVE THIS ADDR.
(2)  033650  005037  033674          CLR     $CNT            ;CLR CNT AT ADDR.
(2)  033654  000403                  BR      2$
(2)  033656  005237  033674  1$:     INC     $CNT            ;OVERFLOW?
(2)  033662  100402                  BMI     3$              ;YES-ERROR RETURN
(2)  033664  062705  000004  2$:     ADD     #4,R5           ;NO-NON ERROR RETURN
(2)  033670  000205          3$:     RTS     R5              ;RETURN.
(2)
(2)  033672  000000          $SAD:   .WORD   0               ;CONTAINS LOOP ADDR.
(2)  033674  000000          $CNT:   .WORD   0               ;# OF TIMES AT ADDR.
```

```
(2)
(2)                                     ;*
(2)                                     ;* THIS ROUTINE REPLACES WHAT THE USER WOULD ORDINARILY
(2)                                     ;*USE FOR A RESET.  FIRST,WE DO A RESET INSTRUCTION.
(2)                                     ;*THEN WE CLR ''.DVLST'' WHICH FORCES US TO RESET BOTH THE
(2)                                     ;*KMC AND DMC AS SOON AS A DEVICE IS REFERENCED.
(2)                                     ;*
(2)                                     ;*      CALL=JSR        PC,$RESET       ;REPLACES 'RESET INSTRUCTION
(2)                                     ;*              ;RETURNS HERE.
(2)                                     ;*
(2)  033676  000005             $RESET: RESET                   ;RESET THE WORLD.
(3)
(3)                                     ;*      MOV     @2$,1$  ;/READ DEVICE REG 2$,PUT DATA IN 1$.
(2)  033710  005737  032724             TST     $AERR           ;IF NO ERROR,LOOP
(2)  033714  001004                     BNE     10$             ;THERE WAS AN ERROR.
(2)  033716  062737  000002  033732     ADD     #2,2$           ;UPDATE DEVICE ADDR.
(2)                                                             ;YOU SEE ,WE HAVE TO PROTECT OUR SELF!
(2)                                                             ;IF 2$ CONTAINED A VALID ADDR,WE
(2)                                                             ;MUST KEEP TRYING UNTIL WE GENERATE
(2)                                                             ;AN INVALID ADDR.
(2)  033724  000764                     BR      $RESET
(2)  033726                     10$:
(2)  033726  000207                     RTS     PC
(2)  033730  000000             1$:     .WORD   0               ;JUNK LOC.
(2)  033732  160000             2$:     .WORD   160000          ;DUMB ADDR. FORCES INIT OF DMC/KMC.
(2)
(2)
(2)                                     ;
(2)                                     ;SDELAY- ROUTINE TO GIVE A MINOR DELAY.
(2)                                     ;        IS NOT TIME DEPENDENT CODE SENCE
(2)                                     ;        NOT USED TO GET SPECIFIC TIME BUT
(2)                                     ;        JUST A LITTLE DELAY.
(2)
(2)                                     ;       THAT IS UNLESS A REAL TIME CLOCK IS PRESENT!
(2)                                     ;       THEN WE'LL GENERATE A TIME BETWEEN 16MS TO 32 MS
(2)                                     ;
(2)                                     ;
(2)                                     ;       CALL -   JSR PC,  SDELAY
(2)                                     ;
(2)  033734                     SDELAY:
(2)  033734  005737  034016             TST     RTCCSR          ;CLOCK PRESENT?
(2)  033740  100016                     BPL     10$
(2)  033742  012737  000002  034006     MOV     #2,TIME
(2)  033750  052777  000115  000040     BIS     #115,@RTCCSR    ;START CLOCK
(2)  033756  005037  177776             CLR     PS
(2)  033762  005737  034006     1$:     TST     TIME
(2)  033766  001375                     BNE     1$
(2)  033770  005077  000022             CLR     @RTCCSR         ;STOP CLOCK
(2)
(2)  033774  000207                     RTS PC
(2)  033776  105237  034006     10$:    INCB    TIME
(2)  034002  001375                     BNE     10$
(2)  034004  000207                     RTS     PC
(2)
(2)  034006  000000             TIME:   .WORD   0
(2)
```

```
(2)  034010  005337  034006          CLKINT: DEC     TIME
(2)  034014  000002                          RTI
(2)  034016  000000                  RTCCSR: .WORD   0               ;CLOCK CSR IF USED.
(2)                                          ;
(2)                                          ;*
(2)                                          ;*THIS MACRO ALLOWS THE OPERATOR TO TALK TO
(2)                                          ;*ANY DEVICE ON THE I/O BUS
(2)                                          ;*USER MUST START AT THIS ADDR.
(2)                                          ;*HE MUST SAY EITHER 'E'' FOR EXAMINE, OR 'D'' FOR DEPOSIT.
(2)                                          ;*'E'' IS DEFAULT.
(2)                                          ;*NEXT, HE MUST SUPPLY AN ADDR.
(2)                                          ;*NOTE IF ADDR. IS NOT FOUND ON I/O BUS, A HALT
(2)                                          ;*WILL OCCUR.
(2)
(2)  034020                          $UTK:
(2)  034020  005037  001666                  CLR     .DVLS
(2)  034024                          21$:
(3)  034024  104401  034032                  TYPE    ,65$            ;;TYPE ASCIZ STRING
(3)  034030  000405                          BR      64$             ;;GET OVER THE ASCIZ
(3)                                  ;;65$:  .ASCIZ  <200>#E OR D?#
(3)  034044                          64$:
(2)  034044  105777  145074          1$:     TSTB    @$TKS
(2)  034050  100375                          BPL     1$
(2)  034052  117737  145070  034174          MOVB    @$TKB,20$       ;GET INPUT
(2)  034060  104401  034174                  TYPE,   20$             ;ECHO, NEXT MESSAGE.
(2)  034064  142737  000240  034174          BICB    #240,20$        ;STRIP PARITY, LC
(2)  034072  104412                          RDOCT                   ;GET ADDR.
(2)  034074  012637  034172                  MOV     (SP)+,14$
(2)  034100  123727  034174  000104          CMPB    20$,#'D         ;DEPOSIT?
(2)  034106  001411                          BEQ     10$
(2)
(2)  034110  004537  033542                  JSR     R5,$INLP        ;GET DATA
(2)  034114  034172                  2$:     .WORD   14$
(2)  034116  034130                          .WORD   5$
(2)
(3)  034120  013746  034130                  MOV     5$,-(SP)        ;;SAVE 5$ FOR TYPEOUT
(3)  034124  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(2)  034126  000736                          BR      21$             ;LOOP.
(2)  034130  000000                  5$:     .WORD   0
(2)
(2)  034132                          10$:
(3)  034132  104401  034140                  TYPE    ,67$            ;;TYPE ASCIZ STRING
(3)  034136  000404                          BR      66$             ;;GET OVER THE ASCIZ
(3)                                  ;;67$:  .ASCIZ  <200>#DATA= #
(3)  034150                          66$:
(2)  034150  104412                          RDOCT
(2)  034152  012637  034170                  MOV     (SP)+,13$
(2)
(2)  034156  004537  033444          11$:    JSR     R5,$OUTLP       ;OUTPUT ROUTINE.
(2)  034162  034172                  12$:    .WORD   14$             ;DEVICE ADDR.
(2)  034164  034170                          .WORD   13$             ;DATA
(2)  034166  000716                          BR      21$
(2)
(2)  034170  000000                  13$:    .WORD   0
(2)  034172  000000                  14$:    .WORD   0
(2)  034174  100001  042504  044526  20$:    .ASCIZ  <1><200>#DEVICE ADDR- #
```

```
   (2)  034202  042503  040440  042104
   (2)  034210  036522  000040
   (2)                                  .EVEN
   (2)
   (1)
   (1)
   (2)                                  ;
   (2)                                  ;THIS ROUTINE LOOKS THROUGH CURENT .DVLS FOR A/D ADDR.
   (2)                                  ;IF UNFOUND,GENERATES IT. THIS ROUTINE'S WHOLE PURPOSE IS
   (2)                                  ;TO SET UP THE USER PROGRAM TO LINK TO FILE ''DRLPX2'' FOR
   (2)                                  ;SAMPLE TAKEING PURPOSES.
   (2)                                  ;       TO TAKE SAMPLES, THE USER PROGRAM MUST SET UP
   (2)                                  ;       A/D CSR IN BSEL 4,AND 5.
   (2)                                  ;       (2) HE MUST CALL THIS ROUTINE:
   (2)                                  ;               JSR     R5,$PUTS        ;CALL SET UP ROUTINE.
   (2)                                  ;               .WORD   ADCSR           ;ADDR. OF A/D CSR.
   (2)                                  ;               ;RETURNS HERE ;KMC BSEL 3,6,7 PERMINENTLY SET UP
   (2)                                  ;                               ;(UNTILL ONE DOES A RESET)
   (2)                                  ;
   (2)                                  ;       (3)THE USER MUST PUT CODE 006 INTO KMC REG 2 TO
   (2)                                  ;          START CONVERSION  CAUTION*DO WITH MOVB INSTR..
   (2)                                  ;       (4)MONITOR KMC REG 2 FOR CODE 377 (DRLPX2 IS DONE)
   (2)                                  ;       (5)READ KMC REG 4,5 FOR A/D RESULT.
   (2)                                  ;       (6) TO TAKE MORE SAMPLES,SIMPLY PUT A/D CSR INTO
   (2)                                  ;          BSEL 4,5 AND CODE 6 INTO BSEL 2.
   (2)                                  ;
   (2)  034214  012537  034224  $PUTS:  MOV     (5)+,1$                 ;GET ADDR OF ADDR. OF A/D
   (2)  034220  004537  033542          JSR     R5,$INLP
   (2)  034224  000000          1$:     .WORD   0
   (2)  034226  034322                  .WORD   10$
   (2)  034230  113777  033604  145416  MOVB    $OFS,@KMAD6
   (2)  034236  113777  033604  145412  MOVB    $OFS,@KMAD7
   (2)  034244  013737  034224  034264  MOV     1$,2$
   (2)  034252  062737  000002  034264  ADD     #2,2$
   (2)  034260  004537  033542          JSR     R5,$INLP
   (2)  034264  000000          2$:     .WORD   0
   (2)  034266  034322                  .WORD   10$
   (2)  034270  113777  033604  145350  MOVB    $OFS,@KMAD3
   (2)  034276  152777  000340  145350  BISB    #340,@KMAD6
   (2)  034304  152777  000300  145344  BISB    #300,@KMAD7
   (2)  034312  152777  000300  145326  BISB    #300,@KMAD3
   (2)  034320  000205                  RTS     R5
   (2)  034322  000000          10$:    .WORD   0
   (2)
5586
5587            000001                  .END
```

```
ABASE = 170404          2995#  3061
ABR     001330          3061#  3246*  3302   3332   3333   3360   3361   3562   3581   3609   3637   3824   3876
                        3934   3986   4065   4066   4067   4068   4069   4070   4088   4134   4140   4175   4215
                        4332   4333   4377   4378   4402   4456   4469   4519   4868   4932   5042   5043   5044
                        5045   5304   5399   5568
ACDW1 = 000000          3061
ACDW2 = 000000          3061
ACPUOP= 000000          3061
ACR     001332          3061#  3248*  3302   3585   3610   3638   3836   3893   4003   4065   4066   4067   4068
                        4069   4070   4096   4144   4221   4377   4378   4406   4483   4526   4771   4880   4939
                        5042   5043   5044   5045   5311   5569   5576   5577
ADDW0 = 000000          3061
ADDW1 = 000000          3061
ADDW10= 000000          3061
ADDW11= 000000          3061
ADDW12= 000000          3061
ADDW13= 000000          3061
ADDW14= 000000          3061
ADDW15= 000000          3061
ADDW2 = 000000          3061
ADDW3 = 000000          3061
ADDW4 = 000000          3061
ADDW5 = 000000          3061
ADDW6 = 000000          3061
ADDW7 = 000000          3061
ADDW8 = 000000          3061
ADDW9 = 000000          3061
ADEVCT= 000000          3061
ADEVM = 000000          3061
AENV  = 000000          3061
AENVM = 000000          3061
AFATAL= 000000          3061
AMADR1= 000000          3061
AMADR2= 000000          3061
AMADR3= 000000          3061
AMADR4= 000000          3061
AMAMS1= 000000          3061
AMAMS2= 000000          3061
AMAMS3= 000000          3061
AMAMS4= 000000          3061
AMSGAD= 000000          3061
AMSGLG= 000000          3061
AMSGTY= 000000          3061
AMTYP1= 000000          3061
AMTYP2= 000000          3061
AMTYP3= 000000          3061
AMTYP4= 000000          3061
APASS = 000000          3061
APRIOR= 000006          2997#  3061
APRITY  001352          3061#  3264
APTCSU= 000040          5528   5529#
APTENV= 000001          5522   5528   5529#
APTSIZ= 000200          3228   5529#
APTSPO= 000100          5528   5529#
ASR     001326          3061#  3234*  3244   3302   3393   3394   3404   3562   3580   3604   3628   3755   3756
                        3760   3763   3788   3789   3791   3792   3804   3823   3830   3831   3833   3875   3883
```

```
                        3886    3888    3932    3938    3939    3941    3944    3954    3978    3988    3989    3991    3994
                        4065    4066    4067    4068    4069    4070    4087    4090    4100    4133    4136    4173    4177
                        4178    4180    4181    4213    4217    4218    4220    4258    4265    4266    4268    4270    4332
                        4333    4377    4378    4400    4403    4405    4454    4460    4461    4463    4515    4521    4529
                        4538    4575    4577    4618    4639    4641    4689    4697    4699    4733    4760    4764    4832
                        4833    4834    4835    4836    4837    4866    4876    4877    4879    4930    4933    4935    5042
                        5043    5044    5045    5146    5147    5148    5149    5242    5250    5253    5256    5293    5306
                        5307    5309    5348    5351    5353    5364    5396    5403    5406    5416    5449    5454    5471
                        5567    5574
ASWREG= 000000          3061
ATESTN= 000000          3061
AUNIT = 000000          3061
AUSWR = 000000          3061
AVECP2  001344          3061#   3258*
AVECT   001342          3061#   3256
AVECT1= 000344          2996#   3061    3223
AVECT2= 000000          3061
BBR     001336          3061#   3252*   3302    3444    3445    3562    3666    3693    3721    4568    4627    4692    4736
                        4762    4832    4833    4834    4835    4836    4837    4870    5146    5147    5148    5149    5452
                        5571
BCR     001340          3061#   3254*   3302    3670    3696    3724    4578    4645    4709    4742    4832    4833    4834
                        4835    4836    4837    5146    5147    5148    5149    5572    5578    5579
BIT0  = 000001          2993#   3562    4573    4929
BIT00 = 000001          2993#   3793    3995    4182
BIT01 = 000002          2993#
BIT02 = 000004          2993#
BIT03 = 000010          2993#
BIT04 = 000020          2993#
BIT05 = 000040          2993#   3945    4065    4066    4067    4068    4069    4070    4101    5407
BIT06 = 000100          2993#
BIT07 = 000200          2993#   3955    4271    5462
BIT08 = 000400          2993#   5525
BIT09 = 001000          2993#   5522    5525
BIT1  = 000002          2993#   3562    4573    5585
BIT10 = 002000          2993#   3562    4267    4332    4333    4377    4378    4404    5252    5522
BIT11 = 004000          2993#   3562    4516    4563    4576    4620    4640    4698    4878    5042    5043    5044    5045
                        5146    5147    5148    5149    5525    5585
BIT12 - 010000          2993#   3562    3759    3790    3832    3887    3940    3990    4179    4219    4462    4934    5308
                        5352
BIT13 - 020000          2993#   3562    3787    5266    5320    5372    5425    5480    5522
BIT14 = 040000          2993#   3562    3907    4659    5525    5585
BIT15 = 100000          2993#   3562    5257    5585
BIT2  = 000004          2993#   3562
BIT3  = 000010          2993#   3562
BIT4  - 000020          2993#   3562    5585
BIT5  = 000040          2993#   3562    4573    5146    5147    5148    5149    5585
BIT6  = 000100          2993#   3562
BIT7  = 000200          2993#   3562
BIT8  = 000400          2993#   3562    5305
BIT9  = 001000          2993#   3562    5305    5352
BPRITY  001354          3061#   3264*
BPTVEC= 000014          2993#
BSR     001334          3061#   3250*   3302    3419    3420    3562    3665    3686    3714    4130    4254    4332    4333
                        4453    4458    4497    4517    4564    4572    4574    4621    4632    4634    4690    4694    4696
                        4700    4734    4761    4832    4833    4834    4835    4836    4837    4867    4872    4874    5042
                        5043    5044    5045    5146    5147    5148    5149    5242    5251    5293    5348    5396    5449
```

```
                        5460    5461    5570    5575
BVECT   001346          3061#   3260*
BVECT2  001350          3061#   3262*
CKSWR = 104407          5522    5525    5531#
CLKINT  034010          5585#
CR    = 000015          2993#   5528
CRLF  = 000200          2993#   5528
DDISP = 177570          2993#   3061    3228
DF0     032042          3072    3079    3086    3093    3100    3107    3114    3122    3129    3136    3140    3141    3142
                        3143    3150    3157    3164    3171    3178    3185    3192    3199    3206    3213    3220    5582#
DH1     031025          3070    3077    5551#
DH10    031311          3119    5557#
DH12    031450          3134    3162    5559#
DH14    031467          3148    3169    5560#
DH20    031505          3176    5561#
DH23    031547          3197    5562#
DH26    031611          3218    5563#
DH3     031063          3084    5552#
DH4     031121          3091    3127    3183    3190    5553#
DH5     031157          3098    5554#
DH6     031215          3105    5555#
DH7     031253          3112    3155    3204    3211    5556#
DISPLA  001142          3061#   3228*   5522*   5525*
DISPRE  000174          2982#   3228
DRLPX2= ****** G          52#   5585
DSWR  = 177570          2993#   3061    3228
DT1     031640          3071    3078    3177    3198    5567#
DT10    031734          3121    5573#
DT12    031750          3135    3163    5574#
DT14    031756          3149    3170    5575#
DT21    031764          3184    5576#
DT22    031776          3191    5577#
DT24    032010          3205    5578#
DT25    032022          3212    5579#
DT26    032034          3219    5580#
DT3     031652          3085    5568#
DT4     031664          3092    3128    5569#
DT5     031676          3099    5570#
DT6     031710          3106    5571#
DT7     031722          3113    3156    5572#
EMTVEC= 000030          2993#   3228*
EM1     030140          3069    5533#
EM10    030405          3118    5540#
EM11    030432          3126    3182    3189    5541#
EM12    030461          3133    5542#
EM14    030521          3147    5543#
EM15    030561          3154    3203    3210    5544#
EM16    030610          3161    5545#
EM17    030643          3168    5546#
EM2     030173          3076    5534#
EM20    030676          3175    5547#
EM23    030735          3196    5548#
EM26    030774          3217    5549#
EM3     030222          3083    5535#
EM4     030251          3090    5536#
EM5     030300          3097    5537#
```

```
EM6      030327            3104    5538#
EM7      030356            3111    5539#
ERRVEC=  000004            2993#   3228*   5525*
GNS    = ****** U          2982    3232    5266    5320    5372    5425    5480    5490    5494    5531    5532    5585
GTSWR  = 104406            5531#
HT     = 000011            2993#   5528
IOTRD    024640            5487#   5532
IOTT   = 104413            5532#
IOTVEC=  000020            2993#   3228*
KMAD0    001640            3223#   3238    5585*
KMAD1    001642            3223#   3238
KMAD2    001644            3223#   5585*
KMAD3    001646            3223#   5585*
KMAD4    001650            3223#   5585*
KMAD5    001652            3223#
KMAD6    001654            3223#   5585*
KMAD7    001656            3223#   3238    5585*
LF     = 000012            2993#   5528
LOOP     002310            3240#   5179
LPADH    001652            3223#
LPADL    001650            3223#
LPCI     001640            3223#
LPCO     001644            3223#
LPMR     001642            3223#
LPMS1    001654            3223#
LPMS2    001656            3223#
LPSO     001646            3223#
LS210    023410            2987    5241#
LS214    023606            2988    5292#
LS220    024022            2989    5347#
LS224    024206            2990    5395#
LS230    024414            2991    5448#
P      = 000012            3562#   4332#   4333#
PIRQ   - 177772            2993#
PIRQVE=  000240            2993#
PR0    = 000000            2993#
PR1    = 000040            2993#
PR2    - 000100            2993#
PR3    = 000140            2993#
PR4    - 000200            2993#
PR5    - 000240            2993#
PR6    = 000300            2993#
PR7    = 000340            2993#
PS     - 177776            2993#   5585*
PSW    - 177776            2993#
PWRVEC=  000024            2993#   3228*   5530*
RDCHR    104410            5527    5531#
RDLIN  - 104411            5526    5531#
RDOCT  = 104412            5531#   5585
RD1      033344            5585#*
RESVEC=  000010            2993#
RSTART   002436            2986    3266#
RTCCSR   034016            5585#*
SDELAY   033734            5585#
SSTART   002442            3265    3267#
STACK    001100            2993#   3228    3267
```

```
START   001730          2984    3228#
STKLMT= 177774          2993#
SWR     001140          3061#   3228*   3907    4659    5266    5320    5372    5425    5480    5522    5525    5527*   5530*
SWREG   000176          2982#   3228    5527
SW0   = 000001          2993#
SW00  = 000001          2993#
SW01  = 000002          2993#
SW02  = 000004          2993#
SW03  = 000010          2993#
SW04  = 000020          2993#
SW05  = 000040          2993#
SW06  = 000100          2993#
SW07  = 000200          2993#
SW08  = 000400          2993#
SW09  = 001000          2993#
SW1   - 000002          2993#
SW10  = 002000          2993#
SW11  = 004000          2993#
SW12  - 010000          2993#
SW13  - 020000          2993#
SW14  - 040000          2993#
SW15  = 100000          2993#
SW2   = 000004          2993#
SW3   - 000010          2993#
SW4   - 000020          2993#
SW5   - 000040          2993#
SW6   - 000100          2993#
SW7   - 000200          2993#
SW8   = 000400          2993#
SW9   - 001000          2993#
TBITVE= 000014          2993#
TIME    034006          5585#*
TKVEC = 000060          2993#
TPVEC = 000064          2993#
TRAPVE= 000034          2993#   3228*
TRTVEC= 000014          2993#
TST1    002460          3302#
TST10   003210          3562#
TST100  012710          4065#
TST101  013030          4066#
TST102  013150          4067#
TST103  013270          4068#
TST104  013410          4069#
TST105  013530          4070#
TST106  013650          4081#
TST107  013766          4123#
TST11   003316          3562#
TST110  014102          4163#
TST111  014224          4202#
TST112  014344          4251#
TST113  014464          4332#
TST114  014644          4333#
TST115  015024          4377#
TST116  015144          4378#
TST117  015264          4391#
TST12   003424          3562#
```

TST120	015370	4435#
TST121	015606	4508#
TST122	015762	4560#
TST123	016112	4611#
TST124	016326	4677#
TST125	016516	4727#
TST126	016620	4752#
TST127	016726	4832#
TST13	003532	3562#
TST130	017054	4833#
TST131	017202	4834#
TST132	017330	4835#
TST133	017456	4836#
TST134	017604	4837#
TST135	017732	4848#
TST136	020124	4920#
TST137	020236	5042#
TST14	003640	3562#
TST140	020534	5043#
TST141	021032	5044#
TST142	021330	5045#
TST143	021626	5146#
TST144	022140	5147#
TST145	022452	5148#
TST146	022764	5149#
TST15	003746	3562#
TST16	004054	3562#
TST17	004162	3562#
TST2	002572	3326#
TST20	004270	3562#
TST21	004376	3562#
TST22	004504	3562#
TST23	004612	3562#
TST24	004720	3562#
TST25	005026	3562#
TST26	005134	3562#
TST27	005242	3562#
TST3	002640	3355#
TST30	005350	3562#
TST31	005456	3562#
TST32	005564	3562#
TST33	005672	3562#
TST34	006000	3562#
TST35	006106	3562#
TST36	006214	3562#
TST37	006322	3562#
TST4	002704	3388#
TST40	006430	3562#
TST41	006536	3562#
TST42	006644	3562#
TST43	006752	3562#
TST44	007060	3562#
TST45	007166	3562#
TST46	007274	3562#
TST47	007402	3562#
TST5	002766	3414#

```
TST50   007510            3562#
TST51   007616            3562#
TST52   007724            3562#
TST53   010032            3562#
TST54   010140            3562#
TST55   010246            3562#
TST56   010354            3562#
TST57   010462            3562#
TST6    003034            3439#
TST60   010570            3562#
TST61   010676            3562#
TST62   011004            3562#
TST63   011112            3562#
TST64   011220            3573#
TST65   011300            3596#
TST66   011364            3624#
TST67   011450            3658#
TST7    003102            3562#
TST70   011530            3682#
TST71   011614            3710#
TST72   011700            3752#
TST73   011764            3782#
TST74   012070            3818#
TST75   012212            3867#
TST76   012374            3928#
TST77   012542            3974#
TYPDS = 104405            5531#
TYPE  - 104401            3232    5179    5266    5320    5372    5425    5480    5490    5494    5521    5522    5523    5524
                          5527    5528    5530    5531#   5585
TYPOC = 104402            5492    5500    5523    5527    5531#   5585
TYPON = 104404            5531#
TYPOS = 104403            5531#
VECTOR  001660            3223#
VECTPS  001662            3223#
VERSN   001664            3223#
W1      033174            5585#*
W2      033176            5585#*
W3      033200            5585#*
$AERR   032724            5585#*
$APTHD  001000            3059#
$ASTAT- ****** U          5529
$ATYC   027454            5529#
$ATY1   027430            5529#
$ATY3   027436            5528    5529#
$ATY4   027446            5522    5529#
$AUTOB  001134            3061#   5527
$BASE   001256            3061#   3234
$BDADR  001122            3061#   5573
$BDDAT  001126            3061#   3302    3333    3334    3361    3362    3394    3395    3420    3421    3445    3446    3562
                          3585    3586    3610    3612    3638    3640    3670    3671    3696    3698    3724    3726    3763
                          3764    3792    3793    3836    3837    3893    3895    3944    3945    3954    3955    4003    4004
                          4065    4066    4067    4068    4069    4070    4096    4097    4100    4101    4130    4131    4144
                          4145    4181    4182    4221    4222    4254    4255    4270    4271    4332    4333    4377    4378
                          4406    4407    4469    4470    4483    4484    4526    4527    4529    4530    4578    4579    4645
                          4647    4700    4701    4709    4710    4742    4743    4771    4772    4832    4833    4834    4835
                          4836    4837    4880    4881    4939    4940    5042    5043    5044    5045    5146    5147    5148
```

```
                        5149    5311    5312    5364    5365    5416    5417    5471    5472    5567    5568    5569    5570
                        5571    5572    5573    5577    5579
$BELL   001172          3061#   5522
$CDW1   001262          3061#
$CDW2   001264          3061#
$CHARC  027424          5528#*
$CKSWR  026372          5527#   5531
$CMTAG  001100          3061#   3228
$CM1  = 000001          3061#
$CM2  = 000002          3061#
$CM3  = 000001          3061#
$CM4  = 000001          3061#
$CNT    033674          5585#*
$CNTLG  027117          5527#
$CNTLU  027112          5527#
$CPUOP  001230          3061#
$CRLF   001177          3061#   5522    5523    5527    5528
$DATR   033346          5585#*
$DBLK   026000          5524#
$DDW0   001266          3061#
$DDW1   001270          3061#
$DDW10  001312          3061#
$DDW11  001314          3061#
$DDW12  001316          3061#
$DDW13  001320          3061#
$DDW14  001322          3061#
$DDW15  001324          3061#
$DDW2   001272          3061#
$DDW3   001274          3061#
$DDW4   001276          3061#
$DDW5   001300          3061#
$DDW6   001302          3061#
$DDW7   001304          3061#
$DDW8   001306          3061#
$DDW9   001310          3061#
$DEVCT  001212          3061#   3235*
$DEVM   001260          3061#
$DOAGN  023366          5179#
$DTBL   025770          5524#
$ENDAD  023356          3057    5179#
$ENDCT  023332          5179#
$ENDMG  023375          5179#
$ENULL  023372          5179#
$ENV    001222          3061#   5522    5528    5529
$ENVM   001223          3061#   3228    5528    5529
$EOP    023276          5179#
$EOPCT  023324          5179#
$ERFLG  001103          3061#   5522*   5525*
$ERMAX  001115          3061#   3228*   5525*
$ERROR  025242          3228    5522#
$ERRPC  001116          3061#   5522*   5523    5567    5568    5569    5570    5571    5572    5573    5574    5575    5576
                        5577    5578    5579    5580
$ERRTB  001360          3061#   5523
$ERRTY  025430          5522    5523#
$ERTTL  001112          3061#   5522*
$ESCAP  001170          3061#   3228*   5522    5525*
```

```
$ETABL  001222  3061#
$ETEND  001326  3059  3061#
$FATAL  001204  3061#  5529*
$FFLG   027674  5529#*
$FILLC  001156  3061#  5528
$FILLS  001155  3061#  5528
$GDADR  001120  3061#  5573
$GDDAT  001124  3061#  3331*  3332  3357*  3360  3390*  3393  3403*  3404  3416*  3419  3441*  3444
                3562*  3579*  3580  3581  3583*  3603*  3604  3606*  3609  3612  3627*  3628  3634*
                3637  3640  3664*  3665  3666  3668*  3685*  3686  3692*  3693  3698  3713*  3714
                3720*  3721  3726  3754*  3755  3756  3759*  3760  3787*  3788  3789  3790*  3791
                3803*  3804  3822*  3823  3824  3829*  3830  3831  3832*  3833  3834*  3837  3869*
                3876  3890*  3895  3904  3909*  3931*  3932  3933*  3934  3937*  3938  3939  3940*
                3941  3977*  3978  3985*  3986  3987*  3988  3989  3990*  3991  4065*  4066*  4067*
                4068*  4069*  4070*  4086*  4087  4088  4089*  4090  4132*  4133  4134  4135*  4136
                4139*  4140  4172*  4173  4174*  4175  4176*  4177  4212*  4213  4214*  4215  4216*
                4217  4218  4219*  4220  4257*  4258  4264*  4265  4266  4267*  4268  4332*  4333*
                4377*  4378*  4399*  4400  4401*  4402  4403  4404*  4405  4452*  4453  4454  4455*
                4456  4457*  4458  4459*  4460  4461  4462*  4463  4467*  4496*  4497  4514*  4515
                4516*  4517  4518*  4519  4520*  4521  4537*  4538  4563*  4564  4567*  4568  4572
                4573*  4574  4575  4576*  4577  4613*  4627  4642*  4647  4656  4661*  4688*  4689
                4690  4691*  4692  4693*  4694  4695*  4696  4697  4698*  4699  4732*  4733  4734
                4735*  4736  4740*  4759*  4760  4761  4762  4763*  4764  4832*  4833*  4834*  4835*
                4836*  4837*  4865*  4866  4867  4868  4869*  4870  4871*  4872  4873*  4874  4875*
                4876  4877  4878*  4879  4931*  4932  4933  4934*  4935  4937*  4940  5042*  5043*
                5044*  5045*  5146*  5147*  5148*  5149*  5567  5568  5569  5570  5571  5572  5573
                5576  5578
$GET42  023346  5179#
$GTSWR  026442  5527#  5531
$HD  =  000001  2980
$HIBTS  001000  3059#
$HIOCT  026370  5526#*
$ICNT   001104  3061#  5242*  5266*  5293*  5320*  5348*  5372*  5396*  5425*  5449*  5480*  5525*
$ILLUP  030036  5530#
$INLP   033542  3302  3333  3361  3394  3420  3445  3562  3585  3610  3638  3670  3696  3724
                3756  3763  3789  3792  3831  3836  3886  3893  3939  3944  3954  3989  3994
                4003  4065  4066  4067  4068  4069  4070  4096  4100  4130  4144  4178  4181
                4218  4221  4254  4266  4270  4332  4333  4377  4378  4403  4406  4461  4469
                4483  4526  4529  4572  4575  4578  4639  4645  4697  4700  4709  4742  4771
                4832  4833  4834  4835  4836  4837  4877  4880  4933  4939  5042  5043  5044
                5045  5146  5147  5148  5149  5256  5307  5311  5351  5364  5406  5416  5461
                5471  5585#
$INTAG  001135  3061#  5527
$ITEMB  001114  3061#  5522*  5523
$LF     001200  3061#  5522  5527  5528
$LFLG   027673  5529#*
$LOAD   032726  5585#
$LPADR  001106  3061#  3228*  3302*  5525*
$LPAI   032046  5585#
$LPERR  001110  3061#  3228*  5522  5525*
$LPW    033350  5585#
$MADR1  001234  3061#
$MADR2  001240  3061#
$MADR3  001244  3061#
$MADR4  001250  3061#
$MAIL   001202  3059  3061#  3228  5522  5525  5528
```

```
$MAMS1  001232            3061#
$MAMS2  001236            3061#
$MAMS3  001242            3061#
$MAMS4  001246            3061#
$MBADR  001002            3059#
$MFLG   027672            5529#*
$MNEW   027135            5527#
$MSGAD  001216            3061#   5529*
$MSGLG  001220            3061#   5529*
$MSGTY  001202            3061#   5529*
$MSWR   027124            5527#
$MTYP1  001233            3061#
$MTYP2  001237            3061#
$MTYP3  001243            3061#
$MTYP4  001247            3061#
$MXCNT  026266            5525#
$NULL   001154            3061#   5528
$NWTST- 000001            3302#   3326#   3355#   3388#   3414#   3439#   3562#   3573#   3596#   3624#   3658#   3682#   3710#
                          3752#   3782#   3818#   3867#   3928#   3974#   4065#   4066#   4067#   4068#   4069#   4070#   4081#
                          4123#   4163#   4202#   4251#   4332#   4333#   4377#   4378#   4391#   4435#   4508#   4560#   4611#
                          4677#   4727#   4752#   4832#   4833#   4834#   4835#   4836#   4837#   4848#   4920#   5042#   5043#
                          5044#   5045#   5146#   5147#   5148#   5149#
$OCNT   025236            5521#*
$OFS  - 033604            5585#
$OMODE  025240            5521#*
$OUTLP  033444            3332    3360    3393    3404    3419    3444    3562    3580    3581    3604    3609    3628    3637
                          3665    3666    3686    3693    3714    3721    3755    3760    3788    3791    3804    3823    3824
                          3830    3833    3875    3876    3883    3888    3932    3934    3938    3941    3978    3986    3988
                          3991    4065    4066    4067    4068    4069    4070    4087    4088    4090    4133    4134    4136
                          4140    4173    4175    4177    4180    4213    4215    4217    4220    4258    4265    4268    4332
                          4333    4377    4378    4400    4402    4405    4453    4454    4456    4458    4460    4463    4497
                          4515    4517    4519    4521    4538    4564    4568    4574    4577    4618    4621    4627    4632
                          4634    4641    4689    4690    4692    4694    4696    4699    4733    4734    4736    4760    4761
                          4762    4764    4832    4833    4834    4835    4836    4837    4866    4867    4868    4870    4872
                          4874    4876    4879    4930    4932    4935    5042    5043    5044    5045    5146    5147    5148
                          5149    5242    5250    5251    5253    5293    5304    5306    5309    5348    5353    5396    5399
                          5403    5449    5452    5454    5460    5585#
$OVER   026252            5525#
$PASS   001210            3061#   3228*   3236*   5179*   5242*   5266*   5293*   5320*   5348*   5372*   5396*   5425*   5449*
                          5480*   5525
$PASTM  001006            3059#
$POWER  030044            5530#
$PUTS   034214            5585#
$PWRDN  027676            3228    5530#
$PWRMG  030032            5530#
$PWRUP  027750            5530#
$QUES   001176            3061#   5522    5527    5528
$RDCHR  026654            5527#   5531
$RDDEC= ****** U          5531
$RDLIN  026774            5527#   5531
$RDOCT  026270            5526#   5531
$RDSZ   000010            5527#
$REGAD  001160            3061#
$REGO   001162            3061#
$RESET  033676            4738    5585#
$RTNAD  023370            5179#
```

```
$R2A  = ****** U        5531
$SAD    033672          5585#*
$SAVRE= ****** U        5531
$SAVR6  030042          5530#*
$SCOPE  026010          3228    5525#
$SETUP= 000117          3226#   3228    5179    5522    5525    5527
$STUP = 177777          3226#
$SVLAD  026216          5525#
$SVPC = 000234          3057#
$SWR    167400          2934#   2980    2981    3061    3228    3302    3326    3355    3388    3414    3439    3562    3573
                        3596    3624    3658    3682    3710    3752    3782    3818    3867    3928    3974    4065    4066
                        4067    4068    4069    4070    4081    4123    4163    4202    4251    4332    43_3    4377    4378
                        4391    4435    4508    4560    4611    4677    4727    4752    4832    4833    4834    4835    4836
                        4837    4848    4920    5042    5043    5044    5045    5146    5147    5148    5149    5179    5522
                        5525    5530
$SWREG  001224          3061#   3228
$SWRMK= 000000          2981    5525
$TESTN  001206          3061#   3302*   5525*
$TIMES  001166          3061#   3228*   3302*   3326*   3355*   3388*   3414*   3439*   3562*   3573*   3596*   3624*   3658*
                        3682*   3710*   3867*   4065*   4066*   4067*   4068*   4069*   4070*   4081*   4508*   4611*   4727*
                        4752*   4832*   4833*   4834*   4835*   4836*   4837*   5042*   5043*   5044*   5045*   5146*   5147*
                        5148*   5149*   5179*   5525*
$TKB    001146          3061#   5527    5585
$TKS    001144          3061#   5527*   5585
$TLKR   033202          5585#
$TLKW   033066          5585#
$TMDAT  001356          3061#   3874*   3875    3882*   3883    3886    3887*   3888    3994    3995    4178    4179*   4180
                        4617*   4618    4620*   4621    4631*   4632    4633*   4634    4639    4640*   4641    4929*   4930
                        5042*   5043*   5044*   5045*   5146*   5147*   5148*   5149*   5242*   5250    5251    5252*   5253
                        5256    5257    5293*   5303*   5304    5305*   5306    5307    5308*   5309    5348*   5350*   5351
                        5352*   5353    5396*   5398*   5399    5402*   5403    5406    5407    5449*   5451*   5452    5453*
                        5454    5459*   5460    5461    5462
$TMP0   001164          3061#   5576    5577    5578    5579    5580
$TN   - 000147          2980#   3302#   3326#   3355#   3388#   3414#   3439#   3562#   3573#   3596#   3624#   3658#   3682#
                        3710#   3752#   3782#   3818#   3867#   3928#   3974#   4065#   4066#   4067#   4068#   4069#   4070#
                        4081#   4123#   4163#   4202#   4251#   4332#   4333#   4377#   4378#   4391#   4435#   4508#   4560#
                        4611#   4677#   4727#   4752#   4832#   4833#   4834#   4835#   4836#   4837#   4848#   4920#   5042#
                        5043#   5044#   5045#   5146#   5147#   5148#   5149#
$TOUT   033636          5585#
$TPB    001152          3061#   5528*
$TPFLG  001157          3061#   5528
$TPS    001150          3061#   5528
$TRAP   030054          3228    5531#
$TRAP2  030076          5531#
$TRP  = 000014          5531#   5532#
$TRPAD  030110          5531#
$TSTM   001004          3059#
$TSTNM  001102          3061#   3302*   5179*   5522    5525*
$TTYIN  027102          5527#
$TYPBN= ****** U        5531
$TYPDS  025564          5524#   5531
$TYPE   027146          5528#   5529    5531
$TYPEC  027360          5527    5528#
$TYPEX  027426          5528#
$TYPOC  025040          5521#   5531
$TYPON  025054          5521#   5531
```

```
$TYPOS  025014            5521#   5531
$UNIT   001214            3061#
$UNITM  001010            3059#
$USWR   001226            3061#
$UTK    034020            5585#
$VECT1  001252            3061#
$VECT2  001254            3061#
$XTSTR  026022            5525#
$$GET4= 000000            5179#
$OFILL  025237            5521#*
$40CAT= ****** U          5522    5525
.       = 034324          2982#   2983#   3057#   3058#   3059#   3061#   3223#   3228    3232#   5179    5266#   5320#   5372#
                          5425#   5480#   5494#   5522    5523#   5524#   5525    5527#   5528    5529#   5530    5565#   5585#
.DVLS   001666            3223#   3238*   3266*   5241*   5292*   5347*   5395*   5448*   5585*
.$ASTA= ****** U          5529
.$X     = 001000          3059#
```

```
ADTST   3277#  3302
AREPT   5047#
BREPT   5151#
BRTM    4781#  4832  4833  4834  4835  4836  4837
CADT    4953#  5042  5043  5044  5045
CADTB   5072#  5146  5147  5148  5149
CBC     3562#
CBTC    4342#  4377  4378
CBT25   4278#  4332  4333
COMMEN  2993#  3338  3341  3365  3368  3397  3401  3424  3428  3449  3453  3562  3589  3593  3616
        3621   3644  3649  3674  3678  3702  3707  3730  3735  3767  3771  3796  3801  3840  3847
        3898   3902  3947  3950  3958  3963  3997  4000  4007  4015  4065  4066  4067  4068  4069
        4070   4105  4110  4147  4151  4184  4187  4224  4236  4273  4275  4332  4333  4377  4378
        4409   4412  4472  4479  4488  4494  4532  4535  4581  4594  4649  4653  4703  4706  4713
        4716   4746  4749  4774  4778  4832  4833  4834  4835  4836  4837  4884  4895  4943  4949
        5042   5043  5044  5045  5146  5147  5148  5149  5260  5264  5315  5318  5367  5370  5419
        5423   5474  5478  5508  5514
CSRDTA  3464#  3562
DFC     3062#  3072  3079  3086  3093  3100  3107  3114  3122  3129  3136  3150  3157  3164  3178
        3185   3192  3199  3206  3213  3220
ECALL   3501#  3562
ECB     3034#  3338  3341  3365  3368  3397  3401  3424  3428  3449  3453  3562  3589  3593  3616
        3621   3644  3649  3674  3678  3702  3707  3730  3735  3767  3771  3796  3801  3840  3847
        3898   3902  3947  3950  3958  3963  3997  4000  4007  4015  4065  4066  4067  4068  4069
        4070   4105  4110  4147  4151  4184  4187  4224  4236  4273  4275  4332  4333  4377  4378
        4409   4412  4472  4479  4488  4494  4532  4535  4581  4594  4649  4653  4703  4706  4713
        4716   4746  4749  4774  4778  4832  4833  4834  4835  4836  4837  4884  4895  4943  4949
        5042   5043  5044  5045  5146  5147  5148  5149  5260  5264  5315  5318  5367  5370  5419
        5423   5474  5478  5508  5514
ENDCOM  2993#  3338  3341  3365  3368  3397  3401  3424  3428  3449  3453  3562  3589  3593  3616
        3621   3644  3649  3674  3678  3702  3707  3730  3735  3767  3771  3796  3801  3840  3847
        3898   3902  3947  3950  3958  3963  3997  4000  4007  4015  4065  4066  4067  4068  4069
        4070   4105  4110  4147  4151  4184  4187  4224  4236  4273  4275  4332  4333  4377  4378
        4409   4412  4472  4479  4488  4494  4532  4535  4581  4594  4649  4653  4703  4706  4713
        4716   4746  4749  4774  4778  4832  4833  4834  4835  4836  4837  4884  4895  4943  4949
        5042   5043  5044  5045  5146  5147  5148  5149  5260  5264  5315  5318  5367  5370  5419
        5423   5474  5478  5485  5508  5514
ERROR   2993#  3339  3366  3398  3425  3450  3562  3590  3617  3645  3675  3703  3731  3768  3797
        3841   3899  3948  3959  3998  4008  4065  4066  4067  4068  4069  4070  4106  4148  4185
        4225   4274  4332  4333  4377  4378  4410  4473  4489  4533  4582  4650  4704  4714  4747
        4775   4832  4833  4834  4835  4836  4837  4885  4945  5042  5043  5044  5045  5146  5147
        5148   5149  5262  5316  5368  5420  5475  5585
ESCAPE  2993#
GETPRI  2993#
GETSWR  2993#
MOVEI    170#  3302  3333  3361  3394  3420  3445  3562  3585  3610  3638  3670  3696  3724  3756
        3763   3789  3792  3831  3836  3886  3893  3939  3944  3954  3989  3994  4003  4065  4066
        4067   4068  4069  4070  4096  4100  4130  4144  4178  4181  4218  4221  4254  4266  4270
        4332   4333  4377  4378  4403  4406  4461  4469  4483  4526  4529  4572  4575  4578  4639
        4645   4697  4700  4709  4742  4771  4832  4833  4834  4835  4836  4837  4877  4880  4933
        4939   5042  5043  5044  5045  5146  5147  5148  5149  5256  5307  5311  5351  5364  5406
        5416   5461  5471  5585
MOVEM    157#  3332  3360  3393  3404  3419  3444  3562  3580  3581  3604  3609  3628  3637  3665
        3666   3686  3693  3714  3721  3755  3760  3788  3791  3804  3823  3824  3830  3833  3875
        3876   3883  3888  3932  3934  3938  3941  3978  3986  3988  3991  4065  4066  4067  4068
        4069   4070  4087  4088  4090  4133  4134  4136  4140  4173  4175  4177  4180  4213  4215
```

```
ZP29      3850#  3867
ZP3E      4065#  4066#  4067#  4068#  4069#  4070#
ZP3EA     4072#  4081
ZP30      3918#  3928
ZP31      3965#  3974
ZP310     4380#  4391
ZP311     4415#  4435
ZP312     4498#  4508
ZP33      4113#  4123
ZP34      4154#  4163
ZP35      4190#  4202
ZP36      4239#  4251
ZP37      4332#  4333#
ZP38      4335#
ZP39      4377#  4378#
ZP42      4544#  4560
ZP43      4597#  4611
ZP44      4666#  4677
ZP45      4719#  4727
ZP46E     4832#  4833#  4834#  4835#  4836#  4837#
ZP47      4839#  4848
ZP6E      4901#  4920
ZP61E     5042#  5043#  5044#  5045#  5146   5147   5148   5149
ZZ1       3456#  3562   3573   3596   3624   3658   3682   3710
Z1        3302#
Z2        3303#  3326
Z3        3344#  3355
Z4        3371#  3388
Z6        3406#  3414
Z7        3432#  3439
$CAI.      747#  5585
$DMAST    1792#
$DMDT     2821#
$MMAST     761#
$$CMRE    3061#
$$CMTM    3061#
$$ESCA    2993#
$$NEWT    2993#  3302   3326   3355   3388   3414   3439   3562   3573   3596   3624   3658   3682   3710   3752
          3782   3818   3867   3928   3974   4065   4066   4067   4068   4069   4070   4081   4123   4163   4202
          4251   4332   4333   4377   4378   4391   4435   4508   4560   4611   4677   4727   4752   4832   4833
          4834   4835   4836   4837   4848   4920   5042   5043   5044   5045   5146   5147   5148   5149
$$SET     5531#  5532
$$SETM    3228#
$$SKIP    2993#
.EQUAT    2930#  2993
.HEADE    2929#  2980
.KMADR      55#  3223
.KSIS      184#  3238
.LOADL     458#  5585
.LPAIN     209#  5585
.PUTCS     417#  5585
.RESET     328#  5585
.SETTR    2929#
.SETUP    2929#  3226
.SWRHI    2930#  2981
.SWRLO    2981#
```

```
.TRMTR   2929#
.UTK      698#   5585
.$ACT1   2932#   3057
.$APTB   2932#   3061#
.$APTH   2932#   3059
.$APTY   2932#   5529
.$CATC   2930#   2982
.$CMTA   2930#   3061
.$EOP    2931#   5179
.$ERRO   2931#   5522
.$ERRT   2931#   5523
.$INLP    651#   5585
.$MMAC    141#
.$OUTL    609#   5585
.$POWE   2930#   5530
.$RDOC   2929#   2932#   5526
.$READ   2931#   5527
.$SCOP   2931#   5525
.$TLKW    510#   5585
.$TOUT   3223#   5585
.$TRAP   2929#   5531
.$TYPD   2931#   5524
.$TYPE   2931#   5528
.$TYPC   2930#   5521


. ABS.  034324     000     OVR   RO    ABS   GBL   D
        000000     001     CON   RW    REL   LCL   I


 ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

CRLPGB,CRLPGB/CRF-DRLPA.MAC,CRLPGB
RUN-TIME: 53 47 2 SECONDS
RUN-TIME RATIO: 404/103=3.9
CORE USED:  46K  (91 PAGES)
```